А. Б. Мерков

РАСПОЗНАВАНИЕ ОБРАЗОВ

Введение в методы статистического обучения

URSS

Введение в методы статистического обучения

А.Б.Мерков

версия: 31 декабря 2014*

Аннотация

Предлагаемый текст, за вычетом приложений, приблизительно соответствует полугодовому курсу по общим методам статистического обучения. Он, по мнению автора, описывает некоторую центральную часть статистического обучения, отсекая фундамент, пристройки и надстройки. С одной стороны, текст не содержит теории статистического обучения, хотя ссылается на нее, с другой стороны, не содержит методов, сильно зависящих от специфики данных, например, изображений, речи или текстов, с третьей, посвящен почти исключительно задаче распознавания, а с четвертой, не привязан ни к каким программным реализациям машинного обучения. Предпринята попытка компактно и единообразно, хотя бы и не полно, изложить основные современные "универсальные" методы распознавания и используемый в них математический аппарат. Заметная часть описываемых методов строго обоснована: простые технические детали доказательств сформулированы и предложены в качестве упражнений, более сложные, но не слишком громоздкие, доказательства предъявлены. Ради компактности изложения количество иллюстраций — как картинок, так и примеров — минимизировано, зато приводятся многочисленные литературные ссылки, в том числе и ссылки на доступные электронные копии статей.

В приложениях рассматриваются не столь универсальные методы распознавания, основанные на построении вероятностных моделей для специфических данных. Они могут служить основой для еще одного полугодового курса.

В незавершенном виде этот текст был издан в виде книги [84]. Дописанные потом главы, дополненные для читабельности копией вводной главы из [84], были изданы отдельной книгой [85].

*Это — расширенный конспект курса лекций (2008) и семинаров (2002–2005) по распознаванию. Он постоянно меняется, в нем исправляются старые ошибки и появляются новые. Бдите!

Содержание

1 Введение

Никто не обнимет необъятного.
К.П.Прутков

Этот раздел является чем-то вроде аннотированного и иллюстрированного, но не претендующего на полноту, предметного указателя по методам статистического обучения. Некоторые важные понятия только декларируются, а разъясняются (или не разъясняются) в последующих разделах. Неполнота усугубляется тем, что в этой еще не стабилизировавшейся области деятельности

одни и те же понятия часто называют разными словами и по-разному переводят на разные языки. Наряду с русскими терминами, иногда неканоническими, приводятся их англоязычные прототипы.

Типичная задача статистического обучения (*statistical learning*, или *machine learning*, или *pattern recognition*, традиционный неправильный перевод последнего варианта — *распознавание образов*), в самом общем виде выглядит так. Есть некоторое количество *объектов*, с какими-то *наблюдаемыми свойствами* и какими-то *ненаблюдаемыми, но известными*. Построить алгоритм, правильно вычисляющий эти *ненаблюдаемые свойства* по этим *наблюдаемым*, причем не только для заранее предъявленных *объектов*, но и для любых других. Или хотя бы алгоритм, *ошибающийся* не очень *часто* и не очень *сильно*. Причем нужно не однократно построить такой алгоритм, а создать алгоритм более высокого уровня (мета-алгоритм, метод, технологию, ...), строящий вычисляющий алгоритм по любому предъявленному набору объектов, чтобы этот алгоритм *часто* получался *приемлемым*. Саму задачу будем называть *распознаванием* (*recognition*), решающий ее алгоритм — *распознавателем* (*recognizer*, *learner*), а построение этого алгоритма — *обучением* (*learning*, *training*, *fitting*) распознавателя.

Для формализации постановки задачи нужно придать смысл всем выделенным словам предыдущего абзаца. Но сперва приведем неформальные примеры таких задач:

Узнавание образцов. Имеется некоторое количество картинок, на каждой из которых нарисована кошка (или треугольник, или жираф, или самолет, или конкретный человек) и, возможно, некоторое количество картинок, на каждой из которых она (он) отсутствует. Построить алгоритм, определяющий наличие кошки (треугольника и т.д.) на картинке.

Распознавание рукописных букв. Имеется некоторое количество картинок, на каждой из которых нарисована буква, и известно, какая именно. Построить алгоритм, узнающий нарисованные буквы.

Распознавание рукописного текста. Имеется некоторое количество отсканированных страниц рукописей и параллельных текстовых файлов с правильно прочитанным содержимым рукописей. Построить алгоритм, читающий рукописный текст с листа.

Распознавание голосовых команд. Имеется некоторое количество звуковых файлов, содержащих записи произнесения (utterance) голосовых команд из конечного ассортимента (например, слов “да” и “нет” или цифр для голосового набора номера) и знание, в каком файле какая команда. Построить алгоритм, понимающий голосовые команды.

Распознавание речи. Имеется некоторое количество звуковых файлов, содержащих записи естественной речи на каком-то языке, и текстовых файлов — их расшифровок. Построить алгоритм, распознающий речь и записывающую ее в виде текста.

Медицинская диагностика. Имеется некоторое количество историй болезни и приложенных к ним результатов обследований больных. Построить алгоритм, по результатам обследований нового больного ставящий ему диагноз, назначающий лечение и/или прогнозирующий результаты лечения. Или, что более реально, подсказывающий врачу наиболее правдоподобные диагнозы.

Геологическая диагностика. Про некоторое количество разработанных нефтяных месторождений известны данные их предварительной геологической разведки (например, сейсмограммы) и результаты их эксплуатации. Построить алгоритм, предсказывающий эксплуатационные характеристики, в первую очередь мощность, разведанных, но еще не вскрытых месторождений.

Экономическое прогнозирование. Имеются данные о еженедельных объемах продаж нескольких тысяч видов товаров в нескольких сотнях магазинов за несколько лет. Построить алгоритм, предсказывающий спрос на ближайший месяц.

Кроме распознавания теория статистического обучения занимается и другими задачами, например, задачей *кластеризации* (синоним: кластерный анализ, *cluster analysis*). Есть некоторое количество *объектов* с какими-то *наблюдаемыми свойствами*. Построить алгоритм, разбивающий объекты на группы, называемые *кластерами* (*cluster*), и алгоритм, определяющий, какому кластеру принадлежит объект, так чтобы объекты внутри каждого кластера были *похожи друг на друга*, а объекты из разных кластеров — *непохожи*, причем чтобы это было как правило (т.е. статистически) верно не только для заранее предъявленных объектов, но и для любых других, предъявляемых впоследствии. Кластеризация и ее аналоги часто возникают в качестве вспомогательных задач в распознавании, сжатии данных и др.

В дальнейшем обсуждается в основном обучение обучающихся (тавтология...) алгоритмов, причем достаточно общих, не использующих специфики задачи. Подробности того, что предшествует обучению, как можно его проводить, что за ним следует и как все это интерпретировать с теоретико-вероятностной точки зрения, хорошо описаны в книгах [14, 15]. Теоретические оценки ошибок распознавания, гарантирующие, что обучение вообще возможно, подробно рассмотрены в книге [114]. Основы теории и классические методы распознавания изложены в книге [117]. Практически в любой толстой книге, содержащей в названии слова "pattern recognition", "machine learning" (например, [15]) или "statistical learning", (например, [50] или [113]) описаны разные распознающие системы, способы их обучения, примеры применения и оценки качества работы, хотя одни и те же вещи в разных книгах могут называться по-разному.

1.1 Постановка задач распознавания и обучения распознавателя

1.1.1 Входы и выходы

Распознаваемые *объекты* кодируются наборами (векторами) *признаков* (*feature*), либо являющихся числовыми, либо принимающих конечное множество значений. Допустимые значения векторов признаков образуют *пространство признаков* $\mathcal{X}$. Например, если удается закодировать объекты d числовыми признаками, то $\mathcal{X}$ — это подмножество стандартного d-мерного евклидова пространства $\mathbb{R}^d$. Это не всегда возможно, а даже когда возможно, не всегда просто. Во-первых, объекты могут быть неограниченно длинными (как распознаваемая речь или рукопись) и либо их нужно научиться разбивать на части, помещающиеся в пространство фиксированной размерности, (сегментировать) и обрабатывать по частям, либо наоборот, отказаться от идеи кодировать объект фиксированным числом признаков. Во-вторых, один и тот же объект может быть оцифрован по-разному (страница может быть отсканирована с разными

разрешениями или по-разному положена на сканер) и хорошо бы уметь преобразовывать разные представления (в данном примере — разные картинки) в один и тот же вектор признаков. В-третьих, какие-то свойства объектов, даже если они традиционно кодируются числами, могут быть существенно нечисловыми: наличие или отсутствие чего-либо, группа крови, артикул товара, словесное описание и т.п.. И в-четвертых, способ кодирования и даже размерность d может зависеть от множества предъявленных объектов.

Для простоты формулировок мы пока не будем рассматривать неограниченно длинные объекты и их сегментацию. Способы унификации представлений объектов (например, как пересчитать растровую картинку любого размера на растр фиксированного размера и какой размер разумно фиксировать, или вообще не пересчитывать ее в растровую, а разложить по какому-либо базису — тригонометрическим многочленам, wavelet'ам и т.п.) существенно зависят от конкретной задачи и мы их также не рассматриваем. А вот про кодирование дискретных признаков есть общие рекомендации:

- Двоичные признаки (например, нет или да) кодировать традиционно: числом, принимающим значение 0 или 1, соответственно.

- Признаки, принимающие конечное число $k > 2$ неарифметических значений (группа крови, категория товара, ...) кодировать k независимыми числовыми признаками, называемыми *вторичными* или *производными* (*derived features*): j-й вторичный признак x^j равен 1, если исходный признак принимал j-е значение, и равен 0 в противном случае. Вторичные признаки удовлетворяют соотношению $\sum_{j=1}^{k} x^j = 1$. Этот способ кодирования легко обобщается на случай, когда значение исходного признака известно не достоверно, а с как-то оцененной вероятностью.

- Признаки, принимающие конечное упорядоченное множество значений, (являющиеся дискретизацией числовых признаков, например, признак (холодный, нормальный, горячий) вместо температуры, или оценки успеваемости по пятибалльной шкале) можно кодировать и одним числом (приблизительно угаданной температурой), но надежнее, хотя и дороже, набором нулей и единиц из предыдущего пункта.

- Целочисленные признаки тоже можно считать действительнозначными (например, количество проданных оранжевых пиджаков 56-го размера), но если они принимают заранее известное небольшое число значений (например, число конечностей у членистоногих — от 0 до 10), лучше кодировать их набором нулей и единиц из предыдущих пунктов.

Если объекты удалось закодировать набором из d числовых признаков, т.е. точкой стандартного евклидова пространства $\mathbb{R}^d$, признаки хорошо бы отнормировать так, чтобы они лежали в каком-либо заранее известном компакте, например, в единичном кубе или шаре.

Ответы (*response*) распознавания — и ожидаемые, и вычисленные распознавателем — кодируются (если временно забыть про распознавание длинных объектов) так же, как признаки объектов, т.е. точками (векторами) в некотором *пространстве ответов* $\mathcal{Y}$, например, в q-мерном евклидовом пространстве $\mathbb{R}^q$. Тогда получается, что распознаватель нужно учить вычислять некоторую функцию $f : \mathbb{R}^d \to \mathbb{R}^q$, про которую известны только ее значения в конечном числе точек. Но для некоторых ответов распознавания специального вида, например, принимающих ровно два значения, есть специфические методы обучения. Иногда бывает удобно, по крайней мере теоретически, разбить распознаватель на несколько более простых, каждый из которых вычисляет ровно один числовой или двузначный дискретный ответ.

1.1.2 Классификация и регрессия

Распознавание численной (скалярной или векторной) характеристики объекта называется *регрессией*. Строгое математическое определение регрессии — это условное математическое ожидание одной случайной величины относительно другой, и во многих задачах регрессия распознавательная является-таки регрессией математической.

Распознавание качественной (дискретной) характеристики объекта называется *классификацией*, число возможных значений q — числом классов, а множество объектов, для которых эта характеристика принимает j-е значение - j-м классом. Ответом распознавателя для каждого объекта лучше считать не номер класса, к которому распознаватель относит объект, а более полное знание, исходя из которого номер класса легко посчитать, — q-мерный вектор "уверенностей" (*confidence*) в принадлежности объекта каждому из классов. Тем самым, классификация превращается в специальный случай регрессии[1].

В теоретически удобных случаях, когда пространство всех возможных объектов разбито на классы и на нем определена вероятность, "уверенности" в кавычках можно заменить вероятностями без кавычек и для них выполнены вероятностные нормировки (каждая из них принимает значения от 0 до 1, сумма всех равна 1). На практике задачу классификации иногда обобщают на случай классов, объединение которых не равно всему пространству, (такая задача сводится к "правильно поставленной" задаче введением дополнительного класса), и даже на случай пересекающихся классов. При этом "уверенности" могут не быть вероятностями в каком-либо строгом смысле. Например, при распознавании рукописных букв на вход распознавателя может попасть вообще не буква. Кроме того, некоторые написания букв 'О' заглавное, 'о' строчное и цифры '0' абсолютно неразличимы, но считать, что такая замечательно написанная буква 'о' является буквой 'о' с вероятностью всего лишь около $\frac{1}{3}$ не всегда полезно для распознавания слова в целом.

В учебной литературе чаще всего изучается задача классификации с двумя более-менее равноправными непересекающимися классами, покрывающими все пространство, или, что почти то же самое, отделение объектов одного класса от всех остальных. В этом случае распознаватель вычисляет ровно одно число ("уверенность" в принадлежности объекта первому классу) и для принятия решения достаточно сравнить эту "уверенность" с некоторым порогом.

Классы могут быть и неравноправны: объекты одного класса могут встречаться на несколько порядков реже, чем объекты другого класса. В этом случае распознавать, какому классу объект принадлежит с большей вероятностью, довольно бессмысленно: почти всегда ко второму. Осмысленно обучать распознаватель оценивать вероятность принадлежности первому (очень редкому) классу.

Дополнительный класс "все остальное" может быть равноправен с настоящими классами (одним или более), а может быть и неравноправен еще и в следующем смысле: объекты дополнительного класса могут быть представлены при обучении классификатора, а могут и отсутствовать. То есть можно обучать классификатор узнаванию буквы 'а', не предъявляя ему ни других букв, ни знаков препинания, ни клякс. Другой пример: можно обучать систему голосового набора телефонных номеров распознаванию цифр, не предъявляя ей остальных слов, произносимых при разговоре с телефонным оператором, например, "пожалуйста".

Классификация с любым числом классов может быть сведена, причем разными способами, к решению конечного числа задач двухклассовой классифика-

[1] Даже в название некоторых методов обучения классификации входит слово "регрессия", см. раздел 2.2.2.

ции. Иногда оказывается, что обучить и скомбинировать несколько двухклассовых классификаторов легче, чем обучить один многоклассовый. А иногда — наоборот, см., например, раздел 4.2.7.

Получилось, что регрессия (распознавание непрерывных величин) ущемлена в правах по сравнению с классификацией (распознаванием дискретных величин): при наличии вероятностной модели в дискретном случае можно предсказывать распределение вероятностей ответов, а в непрерывном — только сам ответ (например, его математическое ожидание) без каких-либо оценок уверенности в нем. Равноправие можно частично восстановить, предположив, что распределение принадлежит к какому-либо конечномерному семейству, и добавив к пространству ответов дополнительные прямые сомножители, описывающие какие-либо еще параметры распределения. Реальный пример таких параметров — дисперсия в случае одномерной регрессии и матрица ковариации в случае многомерной. А в широко распространенном частном случае, когда распределение является фиксированной центрально симметричной мерой, сдвинутой на какой-то вектор, например, гауссовым распределением с фиксированной матрицей ковариации, единственный распознаваемый (векторный) параметр — это в точности математическое ожидание.

1.1.3 Пример: распознавание методом ближайших соседей

Рассмотрим пример распознавателя, который и обучать-то почти не надо. Пусть пространство признаков $\mathcal{X}$ — метрическое, пространство ответов $\mathcal{Y}$ — любое, и имеется *обучающий набор* из N объектов с известными признаками и ответами, т.е. набор $T = ((x_1, y_1), \ldots, (x_N, y_N)) \in (\mathcal{X} \times \mathcal{Y})^N$. В качестве ответа распознавателя для объекта x берется ответ y_i его ближайшего в смысле метрики на $\mathcal{X}$ "соседа" x_i из обучающего набора T. Неформальное обоснование такого ответа: "за неимением дополнительной информации предположим, что с изменением признаков ответ меняется медленно".

Нужно еще как-то определить ответ в случае наличия нескольких равноудаленных ближайших соседей. Это можно сделать следующими способами:

- ответ любого из ближайших соседей;
- если пространство ответов $\mathcal{Y}$ — линейное, например, $\mathbb{R}$ (одномерная регрессия), то ответы ближайших соседей можно усреднить;
- если пространство ответов $\mathcal{Y}$ — конечное с небольшим числом элементов, например, $\{0, 1\}$ (двухклассовая классификация), то из ответов ближайших соседей можно выбрать самый частый.

После того, как выбран способ усреднения в случае, если ближайших соседей оказывается несколько, *метод ближайшего соседа* (*NN*, *nearest neighbor*) естественно обобщается до *метода k ближайших соседей* (k-*NN*) при $k > 1$.

Распознавание методом $k = 1$ ближайшего соседа не делает ни одной ошибки на предъявленном ему наборе T, но может ошибаться на неизвестных ему векторах признаков. Распознавание методом $k > 1$ ближайших соседей не обязательно безошибочно распознает точки обучающего набора, зато при небольших k, как правило, меньше ошибается на неизвестных ему векторах. Впрочем, что значит "меньше ошибается" еще не определено. При N ближайших соседях распознаватель дает постоянный ответ, не зависящий от признаков.

Хотя сравнивать этот метод распознавания пока не с чем, анонсируем его достоинства и недостатки. Обучение распознавателя при методе k ближайших соседей тривиально и сводится к запоминанию обучающего набора. Распознавание тоже тривиально, но очень трудоемко, и трудоемкость растет пропорционально произведению объема обучающего набора и размерности пространства

признаков. Существует огромное количество методов ускорения поиска ближайших соседей для классических метрик (например, евклидовой), см. ссылки в [75], но особо быстрым он не получается. Однако в некоторых задачах он успешно применяется, особенно при специальным образом построенной метрике на пространстве признаков (например, [49]).

В разделе 1.2.2 будет показан пример того, что при естественных дополнительных предположениях асимптотически — при росте обучающего набора — распознавание методом ближайшего соседа теоретически должно работать с ошибкой, близкой к минимально возможной. Но при очень большом обучающем наборе оно практически не работоспособно из-за чрезмерной требуемой памяти и времени. А при малом обучающем наборе, не заполняющем достаточно плотно пространство признаков, метод ближайшего соседа работает плохо. В частности, если пространство признаков $\mathcal{X}$ — евклидово большой размерности (скажем, больше 40) и допустимые векторы признаков не сосредоточены вблизи какой-нибудь маломерной (скажем, 20-мерной) поверхности, то никаких шансов плотно заполнить интересную для распознавания область пространства обучающим набором нет. В литературе этот эффект известен под названием "*проклятие размерности*" (*curse of dimensionality*).

Кроме тривиального запоминания обучающего набора и, возможно, построения вспомогательных структур для ускорения распознавания, метод k ближайших соседей есть чему учить нетривиально: подбору наилучшего значения k. Кроме того ближайшие соседи зависят от используемой метрики и при разных метриках получаются разные распознаватели — какие-то лучше, какие-то хуже. А в каком смысле лучше или хуже, и как обучать?

1.1.4 Формальная постановка задачи обучения распознавателя: минимизация эмпирического риска

Задачу обучения можно формализовать по-разному. Далее в качестве основной формализации фигурирует *минимизация эмпирического риска* (см., например, [115]). Для сравнения в разделах 1.2.4 и 1.2.6 будут также приведены и иногда будут использоваться формализации в терминах вероятностных моделей и соответствующие им методы обучения: байесовский[2], максимизации апостериорной вероятности и максимизации правдоподобия.

Имеется:

пространство (векторов) признаков $\mathcal{X}$**,** точками которого кодируются распознаваемые объекты, например, d-мерное евклидово пространство $\mathbb{R}^d$;

пространство ответов $\mathcal{Y}$**,** точками которого кодируются результаты распознавания, например, q-мерное пространство $\mathbb{R}^q$;

пространство $\mathcal{F}$ ***распознавателей*** $f : \mathcal{X} \to \mathcal{Y}$**,** например, в случае евклидовых пространств $\mathcal{X}$ и $\mathcal{Y}$, — непрерывных, дважды дифференцируемых, линейных, полиномиальных и т.п.;

пространство $\mathcal{P}$ ***распределений*** **(вероятностных мер) на** $\mathcal{X} \times \mathcal{Y}$**,** например, в случае евклидовых пространств $\mathcal{X}$ и $\mathcal{Y}$, — абсолютно непрерывных по мере Лебега, возможно еще и со всюду положительной и/или гладкой плотностью, гауссовых смесей и т.п., удовлетворяющих каким-то специфическим для задачи условиям;

[2]Именем Томаса Байеса называются несколько *совершенно разных* распознавателей, методов обучения, и т.п., к которым он прямого отношения не имеет. Иногда это название намекает на использование известной *формулы Байеса* для условных вероятностей, а иногда просто является синонимом слова "вероятностный".

функция *штрафа* $E\colon \mathcal{Y}\times\mathcal{Y}\times\mathcal{X}\to\mathbb{R}$**,** называемая также функцией *ошибки*, *потерь*, *риска*, *loss function*, *error function*, ..., как правило, неотрицательная и равная 0 при совпадении первого параметра (прогнозируемого ответа) и второго (истинного ответа) и редко зависящая от третьего параметра (вектора признаков); например, в случае евклидова пространства $\mathcal{Y}$ применяется *квадратичный штраф* $E(r,y,x)=\|r-y\|^2$, а в случае дискретного пространства — *0-1-штраф* $E(r,y,x)=\begin{cases}0 & \text{при } r=y\\ 1 & \text{при } r\neq y\end{cases}$; далее будут рассматриваться только функции штрафа вида $E:\mathcal{Y}\times\mathcal{Y}\to\mathbb{R}$, от точки пространства признаков $\mathcal{X}$ не зависящие[3];

обучающий набор $T=((x_1,y_1),\ldots,(x_N,y_N))$**,** состоящий из пар (вектор признаков, ответ) $(x_i,y_i)\in\mathcal{X}\times\mathcal{Y}$, которые считаются значениями независимых случайных величин с одним и тем же, но совершенно неизвестным, распределением $\pi\in\mathcal{P}$.

Обратите внимание на то, что не требуется, чтобы по значению признаков x правильный ответ y был определен однозначно. Определено лишь, хотя и неизвестно, зависящее от совместного распределения π и признаков x распределение $\pi(y|x)$ вероятностей правильного ответа, в случае непрерывных распределений и непрерывного пространства ответов имеющее плотность

$$p_{\pi;x}(y)=p_\pi(y|x)=\frac{p_\pi(x,y)}{p_\pi(x)}=\frac{p_\pi(x,y)}{\int_{y'\in\mathcal{Y}}p_\pi(x,y')dy'}. \tag{1}$$

Хочется по $\mathcal{X}$, $\mathcal{Y}$, $\mathcal{F}$, $\mathcal{P}$, E и T построить распознаватель $f\in\mathcal{F}$, минимизирующий математическое ожидание штрафа (ожидание риска, ошибки, *средний риск*,...)[4] по распределению π

$$E_\pi(f)=\int_{(x,y)\in\mathcal{X}\times\mathcal{Y}}E(f(x),y,x)d\pi(x,y)\to\min_{f\in\mathcal{F}}, \tag{2}$$

или обещающий про штраф что-нибудь еще хорошее, например, что большим он бывает редко, т.е. при некоторых $\epsilon,\eta>0$

$$\pi(\{(x,y)\in\mathcal{X}\times\mathcal{Y}|E(f(x),y,x)>\epsilon\})<\eta. \tag{3}$$

Такое желание выглядит странным и нереалистичным, поскольку про распределение π, от которого все зависит, почти ничего не известно и наоборот, известные обучающие данные T ни на что не влияют. На самом деле про π известно, что обучающие данные T являются случайными с распределением π, что позволяет методом Монте-Карло приблизить ожидание штрафа (2) суммой

$$\int_{(x,y)\in\mathcal{X}\times\mathcal{Y}}E(f(x),y,x)d\pi(x,y)\approx E(f,T)=\frac{1}{N}\sum_{i=1}^{N}E(f(x_i),y_i,x_i)\ , \tag{4}$$

называемой *средним штрафом обучения* или *средней ошибкой обучения* (*training error*), *эмпирическим риском* и т.д.. Теперь можно попробовать подменить минимизацию интеграла $E_\pi(f)$ *минимизацией эмпирического риска*

$$E(f,T)=\frac{1}{N}\sum_{i=1}^{N}E(f(x_i),y_i,x_i)\to\min_{f\in\mathcal{F}} \tag{5}$$

[3]Хотя в приведенных простых примерах функция штрафа симметрична по первым двум аргументам, вообще говоря, это не требуется.

[4]Здесь и далее одной и той же буквой E обозначаются разные функции штрафа (ошибки, Error) и их усреднения или суммы по обучающим наборам, различаемые количеством и типами параметров.

— задача вполне разрешимая при не слишком сложной функции штрафа E и несложном пространстве распознавателей $\mathcal{F}$.

Но нужно понимать, что распознаватель, полученный в результате обучения минимизацией штрафа (5) зависит от обучающего набора T, а какие-либо оценки для приближения (4) справедливы для функций f, не зависящих от набора T. Ожидание штрафа (2) можно оценить, взяв *набор тестовых данных* $T' = ((x'_1, y'_1), \ldots, (x'_{N'}, y'_{N'}))$, аналогичных обучающим и независимых от них, и посчитав для них *средний штраф тестирования* (*среднюю ошибку на тесте*, *test error*) $E(f, T')$. И если тестовые данные действительно независимы, то отклонение среднего штрафа тестирования $E(f, T')$ от его ожидания $E_\pi(f)$ можно оценить стандартными методами статистики. Результат будет неожиданно плохим: $E(f, T') > E(f, T)$, особенно при большом пространстве распознавателей и малом количестве обучающих векторов. Что делать?...

Прежде, чем обсуждать, что делать, приведем пример алгоритма распознавания, обучаемого методом минимизации эмпирического риска.

1.1.5 Пример: распознающие деревья

Распознающее дерево (*recognition tree* или *дерево решений*, *decision* tree) — это распознаватели следующего вида. Для распознаваемого объекта проводится конечная последовательность сравнений значений его признаков с константами на равенство или неравенство, причем от результатов каждого сравнения зависит, что делать дальше: продолжать сравнивать что-то с чем-то или давать какой-то ответ распознавания. То есть, распознавание реализуется как двоичное дерево вложенных операторов вида

```
if ( x[j] ?? t[k] ) then    ...
                    else    ...
```

завершающихся листьями

```
return ( r[l] )
```

где x — вектор признаков, t — массив пороговых значений, r — массив возможных ответов, а через "??" обозначена операция сравнения. Обучение дерева состоит в выборе его структуры, операций сравнения, сравниваемых признаков и порогов в каждой вершине ветвления, и ответов в каждом листе. В терминах отображения $f : \mathcal{X} \to \mathcal{Y}$ из пространства признаков в пространство ответов, распознающее дерево вычисляет функцию, кусочно-постоянную на параллелепипедах, не обязательно ограниченных, стороны которых параллельны координатным гиперплоскостям, и которые получаются процессом последовательного "разрезания пополам" пространства признаков.

Распознающими деревьями являются, например, почти все определители растений. Эти деревья были, в основном, обучены в XVIII–XIX веках без применения вычислительной техники, причем при обучении допустимые ответы не были известны заранее, т.е. сначала решалась не задача распознавания, а задача кластеризации.

Алгоритмы машинного обучения деревьев для классификации и регрессии появились в 1960-х годах и достигли расцвета к концу 1980-х (CART [20], C4.5 [92]). Несмотря на очевидные достоинства деревьев — простоту распознавания, независимость от масштабирования признаков и возможность содержательной интерпретации признаков — их редко применяют в качестве самостоятельных распознавателей, так как хорошего качества распознавания (т.е. малой ошибки на тесте) от них добиться обычно не удается. Зато их широко применяют в качестве сырья в методах построения "хороших" распознавателей из "недостаточно хороших", см. раздел 5.

Не вдаваясь в детали изощренных алгоритмов обучения деревьев, построим простой жадный алгоритм, минимизирующий (локально, как и всякий жадный алгоритм) ошибку обучения (5). Для простоты будем считать, что множества значений каждого признака упорядочены (вообще говоря, для построения деревьев упорядоченность конечнозначных признаков не требуется), и все сравнения производятся на строгое неравенство "$<$".

Сначала обучим на обучающем наборе T дерево с одним листом, т.е. постоянную функцию $f_1(x) = r$, минимизирующую суммарную ошибку обучения

$$E(f_1, T) = \sum_{i=1}^{n} E(f(x_i), y_i) \ . \tag{6}$$

Как это делать — зависит от конкретного вида функции ошибки. Например, для квадратичной ошибки (регрессия) минимум по r достигается в среднем арифметическом ответов обучающего набора

$$r = \frac{1}{N} \sum_{i=1}^{n} y_i \ ,$$

а для 0-1-ошибки (классификация) — в любом из наиболее часто встречающихся ответов.

Затем для каждого возможного разбиения пространства признаков $\mathcal{X}$ на полупространства $x^j < x_i^j$ и $x^j \geq x_i^j$, $j = 1, \ldots, d$, $i = 1, \ldots, N$, такого что в каждом полупространстве имеются векторы из обучающего набора, обучим распознающие деревья с одним ветвлением и двумя листьями (такие деревья иногда называются *пнями*, *stump*), т.е. кусочно-постоянные функции принимающие на этих полупространствах значения $r_<$ и $r_\geq$. Каждое из этих двух значений ищется независимо минимизацией суммарной ошибки (6), причем для вычисления каждого достаточно минимизировать сумму по обучающим векторам, попадающим в соответствующее полупространство. Из получившихся не более $d \times (N-1)$ пней выбираем тот, у которого суммарная ошибка минимальна.

Заготовим список допустимых разбиений, изначально пустой. Полупространства, а в дальнейшем более мелкие части пространства, на которых распознаватель дает постоянный ответ, будем называть областями.

Затем итеративно чередуются три шага:

- добавление в список допустимых разбиений не более чем двух разбиений, каждое из которых минимизирует суммарную ошибку обучения по разбиениям одной из двух вновь образованных областей, содержащим больше одного обучающего вектора в каждой своей "половине";
- выбор из списка допустимых разбиений разбиения с минимальной суммарной ошибкой обучения и соответствующая замена листа на ветвление и два листа в дереве распознавания;
- удаление использованного разбиения из списка.

Поскольку количество листьев в дереве не может превысить количество обучающих векторов, процесс заведомо конечен. Обычно его останавливают досрочно по соображениям, приводимым в разделе 1.1.6. Например, при достижении заранее заданного числа листьев.

Из-за жадности алгоритма обучения получающиеся распознаватели с $k > 2$ листьями не обязательно реализуют минимум ошибки обучения по всем деревьям с k вершинами.

1.1.6 Способность распознавателя к обобщению и регуляризация

Казалось бы, чем больше пространство допустимых распознавателей $\mathcal{F}$, тем лучший распознаватель в нем можно найти. Но только не описанным выше способом (5) минимизации средней ошибки обучения. Например, если допустимы распознаватели, вычисляющие любой набор значений в любом наборе из N точек $\mathcal{X}$, то ошибку обучения можно свести к нулю, обеспечив чтобы $f(x_i) = y_i$ для всех обучающих векторов (x_i, y_i). Таких распознавателей с нулевой ошибкой может быть много, какие из них действительно хорошие, а какие — плохие, остается только гадать. Крайний пример очень плохого, хотя и идеально обученного распознавателя: распознаватель f, такой что $f(x_i) = y_i$ и $f(x)$ принимает взятые с потолка случайные значения при x вне обучающего набора. Распознаватели, имеющие малую ошибку на обучающем наборе и большую вне его, называются неспособными к обобщению (результатов обучения) и довольно бесполезны. Другое название неспособности распознавателя к обобщению — *переобучение* (*overfitting*).

Проверить, насколько распознаватель способен к обобщению, можно сравнив его среднюю ошибку при обучении со средней ошибкой на независимом тесте. Но хочется сразу организовать обучение так, чтобы получить хорошо обобщающий распознаватель. Грубый способ состоит в том, чтобы очень сильно ограничить пространство допустимых распознавателей: настолько сильно, чтобы плохих распознавателей с малой ошибкой обучения в нем быть не могло. Например, когда пространства $\mathcal{Y}$ и $\mathcal{F}$ конечномерные топологические (в частности, евклидовы), полезно обеспечить, чтобы $\dim(\mathcal{F}) < N \dim(\mathcal{Y})$, поскольку в ситуации общего положения коразмерность множества распознавателей с нулевой ошибкой при обучении равна $N \dim(\mathcal{Y})$.

Можно рассматривать параметрическое семейство ограниченных подпространств пространства распознавателей и экспериментально подбирать значение параметра, при котором обученный распознаватель имеет достаточно малую среднюю ошибку на независимом тесте. На самом деле существенна не ограниченность подпространств, а ограничение на их *размерности Вапника-Червоненкиса*[5] (*VC-dimension*), но в простых случаях метрической ограниченности тоже достаточно. Такой подход называется *структурной минимизацией риска*структурная минимизация риска (*structural risk minimization*) с нетрадиционным для математики использованием слова "структура" (покрытие пространства расширяющейся последовательностью подпространств обычно называется не структурой, а фильтрацией). Теория структурной минимизации риска и оценки отклонений средней ошибки от ошибки обучения подробно изложены в книгах [114, 113].

[5] Размерность Вапника-Червоненкиса (или комбинаторная размерность) $\dim_{\mathrm{VC}}(\mathcal{F})$ определена для множеств $\mathcal{F}$ функций $\mathcal{X} \to \mathcal{Y}$. Ее формальное определение зависит от пространства-образа $\mathcal{Y}$. Приведем его для простейшего случая $\mathcal{Y} = \{0, 1\}$, соответствующего двухклассовой классификации, причем для удобства так и будем называть функции $f \in \mathcal{F}$ классификаторами.

Определение. Размерностью Вапника-Червоненкиса семейства двухклассовых классификаторов $\mathcal{F}$ на $\mathcal{X}$ называется наибольшее число n (или ∞, если наибольшего нету), такое что найдутся n точек $x_1, \ldots, x_n \in \mathcal{X}$, которые классификаторы из семейства $\mathcal{F}$ могут классифицировать всеми 2^n возможными способами.

В простых и естественных случаях размерность Вапника-Червоненкиса совпадает с топологической размерностью или немного превышает ее. Но имеются и примеры, когда размерность Вапника-Червоненкиса намного больше топологической.

Хорошие примеры.

- Пространство $\mathcal{F}$ функций на $\mathcal{X} = \mathbb{R}$ вида $f_t(x) = \chi_{\{(x,t)|x-t\geq 0\}}(x, t)$ при $t \in \mathbb{R}$ одномерно и $\dim_{\mathrm{VC}}(\mathcal{F}) = 1$ (очевидно; здесь и далее через χ_S обозначается индикаторная функция множества S).
- Пространство $\mathcal{F}$ функций на $\mathcal{X} = \mathbb{R}$ вида $f_{w,b}(x) = \chi_{\{(x,w,b)|wx+b\geq 0\}}(x, w, b)$ при $w, b \in \mathbb{R}$ тоже одномерно (а не двумерно, как может показаться), но $\dim_{\mathrm{VC}}(\mathcal{F}) = 2$ (очевидно).

Например, пространство распознающих деревьев естественно представимо в виде объединения подпространств, состоящих из деревьев с не более, чем k листьями. А для пространства распознавателей $\mathcal{F}$, параметризованных евклидовым пространством $\mathcal{W}$, т.е. состоящего из распознавателей вида $f(x)=F(w,x)$ для фиксированной функции $F : \mathcal{W} \times \mathcal{X} \to \mathcal{Y}$ и параметра $w \in \mathcal{W}$, в качестве подпространств можно брать шары с центром в нуле, то есть при обучении вместо задачи (5) решать семейство задач

$$\sum_{i=1}^{N} E(F(w,x_i),y_i) \to \min_{\|w\|\le C}, \tag{7}$$

зависящих от параметра $C > 0$. Здесь и далее вместо средней ошибки (как в задаче (5)) минимизируется *суммарная ошибка обучения*, чтобы не возиться с множителем $\frac{1}{N}$.

Более гуманный на вид способ обучения состоит в том, чтобы не запрещать, а штрафовать. Например, можно пространство параметров распознавателя $\mathcal{W}$ считать евклидовым (или банаховым, причем хоть бы и $\mathbb{R}^n$, но с неевклидовой нормой) и назначить штраф, пропорциональный норме параметра (или в более общем виде, какую-то непрерывную функцию ψ с компактными множествами подуровня $\{w|\psi(w) \le C\}$), то есть при обучении вместо задачи (4) решать задачу

$$\psi(w) + E(F(w,\cdot),T) = \psi(w) + \sum_{i=1}^{N} E(F(w,x_i),y_i) \to \min_{w\in\mathcal{W}}, \tag{8}$$

В общематематической науке такой способ решения обратных задач — минимизация функционала (8) вместо решения относительно w системы уравнений $F(w,x_i) = y_i$ — называется *регуляризацией* по Тихонову (см. [108]). В качестве функции штрафа можно взять норму $\psi(w) = \epsilon\|w\|$ или квадрат нормы $\psi(w) = \epsilon\|w\|^2$ с малым коэффициентом ϵ, причем норма $\|\cdot\|$ не обязана быть евклидовой.

Множества решений семейства задач с ограничением (7) и семейства задач с регуляризацией, например,

$$\epsilon\|w\|^2 + \sum_{i=1}^{N} E(F(w,x_i),y_i) \to \min_{w\in\mathcal{W}}, \tag{9}$$

- Пространство $\mathcal{F}$ функций на $\mathcal{X} = \mathbb{R}^d$ вида $f_w(x) = \chi_{\{(x,w)|(w,x)\ge 0\}}(x,w)$ при $w \in \mathbb{R}^d$ $(d-1)$-мерно, а $\dim_{\mathrm{VC}}(\mathcal{F}) = d$ (**упражнение!**).

Плохой пример. Пространство $\mathcal{F}$ функций на $\mathcal{X} = \mathbb{R}$ вида $f_t(x) = \phi(x+t)$ при $t \in [0,1]$ одномерно при любой непостоянной функции $\phi : \mathbb{R} \to \{0,1\}$. Возьмем функцию ϕ, на каждом единичном отрезке $[n, n+1]$ равную n-й цифре записи дробной части ее аргумента в виде двоичной дроби; формально

$$\phi(\tau) = \left(\left\lfloor 2^{\lfloor\tau\rfloor}\,(\tau - \lfloor\tau\rfloor) \right\rfloor \bmod 2 \right)$$

Тогда $\dim_{\mathrm{VC}}(\mathcal{F}) = \infty$. Действительно, последовательность значений f_t в целых точках $1, 2, \ldots$ — это последовательность цифр в представлении t в виде двоичной дроби, и тем самым, может быть любой.

Получение оценок сверху для отклонения средней ошибки распознавателя от ошибки обучения через размерность Вапника-Червоненкиса довольно громоздко и здесь не приводится. Бесконечность размерности Вапника-Червоненкиса семейства распознавателей, строго говоря, еще не означает невозможности обучения распознавателя этого семейства с небольшой средней ошибкой путем минимизации ошибки обучения. Невозможность возникает, когда распознаватель можно научить всем возможным ответам не на каких-то исключительно неудачных обучающих наборах, а на заметной доле от всех обучающих наборов.

или

$$\epsilon\|w\| + \sum_{i=1}^{N} E(F(w, x_i), y_i) \to \min_{w \in \mathcal{W}}, \tag{10}$$

при всевозможных значениях параметров $C > 0$ и $\epsilon \geq 0$, соответственно, при некоторых дополнительных условиях почти совпадают. А именно

Предложение 1

Каждое решение задачи (9) или (10) при $\epsilon \geq 0$ является решением задачи (7) при некотором $C \geq 0$.

Если функция ошибки $E(F(w, x), y)$ выпукла по w при любых x и y и норма w также выпукла, то каждое решение задачи (7) при $C > 0$ является решением задачи (9) или (10) при некотором $\epsilon \geq 0$.

Доказательство получается при применении к задаче (7) идеи метода множителей Лагранжа (для формального применения метода требуется гладкость функции ошибки и сфер в выбранной норме). Технические детали оставляется в качестве **упражнения**. □

Упражнение. Приведите пример невыпуклой по w функции ошибки, при которой задача (7) имеет решения, отличные от решений задачи (9) и даже не лежащие в их замыкании.

1.1.7 Подбор параметров регуляризации

Иногда можно теоретически найти функцию регуляризации ψ или, что проще, константу C, гарантирующую, что минимизация выражения (8) или, соответственно, (7) обеспечивает малость средней ошибки (2), но теоретические оценки слишком пессимистичны. Чаще берут простую функцию регуляризации (квадрат нормы параметра, количество листьев дерева и т.п.) с коэффициентом, который подбирают эмпирически. Для этого распознаватель обучают при разных значениях коэффициента регуляризации на обучающем наборе T, оценивают среднюю ошибку на не пересекающемся с ним *оценочном наборе* (*validation set*, часто используется калька "*валидационный набор*") T'' и выбирают значение коэффициента регуляризации, минимизирующее эту ошибку. Найденный минимум ошибки уже не является правильной оценкой средней ошибки распознавателя, обученного при оптимальном значении коэффициента регуляризации, поскольку этот распознаватель зависит и от обучающего набора T, и от оценочного набора T'': для несмещенной оценки средней ошибки нужен третий набор данных — тестовый.

Этот метод подбора параметра совершенно не зависит от метода обучения распознавателя. Например, так можно подбирать оптимальное число соседей k в методе k ближайших соседей, которое тоже является в некотором смысле параметром регуляризации, напоминающим $\frac{1}{\|w\|}$ для линейных распознавателей: чем больше k, тем меньше изменяется ответ распознавателя при изменении признаков.

Когда обучающих данных мало и жалко отделять от них часть только для оценки коэффициента регуляризации, применяется метод *кросс-валидации*. k-кратная кросс-валидация состоит в том, что обучающий набор разделяют на k примерно равных частей, для каждого значения коэффициента регуляризации обучают k распознавателей, каждый раз выбрасывая одну k-ую часть обучающего набора, оценивают среднюю ошибку каждого распознавателя на выброшенной при его обучении k-й части, усредняют эти k оценок и минимизируют результат усреднения. Аналогичную процедуру можно применять и для тестирования (*кросс-тестирование*), хотя измеренная таким образом средняя

ошибка тестирования будет относиться не к конкретному распознавателю, а к способу обучения.

N-кратная кросс-валидация или кросс-тестирование на обучающем наборе длины N называется также *методом скользящего контроля* (*LOO*, *Leave-One-Out*).

При этом обучается наибольшее число распознавателей, т.е. это, казалось бы, самый медленный способ. Но некоторые методы обучения распознавателей позволяют переучивать их при замене всего лишь одного обучающего вектора гораздо быстрее, чем обучать с нуля.

Теорема 1 *Среднее арифметическое этих N ошибок является насмещенной оценкой для средней ошибки (2) на всем пространстве, усредненной по всем распознавателям, обученным на $N-1$ векторе.*

Доказательство состоит в расписывании (в одну строчку) четырех упомянутых усреднений (несмещенность оценки — это тоже некоторое утверждение про ее усреднение). Оставляется в качестве **упражнения**. □

1.2 Обучение распознавателей и вероятностные модели

Описанная в разделе 1.1.4 постановка задачи обучения распознавателей с одной стороны, предполагает наличие некоторой вероятностной модели (распределение π на пространстве $\mathcal{X}\times\mathcal{Y}$), а с другой стороны, ничего про это распределение не предполагает (заложенное в формулировке ограничение $\pi \in \mathcal{P}$ пока никак не использовалось). Этим она отличается от типичной постановки задач статистики, когда про вероятностную модель что-то заранее известно или предполагается (принадлежность некоторому классу $\mathcal{P}$ и, возможно, еще что-то), на основании обучающего набора это априорное знание уточняется ("модель обучается"), и полученное апостериорное знание используется для прогнозирования (распознавания). В этом разделе будет приведен пример пользы, хотя и чисто теоретической, от знания распределения π, будут сформулированы несколько вариантов такой "классической" постановки задачи обучения и показано, что некоторые из них эквивалентны обучению минимизацией эмпирического риска.

1.2.1 Байесовский классификатор и байесовская регрессия

Предположим, что распределение π известно. Тогда можно предъявить самый лучший распознаватель, решающий задачу (2) в пространстве всех возможных распознавателей, хотя и нет гарантии, что он будет принадлежать подпространству $\mathcal{F}$ допустимых распознавателей.

Рассмотрим задачу классификации: классификатор должен про каждую точку $x \in \mathcal{X}$ в пространстве признаков отвечать, какому классу $y \in \mathcal{Y}$ она принадлежит. Будет ли он предварительно вычислять распределение условных вероятностей $P\{y|x\}$ на множестве классов — его внутреннее дело. В качестве функции ошибки (штрафа) возьмем 0-1-штраф:

$$E(r, y, x) = 1 - \delta_y^r \ , \tag{11}$$

где δ_a^b — *символ Кронекера*, т.е. штраф равен 0 для правильного ответа и 1 для неправильного. Если при такой постановке задачи известно распределение π на $\mathcal{X}\times\mathcal{Y}$ или хотя бы условные распределения $p_\pi(y|x)$, то легко построить классификатор f_π с минимально возможным ожиданием ошибки $E_\pi(f_\pi)$. А именно, при каждом векторе признаков x классификатор должен давать наиболее вероятный ответ, т.е. ответ y, максимизирующий условную вероятность $p_\pi(y|x)$,

или, что то же самое, минимизирующий условное ожидание ошибки по y при данном x:

$$\begin{aligned} f_\pi(x) &= \arg\min_{r\in\mathcal{Y}} \int_{y\in\mathcal{Y}} E(r,y,x)dp_\pi(y|x) \\ &= \arg\min_{r\in\mathcal{Y}}(1-p_\pi(y|x)) = 1-\arg\max_{r\in\mathcal{Y}} p_\pi(y|x)\ . \end{aligned} \tag{12}$$

(При конечном пространстве $\mathcal{Y}$ интеграл $\int_{y\in\mathcal{Y}} E(r,y,x)p_\pi(y|x)dy$ превращается в сумму $\sum_{y\in\mathcal{Y}} E(r,y,x)p_\pi(y|x)$.) Ожидание ошибки (11) по всему пространству

$$\begin{aligned} E_\pi(f_\pi) &= \int_{(x,y)\in\mathcal{X}\times\mathcal{Y}} E(f_\pi(x),y,x)d\pi(x,y) \\ &= \int_{x\in\mathcal{X}} \left(\int_{y\in\mathcal{Y}} E(f_\pi(x),y,x)p_\pi(y|x)dy\right) p_\pi(x)dx \end{aligned} \tag{13}$$

при этом тоже оказывается минимальным.

Этот классификатор называется *байесовским*, вероятно, потому, что если про распределение π известны вероятности классов $p_\pi(y)$ и условные распределения $p_\pi(x|y)$, то максимизируемая вероятность $p_\pi(y|x)$ выражается через них по формуле Байеса. Его ценность не в том, что его можно применить (ведь распределение π неизвестно), а в том, что он дает оценку минимально возможной ошибки классификации, и напоминает о том, что знать или уметь оценивать распределение π полезно. Такой же поточечно оптимальный классификатор можно строить не для неизвестного истинного распределения π, а для какой-то его оценки, и он тоже называется байесовским.

Аналогично в задаче регрессии при известном распределении π поточечной по x минимизацией ожидания ошибки по условному распределению $p_\pi(y|x)$

$$f(x) = \arg\min_{r\in\mathcal{Y}} \int_{y\in\mathcal{Y}} E(r,y,x)p_\pi(y|x)dy \tag{14}$$

строится *байесовская регрессия*, минимизирующая ожидание ошибки (2). При квадратичной функции ошибки $E(r,y,x)=(r-y)^2$ точка минимума считается явно:

$$0 = \frac{d}{dr}\int_{y\in\mathcal{Y}} (r-y)^2 p_\pi(y|x)dy = 2\int_{y\in\mathcal{Y}} (r-y)p_\pi(y|x)dy = 2\left(r - \int_{y\in\mathcal{Y}} yp_\pi(y|x)dy\right),$$

значит

$$f_\pi(x) = \int_{y\in\mathcal{Y}} yp_\pi(y|x)dy, \tag{15}$$

т.е. в точности равна условному математическому ожиданию (в математической терминологии — регрессии) истинного ответа относительно вектора признаков. При других функциях штрафа байесовская регрессия, вообще говоря, отличается от регрессии в математическом смысле. То, что функции штрафа могут определяться вероятностными соображениями, будет показано в разделе 1.2.6.

Упражнение. Вычислите байесовскую регрессию для кусочно-линейной функции штрафа $E(r,y)=|r-y|$.

1.2.2 Пример: асимптотика ошибок метода ближайшего соседа

Обозначим через E_N усреднение по всевозможным обучающим наборам

$$T = ((x_1,y_1),\ldots,(x_N,y_N)) = ((x_1,\ldots,x_N),(y_1,\ldots,y_N)) = (\mathbf{X},\mathbf{Y})$$

длины N ошибки двухклассового классификатора (т.о. $\mathcal{Y} = \{0, 1\}$), "обученного" методом ближайшего соседа. Попробуем сравнить асимптотику E_N по N с ошибкой E_B байесовского классификатора (раздел 1.2.1). Будем предполагать и распределение π и условные вероятности $P\{0|x\} = p_\pi(0|x)$ и $P\{1|x\} = 1 - p_\pi(0|x)$ непрерывными по x. Байесовский классификатор в точке x дает ответ 0, если $P\{0|x\} > P\{1|x\}$, и ответ 1, если $P\{0|x\} < P\{1|x\}$ (какой ответ он дает при $P\{0|x\} = P\{1|x\}$, несущественно). Тогда вероятность ошибки байесовского классификатора в точке x равна $E_B(x) = \min(P\{0|x\}, P\{1|x\})$.

Пусть $(x_n, y_n) \in T = (\mathbf{X}, \mathbf{Y}))$, а (x, y) — независимая от обучающего набора случайная (с распределением π) точка пространства $\mathcal{X} \times \mathcal{Y}$. Тогда ожидание по $\mathbf{Y}$ вероятности несовпадения значений y и y_n равно

$$P\{y \neq y_n\} = P\{y = 0, y_n = 1\} + P\{y = 1, y_n = 0\} = P\{0|x\}P\{1|x_n\} + P\{1|x\}P\{0|x_n\}$$

Если плотности распределений непрерывны и точка x_n близка к точке x, то

$$P\{y \neq y_n\} \approx 2P\{0|x\}P\{1|x\},$$

а значит и ожидание по ответам $\mathbf{Y}$ обучающего набора ошибки метода ближайшего соседа

$$E_N(x) = P\{y \neq y_n\} \approx 2P\{0|x\}P\{1|x\} = 2(1 - E_B(x))E_B(x) \ . \tag{16}$$

Кроме того, по построению байесовского классификатора

$$E_B(x) \leq E_N(x) \ . \tag{17}$$

Беря ожидания приближенного равенства (16) и неравенства (17) по x и $\mathbf{X}$, переходя к пределу при $N \to \infty$ (аккуратное обоснование предельного перехода см. в [27]) и учитывая вогнутость функции $f(t) = 2(1 - t)t$, получаем двустороннюю оценку

$$E_B \leq \lim_{N\to\infty} E_N \leq 2(1 - E_B)E_B \leq 2E_B \ .$$

Имеются аналогичные оценки ошибок классификаторов на k ближайших соседях, в том числе и многоклассовых, см. [27]. Напоминаем еще раз, что эти оценки — всего лишь асимптотические.

1.2.3 Классификация моделей и методов обучения

В классическом статистическом подходе к обучению распознавателей используется специфическая терминология.

Параметрические и непараметрические модели и методы обучения

Методы обучения, т.е. нахождения достаточно хорошей распознающей функции $f \in \mathcal{F}$, традиционно подразделяются на параметрические и непараметрические в соответствии с тем, просто или сложно устроено пространство $\mathcal{F}$. *Параметрические* — это те методы, в которых $\mathcal{F} = \{F(w, \cdot)|w \in \mathcal{W}\}$ для некоторого достаточно удобного (например, евклидова) пространства параметров $\mathcal{W}$ и некоторой функции $F\colon \mathcal{W} \times \mathcal{X} \to \mathcal{Y}$, а *непараметрические* — это методы, в которых пространство $\mathcal{F}$ не зафиксировано заранее, а зависит от обучающего набора T. Аналогично, *параметрические модели* — это просто устроенные и допускающие удобную параметризацию пространства моделей $\mathcal{P}$ (например, гауссовы распределения в $\mathbb{R}^d$, определяемые центром и матрицей ковариаций, т.е. $d + \frac{d(d+1)}{2}$ числами), а *непараметрические* — это модели с более сложным пространством распределений.

На самом деле разница между параметрическими и непараметрическими методами — только в употребляемых словах.

Пример параметрических методов — методы обучения линейных распознавателей, которых даже для простейшей линейной регрессии ($\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \mathbb{R}$, $\mathcal{W} = \mathbb{R} \times \mathbb{R}^d$, $F(w, x) = w^0 + \sum_{j=1}^{d} w^j x^j$) довольно много. Подробнее эти методы рассматриваются в разделе 2.

Классический пример непараметрического метода — метод ближайших соседей (см. раздел 1.1.3). Пространство распознавателей $\mathcal{F}$, получающихся при этом методе, легко описывается, только оно не конечномерно.

Философский вопрос: метод распознающих деревьев (раздел 1.1.5) — параметрический? А метод распознающих пней?

Дискриминантные и порождающие модели и методы обучения

Можно обучать распознаватели, минимизируя среднюю ошибку обучения $E(f, T)$, оценивая качество обучения по средней ошибке тестирования $E(f, T')$ и полностью забыв про вероятностную модель. Такие методы обучения называются *дискриминантными* (калька с английского *discriminative methods*, буквально — различающие, канонический перевод неизвестен, название происходит от двухклассовой классификации, когда вычисляемая распознавателем функция f в точности различает классы и поэтому называется дискриминантом).

Можно, наоборот, пытаться все-таки восстановить распределение π. Такие методы обучения и сами вероятностные модели называются *порождающими* (*generative methods*, канонический перевод неизвестен, название связано с тем, что распознаваемые объекты порождаются в соответствии с распределением π). При поиске оптимального распределения заодно обнаруживаются "правильное" подпространство в пространстве распознавателей $\mathcal{F}$ и "правильная" функция ошибки E.

Есть и промежуточный вариант: пытаться восстановить не все распределение $\pi(x, y)$, а только условные распределения $\pi(y|x)$ (в непрерывном случае определяемые плотностью (1)). Такие методы обучения и модели снова называются *дискриминантными*, хотя, поскольку ответ *y порождается* признаками x в соответствии с распределением $\pi(y|x)$, возможна терминологическая путаница. Другим источником путаницы является то, что самый классический метод обучения порождающей модели унаследовал название от получающегося в результате классификатора — *дискриминанта* Фишера (раздел 2.2.1).

1.2.4 Обучение порождающих и дискриминантных моделей

Порождающие модели

Буквальное применение классической теории вероятностей для обучения порождающих моделей и его упрощенные приближения таковы:

Байесовское обучение. Предположим, что на пространстве моделей $\mathcal{P}$ задана вероятностная мера μ (*априорное распределение*), т.е. вероятностное распределение на пространстве вероятностных распределений. Для любого модели $\pi \in \mathcal{P}$ условная вероятность (или ее плотность) обучающего набора T равна

$$p\{T|\pi\} = \prod_{i=1}^{N} p_\pi(x_i, y_i) \ . \tag{18}$$

Тогда по формуле Байеса для любого подмножества $\Pi \in \mathcal{P}$

$$\mu'(\Pi) = P\{\Pi|T\} = \frac{\int_{\pi \in \Pi} p\{T|\pi\} d\mu(\pi)}{\int_{\pi \in \mathcal{P}} p\{T|\pi\} d\mu(\pi)} \tag{19}$$

вычисляется *апостериорное распределение* μ' на пространстве $\mathcal{P}$. Если же априорное распределение непрерывно и имеет плотность p_μ, то апостериорное распределение имеет плотность

$$p_{\mu'}(\pi) = p\{\pi|T\} = \frac{p\{T|\pi\}p_\mu(\pi)}{\int_{\pi\in\mathcal{P}} p\{T|\pi\}p_\mu(\pi)d\pi} \ . \tag{20}$$

На этом байесовское обучение закончено. Его ответом является не какой-то один распознаватель, а распределение вероятностей на пространстве распределений вероятностей (моделей)!

Для распознавания результат байесовского обучения применяется так: вычисляется ожидание по $\mu'(\pi)$ распределения $p_\pi(x,y)$

$$P_{\mu,T}(Z) = \int_{\pi\in\mathcal{P}} p_\pi(Z)d\mu'(\pi) \ , \ Z \subset \mathcal{X}\times\mathcal{Y} \tag{21}$$

или его плотности

$$p_{\mu,T}(x,y) = \int_{\pi\in\mathcal{P}} p_\pi(x,y)p_{\mu'}(\pi)d\pi \ , \tag{22}$$

если у всех распределений π есть плотность $p_{\mu'}$, после чего для заданного вектора признаков x вычисляется условное распределение

$$P_{\mu,T}(Y) = \frac{P_{\mu,T}(\{x\}\times Y)}{P_{\mu,T}(\{x\}\times\mathcal{Y})} \ , \ Y \subset \mathcal{Y} \tag{23}$$

или, если существует, его плотность

$$p_{\mu,T}(y|x) = \frac{p_{\mu,T}(x,y)}{\int_{y\in\mathcal{Y}} p_{\mu,T}(x,y)dy} \ , \tag{24}$$

(формула (1)) и выдается в качестве ответа.

Или, если нужно дать конкретный ответ, вычисляется еще точка максимума (плотности), или ожидание по y вычисленного распределения $p_{\mu,T}(y|x)$, или точка минимума ожидания какой-либо функции штрафа $E(r,y,x)$:

$$\int_{y\in\mathcal{Y}} E(r,y,x)p_{\mu,T}(y|x)dy \to \min_{r\in\mathcal{Y}} \ . \tag{25}$$

Для нахождения точки максимума (плотности) вероятности ответа достаточно для любого вектора признаков x и любой пары ответов y' и y'' уметь вычислять отношение

$$\frac{p_{\mu,T}(y'|x)}{p_{\mu,T}(y''|x)} \ .$$

Упражнение. Покажите, что

$$\frac{p_{\mu,T}(y'|x)}{p_{\mu,T}(y''|x)} = \frac{\int_{\pi\in\mathcal{P}} p_\pi(x,y')\prod_{i=1}^N p_\pi(x_i,y_i)d\mu(\pi)}{\int_{\pi\in\mathcal{P}} p_\pi(x,y'')\prod_{i=1}^N p_\pi(x_i,y_i)d\mu(\pi)} \ . \tag{26}$$

Замечание. Хотя формула (26) проста и наглядна, вычислительно она крайне неэффективна, поскольку в ней обучение не отделено от распознавания, и при каждом распознавании нужно просматривать весь обучающий набор.

Байесовский подход замечателен всем, кроме двух "мелочей": очень редко бывает известно априорное распределение и очень редко удается все проинтегрировать честно. Аналитически интегрировать обычно невозможно, а численно — слишком трудоемко. Есть, правда, несколько общеизвестных и небесполезных случаев, в которых байесовское обучение можно провести аналитически:

например, когда пространство моделей $\mathcal{P}$ конечно, или когда пространства $\mathcal{X}$ и $\mathcal{P}$ евклидовы и все распределения $\pi \in \mathcal{P}$ и μ являются гауссовыми. Можно также пытаться интегрировать приближенно, например, для непрерывных распределений использовать *аппроксимацию Лапласа*[6], хотя гарантий получения сколько-нибудь хорошей модели при этом уже нет.

Когда нет достаточных оснований выбрать априорное распределение "потому, что...", его выбирают из соображений "для того, чтобы...": например, для того, чтобы все удалось проинтегрировать аналитически, для того, чтобы распознаватель распознавал быстро, или чтобы он был устойчив к возможным ошибкам в обучающем наборе. Чем больше обучающий набор, тем меньше получающийся распознаватель зависит от априорного распределения.

Другой способ применения байесовского обучения — *метод Гиббса* — состоит в том, чтобы вместо усреднения по пространству моделей $\mathcal{P}$ (21) или (22) брать в нем случайную в соответствии с апостериорным распределением μ' модель π и по ней вычислять ответ $p_\pi(y|x)$.

Максимизация апостериорной вероятности. Предположим, что пространство моделей $\mathcal{P}$ достаточно простое, например, евклидово, априорное распределение μ на нем имеет плотность $p_\mu(\pi)$, и каждое распределение $\pi \in \mathcal{P}$ имеет некоторую плотность $p_\pi(x, y)$. Не всегда, но довольно часто, плотность апостериорного распределения имеет единственную точку глобального максимума или несколько точек максимума, переводимых друг в друга симметриями пространства $\mathcal{P}$ и почти полностью сосредоточена возле точки (точек) максимума. Тогда интеграл (22) приблизительно равен плотности $p_\pi(x, y)$ в точке (точках) максимума π плотности апостериорного распределения, а значит при обучении достаточно вычислить не все апостериорное распределение, а только точку его максимума.

Получившееся обучение *методом максимизации апостериорной вероятности* (*MAP*, *maximum of aposteriori probability*) состоит в решении максимизационной задачи

$$p(T|\pi;\mu) = p_\mu(\pi)\prod_{i=1}^{N} p_\pi(x_i, y_i) \to \max_{\pi\in\mathcal{P}} \ . \tag{27}$$

Ее решение — распределение π на пространстве $\mathcal{X} \times \mathcal{Y}$ — позволяет для любого вектора признаков x вычислить условное распределение ответа $\pi(y|x)$ и/или его ожидание, точку максимума его плотности $p_\pi(y|x)$, если оно непрерывно, или точку минимума некоторого штрафа.

Максимизация правдоподобия. И байесовское обучение, и метод максимизации апостериорной вероятности предполагают известным некоторое априорное распределение вероятности на пространстве моделей. А что делать, если никаких априорных идей нет? Можно считать, что априори все модели равновероятны, если пространство моделей дискретно, или плотность априорной вероятности моделей постоянна, если оно непрерывно (это называется *non-informative prior*). На некомпактном пространстве моделей $\mathcal{P}$ таких распределений обычно нет, так что байесовское обучение, состоящее только в уточнении априорных знаний с помощью наблюдений, формально невозможно. А метод максимизации апостериорной вероятности применим всегда и превращается в максимизацию плотности условной вероятности обучающего набора T

[6] Аппроксимация Лапласа для распределения — это приближение его гауссовым, логарифм плотности которого является, с точностью до прибавления константы, квадратичной аппроксимацией логарифма плотности распределения в точке ее максимума.

$$p(T|\pi) = \prod_{i=1}^{N} p_\pi(x_i, y_i) \to \max_{\pi \in \mathcal{P}}, \tag{28}$$

называемой также *правдоподобием* (*likelihood*) модели π. Получившийся метод обучения называется *методом наибольшего правдоподобия* (*ML, maximum likelihood*).

Хотя формально метод наибольшего правдоподобия применим при любом пространстве моделей $\mathcal{P}$, для успешного обучения пространство $\mathcal{P}$ приходится сильно ограничивать. Это необходимо для того, чтобы не получить плотность вероятности p_π, сколько-нибудь отличную от нуля только возле точек обучающего набора, т.е. не построить вероятностную модель, объявляющую все, кроме того, чему ее учили, практически невозможным, а значит, неинтересным. Задачи, в которых естественно возникают такие сильные ограничения, действительно встречаются, но в большинстве остальных ситуаций метод наибольшего правдоподобия работает хуже других методов. При любом априорном распределении асимптотически, при росте обучающего набора, метод наибольшего правдоподобия сходится к методу максимизации апостериорной вероятности (см. [72]).

Обучение порождающих моделей и минимизация штрафа.

Функция штрафа $E(r, y, x)$ при обучении условного распределения $\pi(y|x)$ не понадобилась: она была нужна только при желании превратить найденное условное распределение в конкретный ответ, зависящий от x. Легко проверить, что каждый такой ответ, упомянутый на стр. 21, получается минимизацией некоторой функции штрафа в формуле (25):

Упражнение. Докажите, что

- при классификации (дискретном пространстве ответов $\mathcal{Y}$) наиболее вероятный ответ получается при минимизации 0-1-штрафа;
- при регрессии ожидание распределения $\pi(y|x)$ получается при минимизации квадратичного штрафа;
- при регрессии точка максимума по y плотности $p_\pi(y|x)$ получается при минимизации любого штрафа $E(r, y, x)$, являющегося строго убывающей функцией от $p_\pi(r|x)$.

Нетривиальные функции штрафа, явно не укладывающиеся в вероятностную модель, возникают при асимметрии, вполне встречающейся в жизни. Например, при двухклассовой классификации ошибки первого и второго рода (прогнозировать принадлежность одному классу, при том, что на самом деле объект принадлежит другому) могут штрафоваться по-разному. Или при регрессии достаточно точный прогноз ($-\epsilon_1 < r - y < \epsilon_2$) может не не штрафоваться вообще, сильный недобор ($r - y \leq -\epsilon_1$) может стоить 1 у.е., а сильный перебор ($r - y \geq \epsilon_2$) может стоить 10 у.е.

Априорный выбор пространства распознавателей $\mathcal{F}$ также не имеет никакого отношения к обучению порождающих моделей: какой распознаватель получится, такой получится. Если же пространство $\mathcal{F}$ все-таки фиксировано, то после обучения модели — распределения π на $\mathcal{X} \times \mathcal{Y}$ — можно было бы обучать распознаватель, минимизируя ожидание штрафа (ошибки, риска, потерь,...)

$$E(f, \pi) = \int_{(x,y) \in \mathcal{X} \times \mathcal{Y}} E(f(x), y, x) d\pi(x, y) \to \min_{f \in \mathcal{F}}. \tag{29}$$

На практике такое двухступенчатое обучение применимо только при очень простых моделях или небольших конечных пространствах признаков. При наличии априорных соображений о пространстве распознавателей чаще стараются обучать дискриминантные модели.

Дискриминантные модели

Все три описанные для порождающих методов идеи, как найти самое лучшее распределение или как вместо поиска наилучшего усреднять по всем распределениям (ML, MAP и байесовский подход) применяются и при обучении дискриминантных моделей. Подробнее это будет описано в разделе 1.2.6. А пока рассмотрим примеры порождающих моделей.

1.2.5 Пример: наивный байесовский метод

Рассмотрим задачу q-классовой классификации с конечным d-мерным пространством признаков $\mathcal{X} = \{(x^1, \ldots, x^d)\}$, в котором j-й признак принимает M^i значений. Такое пространство признаков состоит из $D = \prod_{j=1}^{d} M^j$ точек. Каждая порождающая модель π — это конечный набор вероятностей π_{lk} сочетаний l-го возможного набора признаков и k-го ответа. Пространство всех таких моделей является $(Dq-1)$-мерным симплексом $\pi_{lk} \geq 0$, $\sum_{l,k} \pi_{lk} = 1$ в Dq-мерном евклидовом пространстве. Максимизации в методах MAP и ML из раздела 1.2.4 проводятся по конечным множествам, так что, казалось бы, на любом обучающем наборе любым из этих методов можно воспользоваться. Это верно, если число D не слишком велико само по себе и мало по сравнению с длиной N обучающего набора. Ведь по N ответам нужно обучить $Dq-1$ параметров и надеяться, что обучились чему-либо большему, чем распознаванию самого обучающего набора.

В реальных задачах число D неприемлемо велико. Например, если имеется всего лишь $d = 60$ бинарных признаков, то $D = 2^d = 2^{60} \approx 10^{18}$ — по нынешним временам слишком большое число переменных даже для двухклассовой классификации ($q = 2$).

Обучение порождающей модели становится реальным, если заранее резко сократить размерность пространства моделей $\mathcal{P}$. Например, можно ограничится моделями, в которых признаки условно независимы при фиксированных ответах, т.е. для каждого возможного ответа y условные вероятности $p((x^1, \ldots, x^d)|y)$ представимы в виде произведения

$$p((x^1, \ldots, x^d)|y) = \prod_{j=1}^{d} p(x^j|y) \ . \tag{30}$$

Такие модели называются *наивными байесовскими*. Они задаются q вероятностями ответов $\pi_k = p(y = k)$ и qD', где $D' = \sum_{j=1}^{d} M^j$, условными вероятностями $\pi^j_{mk} = p(x^j = m|y = k)$. С учетом вероятностных нормировок их пространство всего лишь $((q-1) + q(D'-d))$-мерно и является прямым произведением $(q-1)$-мерного симплекса

$$s^q = \left\{ (\pi_1, \ldots, \pi_q) \in \mathbb{R}^q_+ \mid \sum_{k=1}^{q} \pi_k = 1 \right\}$$

и qd симплексов

$$s^{M^j}_{jk} = \left\{ (\pi^j_{1k}, \ldots, \pi^j_{M^j k}) \in \mathbb{R}^{M^j}_+ \mid \sum_{m=1}^{M^j} \pi^j_{mk} = 1 \right\} \ .$$

В том же примере двухклассовой классификации с 60 бинарными признаками размерность пространства наивных байесовских моделей равна всего лишь 121.

Обучать наивную байесовскую модель можно всеми методами из раздела 1.2.4, хотя интегралы в формулах байесовского обучения (19) и распознавания (23) берутся аналитически довольно редко. Рассмотрим простой случай равномерного априорного распределения на пространстве моделей. При этом максимизация апостериорной вероятности совпадает с максимизацией правдоподобия.

Байесовское обучение дискретной наивной байесовской модели с равномерным априорным распределением

Обозначим через $S^d(r)$ и $s^d(r)$ следующие d- и $(d-1)$-мерный симплексы в d-мерном евклидовом пространстве, координаты в котором будем обозначать нижними индексами, чтобы не путать с показателями степени:

$$\begin{aligned} S^d(r) &= \left\{ x \in \mathbb{R}^d_+ \mid \sum_{j=1}^d x_j \leq r \right\} \\ s^d(r) &= \left\{ x \in \mathbb{R}^d_+ \mid \sum_{j=1}^d x_j = r \right\} . \end{aligned}$$

Обозначим через $\mathbf{E}_S(f)$ среднее арифметическое вещественнозначной функции f на многограннике $S \subset \mathbb{R}^d$ по мере Лебега размерности $\dim(S)$ (это частный случай ожидания случайной величины f).

Лемма 1 *Среднее арифметическое монома $x_1^{n_1} \cdots x_d^{n_d}$ по симплексу $s^d(1)$, равно*

$$\mathbf{E}_{s^d(1)}(x_1^{n_1} \cdots x_d^{n_d}) = \frac{(d-1)! \prod_{j=1}^d n_j!}{\left(\sum_{j=1}^d n_j + (d-1)\right)!} .$$

Доказательство. Достаточно произвести следующую последовательность вычислений: :

1. объем симплекса
$$v(S^d(r)) = \int_{x \in S^d(r)} dx = \frac{r^d}{d!} ;$$

2. средние значения по разным симплексам
$$\begin{aligned} & \mathbf{E}_{s^d(r)} f(x_1, \ldots, x_{d-1}) g(x_d) \\ = \; & \mathbf{E}_{S^{d-1}(r)} f(x_1, \ldots, x_{d-1}) g(r - \sum_{j=1}^{d-1} x_j) \\ = \; & \frac{\int_{x \in S^{d-1}(r)} f(x_1, \ldots, x_{d-1}) g(r - \sum_{j=1}^{d-1} x_j) dx}{v(S^{d-1}(r))} ; \end{aligned}$$

3. элементарный интеграл:
$$\int_{x \in S^1(r)} x_1^{n_1} dx = \frac{r^{n_1+1}}{n_1+1} = \frac{r^{n_1+1} n_1!}{(n_1+1)!} ;$$

4. классическое упражнение на многократное интегрирование по частям:
$$\int_{x \in S^1(r)} x_1^{n_1} (r - x_1)^{n_2} dx = \frac{r^{n_1+n_2+1} n_1! n_2!}{(n_1+n_2+1)!} ;$$

5. его обобщение индукцией по размерности:

$$\int_{x\in S^{d-1}(r)} \prod_{j=1}^{d-1} x_j^{n_j}\cdot(r-\sum_{j=1}^{d-1}x_j)^{n_d}dx = \frac{r^{\sum_{j=1}^d n_j+(d-1)}\prod_{j=1}^d n_j!}{\left(\sum_{j=1}^d n_j+(d-1)\right)!}\ .$$

Применяя пункты 1, 2 и 5 в симплексе $s^d(1)$, получаем утверждение леммы. Подробности оставляются в качестве **упражнения**. □

Теперь можно провести байесовское обучение наивной байесовской модели на обучающем наборе $T=((x_1,y_1),\ldots,(x_N,y_N))$. Введем обозначения

$$N_k = \#\{i|y_i=k\},\quad k=1,\ldots,q \tag{31}$$

$$N^j_{mk} = \#\{i|x^j_i=m, y_i=k\},\ j=1,\ldots,d,\ m=1,\ldots,M^j,\ k=1,\ldots,q\ . \tag{32}$$

Тогда плотность условной вероятности

$$\begin{aligned} p(T|\pi) &= \prod_{i=1}^N p((x_i,y_i)|\pi) = \prod_{i=1}^N p(y_i|\pi)p(x_i|y_i,\pi) \\ &= \prod_{i=1}^N\left(p(y_i|\pi)\prod_{j=1}^d p(x^j_i|y_i,\pi)\right) = \prod_{k=1}^q\left((\pi_k)^{N_k}\prod_{j=1}^d\prod_{m=1}^{M^j}(\pi^j_{mk})^{N^j_{mk}}\right) \end{aligned} \tag{33}$$

Применяя лемму 1 в симплексах s^q и $s^{M^j}_{jk}$, и учитывая, что $\sum\limits_{k=1}^q N_k = N$ и $\sum\limits_{m=1}^{M^j} N^j_{mk} = N_k$, получим, что плотность апостериорной вероятности (20)

$$p(\pi|T) = \frac{\prod_{k=1}^q\left((\pi_k)^{N_k}\prod_{j=1}^d\prod_{m=1}^{M^j}(\pi^j_{mk})^{N^j_{mk}}\right)}{\frac{(q-1)!\prod_{k=1}^q N_k!}{(N+q-1)!}\prod_{k=1}^q\prod_{j=1}^d\frac{(M^j-1)!\prod_{m=1}^{M^j}N^j_{mk}!}{(N_k+M^j-1)!}}\ , \tag{34}$$

где числитель является мономом от параметров распределения, а знаменатель от них не зависит.

Предсказываемая обученной моделью совместная вероятность конкретных признаков x и ответа y равна

$$\begin{aligned} P_T(x,y) &= \int_{\pi\in\mathcal{P}} P_\pi(x,y)p(\pi|T)d\pi \\ &= \int_{\pi\in\mathcal{P}} \frac{\prod_{k=1}^q\left((\pi_k)^{N_k}\prod_{j=1}^d\prod_{m=1}^{M^j}(\pi^j_{mk})^{N^j_{mk}}\right)}{\frac{(q-1)!\prod_{k=1}^q N_k!}{(N+q-1)!}\prod_{k=1}^q\prod_{j=1}^d\frac{(M^j-1)!\prod_{m=1}^{M^j}N^j_{mk}!}{(N_k+M^j-1)!}}\pi_y\prod_{j=1}^d\pi^j_{x^jy}d\pi \\ &= \frac{\int_{\pi\in\mathcal{P}}\prod_{k=1}^q\left((\pi_k)^{(N_k+\delta^k_y)}\prod_{j=1}^d\prod_{m=1}^{M^j}(\pi^j_{mk})^{(N^j_{mk}+\delta^k_y\delta^m_{x^j})}\right)d\pi}{\frac{(q-1)!\prod_{k=1}^q N_k!}{(N+q-1)!}\prod_{k=1}^q\prod_{j=1}^d\frac{(M^j-1)!\prod_{m=1}^{M^j}N^j_{mk}!}{(N_k+M^j-1)!}} \\ &= \frac{N_y+1}{N+q}\prod_{j=1}^d\frac{N^j_{x^jy}+1}{N_y+M^j}\ , \end{aligned} \tag{35}$$

где δ^b_a — *символ Кронекера*; при интегрировании опять использовалась лемма 1 и два похожих громоздких знаменателя с факториалами почти сократились. Предсказываемая моделью условная вероятность выглядит менее изящно:

$$P_T(y|x) = \frac{P_T(x,y)}{\sum_{k=1}^q P_T(x,k)} = \frac{(N_y+1)\prod_{j=1}^d\frac{N^j_{x^jy}+1}{N_y+M^j}}{\sum_{k=1}^q(N_k+1)\prod_{j=1}^d\frac{N^j_{x^jk}+1}{N_k+M^j}}\ . \tag{36}$$

Быстрее и точнее вычислять не сами вероятности $P_T(x,y)$, а их логарифмы. Еще проще вычислять логарифмы отношений таких вероятностей для пар ответов. Из формулы (35) следует, что для пары ответов y' и y''

$$\begin{aligned}\Delta_T(y',y''|x) &= \ln\frac{P_T(x,y')}{P_T(x,y'')} \\ &= \ln(N_{y'}+1) - \ln(N_{y''}+1) + \sum_{j=1}^{d}\left(\ln\frac{N^j_{x^j y'}+1}{N_{y'}+M^j} - \ln\frac{N^j_{x^j y''}+1}{N_{y''}+M^j}\right) \\ &= \left(\ln(N_{y'}+1) - \sum_{j=1}^{d}\ln(N_{y'}+M^j) - \left(\ln(N_{y''}+1) - \sum_{j=1}^{d}\ln(N_{y''}+M^j)\right.\right. \\ &+\sum_{j=1}^{d}\left(\ln(N^j_{x^j y'}+1) - \ln(N^j_{x^j y''}+1)\right)\end{aligned}$$

Значение (37) называется *дискриминантом* ответов y' и y'' при векторе признаков x. Если дискриминант положителен, то вектор x с большей вероятностью принадлежит классу y', чем y'', а если отрицателен - наоборот.

По значениям дискриминантов $\Delta_T(y',y''|x)$ при любом зафиксированнном ответе y'' и всех ответах y' можно восстановить условные вероятности (36) (**упражнение!**), но для выбора наиболее вероятного ответа, что во некоторых случаях распознавания является конечной целью, это не требуется: наиболее вероятным является ответ y', максимизирующий дискриминант $\Delta_T(y',y''|x)$ (тоже **упражнение!**).

Замечание. Дискриминант (37) двух фиксированных классов, рассматриваемый как функция от вектора признаков x, является суммой константы и d функций, каждая из которых зависит только от одного признака. Такие распознаватели называются *аддитивными*.

Обучение дискретной наивной байесовской модели с равномерным априорным распределением максимизацией апостериорной вероятности

Теперь вместо интегрирования мономов по симплексам, как при байесовском обучении, будем искать точки максимума тех же мономов по тем же симплексам.

Лемма 2 *Максимум монома* $p(x) = x_1^{n_1}\cdots x_d^{n_d}$ *по симплексу* $s^d(1)$, *достигается в точке*

$$x^* = \left(\frac{n_1}{\sum_{j=1}^d n_j}, \ldots, \frac{n_d}{\sum_{j=1}^d n_j}\right).$$

Доказательство. Поскольку в любой внутренней точке симплекса значение монома положительно, в точке максимума оно тоже положительно, значит если $n_j > 0$, то $x_j^* > 0$. Наоборот, если $n_j = 0$, то $x_j^* = 0$ (что и утверждается лемме), поскольку иначе значение монома в точке $\frac{1}{1-x_j^*}x^* - \frac{x_j^*}{1-x_j^*}e^j$, где e^j — j-й базисный вектор, больше, чем в точке x^*. Так что, возможно, рассматривая вместо исходного симплекса его грань, можно считать, что все $n_j > 0$ и что точка максимума обязана находиться внутри симплекса.

Выписывая лагранжиан $L(x,\lambda) = p(x) + \lambda\left(1 - \sum_{j=1}^d x_j\right)$ и приравнивая нулю его производные, получаем систему уравнений относительно x и λ

$$\begin{cases} 0 = \frac{\partial L}{\partial x_j} = \frac{n_j p(x)}{x_j} - \lambda \qquad j = 1,\ldots,d \\ 0 = \frac{\partial L}{\partial \lambda} = 1 - \sum_{j=1}^d x_j \end{cases}$$

Домножая j-е уравнение из первых d на x_j, суммируя по j и учитывая последнее уравнение, получаем

$$\lambda = p(x)\sum_{j=1}^{d} n_j \ .$$

Подставляя это значение λ в первые d уравнений, получаем утверждение леммы. $\square$

Поскольку априорное распределение на пространстве моделей равномерно, максимум апостериорной плотности (27) достигается там же, где максимум правдоподобия $p(T|\pi)$. Применяя лемму 2 к плотности (33), получаем точку максимума $\pi_k = \frac{N_k}{N}$, $\pi^j_{mk} = \frac{N^j_{mk}}{N_k}$. То есть, обученная модель в точке (x, y) предсказывает вероятность

$$P_T(x,y) \quad = \quad \frac{N_y}{N}\prod_{j=1}^{d}\frac{N^j_{x^j y}}{N_y} = \frac{\prod_{j=1}^{d} N^j_{x^j y}}{(N_y)^{d-1}N} \ , \tag{38}$$

а значит условную вероятность

$$P_T(y|x) \quad = \quad \frac{(N_y)^{1-d}\prod_{j=1}^{d} N^j_{x^j y}}{\sum_{k=1}^{q}(N_k)^{1-d}\prod_{j=1}^{d} N^j_{x^j k}} \tag{39}$$

и дискриминанты

$$\begin{aligned}\Delta_T(y', y''|x) &= \ln\frac{P_T(x,y')}{P_T(x,y'')} \\ &= (1-d)\left(\ln(N_{y'}) - \ln(N_{y''})\right) + \sum_{j=1}^{d}\left(\ln(N^j_{x^j y'}) - \ln(N^j_{x^j y''})\right)\end{aligned} \tag{40}$$

Сравнение методов обучения дискретной наивной байесовской модели

Сравнение формул (35) и (38) показывает, что при одних и тех же априорных предположениях разные методы обучения наивной байесовской модели дают разный результат: байесовское обучение — *верный*, максимизация правдоподобия — *неверный*, да и по вычислительной сложности они мало отличаются. Разница между ответами не слишком велика, когда все числа N^j_{mk} большие, и весьма существенна, когда какие-то из них равны нулю: модель, обученная максимизацией правдоподобия, считает невозможным то, что ей не предъявили во время обучения. По сравнению с ней модель, обученная байесовским методом, ведет себя приблизительно (не в точности) так, как если бы кроме обучающего набора ей еще по одному разу предъявили все возможные сочетания векторов признаков и ответов.

И байесовское обучение, и обучение максимизацией апостериорной вероятности, описанные в этом разделе, легко переносятся с равномерного априорного распределения на распределение с плотностью, пропорциональной моному общего вида

$$p(\pi) \propto \prod_{k=1}^{q}\left((\pi_k)^{n^k}\prod_{j=1}^{d}\prod_{m=1}^{M^j}(\pi^j_{mk})^{n^m_{jk}}\right) \ , \tag{41}$$

в котором показатели даже не обязательно целые, а вещественные неотрицательные для максимизации апостериорной вероятности и не меньшие -1 для байесовского обучения. Действительно, в лемме 2 использовалась только неотрицательность показателей, а не их целочисленность, а лемма 1 верна и при

нецелых показателях с заменой факториалов на Г-функции[7], которые потом точно так же, как факториалы, почти сокращаются.

Упражнение. Проведите байесовское обучение и максимизацию апостериорной вероятности для дискретной наивной байесовской модели с априорным распределением, пропорциональным мономy (41).

Замечание. В рецептурных справочниках по алгоритмам распознавания *наивный метод Байеса* иногда излагается в три с половиной строчки:

- для набора T вычислить N_k и N_{mk}^j по формулам (31–32);
- вычислить для вектора признаков x и всевозможных q ответов y вероятность $P_T(x, y)$ по формуле (35) или (38);
- выбрать ответ y, максимизирующий $P_T(x, y)$.

Непрерывные наивные байесовские модели

Конечность пространства ответов $\mathcal{Y}$ для наивных байесовских моделей существенна (т.е. они применимы для классификации, но не для регрессии), а конечность пространства признаков $\mathcal{X}$ совершенно необязательна. Некоторые признаки могут быть непрерывны, некоторые дискретны — лишь бы для каждого признака x^j условные вероятности или их плотности $p(x^j|y)$ принадлежали конечномерным пространствам, на которых к тому же можно как-то разумно определить априорное распределение. В частности, непрерывные признаки можно дискретизировать, т.е. разбить область допустимых значений каждого признака на непересекающиеся отрезки и заменять значение признака номером содержащего это значение отрезка, причем способ разбиения может не быть фиксирован заранее, а определяться по обучающему набору[8]. Обучение максимизацией апостериорной вероятности обычно не очень сложно. Честное байесовское обучение, требующее интегрирования по многомерным пространствам, реализуемо только при некоторых довольно простых априорных распределениях.

Применимость наивного байесовского метода

Наивный байесовский метод довольно успешно применяется в задачах классификации с очень большим числом признаков, как правило, дискретных, про взаимозависимость которых заранее ничего не понятно. Если множество признаков имеет какую-то структуру, например, признаки являются значениями чего-либо в разные моменты времени (звуковой сигнал) или в разных точках плоскости (оптический сигнал), то про взаимозависимость признаков и ответов можно сделать более разумные предположения, чем в наивной байесовской модели, и получить более точные модели (*скрытые марковские модели*, *марковские поля* и другие). Очень многие методы обучения дают более точные распознаватели, чем наивный байесовский, но по скорости обучения он вне конкуренции.

1.2.6 Обучение дискриминантных моделей (продолжение)

При построении дискриминантных моделей пространство условных распределений $p(y|x)$ отождествляется с пространством распознавателей $\mathcal{Y}$, и в зависимости от того, с какой стороны на него смотрят, получаются два изоморфных, но терминологически совершенно не похожих, подхода.

[7]Напоминание: $\Gamma(x) = \int_0^\infty t^{x-1}e^{-t}dt$ при $x > 0$. Интегрированием по частям получаются тождества $\Gamma(x) = (x-1)!$ при целых x и $\Gamma(x+1) = x\Gamma(x)$ при любых.

[8]То же самое имеет смысл делать для целочисленных признаков, количество допустимых значений которых сравнимо с длиной обучающего набора или превышает ее.

Подход со стороны распознавателей (см., например, [52], первоисточник неизвестен): пусть распознаватели из пространства $\mathcal{F}$ вычисляют не принадлежащие небольшому пространству $\mathcal{Y}$ (например, конечному или $\mathbb{R}$) ответы классификации или регрессии, а принадлежащие большему пространству $\mathcal{P}_\mathcal{Y}$ их распределения (в случае дискретных ответов) или параметры распределений (в случае непрерывных ответов), см. раздел 1.1.2. Пусть также на самом пространстве распознавателей $\mathcal{F}$ определено распределение β. Формально β — обычная вероятностная мера, но ее положено называть не вероятностью, а *доверием* (верой, *belief*) и произносить ритуальные заклинания про то, что это доверие является чьей-то субъективной априорной оценкой адекватности распознавателя. Аналогично, распределение $f(x)$ на $\mathcal{Y}$, где $f \in \mathcal{F}$ и $x \in \mathcal{X}$, положено называть условным распределением доверий к ответам (вер в ответы) $y \in \mathcal{Y}$ при условии результата распознавания (т.е. предсказанного распределения) $f(x)$. Воздержимся от философически-филологических упражнений по теории веры и будем формально обращаться с доверияими как с вероятностями.

Подход со стороны моделей (см., например, [14]) состоит в том, что β — действительно обычная вероятностная мера на пространстве $\mathcal{F}$, состоящем, однако не из самих распознавателей, а из моделей условных распределений $p_f(y|x)$, где $f \in \mathcal{F}$, $x \in \mathcal{X}$ и $y \in \mathcal{Y}$, являющихся неполными моделями распределения $p(x,y) = p(y|x)p(x) \in \mathcal{P}$. Задача обучения состоит в том, чтобы выбрать хорошую в каком-либо смысле модель. Имея такую модель, можно предсказывать распределение вероятностей ответов (или сами ответы) в каждой точке пространства признаков $\mathcal{X}$ точно так же, как и при наличии порождающей модели. Распределение $p(x)$ на пространстве $\mathcal{X}$ и при таком подходе остается неизвестным; известно только, что именно им порождены векторы признаков $x_1, \ldots, x_N$ обучающего набора.

До конца этого параграфа будут продемонстрированы оба варианта терминологии [второй — в квадратных скобках], далее будет использоваться только второй (модели).

Для простоты обозначений ограничимся случаем, когда распределения β на $\mathcal{F}$ и $f(x)$ на $\mathcal{Y}$ непрерывны с плотностями p и $p_{f(x)}$ соответственно. Тогда условная плотность доверия распознавателя f к тому, что последовательности векторов признаков $\mathbf{X}=(x_1, \ldots, x_N)$ обучающего набора $T = ((x_1, y_1), \ldots, (x_N, y_N))$ соответствует последовательность ответов $\mathbf{Y}=(y_1, \ldots, y_N)$ [т.е. вероятность последовательности ответов $\mathbf{Y}$ при последовательности признаков $\mathbf{X}$ в модели f], равна

$$p(\mathbf{Y}|\mathbf{X}, f) = \prod_{i=1}^{N} p_{f(x_i)}(y_i), \tag{42}$$

совместная плотность доверия к ответам обучающего набора и распознавателю f [условная плотность вероятности модели и ответов при данных признаках] равна

$$p(\mathbf{Y}, f|\mathbf{X}) = p(\mathbf{Y}|\mathbf{X}, f)p(f),$$

полная плотность доверия к ответам [плотность вероятности ответов при условии данных признаков] обучающего набора

$$p(\mathbf{Y}|\mathbf{X}) = \int_{f\in\mathcal{F}} p(\mathbf{Y}, f|\mathbf{X})df = \int_{f\in\mathcal{F}} p(\mathbf{Y}|\mathbf{X}, f)p(f)df$$

и плотность доверия к распознавателю f [плотность вероятности модели f] при условии обучающего набора T равна

$$p(f|T) = p(f|\mathbf{X}, \mathbf{Y}) = \frac{p(\mathbf{Y}, f|\mathbf{X})}{p(\mathbf{Y}|\mathbf{X})} = \frac{p(\mathbf{Y}|\mathbf{X}, f)p(f)}{\int_{f\in\mathcal{F}} p(\mathbf{Y}|\mathbf{X}, f)p(f)df}\ . \tag{43}$$

Формулу (43) можно применять для обучения распознавателей по крайней мере двумя способами: байесовским и максимизацией доверия [апостериорной вероятности], ср. с разделом 1.2.4. При байесовском подходе строится усредненный распознаватель [модель]

$$f_T = \int_{f\in\mathcal{F}} f p(f|T) df$$

(чтобы усреднение f_T принадлежало пространству $\mathcal{F}$, достаточно чтобы пространство было замкнуто и выпукло), который замечателен всем, кроме практической невозможности честно его вычислить, кроме отдельных полезных случаев, например, когда все распределения $f(x)$ — гауссовы, а β — тоже некоторого специального вида. А вот максимизацию апостериорного доверия [вероятности] рассмотрим подробнее.

Максимизация выражения (43) по f эквивалентна максимизации его числителя

$$p(\mathbf{Y}, f|\mathbf{X}) = p(f)\prod_{i=1}^{N} p_{f(x_i)}(y_i) \to \max_{f\in\mathcal{F}} \tag{44}$$

или минимизации функции

$$-\ln(p(\mathbf{Y}, f|\mathbf{X})) = -\ln(p(f)) - \sum_{i=1}^{N} \ln(p_{f(x_i)}(y_i)) \to \min_{f\in\mathcal{F}}. \tag{45}$$

Введем функцию штрафа $E(r, y) = -\ln(p_r(y))$ и обозначим через $\psi(f)$ выражение $-\ln(p(f))$. В таких терминах минимизация (45) превращается в

$$\psi(f) + \sum_{i=1}^{N} E(f(x_i), y_i) \to \min_{f\in\mathcal{F}}, \tag{46}$$

т.е. в минимизацию суммарного штрафа с регуляризацией (8) в несколько более общей ситуации. А именно, штраф E теперь определен не на пространстве пар ответов $\mathcal{Y}\times\mathcal{Y}$, а на пространстве пар распределение-ответ $\mathcal{P}_\mathcal{Y}\times\mathcal{Y}$.

Аналогично максимизация правдоподобия (42), для которой не требуется определять априорное распределение на пространстве распознавателей [моделей], обобщает минимизацию средней ошибки (4) без регуляризации.

1.2.7 Пример: регрессия методом наименьших квадратов

Предположим, что пространство ответов — действительные числа ($\mathcal{Y} = \mathbb{R}$), пространство признаков $\mathcal{X}$ — любое, но на нем задано отображение $\phi : \mathcal{X} \to \mathbb{R}^d$. Рассмотрим $d+1$-мерное пространство $\mathcal{F}$ дискриминантных моделей (условных распределений)

$$p_{w,s}(y|x) = (2\pi s)^{-\frac{1}{2}} e^{-\frac{(y-w\phi(x))^2}{2s}}\ , \tag{47}$$

состоящее из гауссовых распределений с общей дисперсией s и линейно зависящими от $\phi(x)$ центрами $w\phi(x) = \sum_{j=1}^{d} w_j\phi^j(x)$. Это соответствует ситуации, когда ответ должен бы определяться вектором признаков однозначно, но к нему добавляется большое количество мелких независимых случайных ошибок, а распределение суммы большого количества независимых случайных величин близко к гауссову по центральной предельной теореме.

Применяя для обучения этой модели на обучающем наборе T самый простой и "заведомо ложный" метод максимизации правдоподобия (42), а лучше — минимизации его минус логарифма

$$-\ln(p(\mathbf{Y}, w, s|\mathbf{X})) = \frac{N\ln(2\pi)}{2} + \frac{N\ln(s)}{2} + \frac{1}{2s}\sum_{i=1}^{N}(y_i - w\phi(x_i))^2 \to \min_{w,s}\ ,$$

получаем, что задача распадается на минимизацию по w суммы квадратов

$$\sum_{i=1}^{N}(y_i - w\phi(x_i))^2 \to \min_w \tag{48}$$

(это — классический *метод наименьших квадратов* Гаусса[9] для приближения функций, известных в конечном числе точек x_i, линейными комбинациями базисных функций $\phi^j(x)$) и последующую минимизацию по s

$$\frac{N\ln(s)}{2} + \frac{1}{2s}\sum_{i=1}^{N}(y_i - w\phi(x_i))^2 \to \min_s ,$$

с очевидным ответом

$$s = \frac{1}{N}\sum_{i=1}^{N}(y_i - w\phi(x_i))^2 .$$

Обученная модель для любого вектора признаков x предсказывает распределение ответа y по формуле (47) с обученными коэффициентами, т.е. в частности предсказывает и ожидание ответа $w\phi(x)$, и его дисперсию s.

Любопытно, что если обучать модели при фиксированном значении дисперсии s (например, $s = 2008$), прогнозы для ожиданий ответов будут теми же, хотя прогноз дисперсии $s = 2008$ — абсолютно не относящимся к делу.

1.2.8 Пример: применение регрессии для классификации с оценкой вероятностей классов

Пространство распределений $\mathcal{P}_\mathcal{Y}$ на пространстве ответов $\mathcal{Y} = \{1, \ldots, q\}$ является $(q-1)$-мерным симплексом

$$\{(y^1, \ldots, y^q) \in \mathbb{R}^q_+ | \sum_{j=1}^{q} y^j = 1\} ,$$

пространство моделей $\mathcal{F}$ можно считать вложенным в пространство функций из пространства признаков $\mathcal{X}$ в $\mathbb{R}^q$, удовлетворяющих вероятностным нормировкам ($f^j \geq 0$ и $\sum_{j=1}^{q} f^j = 1$), где $f^j(x) = p(j|x, f)$. В такой постановке задача классификации оказывается сведена к задаче многомерной регрессии со специфическими ограничениями.

Пространство ответов $\mathcal{Y}$, естественно отображается на вершины симплекса $\mathcal{P}_\mathcal{Y}$, а именно, $k \mapsto (\delta^1_k, \ldots, \delta^q_k)$. Тогда условную вероятность $p(y|x, f)$ формально можно записать в виде

$$p(y|x, f) = \prod_{j=1}^{q}(f^j(x))^{y^j}. \tag{49}$$

Формула (49) определена для любого вектора $y \in \mathcal{P}_\mathcal{Y}$, что позволяет обучать модель не только на достоверных, но и на "стохастических" обучающих объектах, состоящий из пар вида (вектор признаков, распределение вероятностей ответов). Соответствующая формуле (49) функция штрафа

$$E(f(x), y) = -\ln(p(y|x, f)) = -\sum_{j=1}^{q} y^j \ln(f^j(x)) \tag{50}$$

[9] Вычислительные подробности метода наименьших квадратов и обучения этой же модели методом максимизации апостериорной вероятности, рассмотрены в разделе 2.1; здесь это только иллюстрация к обучению дискриминантных моделей.

совпадает с *взаимной энтропией* (*cross entropy*) распределений $f(x)$ и y, определенных на q-элементном множестве $\{1,\dots,q\}$. Получается, что обучение распознавателя методом максимизации апостериорной вероятности (формулы (44), (45), (46)), равносильно решению задачи

$$\psi(f) - \sum_{i=1}^{N}\sum_{j=1}^{q} y_i^j \ln(f^j(x_i)) \to \min_{f\in\mathcal{F}} , \tag{51}$$

а обучение методом максимизации правдоподобия — просто минимизации взаимной энтропии

$$-\sum_{i=1}^{N}\sum_{j=1}^{q} y_i^j \ln(f^j(x_i)) \to \min_{f\in\mathcal{F}} . \tag{52}$$

Для двухклассовой классификации можно обозначить $f(\cdot) = f^1(\cdot)$ и $y_i = y_i^1$ и упростить формулы (51) и (52) до

$$\psi(f) - \sum_{i=1}^{N} \big(y_i \ln(f(x_i)) + (1-y_i)\ln(1-f(x_i))\big) \to \min_{f\in\mathcal{F}} , \tag{53}$$

и

$$-\sum_{i=1}^{N} \big(y_i \ln(f(x_i)) + (1-y_i)\ln(1-f(x_i))\big) \to \min_{f\in\mathcal{F}} , \tag{54}$$

соответственно.

Замечание. Вместо того, чтобы обобщать постановку задачи классификации настолько, чтобы можно было обучать распознаватель на образцах с недостоверной (вероятностной) классификацией, можно было бы ограничиться пространством достоверных ответов $\mathcal{Y} = \{1,\dots,q\}$, но при этом все равно обучаться оценивать их вероятности. Все формулы при таком ограничении и соответствующих ему обозначениях немного упрощаются: штраф (50) превращается в

$$E(f(x), y) = -\ln f^y(x) , \tag{55}$$

а, обучение методом максимизации апостериорной вероятности (51) превращается в

$$\psi(f) - \sum_{i=1}^{N} \ln(f^{y_i}(x_i)) \to \min_{f\in\mathcal{F}} . \tag{56}$$

Таким образом, обучение модели, оценивающей вероятности классов, сводится к минимизации вполне определенной функции штрафа (50) или (55). Конкретные пространства моделей (распознавателей) $\mathcal{F}$ и методы их обучения будут рассматриваться далее, например, в разделе 2.2.2.

1.3 Другие задачи статистического обучения

В этом разделе приведено всего лишь несколько примеров типов задач, похожих на распознавание и решаемых методами статистического обучения. Есть и другие типы таких задач. И наоборот, для приведенных задач есть совершенно другие методы решения.

1.3.1 Обучение с учителем и без

Вспомним метод моделирования распределения $\pi \in \mathcal{P}$ (раздел 1.2.4), которое опять для простоты обозначений будем считать непрерывным. Его плотность $p_\pi(x, y)$ представима в виде

$$p_\pi(x, y) = p_\pi(x)p_\pi(y|x), \text{ где } p_\pi(x) = \int_{y \in \mathcal{Y}} p_\pi(x, y)dy$$

(ср. с формулой(1)). Моделирование распределения можно разбить на два этапа: отдельно моделирование плотности $p_\pi(x)$, описывающей распределение наборов признаков безотносительно к ожидаемым ответам, и отдельно моделирование условной плотности $p_\pi(y|x)$, которую и оценивает распознаватель. Независимость первого этапа от ответов замечательна тем, что для обучения этой части модели не требуется размеченный обучающий набор (признаки и ответы), который может быть весьма ограничен, а годятся любые наборы признаков. Например, при обучении распознаванию речи годятся записи произнесения слов, собираемые автоматически, без их расшифровки, делаемой вручную.

Такое обучение, использующее только векторы признаков и игнорирующее соответствующие им ответы, называется *обучением без учителя* (*unsupervised learning*) в отличие от рассматривавшегося ранее *обучения с учителем* (*supervised learning*). Обучение без учителя применимо не только в качестве промежуточного этапа обучения распознавателя, но и в некоторых других задачах статистического обучения.

1.3.2 Оценка плотности и обнаружение выбросов

Оценка плотности распределения π на пространстве $\mathcal{X}$ по выборке $\mathbf{X} \in \mathcal{X}^N$ — это одна из любимых задач классической статистики. Пример в разделе 1.3.1 показывает, что она может быть полезна при обучении распознаванию. Другой (тривиальный) пример ее применения в обучении распознавателя был приведен в разделе 1.2.5 при оценке вероятностей $p(y)$ всевозможных ответов: в этом случае ответы обучающего набора выступают в роли признаков. И вообще, оценка плотности является частным случаем распознавания, а именно обучением вероятностной модели распределения на пространстве $\{0\} \times \mathcal{Y}$ по обучающему набору $T = ((0, y_1), \ldots, (0, y_N))$.

Оценка плотности может применяться как промежуточный этап и для решения задачи обнаружения *выбросов* (*outlier* detection), являющейся вырожденным случаем задачи двухклассовой классификации: в обучающем наборе представлены только объекты класса “нормальные”, а объекты класса “выбросы” (объекты с аномальными значениями признаков) не представлены. Отсутствие выбросов в обучающем наборе необходимо компенсировать априорной информацией об их вероятности и, возможно, распределении.

Другой подход к обнаружению выбросов больше похож на обучение распознавателя минимизацией эмпирического риска с регуляризацией: в заранее заданном классе “сигналов тревоги” $\mathcal{F} \subset \{f : \mathcal{X} \to \{0, 1\}\}$ на обучающем наборе T минимизировать какую-либо функцию от поданных сигналом ложных тревог и “сложности” сигнальной функции. Пример такого обучения приведен в разделе 4.2.7.

Подробно рассматриваться задача обнаружения выбросов не будет: она приведена только как пример задачи, в которой применяется обучение без учителя.

1.3.3 Кластеризация

Кластеризацией называется следующая задача. Есть некоторое количество N объектов, т.е. обучающий набор векторов признаков. Построить алгоритм

(*кластеризатор*), разбивающий пространство признаков $\mathcal{X}$ на конечное число q областей, называемых *кластерами*, так чтобы векторы, из одного кластера были больше похожи друг на друга, чем векторы из разных кластеров. Конечно, понятие похожести нужно как-то определить, например, для метрического пространства признаков, через расстояние в нем. Таким образом, кластеризатор можно считать, как и классификатор, отображением $f : \mathcal{X} \to \mathcal{Y} = \{1, \ldots, q\}$, а точнее, в отличие от классификатора, — классом таких отображений, переводимых друг в друга перестановками множества $\mathcal{Y}$.

Довольно часто кластеризация применяется при определении понятия классов для последующей классификации. Например, классификация в ботанике и зоологии начиналась с того, что имелось огромное количество описаний живых существ с разным количеством, цветом и формой листьев, лепестков, ног и хвостов, и нужно было разбить их на кучки похожих и предъявить алгоритм отнесения каждого нового существа к какой-то из кучек.

Кластеризация похожа на классификацию, в которой в обучающем наборе отсутствуют ответы — номера классов. Чисто теоретически, можно назначить эти ответы всевозможными способами (очень многими: N^q), обучить N^q классификаторов, выбрать один из самых лучших (которых не менее $q!$) и использовать в качестве кластеризатора. На практике, конечно, обучить столько классификаторов невозможно, но идея использовать в качестве кластеризатора подходящий классификатор вполне применима.

Задачу кластеризации можно разными способами видоизменять. Например, можно, как и в задаче многоклассовой классификации, строить кластеризатор $f : \mathcal{X} \to \mathbb{R}^q$, для каждой точки x пространства признаков вычисляющий не только ответ $\arg\max_j f_j(x)$ (номер кластера), но и уверенность в нем, равную $\max_j f_j(x)$. Такой кластеризатор заодно может решать задачу обнаружения *выбросов* (раздел 1.3.2).

Обучать такой кластеризатор можно следующим образом: потребовать, чтобы его ответы удовлетворяли обычным вероятностным нормировкам $f_j(x) \geq 0$ и $\sum_j f_j(x) = 1$ и для обучающего набора $\mathbf{X}$ в стандартном предположении о независимости обучающих векторов искать "наиболее правдоподобную" кластеризацию

$$\max_{(j_1,\ldots,j_N)\in\mathcal{Y}^N} \prod_{i=1}^{N} f_{j_i}(x_i) = \prod_{i=1}^{N} \max_{j\in\mathcal{Y}} f_j(x_i) \to \max_{f\in\mathcal{F}}. \tag{57}$$

Для решения этой задачи возьмем какой-нибудь метод обучения классификаторов, максимизирующий оценку уверенности классификатора в ответах на обучающем наборе, и какую-нибудь, хоть бы и случайную, кластеризацию. Обучим классификатор классифицировать в соответствии с этим назначением кластеров. Затем каждый вектор признаков из обучающего набора классифицируем с помощью обученного классификатора. Хотя классификатор и обучался на этом же самом обучающем наборе, для некоторых обучающих векторов его ответ может отличаться от того, чему его учили. Для полученного разбиения на классы снова обучим классификатор, затем в соответствии с результатами обучения снова подправим классификацию и т.д. При обучении произведение уверенностей (57) монотонно не убывает. Обучение останавливается, когда это произведение перестает возрастать. При таком обучении классификатор учится классифицировать не так, как задано в каком-то обучающем наборе, а так, как ему удобно максимизировать уверенность в своей работе. Результат обучения кластеризатора сильно зависит от начального разбиения на кластеры.

Подробно эмпирические методы выбора начального разбиения или, что эквивалентно, начального состояния классификаторов, здесь не обсуждаются. Например, можно начинать обучение следующим образом: выбрать из обучающего набора несколько случайных векторов, по одному на кластер, объ-

явить их принадлежащими разным классам и обучить классификатор только на них, после чего продолжать обучение уже на всем обучающем наборе. Во время обучения тоже могут возникнуть неоднозначности, разрешаемые довольно произвольно. К какому кластеру отнести вектор, для которого некоторые из уверенностей в принадлежности к разным кластерам совпадают (варианты: к случайному, к кластеру с меньшим номером, к кластеру с меньшим количеством векторов, что в свою очередь неоднозначно,...)? Что делать, если какой-то кластер стал пустым (варианты: не обращать внимания, продолжать кластеризацию с меньшим числом кластеров, родить этот кластер заново, поместив в него вектор с наименьшей уверенностью,...)?

В качестве примера рассмотрим байесовский q-классовый классификатор (раздел 1.2.1) на d-мерном евклидовом пространстве признаков $\mathcal{X}$, для следующей порождающей модели: вероятности $p(y)$ всех классов равны $\frac{1}{q}$, а плотности условных вероятностей $p(x|y)$ являются гауссовыми с единичной матрицей ковариации

$$p(x|j) = (2\pi)^{-\frac{d}{2}} e^{-\frac{1}{2}(x-\mu_j)^2} \quad , \quad 1 \le j \le q.$$

Обучение модели состоит в подборе q векторов $\mu_1, \ldots, \mu_q$ в пространстве $\mathcal{X}$. При обучении методом наибольшего правдоподобия (формула (28)) на обучающем наборе $\mathbf{X} = (x_1, \ldots, x_N)$ с каким-то распределением $y_1, \ldots, y_N$ обучающих векторов по кластерам, максимизируется по параметрам модели произведение вероятностей

$$\prod_{i=1}^{N} p(x_i, y_i) = \prod_{i=1}^{N} (p(x_i|y_i)p(y_i)) = \frac{1}{q^N} \prod_{i=1}^{N} p(x_i|y_i) \to \max_{\mu} ,$$

т.е. минимизируется сумма

$$\sum_{i=1}^{N} -\ln(p(x_i|y_i)) \to \min_{\mu} ,$$

что эквивалентно минимизации суммы квадратов

$$\sum_{i=1}^{N} (x_i - \mu_{y_i})^2 \to \min_{\mu} . \tag{58}$$

Квадратичная минимизация сводится к очень простой системе линейных уравнений, которая элементарно решается:

Упражнение. Докажите, что

$$\mu_j = \frac{\sum_{i|y_i=j} x_i}{\#\{i|y_i = j\}}$$

— точка глобального минимума суммы (58).

Таким образом, каждый вектор параметров μ_j сдвигается в центр тяжести (среднее арифметическое) обучающих векторов j-го кластера. Затем каждый обучающий вектор x_i переводится в тот кластер j, для которого оценка байесовского распознавателя $p(x_i|j)$ максимальна, т.е. расстояние $\|x_i - \mu_j\|$ минимально. Снова обучается классификатор, снова пересчитываются кластеры, и т.д., пока правдоподобие не перестанет увеличиваться, т.е. распределение обучающих векторов по кластерам не перестанет изменяться.

Этот пример обучения кластеризатора по совместительству является самым распространенным примером описываемого в разделе 1.3.4 векторного квантования. Он обычно называется "*k-means*" (*k средних*), где k — обозначение числа

кластеров (то, что выше обозначалось буквой q), а "means" — память о вычислении средних арифметических. В течение 25 лет он был известен в научном фольклоре, прежде чем его автор Стюарт Ллойд (Stuart Lloyd) опубликовал его в открытой печати: [79]. Как правило, метод k-means сходится быстро, хотя есть и примеры очень медленной сходимости ([5]). Напоминаем, что получающаяся кластеризация зависит от начальной кластеризации и не обязательно оптимальна.

Метод k-means легко обобщить, допустив в качестве плотностей условных вероятностей гауссианы с любыми матрицами ковариации, в том числе и с разными матрицами для разных кластеров.

Количество кластеров k может быть фиксировано заранее, а может, аналогично обучению распознавателей с регуляризацией (8), подбираться минимизацией суммы какого-либо зависящего от k штрафа "за сложность" и убывающей функции от уверенности (57). Стандартных рецептов, как именно штрафовать и как минимизировать по дискретному параметру k, нет.

1.3.4 Векторное квантование и понижение размерности

Очень распространена следующая модификация задачи кластеризации. Одновременно с поиском хорошей кластеризующей функции $f : \mathcal{X} \to \mathcal{Y}$ будем искать "функцию центров" $c : \mathcal{Y} \to \mathcal{X}$, сопоставляющую каждому кластеру y его "типичного представителя" $c(y)$, называемого центром кластера. Хочется, чтобы центр кластера был достаточно близок ко всем объектам этого кластера.

В такой постановке задача кластеризации обычно называется *векторным квантованием* (*vector quantization*). Векторное квантование часто применяется при необратимом сжатии информации для хранения или передачи по линиям связи. Информация любой природы представляется, возможно искусственно, в виде последовательности векторов признаков, лежащих в пространстве признаков $\mathcal{X}$. При кодировании (сжатии) каждый вектор кодируется номером кластера, которому он принадлежит. При декодировании (раздутии) каждый номер кластера заменяется на его центр.

Задачу векторного квантования можно формализовать в том же стиле, что и задачу распознавания (раздел 1.1.4). Имеется:

пространство (векторов) признаков $\mathcal{X}$, точками которого кодируются квантуемые объекты, например d-мерное евклидово пространство $\mathbb{R}^d$ или любое метрическое пространство;

***пространство кластеров* $\mathcal{Y}$,** обычно конечное, хотя возможны и обобщения;

пространство $\mathcal{F}$ кластеризующих функций $f : \mathcal{X} \to \mathcal{Y}$, например, любых функций из $\mathcal{X}$ в $\mathcal{Y}$;

пространство $\mathcal{C}$ функций центров $c : \mathcal{Y} \to \mathcal{X}$, как правило, любых функций из $\mathcal{Y}$ в $\mathcal{X}$;

пространство $\mathcal{P}$ распределений (вероятностных мер) на $\mathcal{X}$, например, в случае для евклидова пространства $\mathcal{X}$, — абсолютно непрерывных по мере Лебега, гауссовых смесей и т.п.;

функция штрафа $E : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, как правило, неотрицательная и равная 0 при совпадении параметров, например, в случае евклидова пространства $\mathcal{X}$ удобно взять $E(x, s) = \|x - s\|^2$;

обучающий набор $\mathbf{X}=(x_1,\dots,x_N)$, где векторы признаков $x_i \in \mathcal{X}$ считаются значениями независимых случайных величин с одним и тем же, но совершенно неизвестным распределением $\pi \in \mathcal{P}$.

Хочется по $\mathcal{X}$, $\mathcal{Y}$, $\mathcal{F}$, $\mathcal{C}$, $\mathcal{P}$, E и $\mathbf{X}$ построить кластеризатор $f \in \mathcal{F}$ и функцию центров $c \in \mathcal{C}$, минимизирующие ожидание штрафа $E_\pi(f)$

$$E_\pi(f,c) = \int_{x\in\mathcal{X}} E(c(f(x)),x)d\pi(x) \to \min_{f\in\mathcal{F},c\in\mathcal{C}}, \tag{59}$$

где $\pi \in \mathcal{P}$. Как и в случае распознавания, можно сложную задачу минимизации интеграла подменить задачей минимизации приближающей его суммы

$$E_\pi(f,c) \approx E(f,c,\mathbf{X}) = \frac{1}{N}\sum_{i=1}^{N} E(c(f(x_i)),x_i) \to \min_{f\in\mathcal{F},c\in\mathcal{C}}, \tag{60}$$

но тогда, чтобы обеспечить малость ожидания штрафа (59) может понадобиться регуляризация, ср. с разделом 1.1.6. Обратите внимание на то, что отображения f и c входят в формулы (59) и (60) только в виде композиции $c \circ f$.

Дальше, как и в случае распознавания, можно забыть про вероятностную модель и решать задачу, подобную (60), а качество полученного квантования оценивать экспериментально, вычисляя штраф $E(f,c,\mathbf{X}')$ на независимом *тестовом наборе* $\mathbf{X}'$. А можно продолжать изучать вероятностную модель и перенести на случай кластеризации и векторного квантования соображения из раздела 1.2.4 об апостериорной вероятности или байесовском обучении.

В таком виде задача обучения осмысленна и полезна не только при конечном пространстве $\mathcal{Y}$, но при любом. Например, если $\mathcal{X}$ и $\mathcal{Y}$ — евклидовы пространства, $\dim(\mathcal{Y}) < \dim(\mathcal{X})$, а класс $\mathcal{C}$ состоит из гладких или линейных функций, то обучение квантованию равносильно поиску $\dim(\mathcal{Y})$-мерного подмножества (гладкой поверхности или плоскости, соответственно), хорошо аппроксимирующего встречающиеся векторы признаков. Только называется эта задача уже не квантованием, а *понижением размерности* (*dimension reduction*).

Квантование и понижение размерности можно рассматривать как частный случай задачи регрессии: пространство ответов $\mathcal{Y}$ совпадает с пространством признаков $\mathcal{X}$, и нужно приблизить тождественное отображение с помощью отображений вида $c \circ f$, множество значений которых либо конечно (квантование), либо маломерно (понижение размерности).

Огромное количество методов обучающегося векторного квантования (*LVQ*, *learning vector quantization*) и понижения размерности с помощью *самоорганизующихся отображений* (*SOM*, *self-organizing maps*), описаны в книге [68]. Такие методы классической статистики, как *факторный анализ* (factor analysis) и *анализ главных компонент* (principal component analysis) также являются методами понижения размерности, работающими в предположении гауссовости распределений $p(x|y)$, см., например, [15].

2 Линейные распознаватели: обзор

Линейными называются распознаватели, у которых обучающаяся часть является линейным отображением, хотя не обязательно прямо из пространства признаков $\mathcal{X}$ в пространство ответов $\mathcal{Y}$ (здесь и далее используются обозначения раздела 1.1.4). В самом общем виде это распознаватели, представимые в виде композиции

$$\mathcal{X} \xrightarrow{\phi} \mathcal{X}' \xrightarrow{f} \mathcal{Y}' \xrightarrow{\sigma} \mathcal{Y},$$

где пространства $\mathcal{X}$ и $\mathcal{Y}$ — любые, $\mathcal{X}'$ и $\mathcal{Y}'$ — линейные (точнее, аффинные, но начало координат, да и вся система координат, обычно зафиксированы, так что их можно считать даже не просто векторными, а стандартными евклидовыми), отображения ϕ и σ фиксированы, а отображение f ищется (обучается) в некотором пространстве $\mathcal{F}$, состоящем из линейных (опять-таки, аккуратно говоря, аффинных) операторов $\mathcal{X}' \to \mathcal{Y}'$. То есть, обучаемые функции имеют вид $f(x) = wx$ или $f(x) = wx + b$, соответственно.

В *линейной регрессии*, как правило, $\mathcal{Y}' = \mathcal{Y} = \mathbb{R}$ и σ — тождественное отображение. В *линейной классификации* пространство ответов $\mathcal{Y}$, конечно же, не является линейным, а σ является какой-либо пороговой функцией, например, для двухклассовой классификации с ответами $\mathcal{Y} = \{0, 1\}$ в качестве функции σ можно брать *θ-функцию Хевисайда* $\theta(t) = \frac{1}{2}(1 + \operatorname{sign}(t))$, игнорируя недопустимое значение $\theta(0) = \frac{1}{2}$ или произвольно переопределяя значение в нуле. Для оценки вероятностей классов, когда пространство ответов $\mathcal{Y}$ является симплексом в аффинном пространстве, в качестве σ применяются функции специального вида, см. раздел 2.2.2.

В разделах 2.1 и 2.2 будет считаться, что признаки уже закодированы набором чисел фиксированной длины, $\mathcal{X} = \mathcal{X}'$ и преобразование $\phi : \mathcal{X} \to \mathcal{X}'$ тождественно. Нетривиальные преобразования ϕ будут рассмотрены в разделе 2.3.

Линейные распознающие системы активно изучались в середине 20-го века, когда для работы с более сложными алгоритмами не хватало вычислительных мощностей (цифровых вычислительных машин не было вообще). С развитием вычислительной техники интерес сместился в сторону более универсальных нелинейных алгоритмов. Однако и в 21-м веке линейные обучающиеся системы работают не хуже более сложных нелинейных в следующих ситуациях:

- когда пространство допустимых распределений вероятностей $\mathcal{P}$ состоит только из очень простых распределений, например, из гауссовых смесей с фиксированной матрицей ковариации, и линейность оптимальных распознающих функций можно доказать;
- когда размерность пространства признаков $\mathcal{X}$ очень велика по сравнению с количеством доступных обучающих данных, и даже в классе линейных распознавателей легко полностью устранить ошибку обучения (4), но трудно бороться с переобучением;
- когда удается отобразить пространство настоящих признаков $\mathcal{X}$, само не обязательно линейное, в линейное *пространство вторичных признаков* (также называемое *спрямляющим пространством*) $\mathcal{X}'$ большой по сравнению с обучающим набором размерности и, тем самым, свести ситуацию к предыдущему пункту.

Остаток этого раздела состоит из примеров несложных линейных систем распознавания, рассматриваемых как иллюстрации к общим принципам из раздела 1, и примеров построения пространств вторичных признаков. Большая часть примеров заимствована из книг [14, 50] и конспекта лекций [98]. Некоторые более сложные линейные распознаватели подробнее рассматриваются в последующих разделах.

Иногда для простоты будут рассматриваться однородные линейные распознаватели $f(x) = wx$ вместо аффинных $f(x) = wx + b$. Конечно, от аффинных операторов всегда можно перейти к линейным, увеличив размерность пространства-прообраза $\mathcal{X}'$ на 1 и приписав к векторам $x = (x^1, \ldots, x^d)$ дополнительную координату $x^0 = 1$, а к матрицам операторов w — дополнительный столбец $w_0 = b$. Но при этом теряется инвариантность методов обучения распознавателей относительно сдвигов в пространстве признаков $\mathcal{X}$.

Далее будем обозначать через $\bar{x}$ вектор x с приписанной единицей

$$\bar{x} = \begin{pmatrix} 1 \\ x \end{pmatrix} \in \mathbb{R} \times \mathcal{X}' \,. \tag{61}$$

Парой букв w и b будем обозначать аффинный оператор $\mathcal{X}' \to \mathcal{Y}'$, а одной буквой $\bar{w}$ — эквивалентный ему линейный оператор

$$\bar{w} = (\ b\ w\ ) : \mathbb{R} \times \mathcal{X}' \to \mathcal{Y}' \,. \tag{62}$$

В случае одномерного пространства $\mathcal{Y}'$ (т.е. в задачах одномерной регрессии и одно- или двухклассовой классификации) наряду с операторным обозначением $f(x) = \bar{w}\bar{x}$ будет использоваться скалярное произведение $f(x) = (\bar{w}, \bar{x})$ или $f(x) = b + (w, x) = w^0 + (w, x)$.

2.1 Линейная регрессия

2.1.1 Минимизация квадратичной ошибки

Задача (48) минимизации квадратичной ошибки обучения для одномерной линейной регрессии превращается в

$$E(\bar{w}, T) = \sum_{i=1}^{N} (y_i - \bar{w}\bar{x}_i)^2 \to \min_{\bar{w}} . \tag{63}$$

Минимизируемый функционал $E(\bar{w}, T)$ квадратичен и неотрицательно определен по $\bar{w}$ (тривиальное **упражнение**), поэтому у него всегда есть решение, хотя не обязательно единственное. В случае неединственности решения образуют аффинное подпространство пространства операторов $\bar{w}$. Задача (63) эквивалентна системе линейных уравнений

$$\frac{\partial E(\bar{w}, T)}{\partial \bar{w}} = 0.$$

Чтобы выписать решение явно, перейдем к матричным обозначениям. Обозначим через $\mathbf{X} \in (\mathbb{R} \times \mathcal{X})^N \simeq \mathbb{R}^{(d+1)N}$ и $\mathbf{y} \in \mathcal{Y}^N \simeq \mathbb{R}^N$ матрицы

$$\mathbf{X} = (\bar{x}_1, \ldots, \bar{x}_N) = \begin{pmatrix} 1 & \cdots & 1 \\ x_1^1 & \cdots & x_N^1 \\ \vdots & \ddots & \vdots \\ x_1^d & \cdots & x_N^d \end{pmatrix}$$

и

$$\mathbf{y} = (y_1, \ldots, y_N)$$

соответственно[10]. При одномерном пространстве ответов $\mathcal{Y}$ линейный оператор $\bar{w}$ является просто ковектором (вектором-строкой)

$$\bar{w} = (w_0, w_1, \ldots, w_d).$$

Тогда формула (63) записывается в виде

$$E(\bar{w}, T) = (\mathbf{y} - \bar{w}\mathbf{X})(\mathbf{y} - \bar{w}\mathbf{X})^\top \to \min_{\bar{w}}, \tag{64}$$

[10]Напомним, что в принятых обозначениях *векторы* признаков считаются векторами-столбцами и их координаты обозначаются верхними индексами, а номера обучающих векторов — нижними. Широко распространена и транспонированная система обозначений, идущая не от линейной алгебры, а от устройства файлов с данными и электронных таблиц, в которой векторы признаков являются строками. В такой системе обозначений все последующие матричные вычисления нужно транспонировать.

а условие минимума[11] —

$$0 = \frac{\partial E(\bar{w}, T)}{\partial \bar{w}} = -2\mathbf{X}(\mathbf{y} - \bar{w}\mathbf{X})^\top, \tag{65}$$

то есть

$$\bar{w}\mathbf{X}\mathbf{X}^\top = \mathbf{y}\mathbf{X}^\top. \tag{66}$$

Матрицы $\mathbf{X}\mathbf{X}^\top$ и $\mathbf{y}\mathbf{X}^\top$, точнее говоря, матрицы $\frac{1}{N}\mathbf{X}\mathbf{X}^\top$ и $\frac{1}{N}\mathbf{y}\mathbf{X}^\top$, имеют простой смысл — это эмпирические оценки корреляций

$$\left(\int_{(x,y)\in\mathcal{X}\times\mathcal{Y}} x^k x^l d\pi(x,y)\right)_{1\leq k,l\leq d}$$

между входными признаками и

$$\left(\int_{(x,y)\in\mathcal{X}\times\mathcal{Y}} y x^k d\pi(x,y)\right)_{1\leq k\leq d}$$

между ответом и входными признаками, соответственно. Если матрица $\mathbf{X}\mathbf{X}^\top$ не вырождена, то решение задачи (63) задается явной формулой

$$\bar{w} = \mathbf{y}\mathbf{X}^\top(\mathbf{X}\mathbf{X}^\top)^{-1}, \tag{67}$$

а если матрица вырождена, то решение существует, но не единственно. На практике непосредственно решать систему уравнений (66) относительно $\bar{w}$ быстрее и лучше, чем вычислять обратную матрицу и подставлять ее в формулу (67).

В случае q-мерного пространства ответов $\mathcal{Y}$ вместо стандартной квадратичной ошибки — суммы квадратов покоординатных разностей — может применяться квадратичная ошибка более общего вида $E(r,y) = A(r-y, r-y)$ с любой положительно определенной квадратичной формой A (переходящая в стандартную при некоторой линейной замене координат). В матричном виде эту функцию ошибки удобно выразить немного неожиданным образом через след:

$$E(r,y) = (r-y)^\top A(r-y) = \operatorname{tr}\left(A(r-y)(r-y)^\top\right).$$

Тогда вместо формулы (64) суммарная ошибка записывается в виде

$$E(\bar{w}, T) = \operatorname{tr}\left(A(\bar{w}\mathbf{X} - \mathbf{y})(\bar{w}\mathbf{X} - \mathbf{y})^\top\right),$$

и дифференцированием по $\bar{w}$ ее минимизация сводится к решению системы

$$A\mathbf{X}(\bar{w}\mathbf{X} - \mathbf{y})^\top = 0,$$

где матрица $\mathbf{y}$ уже q-строчная:

$$\mathbf{y} = (y_1, \ldots, y_N) = \begin{pmatrix} y_1^1 & \cdots & y_N^1 \\ \vdots & \ddots & \vdots \\ y_1^q & \cdots & y_N^q \end{pmatrix}.$$

Независимо от конкретной невырожденной квадратичной формы A эта система равносильна системе (66).

Упражнение. Если $\sum_{i=1}^N x_i = 0$ и $\sum_{i=1}^N y_i = 0$, то во всех решениях задачи минимизации квадратичной ошибки (63) $w_0 = 0$.

[11] Здесь и далее уравнения и переходы между ними, не всегда очевидны (="видны глазом"), но всегда легко проверяются прямым вычислением в одну-две строки.

2.1.2 Минимизация квадратичной ошибки с регуляризацией

Различные мотивации обучения регрессии посредством минимизации регуляризованной *квадратичной ошибки* (формулы (8), (46)) в линейном случае обычно превращаются в минимизационную задачу

$$\psi(\|\bar{w}\|) + \sum_{i=1}^{N}(y_i - \bar{w}\bar{x}_i)^2 \to \min_{\bar{w}}, \tag{68}$$

где $\|\cdot\|$ — некоторая норма в пространстве коэффициентов, например, l^p-норма, т.е. корень p-й степени из суммы p-х степеней координат при $p > 0$ и количество ненулевых координат при $p = 0$. Эта норма совершенно не обязана быть симметричной по координатам. Действительно, один признак может измеряться в метрах (или милях), а другой — в тоннах (или унциях), и было бы странно одинаково штрафовать коэффициенты при них, к тому же независимо от принятых единиц измерения. На практике предпочитают все же симметричные нормы, а признаки, наоборот, предварительно нормируют, например, уравнивая их дисперсии на обучающем наборе[12].

Интерес представляют те функции регуляризации $\psi(\|\cdot\|)$, при которых либо задача (68) имеет серьезную вероятностную мотивацию, либо ее легко решать, либо ее решения обладают какими-то хорошими свойствами. Рассмотрим несколько самых известных примеров.

Квадрат l^2-нормы $\epsilon\|\bar{w}\|_2^2$ или $\epsilon\|w\|_2^2$ (*гребневая регрессия*, *ridge regression* [56])

Этот пример соответствует обучению методом максимизации апостериорной вероятности дискриминантной модели

$$p_{\bar{w}}(y|x) = (2\pi)^{-\frac{1}{2}} e^{-\frac{(y-\bar{w}\phi(x))^2}{2}} \tag{69}$$

(ср. с моделью (47)) при сферическом гауссовом распределению с центром 0 и дисперсией $\frac{1}{\epsilon}$ априорной вероятности коэффициентов распознавателя. Получающаяся минимизационная задача

$$\epsilon\|\bar{w}\|_2^2 + \sum_{i=1}^{N}(y_i - \bar{w}\bar{x}_i)^2 \to \min_{\bar{w}}\ , \tag{70}$$

как и задача (63), является квадратичной и положительно определенной (в отличие от задачи (63) она строго положительно определена) и решается явно. Решение совершенно аналогично решению задачи (63), и ответ задается формулой

$$\bar{w} = \mathbf{y}\mathbf{X}^\top(\epsilon I_{d+1} + \mathbf{X}\mathbf{X}^\top)^{-1} \tag{71}$$

(ср. с формулой (67)), где через I_n обозначена матрица тождественного оператора в $\mathbb{R}^n$, т.е. единичная матрица размера $n \times n$. Заметим, что при таком подходе свободный член распознавателя, спрятанный в нулевой столбец w_0 его матрицы (62), тоже штрафуется.

Чаще распознаватель обучают без штрафа за свободный член. Обозначим через $I_n^0 = 0_1 \oplus I_{n-1}$ оператор проекции на $(n-1)$-мерное подпространство в $\mathbb{R}^n$, порожденное всеми координатами, кроме самой первой. Тогда распознаватель обучается решением задачи

[12] Более радикальная нормировка — применение линейного преобразования, переводящего матрицу ковариаций признаков на обучающем наборе в единичную, — чересчур трудоемка.

$$\epsilon\|w\|_2^2 + \sum_{i=1}^{N}(y_i - \bar{w}\bar{x}_i)^2 = \epsilon\|\bar{w}I_{d+1}^0\|_2^2 + \sum_{i=1}^{N}(y_i - \bar{w}\bar{x}_i)^2 \to \min_{\bar{w}} \tag{72}$$

вместо задачи (70) и аналогично формуле (71) результат обучения предъявляется явно:

$$\bar{w} = \mathbf{y}\mathbf{X}^\top(\epsilon I_{d+1}^0 + \mathbf{X}\mathbf{X}^\top)^{-1}.$$

Упражнение. Докажите, что матрица $\epsilon I_{d+1}^0 + \mathbf{X}\mathbf{X}^\top$ обратима.

l^1-норма $\epsilon\|w\|_1$ (*лассо*, *LASSO*, *lasso regression* [110])

Этот пример соответствует обучению методом максимизации апостериорной вероятности дискриминантной модели (69) при многомерном распределении Лапласа с центром в 0 и плотностью

$$p(w) = \left(\frac{a}{2}\right)^{qd} e^{-a\|w\|_1}$$

априорной вероятности коэффициентов распознавателя (исключая свободный член). Получающаяся минимизационная задача

$$\epsilon\|w\|_1 + \sum_{i=1}^{N}(y_i - \bar{w}\bar{x}_i)^2 \to \min_{\bar{w}} \tag{73}$$

выпукла и, хотя аналитического решения предъявить нельзя, методы решения таких задач, называемых задачами квадратичного программирования, довольно хорошо изучены. В частности, разработан специфический алгоритм (модификация алгоритма *LARS*, см. [35]) для решения сразу всего семейства задач (73) при всех значениях параметра ϵ.

Решение задачи (73) часто бывает *разреженным*, т.е. многие элементы матрицы оператора $\bar{w}$ равны 0. А разреженность решения, то есть независимость результата распознавания от некоторых входных признаков, и ускоряет обучение и работу распознавателя, и дает важную информацию о самой задаче распознавания. Особенно актуально это, если размерность d пространства признаков намного больше количества N обучающих векторов.

Чтобы понять причину возможной разреженности решения удобно перейти от семейства задач (73), зависящих от параметра $\epsilon \geq 0$, к семейству задач

$$\sum_{i=1}^{N}(y_i - \bar{w}\bar{x}_i)^2 \to \min_{\|w\|_1 \leq C}, \tag{74}$$

зависящих от параметра $C > 0$, имеющему те же решения (предложение 1). Множество вида $\|w\|_1 \leq C$ — это выпуклый многогранник в пространстве коэффициентов (конкретнее, выпуклая оболочка $2qd$ точек, лежащих на координатных осях, например, в трехмерном пространстве — октаэдр). Он имеет конечное число (а именно, $3^{qd} - 1$) граней разных размерностей, направления которых не зависят от C. Минимизируемая функция является неотрицательно определенной квадратичной формой от $\bar{w}$, и ее множества уровня являются семейством концентрических эллипсоидов или, если форма вырождена, эллипсоидальных цилиндров. Их центр (соответственно, ось) почти всегда отличен от центра многогранника, т.е. начала координат (соответственно, не проходит через него). Решение задачи (74) — это точка, в которой многогранник опирается на минимальный из пересекающихся с ним эллипсоидов семейства. Почти всегда касательная плоскость в начале координат к проходящему через начало координат эллипсоиду уровня не параллельна никакой грани многогранника, кроме нульмерных. Тогда если многогранник очень мал (C мало, а ϵ велико),

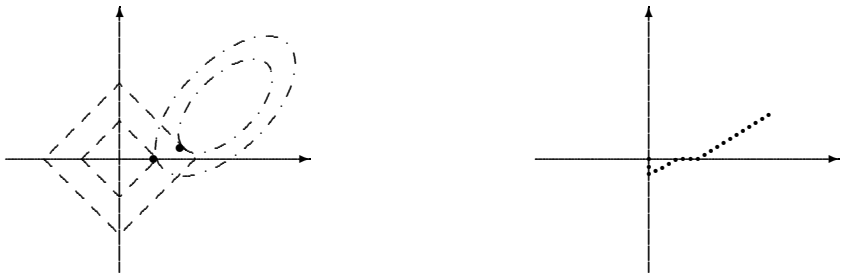

Рис. 1: Разреженность решения задачи (74). Обе картинки изображают пространство коэффициентов распознавателя. Слева штриховые линии — это линии уровня l^1-нормы, штрих-пунктирные — линии уровня суммарной ошибки на обучающем наборе, жирными точками показана пара решений, разреженное и не разреженное. Справа (несколько утрированно) изображено все семейство решений при различных C.

то опорный эллипсоид проходит через его вершину. При этом у решения задачи (74) будет только одна ненулевая координата. По мере роста многогранника (увеличения C) опорная точка будет переходить на другие грани, вообще говоря, большей размерности (т.е. решение будет иметь все больше ненулевых координат, см. рис. 1) и когда многогранник пройдет через центр эллипсоида, решение уже почти наверняка не будет разреженным.

В семействе регуляризирующих штрафов, зависящих от l^p-нормы коэффициентов, случай $p = 1$ — пограничный. Минимизационная задача (68) выпукла (и значит, хорошо решается алгоритмически) при $p \geq 1$, а решение разрежено при $p \leq 1$.

Комбинация l^1 и l^2-норм (*elastic net* [127])

Регуляризирующий штраф при обучении линейной регрессии совершенно не обязан быть какой-то l^p-нормой коэффициентов. Например, возможно обучение (68) с штрафом вида $\psi(\bar{w}) = \epsilon_1\|w\|_1 + \epsilon_2\|w\|_2^2$, обладающее при положительных ϵ_1 и ϵ_2 свойствами, промежуточными между гребневой регрессией и лассо. Минимизационная задача (68) при таком штрафе выпукла, и при любом фиксированном ϵ_2 все семейство ее решений при разных ϵ_1 вычисляется, как и в методе лассо, модификацией алгоритма LARS. Действительно, добавка слагаемого $\epsilon_2\|w\|_2^2$ в обучение лассо (73) равносильна добавлению к обучающему набору вектора

$$(x_0, y_0) = \left(\begin{pmatrix} 0 \\ \sqrt{\epsilon_2} \\ \vdots\sqrt{\epsilon_2} \end{pmatrix}, 0 \right) .$$

Получающиеся распознаватели, как и в методе лассо, разрежены, хотя и не так сильно. Качественно это можно понять, рассмотрев вырожденную ситуацию, когда несколько признаков принимают почти одинаковые значения на каждом обучающем векторе[13]. Тогда, как и при обучении методом гребневой регрессии, за счет квадратичного слагаемого регуляризации обученный распознаватель будет зависеть от всех этих признаков с почти одинаковыми коэффициентами, а распознаватель, обученный методом лассо, почти всегда будет зависеть только от одного из этих признаков.

[13]Такая ситуация реально встречается при анализе экспрессий генов после нормировки признаков.

В статье [127] приводятся примеры задач, в которых метод elastic net[14] дает меньшую ошибку распознавания, чем гребневая регрессия и лассо, а его решения, менее разреженные по сравнению с решениями лассо, более адекватно (не с точки зрения минимизации эмпирического риска, а с точки зрения содержания конкретной задачи) находят существенные для распознавания признаки.

l^0-норма $\epsilon\|w\|_0$ (*subset selection*)

В этом примере штрафуется в точности количество ненулевых коэффициентов распознавателя, так что меняя параметр ϵ можно добиться любой желаемой разреженности. Удобнее сразу перейти к задаче с ограничением

$$\sum_{i=1}^{N}(y_i - \bar{w}\bar{x}_i)^2 \to \min_{\|w\|_0 \le C}. \tag{75}$$

Простейший алгоритм ее решения — перебор всех C-элементных подмножеств элементов матрицы w, т.е. dq-элементного множества, и поиска минимума квадратичной ошибки при остальных элементах матрицы, приравненных нулю, — является экспоненциально медленным по произведению dq размерностей пространств входов и выходов. Известны и более быстрые алгоритмы: например, метод "leaps and bounds" ([43]) работоспособен при десятках (до 50) коэффициентов. Но все-таки методы непрерывной оптимизации, дающие разреженность коэффициентов в качестве побочного эффекта, работают быстрее прямой дискретной оптимизации.

2.1.3 Минимизация других ошибок

Наряду с применением минимизации квадратичной ошибки и разных регуляризирующих штрафов для обучения распознавателей можно минимизировать и другие функции ошибок. Например, применяются следующие функции ошибок (см. рис. 2):

линейная $E(r,y) = |y-r|$;

степенная [15] $E(r,y) = |y-r|^p$, $p > 0$;

ошибка Хьюбера ([60]) $E(r,y) = \begin{cases} |y-r|^2 & \text{при } |y-r| \le \epsilon \\ \epsilon(2|y-r| - \epsilon) & \text{при } |y-r| > \epsilon \end{cases}$;

линейная ϵ-нечувствительная $E(r,y) = \max(0, |y-r| - \epsilon)$;

квадратичная ϵ-нечувствительная $E(r,y) = (\max(0, |y-r| - \epsilon))^2$.

То, что во всех примерах ошибка зависит только от разности прогноза и истинного ответа, не случайно — это самый распространенный подход —, но и не строго обязательно. Разнообразие применяемых функций ошибки связано с попытками совместить несколько несовместимых желаний:

- функция ошибки должна быть непрерывной (для устойчивости алгоритмов обучения);
- функция ошибки должна быть выпуклой, желательно строго выпуклой (для единственности решений минимизационной задачи);

[14] Выражение “elastic net” (“эластичная сеть”) в качестве названия метода линейной регрессии весьма неудачно, поскольку традиционно оно относится к построению самоорганизующихся отображений, и используется довольно неформально.

[15] На практике из степенных применяются только квадратичная или линейная ошибки для регрессии и “ошибка нулевой степени”, равная 0-1-ошибке, для классификации.

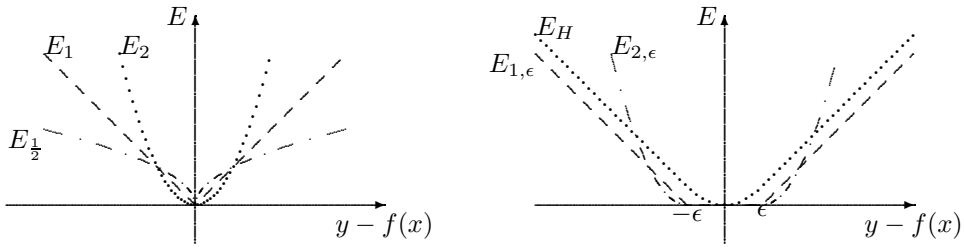

Рис. 2: Функции ошибок, применяемые при обучении регрессии. Слева — степенные ошибки E_p, справа — ошибка Хьюбера E_H и ϵ-нечувствительные ошибки $E_{p,\epsilon}$ разных степеней. Все эти ошибки зависят только от разности прогноза и истинного ответа.

- функция ошибки должна быть дифференцируемой, желательно дважды дифференцируемой (для применения градиентных алгоритмов);
- функция ошибки должна быть достаточно простой, желательно линейной (для применения быстрых оптимизационных алгоритмов);
- ошибка должна быть очень мала, желательно равна 0, там, где она неизбежна (в пределах погрешности измерения или оцифровки), или там, где она приемлема (в пределах допустимой погрешности ответа);
- ошибка не должна быть очень велика нигде (чтобы один сумасшедший обучающий вектор, естественно, неверный, не мог испортить результат длительного кропотливого обучения).

Самое популярное в настоящее время сочетание — ϵ-нечувствительная ошибка (обычно линейная, редко квадратичная или ошибка Хьюбера) и квадратичный регуляризирующий штраф. Называется оно *регрессией методом опорных векторов* (*SVR*, *support vector regression*), подробно рассматриваемой в разделе 4.2.4. При $d \gg N$ часто применяется двойственное к этому сочетание, а именно, квадратичная ошибка и линейный регуляризирующий штраф (lasso (стр. 43)).

2.2 Линейная классификация

Напоминаем, что при классификации можно пытаться оценить вероятность $P\{j|x\} = f^j(x)$ принадлежности входного вектора признаков x каждому из q классов, а можно, хотя и не всегда хорошо, просто предсказать, какому классу этот вектор принадлежит. Далее рассматриваются оба случая, каждый несколькими способами. В первом случае кажется понятным, что именно может быть линейным — функции $f^j(x)$, — и тогда задача сводится к линейной регрессии. На самом деле обычно линейными бывают не прогнозы вероятностей $f^j(x)$, а другие функции, через которые они выражаются. Во втором случае вообще не понятно, что может быть линейного в функции с конечным числом значений. Ответ: линейными (конкретнее, гиперплоскостями или многогранниками в них) могут быть границы между прообразами разных значений, а также функции, положительные на векторах одного класса и отрицательные на векторах другого (*дискриминанты*).

2.2.1 Линейный дискриминантный анализ (*дискриминант Фишера*)

Построим для q-классовой классификации с d-мерным евклидовым пространством признаков $\mathcal{X}$ простую (и, вообще говоря, не соответствующую действительности) порождающую модель (раздел 1.2.4). А именно, предположим, что для любого (j-го) из q классов распределение векторов признаков этого класса — гауссово с центром $\mu_j \in \mathcal{X}$ и неизвестной, но одинаковой для всех классов матрицей ковариации A, симметричной и предположительно невырожденной. Вероятность j-го класса обозначим через π_j и будем считать, что $\pi_j > 0$ (иначе j-й класс можно было бы выкинуть из модели). То есть[16]

$$p_j(x) = p(x|y=j) = (2\pi)^{-\frac{d}{2}}|A|^{-\frac{1}{2}}e^{-\frac{1}{2}(x-\mu_j)^\top A^{-1}(x-\mu_j)},$$
$$p(x,y) = \pi_y p_y(x) = \pi_y(2\pi)^{-\frac{d}{2}}|A|^{-\frac{1}{2}}e^{-\frac{1}{2}(x-\mu_y)^\top A^{-1}(x-\mu_y)} \tag{76}$$

и

$$p(x) = \sum_{j=1}^{q} p(x,j) = \sum_{j=1}^{q} \pi_j(2\pi)^{-\frac{d}{2}}|A|^{-\frac{1}{2}}e^{-\frac{1}{2}(x-\mu_j)^\top A^{-1}(x-\mu_j)}.$$

Тогда условные вероятности $P\{y|x\}$, которые и должен оценивать классификатор, равны

$$\begin{aligned} P\{y|x\} &= \frac{p(x,y)}{p(x)} = \frac{\pi_y e^{-\frac{1}{2}(x-\mu_y)^\top A^{-1}(x-\mu_y)}}{\sum_{j=1}^{q}\pi_j e^{-\frac{1}{2}(x-\mu_j)^\top A^{-1}(x-\mu_j)}} \qquad (77)\\ &= \frac{e^{\mu_y{}^\top A^{-1}x - \frac{1}{2}\mu_y{}^\top A^{-1}\mu_y + \ln(\pi_y)}}{\sum_{j=1}^{q} e^{\mu_j{}^\top A^{-1}x - \frac{1}{2}\mu_j{}^\top A^{-1}\mu_j + \ln(\pi_j)}}, \end{aligned}$$

то есть имеют вид

$$P\{y|x\} = \frac{e^{\bar{w}^y\bar{x}}}{\sum_{j=1}^{q} e^{\bar{w}^j\bar{x}}} \tag{78}$$

для некоторого аффинного оператора $\bar{w}: \mathcal{X} \to \mathbb{R}^q$.

Обучение модели состоит в оценке ее параметров π_j, μ_j при $1 \le j \le q$ и A по имеющемуся обучающему набору T. Это можно сделать методом наибольшего правдоподобия, т.е. решением задачи

$$-\ln(p(T)) = -\sum_{i=1}^{N}\ln(p(x_i,y_i)) \to \min_{\pi,\mu,A},$$

где плотности вероятностей $p(x_i,y_i)$ вычисляются по формуле (76). Поскольку сумма вероятностей π_j всех классов равна 1, нужно написать лагранжиан

$$\begin{aligned} L(T,\pi,\mu,A,\lambda) &= -\ln(p(T)) + \lambda\left(\sum_{j=1}^{q}\pi_j - 1\right) \qquad (79)\\ &= -\sum_{i=1}^{N}\left(\ln(\pi_{y_i}) - \frac{d}{2}\ln(2\pi) - \frac{1}{2}\ln|A| - \frac{1}{2}(x_i-\mu_{y_i})^\top A^{-1}(x_i-\mu_{y_i})\right)\\ &\quad + \lambda\left(\sum_{j=1}^{q}\pi_j - 1\right) \end{aligned}$$

[16] В последующих формулах используются матричные обозначения: значение квадратичной формы A^{-1} на паре векторов признаков x_1 и x_2 записывается как $x_2{}^\top A^{-1}x_1$, а индексы элементов матриц A^{-1} и A пишутся снизу. Можно писать и в виде скалярного произведения с действием оператора $(A^{-1}x_1, x_2)$, и в тензорном виде $A^{-1}(x_1,x_2)$ — кто как привык.

и приравнять нулю его производные по всем переменным λ, π_j, μ_j и A. Удобно дифференцировать не по элементам матрицы A, а по диагональным и наддиагональным элементам матрицы A^{-1} (поддигональные элементы определяются симметричностью матрицы и не считаются независимыми переменными).

Упражнение. Докажите, что для обратимой матрицы M общего вида

$$\frac{\partial}{\partial\left(M^{-1}\right)_{kl}}(-\ln|M|) = \frac{\partial}{\partial\left(M^{-1}\right)_{kl}}(\ln|M^{-1}|) = M_{lk}\ ,$$

а для симметричной матрицы A^{-1}, параметризованной своей верхне-треугольной частью,

$$\frac{\partial}{\partial\left(A^{-1}\right)_{kl}}(\ln|A^{-1}|) = (\#\{k,l\})\,A_{lk}\ , \qquad k \le l\ ,$$

т.е. при производных по недиагональным элементам, появляющимся в матрице дважды, появляется коэффициент 2.

Получающаяся система уравнений

$$\begin{aligned}
0 = \frac{\partial L}{\partial \lambda} &= \sum_{j=1}^{q} \pi_j - 1 \\
0 = \frac{\partial L}{\partial \pi_j} &= -\frac{\#\{i|y_i = j\}}{\pi_j} + \lambda \\
0 = \frac{\partial L}{\partial \mu_j} &= -\sum_{i|y_i=j} A^{-1}(x_i - \mu_j) \\
0 = \frac{\partial L}{\partial A^{-1}_{kl}} &= (\#\{k,l\})\left(-\frac{N}{2}A_{lk} + \frac{1}{2}\sum_{i=1}^{N}(x_i^k - \mu_{y_i}^k)(x_i^l - \mu_{y_i}^l)\right)
\end{aligned}$$

легко решается:

$$\begin{aligned}
\lambda &= N \\
\pi_j &= \frac{\#\{i|y_i = j\}}{N} && (80) \\
\mu_j &= \frac{\sum_{i|y_i=j} x_i}{\#\{i|y_i = j\}} && (81) \\
A_{kl} &= \frac{\sum_{i=1}^{N}(x_i^k - \mu_{y_i}^k)(x_i^l - \mu_{y_i}^l)}{N}, && (82)
\end{aligned}$$

и получаются традиционные в статистике оценки вероятности как частоты, математического ожидания как центра тяжести и ковариации. Впрочем, оценка ковариации при методе максимизации правдоподобия получается заниженной: несмещенная оценка в $\frac{N}{N-q}$ раз больше (**упражнение!**). Матрица A может оказаться вырожденной (и обязательно окажется вырожденной при $N \le d$).

Упражнение. Рассмотрите аккуратно случай вырожденной матрицы ковариации A. Вот где проявляется слабость обучения методом наибольшего правдоподобия! Априорного распределения не хватает.

Плотность совместного распределения (76) оценена, значит получены и прогнозы условных вероятностей (77), а заодно обучен байесовский классификатор (раздел 1.2.1). А при чем тут дискриминантный анализ, да еще и линейный?

Дискриминантами называются функции, различающие классы, то есть такие функции $\Delta_{ij}(x)$, что неравенство $p_i(x) > p_j(x)$ (вектор x скорее принадлежит i-му классу, чем j-му) равносильно неравенству $\Delta_{ij}(x) > 0$. В частности, для любого классификатора, оценивающего условные плотности вероятностей

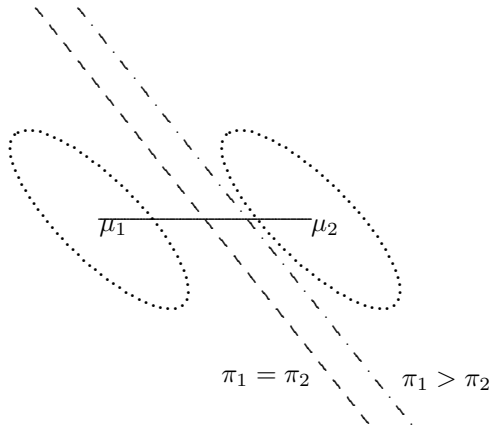

Рис. 3: Разделяющие плоскости дискриминанта Фишера (83) для двух классов и разных соотношений между вероятностями π_j

$p_j(x)$ или совместные плотности вероятностей $p(x,j)$, дискриминантами являются попарные разности $p_i(x) - p_j(x)$ или $p(x,i) - p(x,j)$. Для описываемого классификатора более удобными дискриминантами, причем линейными, являются функции

$$\begin{aligned}\ln\frac{p_i(x)}{p_j(x)} &= \ln\frac{\pi_i}{\pi_j} - \frac{1}{2}\Big((x-\mu_i)^\top A^{-1}(x-\mu_i) - (x-\mu_j)^\top A^{-1}(x-\mu_j)\Big)\\ &= \ln\frac{\pi_i}{\pi_j} - \frac{1}{2}\left(\mu_i{}^\top A^{-1}\mu_i - \mu_j{}^\top A^{-1}\mu_j\right) + (\mu_i-\mu_j)^\top A^{-1}(x)\\ &= \ln\frac{\pi_i}{\pi_j} + (\mu_i-\mu_j)^\top A^{-1}\left(x - \frac{\mu_i+\mu_j}{2}\right). \qquad (83)\end{aligned}$$

При $A = I_d$ и $\pi_i = \pi_j$ множество уровня 0 дискриминанта (83) — это гиперплоскость, проходящая через середину отрезка $[\mu_i, \mu_j]$ перпендикулярно ему. При $\pi_i \neq \pi_j$ гиперплоскость смещена относительно середины, а при $A \neq I_d$ она не перпендикулярна, а ее направление сопряжено направлению отрезка $[\mu_i, \mu_j]$ относительно квадратичной формы A^{-1} (см. рис. 3). Еще одно полезное геометрическое соображение: для классификации любого d-мерного вектора признаков достаточно знать его проекцию (при $A = I_d$ — ортогональную, в общем случае — вдоль сопряженной относительно A^{-1} плоскости) на не более чем $(q-1)$-мерную линейную оболочку точек μ_j.

Вариации на тему дискриминанта Фишера

Если распределения векторов признаков для разных классов считать гауссовыми, но не с общей матрицей ковариаций, а с разными, то модель тоже можно обучить методом наибольшего правдоподобия. При этом оценки (80) и (81) остаются неизменными, а оценка (82) естественно распадается на q независимых оценок

$$A_j^{kl} = \frac{\sum_{i|y_i=j}(x_i - \mu_{y_i})^k (x_i - \mu_{y_i})^l}{\#\{i|y_i = j\}}.$$

Но аналог дискриминанта (83) будет уже не линейным, а квадратичным, и так и называется *квадратичным дискриминантом Фишера* в отличие от обычного (линейного) дискриминанта Фишера.

Хотя модель с разными распределениями признаков для разных классов можно обучить точнее (с большим правдоподобием), чем модель с общей матрицей ковариаций, получающийся из нее квадратичный дискриминант может ошибаться на независимом тесте не меньше, а больше линейного дискриминанта. Это — типичный пример переобучения распознавателя при обучении без регуляризации. Размерность пространства моделей с общей матрицей ковариаций равна $(q-1)+qd+\frac{d(d+1)}{2}$, а размерность пространства моделей с разными матрицами равна $(q-1)+q\left(d+\frac{d(d+1)}{2}\right)$, т.е. почти в q раз больше[17]. Для того, чтобы ошибка на тесте для обученной модели из большого пространства была столь же близка к ошибке обучения, как и для модели из маленького пространства, модель из большого пространства нужно либо обучать на гораздо большем обучающем наборе, либо обучать с регуляризацией. С этой точки зрения принудительное приравнивание матриц ковариаций для разных классов можно рассматривать как регуляризацию, причем при малом обучающем наборе даже его может оказаться недостаточно.

Возможны и другие, более радикальные способы уменьшения размерности пространства моделей. Например, матрицы ковариаций (и различные, и общую) можно принудительно считать диагональными, скалярными или даже единичными. При этом если матрица ковариаций общая, дискриминанты тоже будут получаться линейными.

Вообще говоря, все эти модели можно обучать не только максимизацией правдоподобия, но и другими методами раздела 1.2.4, в которых в качестве регуляризации выступает априорное распределение на пространстве моделей.

Конструкция дискриминанта (83), оценки (80), (81), (82) и геометрическая интерпретация разделяющих гиперплоскостей идут от работы Р.Фишера 1936 года [36]. Линейный (и квадратичный) дискриминантный анализ был полезен в докомпьютерные времена из-за явных формул для ответа и наглядной геометрической интерпретации. Более общий, хотя вычислительно более сложный, метод из раздела 2.2.2 тоже дает линейные дискриминанты, не требует никаких сомнительных предположений о распределении векторов признаков каждого класса и на практике приводит к лучшему распознаванию.

2.2.2 Логистическая регрессия (*logistic regression*)

В этом разделе вместо обучения порождающей модели из раздела 2.2.1 максимизацией правдоподобия мы будем обучать дискриминантную модель в том же классе условных распределений (78) максимизацией апостериорной вероятности. Обучение более простой модели ($q(d+1)$ параметров[18] вместо $(q-1)+qd+\frac{d(}$ и сразу с регуляризацией (априорным распределением) дает надежды получить лучший классификатор, чем дискриминант Фишера. Пользуясь установленной в конце раздела 1.2.6 эквивалентностью максимизации апостериорной вероятности (43) для дискриминантной модели и минимизации ошибки обучения с регуляризацией (46), мы будем вести изложение в терминах регуляризации и минимизации ошибки.

В разделе 1.2.8 было показано, что дискриминантную модель для классификации — набор функций $f^y(x)$, оценивающих условные вероятности $P(y|x)$, — "правильно" обучать минимизируя взаимную энтропию (см. формулы (51), (52)). Но нужно еще проследить, чтобы прогноз $f(x)$ удовлетворял вероятностным нормировкам. Это обычно делают следующим образом. Неотрицательность прогнозов $f^j(x)$ обеспечивают представлением (пока — предваритель-

[17]Как уже отмечалось в разделе 1.1.6, на самом деле существенна не топологическая размерность, а размерность Вапника-Червоненкиса.

[18]На самом деле, как будет показано чуть ниже, достаточно $(q-1)(d+1)$ параметров

ным) их в виде экспонент от других функций, т.е. функций вида $e^{h^j(x)}$. При этом вместо неотрицательности получается строгая положительность, но во-первых, положительность необходима для вычисления взаимной энтропии, а во-вторых, с точки зрения практических вычислений $e^{-100} = 0$. А равенство единице суммы ответов обеспечивается принудительным нормированием:

$$f^j(x) = \frac{e^{h^j(x)}}{\sum_{k=1}^{q} e^{h^k(x)}}. \tag{84}$$

Поскольку отношение (84) не меняется при одновременном прибавлении одной и той же функции ко всем h^j, можно считать, например, что $h^q = 0$, или что $\sum_{j=1}^{q} h^j = 0$.

Дискриминантная модель (84), обучаемая минимизацией ошибки вида (51), называется *логистической регрессией* (*logistic regression*). Если же функции $h^j(x) = \bar{w}^j\bar{x}$ линейны, то она называется *линейной логистической регрессией*

$$f^j(x) = \frac{e^{\bar{w}^j\bar{x}}}{\sum_{k=1}^{q} e^{\bar{w}^k\bar{x}}}\ . \tag{85}$$

В дальнейшем будем считать, что $\bar{w}^q = 0$.

Несмотря на нелинейность функций (85), множества уровней их попарных отношений $\frac{f^j(x)}{f^i(x)}$ являются гиперплоскостями $(\bar{w}^j - \bar{w}^i)\bar{x} + b = 0$. Следовательно, множества вида $\{x | \forall i \in \{1, \ldots, q\}\ f^j(x) \geq f^i(x)\}$ при всех j являются выпуклыми многогранниками.

Обучение такой модели с, например, часто применяемым квадратичным по w (а не по $\bar{w}$) регуляризатором $\psi(\bar{w}) = \epsilon\|w\|_2^2$, сводится к решению задачи

$$L(\bar{w}, T, \epsilon) = \epsilon\|w\|_2^2 - \sum_{i=1}^{N}\left(\left(\sum_{j=1}^{q} y_i^j \bar{w}^j \bar{x}_i\right) - \ln\left(\sum_{j=1}^{q} e^{\bar{w}^j\bar{x}_i}\right)\right) \to \min_{\bar{w}^q=0} \tag{86}$$

(это просто подстановка формулы (85) в формулу (51) и учет соотношения $\sum_{j=1}^{q} y_i^j = 1$). В отличие от задачи линейной регрессии, явного решения задачи (86) с нелинейными производными по $\bar{w}$, аналогичного формуле (71), нет. Но производные по $\bar{w}$ выписываются явно, и можно проверить прямым вычислением, что матрица вторых производных (гессиан) положительно определена (**упражнение!**). Значит задача (86) строго выпукла и имеет единственное решение. Это решение можно искать методом Ньютона (*Ньютона-Рафсона*) второго порядка[19] или каким-либо методом градиентного спуска.

Вместо квадратичного регуляризирующего штрафа в функции (86) можно применять любой другой, например, линейный ([110], см. раздел 2.1.2), часто

[19]Напоминание о методах. Общеизвестный метод Ньютона — это итеративный метод поиска корня дифференцируемой функции одного переменного. Шаг итерации состоит в том, что в текущей точке — приближенном значении корня — функция заменяется своим многочленом Тейлора степени 1, т.е. линейной функцией, и в качестве следующего приближения берется корень этой линейной функции. Этот же метод дословно переносится на решение систем n уравнений от n переменных. Менее общеизвестный метод Ньютона второго порядка — это итеративный метод поиска критической точки дважды дифференцируемой функции (многих переменных). Шаг итерации состоит в том, что в текущей точке — приближенном значении критической — функция заменяется своим многочленом Тейлора степени 2, и в качестве следующего приближения к критической точке берется критическая точка этого квадратичного многочлена. Это в точности равносильно применению обычного метода Ньютона для поиска нулей градиента исходной функции. В окрестности критической точки метод Ньютона сходится быстро, а вдали от нее может и разойтись. Джозеф Рафсон придумал метод Ньютона позже Исаака Ньютона, но опубликовал более чем на 40 лет раньше.

дающий разреженные распознаватели. Типичные алгоритмы обучения линейной логистической регрессии с различными штрафами состоят в последовательном обучении линейных регрессий с квадратичными ошибками, аппроксимирующими логистическую ошибку в точке очередного решения, см., например, [76].

Выпишем для наглядности в простом случае двухклассовой классификации

$$q = 2, \quad \bar{w}^1 = \bar{w}, \quad \bar{w}^2 = 0, \quad y^1 = y, \quad y^2 = 1 - y,$$
$$f^1(x) = f(x) = \frac{e^{\bar{w}\bar{x}}}{1 + e^{\bar{w}\bar{x}}} = \frac{1}{1 + e^{-\bar{w}\bar{x}}}, \quad f^2(x) = \frac{1}{1 + e^{\bar{w}\bar{x}}} \tag{87}$$

минимизируемую функцию (86), условие ее минимума и ее гессиан:

$$L(\bar{w}, T, \epsilon) = \epsilon \|\bar{w} I^0_{d+1}\|_2^2 - \sum_{i=1}^{N} \left(y_i \bar{w} \bar{x}_i - \ln(1 + e^{\bar{w}\bar{x}_i}) \right) \tag{88}$$

$$0 = \frac{\partial L(\bar{w}, T, \epsilon)}{\partial \bar{w}} = 2\epsilon I^0_{d+1} \bar{w}^\top - \sum_{i=1}^{N} \left(y_i \bar{x}_i - \frac{e^{\bar{w}\bar{x}_i}}{1 + e^{\bar{w}\bar{x}_i}} \bar{x}_i \right) \tag{89}$$

$$\frac{\partial^2 L(\bar{w}, T, \epsilon)}{\partial \bar{w} \partial \bar{w}^\top} = 2\epsilon I^0_{d+1} + \sum_{i=1}^{N} \frac{e^{\bar{w}\bar{x}_i}}{(1 + e^{\bar{w}\bar{x}_i})^2} \bar{x}_i \bar{x}_i^\top . \tag{90}$$

Хорошо видно, что гессиан равен линейной комбинации с положительными коэффициентами неотрицательно определенных вырожденных матриц, а значит неотрицательно определен. Чтобы убедиться в том, что он положительно определен, достаточно заметить, что на собственном направлении матрицы I^0_{d+1} с нулевым собственным значением (т.е. на прямой $(t, 0, \ldots, 0)$) каждое из остальных N слагаемых положительно определено.

Упражнение. Для двухклассовой логистической регрессии (84) с пространством ответов $\mathcal{Y} = \{1, -1\}$ вместо $\mathcal{Y} = \{1, 2\}$ и $h^1 = h$, $h^{-1} = 0$ сама регрессия и формула (50) для ошибки выглядят очень компактно:

$$f^y(x) = \frac{1}{1 + e^{-yh(x)}}$$
$$E(f(x), y) = \ln(1 + e^{-yh(x)}) \, .$$

Немного об англостатистическом новоязе. Описанный в этом разделе метод обучения классификаторов называют logistic regression, функции (84) как функции от $h^j(x)$ называют *logistic functions*, логарифмы их попарных отношений — *log-odds* или *logit*, и каждый термин разными авторами используется в нескольких разных смыслах, не только в этих. Под *logistic function* чаще понимают функцию одного переменного $\sigma(t) = \frac{1}{1+e^{-t}}$, называемую также *sigmoid function*. Для двухклассовой классификации-таки $f(x) = \sigma(\bar{w}\bar{x})$ (формула (87)).

Историческая справка. Логистическая регрессия зарождалась в 1950-е годы в работах разных авторов, в нынешнем виде была сформулирована в середине 1960-х ([28]), а алгоритмы обучения продолжают совершенствоваться до сих пор, например, [76].

2.2.3 Перцептрон Розенблатта

Опишем еще один реликтовый, эпохи кибернетики, т.е. 1950-х годов, метод обучения линейных классификаторов, принадлежащий Ф.Розенблатту [97]. Этот

метод был даже реализован программно-аппаратно под названием "*перцептрон*", и его недостатки послужили толчком к развитию как нелинейных методов (в первую очередь, многослойных перцептронов), так и общей теории распознавания.

Рассматривается задача линейной двухклассовой классификации без каких-либо попыток оценить вероятности классов: пространство признаков $\mathcal{X} = \mathbb{R}^d$, пространство ответов $\mathcal{Y} = \{-1, +1\}$, пространство допустимых классификаторов $\mathcal{F} = \{\text{sign}(w, x) | w \in \mathbb{R}^d\}$. В начале раздела 2 описано, как можно повышением размерности пространства признаков свести к линейной классификации аффинную, но сейчас ради геометрической наглядности мы этого не делаем или считаем, что это уже заранее сделано. То, что функция sign может принимать не только два значения из пространства $\mathcal{Y}$, но еще и значение 0, всеми авторами традиционно игнорируется, и уточнение деталей оставляется читателям[20]. Целью обучения перцептрона является безошибочная классификация всех векторов признаков из обучающего набора T. То есть, нужно разделить два конечных набора точек $\{x_i | y_i = -1\}$ и $\{x_i | y_i = +1\}$ гиперплоскостью вида $\{x \in \mathcal{X} | (w, x) = 0\}$, или, что равносильно, найти такой вектор w, чтобы для любого обучающего вектора (x_i, y_i) выполнялось неравенство $y_i(w, x_i) > 0$. Конечно, это не всегда возможно.

Для разделимости гиперплоскостями общего вида $(w, x) + b = 0$ справедливо следующее

Предложение 2 *Обучающие векторы x_i разных классов не разделимы никакой гиперплоскостью $\Leftrightarrow$*
выпуклые оболочки классов пересекаются $\Leftrightarrow$
выпуклая оболочка векторов $y_i x_i$ содержит начало координат $\Rightarrow$
обучающие векторы линейно зависимы.

Доказательство оставляется в качестве **упражнения** по линейной алгебре. □

Если обучающие векторы можно разделить гиперплоскостью $(w, x) = 0$, то их можно разделить и полосой $\{x \in \mathcal{X} | \ |(w, x)| < \rho\}$, где

$$\rho = \min_{1 \le i \le N} y_i(w, x_i) > 0.$$

Ширина этой *разделяющей полосы* равна

$$2\delta = 2\frac{\rho}{\|w\|}.$$

Розенблатт предложил алгоритм обучения (см. алгоритм 1), который так и называется "перцептрон Розенблатта" и, вообще говоря, может зацикливаться.

[20] которыми оно тоже игнорируется...

Алгоритм 1: Перцептрон Розенблатта

1. Вход: обучающий набор $T = ((x_1, y_1), \ldots, (x_N, y_N))$ длины N.
2. Начальные вектор $w = 0$ классификатора $x \mapsto \mathrm{sign}(w, x)$, счетчик его коррекций $k = 0$ и количество ошибочно классифицированных обучающих векторов $n = N$.
3. Пока $n > 0$
   (a) забыть все ошибки: $n = 0$;
   (b) для каждого обучающего вектора (x_i, y_i) для которого $y_i(w, x_i) \leq 0$ (ошибка классификации)
      i. $n = n + 1$ (посчитать ее);
      ii. $w = w + y_i x_i$ (скорректировать классификатор);
      iii. $k = k + 1$ (посчитать коррекцию).
4. Выход: обученный вектор w и число коррекций k.

Теорема 2 (A.Novikoff ([89]) и многие другие) *Если обучающий набор можно правильно расклассифицировать линейной функцией, то алгоритм Розенблатта не зацикливается. Точнее, если обучающий набор можно правильно разделить полосой ширины 2δ и все обучающие векторы лежат в шаре радиуса $R = \max_{1 \leq i \leq N} \|x_i\|$, то алгоритм Розенблатта строит разделяющую функцию (w, x) за не более, чем $\left(\frac{R}{\delta}\right)^2$ коррекций w.*

При этом построенный алгоритмом вектор w может отличаться от направляющего вектора w' разделяющей полосы, гарантирующей сходимость алгоритма, и ширина построенной по w разделяющей полосы тоже может отличаться от 2δ.

Доказательство. Обозначим через $w^{(k)}$ значение вычисляемого алгоритмом Розенблатта вектора w после k шагов коррекции. $w^{(0)} = 0$, а вектор $w^{(k)}$ получается из вектора $w^{(k-1)}$ прибавлением какого-то вектора $y_i x_i$, скалярное произведение которого с w' не меньше δ, скалярное произведение с $w^{(k-1)}$ отрицательно, а длина не больше R. Значит с одной стороны

$$(w^{(k)}, w') = (w^{(k-1)}, w') + (y_i x_i, w') \geq (w^{(k-1)}, w') + \delta \|w'\|,$$

следовательно

$$(w^{(k)}, w') \geq k\delta \|w'\|$$

и с учетом неравенства Шварца $(w^{(k)}, w') \leq \|w^{(k)}\| \|w'\|$ получается оценка снизу

$$\|w^{(k)}\| \geq k\delta \ . \tag{91}$$

С другой стороны

$$\|w^{(k)}\|^2 = \|w^{(k)} + y_i x_i\|^2 \leq \|w^{(k-1)}\|^2 + \|y_i x_i\|^2 \leq \|w^{(k-1)}\|^2 + R^2, \tag{92}$$

следовательно

$$\|w^{(k)}\|^2 \leq kR^2. \tag{93}$$

Оценка снизу (91) совместна с оценкой сверху (93) только при

$$k \le \left(\frac{R}{\delta}\right)^2,$$

то есть, алгоритм обязан остановиться. Теорема доказана. $\square$

Модификации алгоритма Розенблатта, предохраняющие его от зацикливания при несуществовании разделяющей гиперплоскости, большой пользы науке не принесли. Наоборот,

- идея, что разделять нужно не гиперплоскостями, а полосами, причем чем шире полоса, тем лучше,
- наблюдение, что решение w принадлежит линейной оболочке обучающих векторов (конкретнее, представимо в виде линейной комбинации $w = \sum_{i=1}^{N} \alpha_i y_i x_i$ с неотрицательными коэффициентами α_i),
- само выражение $\left(\frac{R}{\delta}\right)^2$
- и независимость алгоритма, условий и утверждения теоремы от размерности пространства признаков

оказались очень важными.

В частности, если обучающих векторов достаточно много, а разделяющая полоса достаточно широка, так что выполнено неравенство $\left(\frac{R}{\delta}\right)^2 < N$, то не все обучающие векторы смогли поучаствовать в вычислении решения w. Это значит, что во-первых, w принадлежит линейной оболочке не более, чем $\left(\frac{R}{\delta}\right)^2$ обучающих векторов, а во-вторых, построенный по этим обучающим векторам классификатор правильно классифицирует остальные $N - \left(\frac{R}{\delta}\right)^2$ обучающих векторов, фактически не участвовавших в его обучении, то есть $1 - \frac{R^2}{N\delta^2}$-ую долю обучающего набора. Поскольку обучающие векторы считаются случайными и независимыми, это дает надежду на то, что и на почти любом независимом тестовом множестве частота ошибки классификации не будет сильно превышать $\frac{R^2}{N\delta^2}$. Подобные оценки, причем доказанные теоремы, а не безответственные спекуляции вроде предыдущей фразы, появились в работе Вапника и Червоненкиса [116].

Алгоритм Розенблатта можно интерпретировать как разновидность градиентного спуска в пространстве линейных классификаторов, т.н. *стохастический градиентный спуск* (известный также под названием *стохастическая аппроксимация* и подробнее обсуждающийся на стр. 83), пытающийся минимизировать сумму ошибок

$$E(r, y) = (-ry)_+ = \begin{cases} -ry & \text{при } ry < 0 \\ 0 & \text{при } ry \ge 0 \end{cases}$$

(рис. 5) по обучающему набору T:

$$\sum_{i=1}^{N} E_P((w, x_i), y_i) \quad = \quad \sum_{i=1}^{N} (-y_i(w, x_i))_+ \to \min_{w \in \mathcal{X}} .$$

Шаг стохастического градиентного спуска состоит в вычитании из вектора w градиента ошибки (вообще говоря, с небольшим положительным коэффициентом, регулирующим скорость спуска и зависящим от номера шага), в данном случае градиента ошибки $E_P((w, x_i), y_i)$. Такой шаг не обязательно уменьшает суммарную ошибку. Но при некоторых естественных дополнительных условиях

на скорость спуска стохастический градиентный спуск, применяемый последовательно к случайно распределенным обучающим векторам, с большой вероятностью минимизирует суммарную ошибку см. теорему 6 на стр. 83. Стохастический градиентный спуск активно применялся в 1950–1960-ые годы, когда оперативной памяти вычислительных машин не хватало на то, чтобы хранить весь обучающий набор. Он снова стал актуален в XXI веке для анализа данных из Интернета, объем которых растет заметно быстрее объемов оперативной памяти отдельных машин.

Один из недостатков алгоритма Розенблатта состоит в том, что хотя для его сходимости требуется существование разделяющей полосы ширины 2δ, его ответ никакой ширины разделяющей полосы не гарантирует. Этот недостаток легко исправить с помощью одной маленькой модификации: зафиксируем положительное число ϵ и будем проводить шаг коррекции не при условии $y_i(w, x_i) \le 0$, а при условии $y_i(w, x_i) \le \epsilon\|w\|$. Тогда если алгоритм сходится, то он определяет разделяющую полосу ширины больше 2ϵ.

Теорема 3 *Если обучающий набор можно правильно разделить полосой ширины 2δ, $R = \max_{1\le i\le N}\|x_i\|$, а $\epsilon < \delta$, то модифицированный алгоритм Розенблатта строит разделяющую полосу ширины больше 2ϵ за не более, чем $2\left(\frac{R}{\delta-\epsilon}\right)^2$ коррекций.*

Доказательство. В обозначениях доказательства теоремы 2 оценка снизу (91) остается неизменной, а вместо оценки (92) получается

$$\|w^{(k)}\|^2 = \|w^{(k)} + y_i x_i\|^2 \le \|w^{(k-1)}\|^2 + 2\epsilon\|w^{(k-1)}\| + R^2. \tag{94}$$

При этом $\|w^{(k)}\|$ растет при росте k намного быстрее, чем квадратный корень из k, как следовало из формулы (93), но все-таки медленнее, чем линейно с коэффициентом δ, как в формуле (91). А именно, для любого числа η, такого что $\epsilon < \eta < \delta$, из оценки (94) следует, что если $\|w^{(k-1)}\| \ge \frac{R^2}{2(\eta-\epsilon)}$ (на самом деле достаточно более слабого условия), то $\|w^{(k)}\| \le \|w^{(k-1)}\| + \eta$. Отсюда легко следует, что

$$\|w^{(k)}\| \le \frac{R^2}{2(\eta-\epsilon)} + k\eta. \tag{95}$$

Оценка сверху (95) совместна с оценкой снизу (91) только при

$$k \le \frac{R^2}{2(\eta-\epsilon)(\delta-\eta)},$$

т.е. модифицированный алгоритм сходится. Минимизируя правую часть последнего неравенства по η, получаем $\eta = \frac{\epsilon+\delta}{2}$ и

$$k \le \frac{2R^2}{(\delta-\epsilon)^2}.$$

□

Упражнение. Какую ошибку пытается минимизировать вышеописанная модификация алгоритма Розенблатта?

2.2.4 Классификаторы с разделяющей полосой (с зазором, *margin classifiers*)

В условиях раздела 2.2.3 можно не пытаться угадывать простой алгоритм, пригодный для обучения, а честно поставить и решить задачу нахождения разделяющей полосы максимальной ширины. Вернемся к обозначениям (61) и др. со

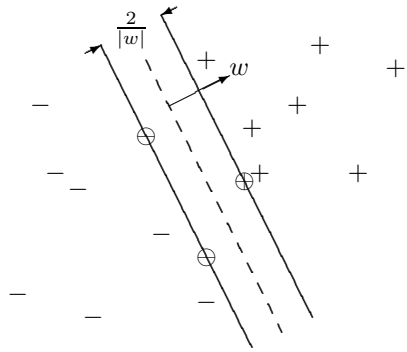

Рис. 4: Максимальная разделяющая полоса в задаче (97). Опорные векторы помечены кружками.

страницы 40, т.е. классификатор будет иметь вид

$$\operatorname{sign}((\bar{w}, \bar{x})) = \operatorname{sign}((w, x) + b) \ . \tag{96}$$

Для удобства вычислений умножим направляющий вектор $\bar{w}$ искомой разделяющей гиперплоскости (и полосы) на такой положительный коэффициент, чтобы для каждого обучающего вектора выполнялось не просто условие $y_i(\bar{w}, \bar{x}_i) > 0$, а условие $y_i(\bar{w}, \bar{x}_i) \geq 1$. Тогда ширина разделяющей полосы равна $\frac{2}{\|\bar{w}\|}$ и ее максимизация равносильна минимизации длины направляющего вектора $\bar{w}$ или, что еще удобнее, ее квадрата:

$$\begin{aligned} &\|\bar{w}\|^2 \to \min_{\bar{w}} \\ &y_i(\bar{w}, \bar{x}_i) \geq 1 \quad \text{при } 1 \leq i \leq N. \end{aligned} \tag{97}$$

Это задача квадратичной минимизации при линейных ограничениях, она выпукла, и, если ограничения совместны, имеет единственное решение. Алгоритмы приближенного решения таких задач сложнее алгоритма Розенблатта, но достаточно хорошо изучены. Решение задачи обладает замечательным свойством: оно выражается через обучающие векторы, лежащие на краю разделяющей полосы (*опорные векторы*, см. рис. 4), которых может оказаться заметно меньше, чем всех обучающих векторов.

Обычно при постановке и решении задачи о разделении полосой идут на некоторое усложнение вычислений ради того, чтобы полоса обитала в пространстве признаков, а не в пространстве на единицу большей размерности, и обучение было инвариантно относительно сдвига в пространстве признаков (ср. с различием между формулами (70) и (72)). Вместо задачи (97) решают очень похожую, но не эквивалентную, задачу

$$\begin{aligned} &\|w\|^2 \to \min_{w,b} \\ &y_i((w, x_i) + b) \geq 1 \quad \text{при } 1 \leq i \leq N. \end{aligned} \tag{98}$$

Для того, чтобы разделение полосой было возможно при любом обучающем наборе, вплоть до такого, в котором присутствуют одинаковые векторы признаков с разными ответами, нужно не полностью запретить ошибки классификации ($y_i((w, x_i)+b) < 0$) и неуверенность ($y_i((w, x_i)+b) < 1$), а лишь ограничить их количество или, что удобнее, добавить к минимизируемой функции штраф

за них. Традиционно задачу обучения формализуют следующим образом:

$$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N} E(\xi_i) \to \min_{w,b,\xi} \tag{99}$$
$$y_i((w,x_i)+b) \geq 1-\xi_i \quad \text{при } 1 \leq i \leq N$$
$$\xi_i \geq 0 \quad \text{при } 1 \leq i \leq N.$$

Здесь $E(\cdot)$ — неотрицательная монотонно неубывающая функция штрафа от неотрицательного аргумента, а $C > 0$ — вынесенный для удобства наружу коэффициент, управляющий соотношением штрафа за ошибки классификации и штрафа за сужение разделяющей полосы. Так обучаемые классификаторы называются *классификаторами с мягким зазором* (*soft margin classifier*[21]).

Фактически вектор признаков x_i классифицируется неправильно в точности при $\xi_i \geq 1$, т.е. правильным штрафом была бы функция

$$E(\xi) = \begin{cases} 0 & \text{при } \xi < 1 \\ 1 & \text{при } \xi \geq 1 \end{cases}$$

Но минимизировать разрывную функцию очень неудобно, поэтому используют какой-либо непрерывный и удобный для вычислений штраф, мажорирующий этот "правильный" штраф. Чаще всего берут кусочно-линейную функцию штрафа $E(\xi) = \max(\xi, 0)$, см., впрочем, рассуждение об ошибках в разделе 2.1.3.

Этот способ обучения классификаторов развивался в работах В.Н.Вапника с соавторами от [117] до [18, 26] и в окончательном виде стал известен как *метод опорных векторов* (*SVM*, *support vector machine*). Метод опорных векторов хорошо описан в статье [26] и пересказан в книге [31], статье [22] и тысячах других статей и десятках книг. Он, конечно, укладывается в общую схему минимизации ошибки с регуляризацией (формула (8)), но имеет и необычное теоретико-статистическое свойство.

Как и в случае построения безошибочно разделяющей полосы (задача (97)) оказывается, что полученная при обучении классификатора решающая функция $f(x) = (w,x) + b$ от многих обучающих векторов вообще не зависит, а зависит только от неправильно классифицированных обучающих векторов и от векторов, правильно или неправильно классифицированных, лежащих в полосе $|f(x)| \leq 1$. Как правило, термин "опорный вектор" распространяется на все такие векторы, теряя при этом свой исходный геометрический смысл. И оказывается, что, во-первых, даже при таком расширенном толковании опорных векторов, их получается достаточно мало (в экспериментах Вапника [113] 3–5% от числа обучающих векторов, в реальных приложениях — до 30–50%), а во-вторых, средняя ошибка на тесте не превышает долю опорных векторов среди всех обучающих.

В отличие от описанного в разделе 2.2.2 метода логистической регрессии, построенный методом опорных векторов классификатор не обучен предсказывать

[21] Из многих словарных значений слова "margin" здесь актуальны значения "край", "полоса" (в пространстве признаков) и "разность между доходами и затратами (прибыль, маржа)" (в пространстве значений линейной функции $(w,x)+b$). Гладко перевести "margin" на русский одним словом не получается. Мы используем термины "разделяющая полоса" в пространстве признаков и "зазор" между значением прогноза $((w,x)+b)$ и порогом принятия решения, т.е. нулем. В англоязычной литературе "margin" может означать саму разделяющую полосу ([50]), половину ширины разделяющей полосы, т.е. $\frac{1}{\|w\|}$ ([114],[15]), расстояние от вектора признаков до разделяющей гиперплоскости ([52]), значение прогноза "необрезанного" классификатора умноженное на правильный ответ, т.е. $y((w,x)+b)$ ([100],[52]), или его минимум по обучающему набору ([52])

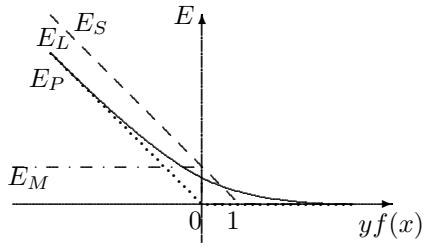

Рис. 5: Функции ошибок, применяемые при обучении двухклассовых классификаторов $\mathrm{sign}(f(x))$ с ответами $\{1,-1\}$: 0-1-ошибка классификации $E_M(t)=\theta(-t)$, $E_P(t)=(-t)_+$, неявно применяемая для перцептрона Розенблатта, взаимная энтропия $E_L(t)=\ln(1+e^{-t})$ для логистической регрессии, и $E_S(t)=(1-t)_+$ для классификаторов с зазором. $E_L(t)$ экспоненциально быстро стремится к 0 при $t\to\infty$ и к $-t$ при $t\to-\infty$. Очевидно, что $E_M(t)\le E_S(t)$ и $E_M(t)\le\frac{E_L(t)}{\ln 2}$.

вероятности классов. Иногда, по наивной аналогии с логистической регрессией, в качестве оценки вероятности принадлежности вектора x классу 1 берут

$$\sigma((w,x)+b)=\frac{1}{1+e^{-((w,x)+b)}}$$

(ср. с концом раздела 2.2.2). Сейчас мы покажем, что действительно эти два классификатора являются решениями очень похожих минимизационных задач (8). Но, конечно, нельзя свойства одного автоматически переносить на другой.

Деля на C и обозначая $\frac{1}{2C}$ через ϵ, минимизационную задачу обучения SVM (99) можно переписать в виде

$$\epsilon\|w\|^2+\sum_{i=1}^{N}E_S(y_i(w,x_i)+b))\to\min_{w,b}\quad, \tag{100}$$

где

$$E_S(t)=(1-t)_+=\begin{cases}1-t & \text{при } t\le 1\\ 0 & \text{при } t>1\end{cases}$$

С другой стороны, с учетом того, что в разделах 1.2.8 и 2.2.2 ответ y принадлежал множеству $\{0,1\}$, а сейчас принадлежит множеству $\{-1,1\}$, выражение (88), минимизируемое при обучении логистической регрессии, превращается в

$$\epsilon\|w\|^2+\sum_{i=1}^{N}E_L(y_i(w,x_i)+b))\to\min_{w,b} \tag{101}$$

с функцией ошибки

$$E_L(t)=\ln(1+e^{-t}).$$

(упражнение на стр. 52). Задачи (100) и (101) различаются только функциями ошибок, причем не очень сильно, см. рис. 5.

Метод опорных векторов является самым распространенным, но не единственным методом обучения классификаторов с разделяющей полосой. В разных методах различаются функции штрафа за ошибки, штрафа за разделяющую функцию (вместо $\|w\|^2$ можно брать $\|w\|_1$ и т.п., см. раздел 2.1.2) и способы построения пространства признаков $\mathcal{X}$, обсуждаемые далее в разделе 2.3.2.

2.3 Пространства признаков для линейных распознавателей

Напомним написанное в начале раздела 2. Для того, чтобы линейные распознаватели работали сколько-нибудь прилично, стандартных способов кодирования признаков (раздел 1.1.1) может оказаться недостаточно. Все-таки линейных функций в маломерных пространствах слишком мало. Невозможно хорошо разделить в $\mathbb{R}$ интервал $\{|x| < 1\}$ и объединение двух интервалов $\{2 < |x| < 3\}$ линейной функцией. Вот квадратичной — пожалуйста.

Линейные методы можно применять для распознавания непосредственно в пространстве исходных признаков когда его размерность достаточно велика по сравнению с количеством доступных обучающих векторов. В остальных случаях бывает полезно кроме (или вместо) исходных признаков использовать достаточно много нелинейно зависящих от них вторичных признаков. Тем самым, с помощью линейных методов строится нелинейный распознаватель. Приведем несколько примеров построения пространств вторичных признаков.

2.3.1 Базисные функции

Зафиксируем конечный набор функций $\phi^j : \mathcal{X} \to \mathbb{R}$, $1 \leq j \leq d'$, называемых *базисными*, или, что то же самое, отображение $\phi : \mathcal{X} \to \mathcal{X}' = \mathbb{R}^{d'}$ пространства исходных признаков $\mathcal{X}$ в пространство вторичных признаков $\mathcal{X}'$. Пусть $f : \mathcal{X}' \to \mathcal{Y}$ — обученный любым методом линейный распознаватель, т.е. либо сама функция f в случае регрессии, либо дискриминанты классов в случае классификации являются линейными. Для определенности будем рассматривать регрессию, тогда f имеет вид $f(x') = (w, x') + b$. Композиция $f \circ \phi : \mathcal{X} \to \mathcal{Y}$ является распознавателем на $\mathcal{X}$ и представима в виде

$$f \circ \phi(x) = (w, \phi(x)) + b = \sum_{j=1}^{d'} w^j \phi^j(x) + b, \tag{102}$$

т.е. линейной комбинации базисных функций и свободного члена (актуального, если среди базисных функций нет констант). В зависимости от того, какие функции выбирать за базисные, можно строить распознаватели разного вида.

Самый наглядный и весьма полезный пример набора базисных функций — мономы от координат в пространстве исходных признаков. Например, можно взять все мономы степени не больше n. Для d-мерного пространства исходных признаков получится $d' = \binom{d+n}{d}$ базисных функций, в том числе константы и координаты в пространстве исходных признаков. Распознаватели (регрессия или дискриминанты при классификации) получаются многочленами степени не более n. Не удивительно (теорема Вейерштрасса о равномерном приближении функций многочленами), что при больших n можно построить распознаватель со сколь угодно малой ошибкой обучения. Но это может оказаться бесполезным и даже вредным, поскольку распознаватель с малой ошибкой обучения может иметь большую среднюю ошибку.

Другой, тоже наглядный и полезный, набор базисных функций можно определить на любом метрическом пространстве исходных признаков $\mathcal{X}$ с метрикой ρ. Возьмем в пространстве $\mathcal{X}$ конечное множество точек-эталонов (центров) $\{c_1, \ldots, c_{d'}\}$ и какую-либо функцию $\chi : \mathbb{R}_+ \to \mathbb{R}_+$ и в качестве базисных функций возьмем функции $\phi^j(x) = \chi(\rho(x, c_j))$. Такие, зависящие только от расстояния, функции называются *радиальными базисными функциями* (*RBF*, *radial basis functions*). Функцию χ обычно берут быстро убывающей, так что в точках x, близких к какому-либо центру c_j и не очень близких к остальным центрам,

распознаватель (102) дает ответ, близкий к ответу в центре c_j. Центры (эталоны) можно фиксировать заранее (например, при распознавании рукописных букв это могут быть образцы написания, взятые из прописей), а можно строить обучающимися методами векторного квантования (разделы 1.3.3 и 1.3.4) или RBF-сетей (раздел 3.3).

И для разнообразия приведем совершенно другой пример базисных функций. Пространство исходных признаков — это пространство строк в конечном алфавите (например, текстов электронных писем), а базисные функции — количества вхождений в строку слов из фиксированного набора (словаря).

Размерность пространства базисных функций может быть как больше, так и меньше размерности пространства исходных признаков (если последнее имеет в каком-нибудь смысле размерность). Это несущественно. Существенно соотношение между размерностью пространства базисных функций и количеством обучающих векторов N. Неформальные и, вообще говоря, неверные соображения (формальная наука изложена в трудах В.Н.Вапника [114, 113, 115]) таковы. Если базисных функций меньше N, то при обучении линейного распознавателя заведомо использовалась не вся информация, доступная при обучении, и распознаватель получился хотя и довольно простой, но, возможно, не очень хороший. Если базисных функций больше N, то их ограничения на обучающий набор линейно зависимы, и линейный по этим базисным функциям распознаватель, обученный на этом наборе, эквивалентен (правда, только на этом же обучающем наборе) некоторому более простому распознавателю, зависящему только от N базисных функций. Из этих соображений вытекает следующая практическая рекомендация. Пространство базисных функций может быть очень большим, даже бесконечномерным (в приведенных примерах это будут пространства всех многочленов, или расстояний до всех точек пространства $\mathcal{X}$, или частоты вхождения всех конечных подстрок, соответственно), а для обучения линейного распознавателя нужно выбрать в нем конечномерное подпространство размерности порядка количества имеющихся обучающих векторов (все-таки с некоторым запасом...) и ограничиться только этим подпространством.

В разделе 2.3.2 описан способ построения таких подпространств удобной размерности.

2.3.2 Ядра[22] (kernels)

Зафиксируем на пространстве исходных признаков $\mathcal{X}$ функцию от двух переменных $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. Для обучающего набора $T = ((x_1, y_1), \ldots, (x_N, y_N))$ определим отображение $\phi_T : \mathcal{X} \to \mathbb{R}^N$ как $\phi_T(x) = (K(x_1, x), \ldots, K(x_N, x))$ и будем считать пространство-образ $\mathbb{R}^N$ пространством вторичных признаков $\mathcal{X}'(T)$ (оно зависит от обучающего набора T) и обучать на нем линейный распознаватель. Функции $\phi_T^j(\cdot) = K(x_j, \cdot)$ играют роль базисных функций. Каждый вектор признаков x переходит в вектор вторичных признаков

$$\phi_T(x) = (K(x_1, x), \ldots, K(x_N, x)). \tag{103}$$

Обученный линейный распознаватель будет иметь вид

$$(w, \phi_T(x)) + b = \sum_{j=1}^{d'} w^j K(x_j, x) + b \tag{104}$$

(ср. с формулой (102)), а коэффициенты w^j и свободный член b будут зависеть только от значений $K(x_i, x_j)$ и ответов y_i.

[22] Термин "ядро" в математике многозначен. Здесь он используется в специфическом смысле, не имеющем отношения к "ядру линейного оператора", но иногда превращающемся в "ядро интегрального оператора" или в "сверточное ядро".

Функция K называется *ядром* (*kernel*), а описанный в предыдущем абзаце способ построения пространства вторичных признаков самой замечательной размерности N (см. рассуждения о размерности пространства базисных функций в конце раздела 2.3.1) иногда называется *kernel trick*. Пространство исходных признаков $\mathcal{X}$ и функция K могут быть абсолютно любыми. Например, $\mathcal{X}$ может быть пространством строк, а ядро $K(x, y)$ — модулем разности их длин, деленным на длину первой строки, увеличенную на 3.14. Более разумное ядро $K(x, y)$ на строках (ср. с примером базисных функций на строках (стр. 61)) — количество словарных слов, входящих в обе строки x и y одновременно. Более общий пример ядра — это любая функция от скалярного произведения в евклидовом пространстве или от попарного расстояния между точками в метрическом.

Но настоящий *kernel trick* начинается тогда, когда в пространстве вторичных признаков вместо стандартного (евклидова) скалярного произведения

$$(\phi_T(x), \phi_T(x')) = \sum_{j=1}^{N} K(x_j, x) K(x_j, x')$$

рассматривают нестандартное и не зависящее от обучающего набора

$$(\phi(x), \phi(x')) = K(x, x'), \tag{105}$$

которое вычисляется намного быстрее (примерно в $2N$ раз). Для этого необходимо, чтобы функция K была похожа на скалярное произведение, а именно, симметрична ($K(x, x') = K(x', x)$) и неотрицательно определена (для любого конечного набора векторов $(x_1, \ldots, x_n)$ матрица $(K(x_i, x_j))$ неотрицательно определена). Эти необходимые условия являются и достаточными (теорема 9, стр. 93). Такие ядра называют *неотрицательно определенными ядрами* (*positive definite kernel*) или *ядрами Мерсера*.

Самый простой пример ядра Мерсера — это скалярное произведение в пространстве исходных признаков $\mathcal{X}$, если, конечно, это пространство евклидово. Для скалярного произведения отображение ϕ_T является линейным отображением из d-мерного пространства в N-мерное. Построенные при помощи этого ядра линейные в пространстве вторичных признаков распознаватели являются линейными и в пространстве исходных признаков, так что применение ядра, казалось бы, ничего не дает. Там не менее, при $N > d$ при некоторых методах обучения распознавателей такая линейная замена признаков улучшает качество обучения, а при $N < d$ ускоряет обучение без потери качества.

Более общие функции от скалярного произведения или от расстояния симметричны, но не обязательно неотрицательно определены. Впрочем, среди них есть полезные семейства неотрицательно определенных, например, многочлены с положительными коэффициентами от скалярного произведения или гауссово ядро $K(x, x') = e^{-\|x-x'\|^2}$.

Самый общий пример ядра Мерсера — это функция, индуцированная из скалярного произведения в каком-то евклидовом или гильбертовом пространстве H на любое пространство исходных признаков $\mathcal{X}$ любым отображением $\phi : \mathcal{X} \to H$. Других ядер Мерсера нет. То есть можно не по ядру Мерсера $K(x, x')$ и обучающему набору из N векторов признаков $x_1, \ldots, x_N$ строить зависящее от обучающего набора N-мерное пространство вторичных признаков и нестандартное скалярное произведение в нем (формула (105)), а наоборот, строить ядро по любому отображению из пространства исходных признаков в универсальное, не зависящее от обучающего набора пространство, по той же формуле $K(x, x') = (\phi(x), \phi(x'))$. Более подробно, а не на уровне произнесения общих слов и декларации результатов, ядра Мерсера описаны в разделе 4.1.

Некоторые методы обучения линейных распознавателей не используют никакой информации про векторы признаков, кроме их попарных скалярных произведений. Поэтому одной лишь заменой скалярного произведения на любое ядро Мерсера из них сразу получается метод распознавания в соответствующем пространстве вторичных признаков. Эта идея впервые была реализована в работе [1][23].

Приведем несколько примеров.

Классификаторы с разделяющей полосой и перцептрон Розенблатта изначально формулируются в терминах скалярного произведения, так что достаточно всюду вместо скалярного произведения векторов признаков (x, y) писать значение ядра $K(x, y)$, а вместо вектора w — его разложение по обучающим векторам (стр. 55), традиционно записываемое в виде $w = \sum_{i=1}^{N} \alpha_i y_i x_i$. Подробнее об этом написано в разделе 4.2.

Для гребневой регрессии с квадратичной ошибкой и квадратичным штрафом (70) явно написанный направляющий вектор распознавателя

$$\bar{w} = \mathbf{y}\mathbf{X}^\top \big(\epsilon I_{d+1} + \mathbf{X}\mathbf{X}^\top\big)^{-1}$$

(формула (71)) и сама регрессия

$$f(\bar{x}) = \bar{w}\bar{x} = \mathbf{y}\mathbf{X}^\top \big(\epsilon I_{d+1} + \mathbf{X}\mathbf{X}^\top\big)^{-1}\bar{x}, \tag{106}$$

казалось бы, не содержат скалярных произведений, но их можно выразить через скалярные произведения. Действительно,

$$\mathbf{X}^\top \big(\epsilon I_{d+1} + \mathbf{X}\mathbf{X}^\top\big) = \epsilon\mathbf{X}^\top + \mathbf{X}^\top\mathbf{X}\mathbf{X}^\top = \big(\epsilon I_N + \mathbf{X}^\top\mathbf{X}\big)\mathbf{X}^\top$$

(тождество), значит

$$\big(\epsilon I_N + \mathbf{X}^\top\mathbf{X}\big)^{-1}\mathbf{X}^\top = \mathbf{X}^\top \big(\epsilon I_{d+1} + \mathbf{X}\mathbf{X}^\top\big)^{-1}$$

(тоже тождество, обратимость обращаемых матриц следует из их положительной определенности), а значит

$$\bar{w} = \mathbf{y}\big(\epsilon I_N + \mathbf{X}^\top\mathbf{X}\big)^{-1}\mathbf{X}^\top$$

и распознающая функция (регрессия) представима в виде

$$f(\bar{x}) = \bar{w}\bar{x} = \mathbf{y}\big(\epsilon I_N + \big(\mathbf{X}^\top\mathbf{X}\big)\big)^{-1}\big(\mathbf{X}^\top\bar{x}\big). \tag{107}$$

В отличие от матрицы корреляций $\mathbf{X}\mathbf{X}^\top$, $\mathbf{X}^\top\mathbf{X}$ — это матрица попарных скалярных произведений обучающих векторов (матрица Грама), а $\mathbf{X}^\top\bar{x}$ — вектор скалярных произведений тестового вектора $\bar{x}$ со всеми обучающими векторами. При замене скалярного произведения на ядро Мерсера распознаватель (107) превращается в

$$f(\bar{x}) = \mathbf{y}\Big(\epsilon I_N + \big(K(\bar{x}_i, \bar{x}_j)\big)_{i,j=1}^{N}\Big)^{-1}\big(K(\bar{x}_i, \bar{x})\big)_{i=1}^{N}. \tag{108}$$

Напоминаем, что если просто подставить в формулы (106) или (107) векторы вторичных признаков (103) вместо векторов исходных признаков, получится распознаватель, не совпадающий с распознавателем (108), требующий больше вычислений, но зато применимый для любого ядра, а не только для ядра Мерсера.

Упражнение. Переформулируйте распознавание методом k ближайших соседей и кластеризацию методом k средних в терминах скалярных произведений и замените скалярные произведения ядрами Мерсера.

[23] Ее перевод [3] гордо занимает первое по алфавиту место в списках литературы большинства статей про распознавание с применением ядер. Интересно, что русскоязычные авторы обычно ссылаются не на оригинальную статью, а на книгу [2].

2.3.3 Слабые распознаватели

Специальным случаем базисных функций являются слабые (т.е. неточные, хотя, возможно, простые и быстрые) распознаватели, не обязательно линейные. Предположим, что их достаточно много (d'), они не очень хороши (т.е. дают слишком большую ошибку на тестах), но все-таки работают лучше случайных (в случае двухклассовой классификации это означает, что вероятность ошибки каждого классификатора меньше $\frac{1}{2}$). Если эти распознаватели обучались независимо, то есть надежда, что можно устроить голосование между ними в случае классификации или их усреднение в случае регрессии (т.е. в обоих случаях вычислить среднее арифметическое их прогнозов), так что их совокупное мнение будет иметь меньшую ошибку, чем каждый из них. Теоретически эта надежда абсолютно не обоснована (наоборот, есть контрпримеры и теоремы о том, что в среднем никакого выигрыша в качестве распознавания не будет), но на практике иногда распознавание улучшается.

Вместо простого вычисления среднего арифметического можно обучать линейный распознаватель, который использует в качестве признаков прогнозы этих слабых распознавателей, т.е. реализует голосование с весами, не обязательно одинаковыми. Для такого обучения кроме общих методов есть и специфические методы совместного обучения слабых распознавателей и их весов в голосовании, называемые *бустингом* (*boosting — усиление*), см. раздел 5.

Обучение весов, например, в задаче двухклассовой классификации, можно проводить аналогично обучению классификатора с разделяющей полосой, в котором все компоненты w^j вектора w предполагаются неотрицательными (ведь каждый слабый распознаватель все-таки лучше случайного), свободный член b отсутствует (голосуют только классификаторы), а в качестве регуляризирующего штрафа берется не евклидова, а, как в методе лассо, l^1-норма вектора w, равная в этом случае просто сумме $\sum_{j=1}^{d'} w^j$. При этом квадратичная минимизационная задача (99) превращается в линейную:

$$\begin{aligned}
\sum_{j=1}^{d'} w^j + C\sum_{i=1}^{N}\xi_i &\to \min_{w,\xi} \\
y_i(w, x'_i) &\geq 1 - \xi_i \quad \text{при } 1 \leq i \leq N \\
w^j &\geq 0 \quad \text{при } 1 \leq j \leq d' \\
\xi_i &\geq 0 \quad \text{при } 1 \leq i \leq N,
\end{aligned}$$

где x'_i — это не сам вектор признаков i-го обучающего объекта, а d'-мерный вектор ответов классификаторов на нем, а $y_i \in \{-1, 1\}$ — как обычно, класс i-го объекта. Подробнее обучение голосованию рассмотрено в разделе 5.3. Обучение весов, например, в задаче двухклассовой классификации, можно проводить аналогично обучению классификатора с разделяющей полосой, в котором все компоненты w^j вектора w предполагаются неотрицательными (ведь каждый слабый классификатор все-таки лучше случайного), свободный член b отсутствует (голосуют только классификаторы), а в качестве регуляризирующего штрафа берется не евклидова, а, как в методе лассо, l^1-норма вектора w, равная в этом случае просто сумме $\sum_{j=1}^{d'} w^j$. При этом квадратичная минимизационная задача (99) превращается в линейную:

$$\begin{aligned}
\sum_{j=1}^{d'} w^j + C\sum_{i=1}^{N}\xi_i &\to \min_{w,\xi} \\
y_i(w, x'_i) &\geq 1 - \xi_i \quad \text{при } 1 \leq i \leq N \\
w^j &\geq 0 \quad \text{при } 1 \leq j \leq d'
\end{aligned}$$

$$\xi_i \geq 0 \quad \text{при } 1 \leq i \leq N,$$

где x'_i — это не сам вектор признаков i-го обучающего объекта, а d'-мерный вектор ответов классификаторов на нем, а $y_i \in \{-1, 1\}$ — как обычно, класс i-го объекта. Подробнее обучение голосованию рассмотрено в разделе 5.3.

Возможен и другой способ улучшения качества слабых распознавателей: использование и всех исходных признаков, и прогнозов слабых распознавателей в качестве вторичных признаков для дальнейшего обучения. При таком подходе слабые распознаватели просто играют роль базисных функций, а то, что они лучше случайных, не учитывается.

3 Нейронные сети

В предыдущих разделах рассматривались простейшие нелинейные методы распознавания (метод ближайших соседей, распознающие деревья) и методы распознавания с линейной обучающейся частью (все методы из раздела 2), возможно, в сочетании с необучаемым нелинейным преобразованием признаков (ядра, базисные функции) и ответов (логистическая регрессия). Ни в том, ни в другом случае не было априорной уверенности в том, что такими методами можно за приемлемое время научиться распознавать что угодно. В этом разделе будут описаны более сложные нелинейные методы распознавания, про которые была уверенность, что их можно распознавать что угодно, поскольку они успешно работают в природе уже миллионы лет. Но обучать их оказалось не так просто. . . .

Речь идет о нейронных сетях — упрощенных математических моделях нервной системы живых существ, в частности, головного мозга человека. Его-то можно обучить чему угодно!

Из всех многочисленных конструкций нейронных сетей рассматриваются только сети прямого распространения сигнала, и то лишь некоторые. Зато описываемые, хотя и очень кратко, методы их обучения, за исключением метода обратного распространения ошибки, не слишком специфичны для нейронных сетей и применимы при обучении самых разных нелинейных распознающих систем.

3.1 Естественные и искусственные нейронные сети

Теория нейронных сетей возникла в 1940–1960-х годах в результате совместных попыток физиологов и кибернетиков (так тогда назывались специалисты по computer science) понять и смоделировать работу мозга. Получилась следующая модель.

Устройство нейронных сетей

Мозг состоит из очень большого числа (порядка 10^{11}) клеток (нейронов), каждая из которых имеет длинный хвост (аксон) и большое число (порядка 10^4) ответвлений (дендритов), касающихся аксонов других нейронов и/или входных рецепторов. Через эти зоны касания (синапсы) может передаваться информация (электрохимический потенциал, что бы это ни значило). Каждый нейрон является простеньким компьютером: потенциал нейрона (и его аксона, играющего роль выхода) является функцией от потенциалов синапсов его дендритов (входов), причем функцией вполне определенного вида. А именно, каждый нейрон имеет два устойчивых состояния (возбужденное и невозбужденное) и соответствующие им значения потенциала, одинаковые для всех нейронов. Каждый

нейрон вычисляет линейную комбинацию потенциалов входных синапсов, сравнивает ее с пороговым значением и переходит в возбужденное (невозбужденное) состояние если эта линейная комбинация больше (соответственно, меньше) порогового значения. Такая модель нейрона впервые была описана в работе [82] в 1943 году[24]. В совокупности мозг вычисляет некоторую вектор-функцию: зависимость потенциалов нейронов (достаточно рассматривать не все нейроны, а только связанные своими аксонами с исполнителями) от потенциалов входных рецепторов. А вся нетривиальность работы мозга состоит в том, что пороговые значения (по одному на нейрон, итого порядка 10^{11}) и коэффициенты линейных комбинаций (по одному на дендрит, итого порядка 10^{15}), вообще говоря, различны и могут изменяться со временем. Это изменение коэффициентов называется обучением.

За последующие полвека модель мозга сильно эволюционировала, но в качестве мотивации искусственных нейронных сетей использовалась именно такая упрощенная устаревшая модель.

В *искусственных нейронный сетях* (*ANN, Artificial Neural Networks*), применяемых для вычислений, в частности, для распознавания, обычно вводятся некоторые "нефизиологические" ограничения и снимаются некоторые "физиологические". Мы ограничимся рассмотрением сетей прямого распространения сигнала.

Нейронная сеть прямого распространения — это связный ориентированный ациклический граф с множеством вершин $V = \{v_1, \ldots, v_S\}$ (*рецепторов* и *нейронов*) и ребер (*синапсов*) $E = \{e_1, \ldots, e_T\}$, снабженный следующей дополнительной структурой:

- множества вершин-истоков (рецепторов) V_{in} и вершин-стоков (*выходных нейронов*) V_{out} нейронной сети (непустые, как у всякого связного ориентированного ациклического графа) упорядочены;
- каждому ребру e приписан параметр $w_e \in \mathbb{R}$ и функция $h_e : \mathbb{R}^2 \to \mathbb{R}$, называемая *функцией проводимости* ребра (синапса);
- каждой вершине $v \in V \setminus V_{\text{in}}$ приписан параметр $w_v \in \mathbb{R}$ и функция $g_v : \mathbb{R}^2 \to \mathbb{R}$, называемая *функцией активации* вершины (нейрона).

Пусть упорядоченные множества $V_{\text{in}} = (v^1_{\text{in}}, \ldots, v^d_{\text{in}})$ и $V_{\text{out}} = (v^1_{\text{out}}, \ldots, v^q_{\text{out}})$ содержат d и q вершин, соответственно. Нейронная сеть вычисляет вектор-функции $f : \mathbb{R}^d \to \mathbb{R}^q$, зависящие также от параметров всех вершин и ребер сети. Для вычисления значения $y = f(x)$ на вершинах и ребрах сети определяется вспомогательная вещественнозначная функция u (*потенциал*), зависящая от вектор-аргумента x и параметров сети w, причем потенциал $u(e)$ каждого ребра e равен потенциалу $u(v)$ вершины v, из которой это ребро выходит ($e \in v_{\text{out}}$). Потенциал определяется индуктивно:

- потенциал j-го истока v^j_{in} равен j-му аргументу x^j, т.е. $u(v^j_{\text{in}}) = x^j$;
- для каждой вершины v, для всех входящих ребер которой входные потенциалы уже определены, вычисляется *активирующая сумма*

$$s(v) = \sum_{e \in v_{\text{in}}} h_e(w_e, u(e)) \tag{109}$$

и потенциал

$$u(v) = g_v\left(w_v, s(v)\right) \ . \tag{110}$$

[24]Этой модели нейрона в точности соответствует перцептрон Розенблатта, раздел 2.2.3.

Связность и ацикличность сети гарантируют корректное вычисление потенциала каждой вершины. Результатом вычисления является вектор-потенциал $y = (y^1, \dots, y^q) = (u(v^1_{\text{out}}), \dots, u(v^q_{\text{out}}))$ вершин-стоков.

Вектор w, составленный из всех чисел w_e и w_v, является параметром сети. Обучение нейронной сети заключается в подборе такого значения w, чтобы выходной вектор y, являющийся функцией $y = f(x) = F(w, x)$ от параметра и входного вектора x, был похож на то, чему сеть пытаются научить. Наоборот, функции h_e и g_v и сам ориентированный граф остаются неизменными.

Для простоты и удобства реализации нейронной сети обычно полагают ее состоящей из нейронов нескольких (немногих) разновидностей, регулярным образом соединенных друг с другом. Чаще всего множество V всех нейронов сети предполагаются разбитым на пронумерованные *слои* $V_0 = V_{\text{in}}, V_1, \dots, V_L = V_{\text{out}}$, так что

- каждое ребро, входящее в нейрон некоторого i-го слоя, выходит из некоторого нейрона $(i-1)$-го слоя;
- для каждого слоя функции активации g_v всех нейронов этого слоя совпадают; совпадают и функции проводимости h_e всех входящих в них ребер.

Такая нейронная сеть называется L-слойной. Послойная градуировка автоматически обеспечивает ацикличность сетей. В практически применяемых нейронных сетях послойная структура может быть более сложной и не вполне регулярной.

Наиболее типичными для многослойных сетей являются следующие функции проводимости ребер и активации нейронов:

- $h_e(w_e, u) = w_e u$, а $g_v(w_v, s) = \text{th}(s - w_v)$ (гиперболический тангенс разности), $g_v(w_v, s) = \sigma(s - w_v)$ (логистическая функция от разности), или какое-нибудь другое гладкое монотонно возрастающее приближение ступенчатой функции; см. раздел 3.2;
- $h_e(w_e, u) = (u - w_e)^2$, а $g_v(w_v, s) = e^{-s/2w_v}$ или какое-нибудь другое гладкое приближение дельта-функции; см. раздел 3.3.

Функция активации *выходных нейронов* (нейронов последнего слоя) обычно определяется задачей, для решения которой нейронная сеть предназначена. Для классификации применяются нейронные сети либо (очень редко) с пороговой функцией активации выходных нейронов $g_v(w_v, s) = \theta(s - w_v)$, либо (чаще всего) с логистической функцией активации $g_v(w_v, s) = \sigma(s - w_v)$ (ср. с логистической регрессией (87) для двух классов); логистическая функция активации применяется и для оценки вероятности чего-либо. Для регрессии применяются сети с линейной функцией активации выходных нейронов: $g_v(w_v, s) = s - w_v$.

Искусственные нейронные сети могут применяться для вычисления функций как от дискретных, так и от непрерывных признаков. В последнем случае для устойчивости сети к ошибкам измерения входных векторов необходима непрерывность функций h_e и g_v, а для обучения сети градиентными методами — еще и их дифференцируемость. В дальнейшем функции h_e и g_v предполагаются гладкими.

Количество L слоев в практически применяемых нейронных сетях на удивление мало: чаще всего встречаются трехслойные сети, иногда даже двухслойные. И наоборот, в одном и том же распознавании иногда участвуют несколько различных сетей, как-то соединенных друг с другом, но обучаемых раздельно. Напоминаем, что сеть может и не иметь никакой регулярной многослойной структуры.

Количество d входов искусственной сети и количество q ее выходных нейронов определяются решаемой задачей и обычно не очень велики. d равно размерности пространства признаков (обычно десятки или сотни), а q меняется от 1 (регрессия) до сотен (многоклассовая классификация). Наоборот, число остальных нейронов, называемых *скрытыми*, бывает порядка тысяч и десятков тысяч, а число ребер доходит до миллионов, хотя это несравнимо меньше, чем в естественной нейронной сети. В искусственных нейронных сетях, как и в мозгу, все вычисления могут происходить параллельно, и тем самым, очень быстро. В реальности нейронные сети моделируются на обычных последовательных компьютерах и работают не слишком быстро, а главное, медленно обучаются, поэтому на количестве вершин и ребер сети приходится экономить. Появляющиеся время от времени аппаратные реализации нейронных сетей (нейрокомпьютеры) позволяют несколько увеличить размеры пригодных для обучения сетей.

Обучение нейронных сетей: анонс

В отличие от устройства нейронных сетей, способ обучения естественных нейронных сетей, пригодный для искусственных, "подсмотреть у природы" не удалось. При обучении искусственных нейронных сетей распознаванию обычно применяются методы минимизации эмпирического риска (раздел 1.1.4), чаще с регуляризацией (раздел 1.1.6), или, если сеть обучают предсказывать вероятности, максимизации апостериорной вероятности (раздел 1.2.4).

Создание нейронной сети для решения какой-либо задачи состоит из двух этапов: построение сети, т.е. ациклического ориентированного графа и функций проводимости и активации, и подбор ее параметров w. Первый этап — очень эвристический и существенно зависящий от конкретной решаемой задачи и объема доступных обучающих данных, и как правило требует многочисленных экспериментов. Для второго этапа — обучения — применяются довольно универсальные методы.

Существенное отличие обучения нейронных сетей от обучения большинства рассмотренных ранее распознавателей, особенно линейных, состоит в том, обучаемых параметров очень много (например, миллион), и оптимальная сеть далеко не единственна, т.е. приходится искать какой-либо из очень большого числа локальных минимумов некоторой функции в очень многомерном пространстве.

В следующих разделах рассматриваются два наиболее распространенных типа многослойных нейронных сетей (MLP и RBF) и на их примере несколько групп методов обучения, применимых не только к нейронным сетям: градиентные, стохастические и максимизация ожидания. Теория нейронных сетей такими сетями не ограничивается, но прочие типы сетей применяются для распознавания исчезающе редко.

3.2 Многослойные перцептроны (MLP)

Слово перцептрон ("восприниматель") появилось в работах Розенблатта 1950-х годов [97] и относилось к распознающей системе, обучающаяся часть которой состояла из одного нейрона с линейной функцией проводимости и ступенчатой функцией активации. Теперь *многослойным перцептроном* (*MLP (Multi-Layer Perceptron)*) называют любую многослойную нейронную сеть с линейной функцией проводимости и общей для всех скрытых нейронов монотонной ограниченной функцией активации g_v, зависящей только от разности $t = s(v) - w_v$, являющейся "сглаженной ступенькой" — как правило, гиперболическим тангенсом $\text{th}(t) = \frac{e^t - e^{-t}}{e^t + e^{-t}}$ или логистической функцией $\sigma(t) = \frac{\text{th}(t/2)+1}{2} = \frac{1}{1+e^{-t}}$.

Функция активации выходных нейронов может тоже быть той же "сглаженной ступенькой", а может быть и тождественной: $g_v(t) = t$. То есть каждый нейрон v вычисляет функцию

$$u(v) = g_v\left(\left(\sum_{e \in v_{\text{in}}} w_e u(e)\right) - w_v\right). \tag{111}$$

Параметры ребер w_e называют их весами, а параметры вершин w_v — смещениями.

Какова функция активации нейронов промежуточных слоев — гиперболический тангенс th или логистическая σ — безразлично вот в каком смысле: для любого многослойного перцептрона с функцией активации th, вычисляющего функцию $F_{\text{th}}(w, \cdot)$, такой же перцептрон, в котором функция активации в промежуточных слоях заменена на σ, вычисляет ту же самую функцию при некотором другом значении параметра w', $F_\sigma(w', \cdot) = F_{\text{th}}(w, \cdot)$, да и к тому же потенциалы всех нейронов промежуточных слоев преобразуются одной и той же линейной функцией $u_\sigma(v) = \frac{u_{\text{th}}(v)+1}{2}$.

Упражнение. Выпишите отображение $w \mapsto w'$ и проверьте, что оно обратимо.

На практике чаще применяется логистическая функция активации, но теоретически удобнее рассматривать гиперболический тангенс, поскольку, если еще и функция активации выходных нейронов является гиперболическим тангенсом или тождественной, то $F_{\text{th}}(0, \cdot) = 0$ и при малых w и x вычисляемая перцептроном функция $f(x) = F_{\text{th}}(w, x)$ близка к линейной.

Для упрощения формализма часто добавляют к нейронной сети один дополнительный вход с потенциалом -1 и соединяют каждый нейрон v с этим входом ребром с приписанным весом w_v. Тогда смещение $-w_v$ попадает в активирующую сумму, и вычисляемая нейроном v функция вместо (111) равна

$$u(v) = g_v\left(\sum_{e \in v_{\text{in}}} w_e u(e)\right). \tag{112}$$

3.2.1 Вычислительные возможности перцептрона и теорема Колмогорова

Следующие предложения, настоятельно рекомендуемые в качестве простых **упражнений**, показывают, что с помощью многослойных перцептронов можно вычислить почти что угодно.

Предложение 3 *(однослойный перцептрон)*

Индикаторную функцию любого полупространства в пространстве признаков (если не обращать внимания на значение функции на границе этого полупространства) можно вычислить однослойным перцептроном с одним нейроном с разрывной ступенчатой функцией активации $g(t) = \theta(t)$.

Индикаторную функцию любого полупространства в пространстве признаков можно приблизить в любом разумном слабом смысле (по мере, в пространстве L^2_{loc} функций, квадрат которых локально интегрируем, и т.п.) однослойным перцептроном с одним нейроном с непрерывной функцией активации $g(t) = \sigma(t)$.

Те же оговорки в скобках подразумеваются и в последующих предложениях.

Предложение 4 *(двухслойный перцептрон)*

Индикаторную функцию любого выпуклого многогранника в пространстве признаков можно вычислить двухслойным перцептроном с ступенчатой функцией активации.

Индикаторную функцию любого выпуклого многогранника в пространстве признаков можно приблизить двухслойным перцептроном с непрерывной функцией активации.

В обоих случаях для вычисления или приближения индикаторной функции параллелепипеда достаточно $2d$ скрытых нейронов и одного выходного.

Предложение 5 *(трехслойный перцептрон)*

Любую конечную линейную комбинацию индикаторных функций параллелепипедов в пространстве признаков можно вычислить трехслойным перцептроном со ступенчатой функцией активации.

Любую непрерывную функцию на любом компакте в пространстве признаков можно равномерно приблизить трехслойным перцептроном с непрерывной функцией активации.

То есть трехслойных перцептронов достаточно для сколь угодно точного решения задач аппроксимации функций, и в частности, задач распознавания. Но чем точнее требуется аппроксимация, тем больше нейронов должно быть в сети.

Для сравнения напомним замечательную теорему, казалось бы, не имеющую никакого отношения к нейронным сетям.

Теорема 4 *(А.Н.Колмогоров, 1957, [69]). Любую непрерывную функцию на d-мерном кубе можно представить в виде суперпозиции непрерывных функций одной переменной и сложения.*

На самом деле теорема Колмогорова утверждает, что любая непрерывная функция на d-мерном кубе вычисляется некоторой нейронной сетью с тождественной функцией проводимости ребер (ср. формулировку теоремы с формулой (111)). Из доказательства теоремы следует явное построение такой сети: она трехслойная, с одним выходным нейроном, $(d+1)(2d+1)$ скрытыми, и с монотонными функциями активации. Но эта сеть не является многослойным перцептроном: в ней функции активации разных нейронов различны, зависят от вычисляемой функции и обязательно недифференцируемы.

Предложение 5 можно усилить:

Предложение 6 *Любую непрерывную функцию на любом компакте в пространстве признаков можно равномерно приблизить двухслойным перцептроном с непрерывной функцией активации.*

Доказательство этого предложения является более сложным **упражнением**, чем предыдущие. Подсказка: предложение можно разбить на два утверждения, общеизвестное и простое. Общеизвестное утверждение: любую непрерывную функцию на компакте в евклидовом пространстве можно приблизить тригонометрическим многочленом вида

$$T(x) = \sum_k c_k e^{i \sum_{j=1}^d c_{kj} x^j} \ .$$

Простое утверждение: любую функцию одного переменного, например, синус или косинус, на любом компакте можно приблизить линейной комбинацией константы и (сглаженных) ступенек — функций активации. И всю двойную

сумму — по тригонометрическим мономам и по аппроксимациям мономов сглаженными ступеньками — может вычислить один-единственный выходной нейрон.

Получается, что теоретически третий слой в предложении 5 — лишний и можно всегда обойтись двухслойным перцептроном. На практике же часто бывает легче обучить многослойную сеть с небольшим количеством нейронов, чем двухслойную с очень большим.

Замечание. Возможность сколь угодно точно приблизить любую функцию с помощью многослойных перцептронов очень вдохновляла исследователей в 1950–1960-ые годы. Но быстро выяснилось, что и из теоретических соображений (угроза переобучения, см. раздел 1.1.6) и из практических (ресурсоемкость) приходится обучать нейронные сети заранее ограниченного размера, тем самым ограничивая класс вычислимых функций.

3.2.2 Конструирование перцептронов (пример)

Типичный распознающий перцептрон — трехслойный (в соответствии с предложением 5). Нейроны первого скрытого слоя разбиты на несколько групп: нейроны внутри каждой группы одинаковы и соединены не со всеми входами сети, а только с каким-то их подмножеством, своим для каждой группы. Это разбиение на группы производится вручную исходя из специфики задачи. Например, при распознавании растровых изображений каждый нейрон первого слоя может иметь свое небольшое "поле зрения", т.е. быть соединенным с рецепторами, измеряющими яркость изображения в некотором количестве (нескольких десятках) смежных точек. Поля зрения разных нейронов одной группы сдвинуты друг относительно друга (обычно поля зрения "соседних" нейронов перекрываются), а нейроны разных групп имеют разные по размеру и по форме поля зрения. Нейроны второго скрытого слоя более-менее одинаковы, но каждый нейрон соединен со случайным не слишком большим подмножеством нейронов первого слоя. Каждый нейрон третьего слоя (выходного) соединен со всеми нейронами второго слоя.

Несколько более сложный пример реальной нейронной сети, использовавшейся для распознавания рукописного ввода в одном из первых наладонных компьютеров 1990-х годов, приводится в статье [123, стр. 4].

3.2.3 Обучение многослойного перцептрона: метод обратного распространения ошибки (error *back-propagation*)

В соответствии с идеологией минимизации эмпирического риска с регуляризацией (раздел 1.1.6, формула (8)) обучение перцептрона, вычисляющего функцию $F(w, x)$ — это поиск вектора весов и смещений w, минимизирующего регуляризованную суммарную ошибку

$$E_T(w) = \psi(w) + \sum_{i=1}^{N} E(F(w, x_i), y_i) \tag{113}$$

на некотором обучающем наборе $T = ((x_1, y_1), \ldots, (x_N, y_N))$. Обучение чаще всего производится методом градиентного спуска и комбинированием его с другими методами. Для его применимости функции активации всех нейронов и функции ошибки E и регуляризации ψ должны быть дифференцируемы. Вычисление градиента не совсем просто из-за огромной размерности параметра w и отсутствия явных формул для производных функции F по w.

Замечательно то, что этот огромной размерности градиент ошибки по параметру для нейронной сети можно вычислять красиво и быстро — за время

того же порядка, что и время работы сети — с помощью другой (*двойственной*) нейронной сети, получающейся из исходной обращением направлений ребер. На этом основании градиентный спуск для нейронных сетей обычно называется не просто градиентным спуском, а методом *обратного распространения ошибки*. Хотя строго говоря, метод обратного распространения — это всего лишь способ вычисления градиента ошибки, и даже не обязательно ошибки, а любой функции от выхода нейронной сети.

Конфигурация двойственной сети однозначно определена конфигурацией исходной сети, а веса и функции активации зависят от весов ребер исходной сети и набора значений $\{u(v)\}$ потенциалов нейронов. А именно,

- нейронами $\check{v}_{\dots}$ двойственной сети являются все скрытые нейроны и входы $v_{\dots}$ исходной сети;
- входами $\check{v}_L^1, \dots, \check{v}_L^q$ двойственной сети являются выходные нейроны исходной сети;
- ребрами $\check{e}_{\dots}$ двойственной сети являются ребра $e_{\dots}$ исходной сети с обращенными направлениями;
- каждому ребру двойственной сети $\check{e}$ приписан вес $\check{w}_{\check{e}} = w_e$, где e — ребро-двойник $\check{e}$;
- каждому скрытому нейрону двойственной сети $\check{v}$ приписана функция активации $\check{g}_{\check{v}}$, являющаяся умножением на $g'_v(g_v^{-1}(u(v))) = g'_v(s(v) - w_v)$, т.е. на производную функции активации g_v двойника v нейрона $\check{v}$ в исходной сети, взятую в прообразе значения $u(v)$ относительно g_v, а каждому выходному нейрону — тождественная функция активации.

Таким образом, двойственная сеть зависит не только от исходной сети, но и от поданного на нее входного вектора.

Для реально используемых функций активации выражение $g'_v(g_v^{-1}(u(v)))$ оказывается очень простым.

Упражнение. Если активирующая функция $g_v(t)$ —

- гиперболический тангенс, то $g'_v(g_v^{-1}(u(v))) = 1 - (u(v))^2$;
- логистическая функция, то $g'_v(g_v^{-1}(u(v))) = u(v)(1 - (u(v)))$;
- тождественная функция, то $g'_v(g_v^{-1}(u(v))) = 1$.

В первых двух случаях оно максимально, когда активирующая сумма (минус смещение) $s(v) = g_v^{-1}(u(v))$ равна 0 и экспоненциально убывает при возрастании модуля активирующей суммы.

На каждый вход $\check{v}_L^j$ двойственной сети подают частную производную функции ошибки по соответствующему выходу исходной сети

$$\left.\frac{\partial E(r,y)}{\partial r^j}\right|_{r=F(w,x)} ,$$

умноженную на $g'_v(g_v^{-1}(u(v_L^j)))$. Например, для квадратичной ошибки

$$E(F(w,x),y) = \|F(w,x) - y\|^2 ,$$

часто применяемой при обучении (многомерной) регрессии, и тождественной функции активации выходных нейронов эта производная домножается на 1 и равна

$$\check{u}(\check{v}_L^j) = 2(F^j(w,x) - y^j) = 2(u(v_L^j) - y^j) ,$$

а для кросс-энтропийной ошибки

$$E(F(w,x),y) = -y \ln F(w,x) - (1-y)\ln(1-F(w,x)) \ ,$$

применяющейся для сети с единственным выходом v_L, оценивающем вероятность чего-либо (ср. с формулой (54)), и логистической функции активации единственного выходного нейрона $g_{v_L}(t) = \frac{1}{1+e^{-t}}$ производная этой ошибки, равная $\frac{1-y}{1-F(w,x)} - \frac{y}{F(w,x)}$, домножается на $F(w,x)(1-F(w,x))$ и получается

$$\check{u}(\check{v}_L) = (1-y)F(w,x) - y(1-F(w,x)) = F(w,x) - y = u(v_L) - y \ ,$$

т.е. с точностью до коэффициента 2 то же самое.

Упражнение. Проверьте вычисления.

С этими значениями потенциалов входов двойственную сеть запускают на счет и вычисляют потенциалы всех ее нейронов. Нетривиально, что для дальнейшего понадобятся посчитанные потенциалы не только выходных, а всех нейронов и входов обеих сетей — и исходной, и двойственной.

Доказательство следующего предложения настоятельно рекомендуется в качестве **упражнения** по осмыслению и вычислению частных производных сложных функций: нужно многократно применить правило дифференцирования композиции функций к уравнению (111). Напомним, что $s(v)$ — это активирующая сумма для нейрона v, в общем виде определенная в формуле (109), а потенциал ребра — это по определению потенциал его начала.

Предложение 7 *Для любого нейрона v исходной сети*

$$-\frac{\partial E(F(w,x),y)}{\partial w_v} = \frac{\partial E(F(w,x),y)}{\partial s(v)} = \check{u}(\check{v}),$$

где $\check{v}$ — нейрон, двойственный к v. Для любого ребра e исходной сети

$$\frac{\partial E(F(w,x),y)}{\partial w_e} = u(e)\check{u}(\check{e})$$

где $\check{e}$ — ребро, двойственное к e.

То есть, для вычисления градиента ошибки перцептрона по параметрам для одного обучающего вектора (x,y) нужно подать вектор x на вход, вычислить и запомнить потенциалы всех нейронов, сформировать двойственную сеть, вычислить производную ошибки в точке $(F(w,x),y)$, подать ее на вход двойственной сети, вычислить двойственные потенциалы всех нейронов, и для каждого ребра сети запомнить произведение исходного и двойственного потенциалов в его концах — и все частные производные ошибки по параметрам посчитаны (производные по w_v — с обратным знаком).

Для вычисления градиента регуляризованной ошибки (113) остается просуммировать вычисленный выше градиент по всем обучающим векторам и прибавить градиент функции регуляризации $\psi(w)$. Обычно применяется квадратичная регуляризация $\psi(w) = \epsilon \sum_e w_e^2$, неформально говоря, штрафующая отклонение вычисляемой перцептроном функции от константы, или $\psi(w) = \epsilon_e \sum_e w_e^2 + \epsilon_v \sum_v w_v^2$, еще и обеспечивающая отсутствие слишком больших по модулю активирующих сумм нейронов, т.е. отсутствие плохо обучаемых нейронов, от параметров которых вычисляемая перцептроном функция почти не зависит, а также гарантирующая отсутствие минимумов выражения (113) на бесконечности. Их градиент вычисляется тривиально (**упражнение**).

После того, как градиент регуляризованной ошибки (113) вычислен, можно сделать шаг градиентного спуска, уменьшающий ее. Подробнее организация градиентного обучения описана в разделе 3.2.4.

Замечание. Послойная структура сети для применимости метода обратного распространения ошибки никак не используется: ацикличности сети достаточно.

Замечание. Метод обратного распространения ошибки обобщается и на сети с функциями проводимости ребер h_e и функциями активации нейронов g_v общего вида (109) и (110). При этом двойственная сеть и аналог предложения 7 выглядят менее изящно: требуется знание не только потенциалов всех нейронов сети, но и их активирующих сумм. Зато лучше видно, что работа двойственной сети — это просто вычисление частных производных композиций функций многих переменных. Отличия общего случая от случая многослойного перцептрона таковы:

- каждому ребру двойственной сети $\check{e}$ приписан вес

$$\left.\frac{\partial h_e(w_e,t)}{\partial t}\right|_{t=u(e)};$$

- каждому скрытому нейрону двойственной сети $\check{v}$ приписана линейная функция активации $\check{g}_{\check{v}}$, являющаяся умножением на

$$\left.\frac{\partial g_v(w_v,t)}{\partial t}\right|_{t=s(v)};$$

- на каждый вход $\check{v}_L^j$ двойственной сети подают частную производную функции ошибки по соответствующему выходу исходной сети, умноженную на

$$\left.\frac{\partial g_{v_L^j}(w_{v_L^j},t)}{\partial t}\right|_{t=s(v_L^j)};$$

- для любого нейрона v исходной сети

$$\frac{\partial E(F(w,x),y)}{\partial w_v} = \left.\frac{\partial g_v(r,s(v))}{\partial r}\right|_{r=w_v} \check{u}(\check{v});$$

- для любого ребра e исходной сети

$$\frac{\partial E(F(w,x),y)}{\partial w_e} = \left.\frac{\partial h_e(r,u(e))}{\partial r}\right|_{r=w_e} \check{u}(\check{e}).$$

Упражнение. Докажите, что так построенная двойственная сеть действительно вычисляет градиент[25] $\nabla_w E(F(w,x),y) = \frac{\partial E(F(w,x),y)}{\partial w}$.

Обучение нейронных сетей градиентными методами с использованием обратного распространения ошибки было придумано относительно поздно — в 1970-е годы — и стало господствующим после публикации статьи [99].

[25] Обозначения $\nabla_x f(x,y)$ и $\frac{\partial f(x,y)}{\partial x}$ для вектора частных производных функции f по переменным x будем считать эквивалентными и во внутристрочных формулах использовать, преимущественно, более компактное "∇".

3.2.4 Обучение нейронных сетей: применение градиентного спуска и стохастических методов

В разделе 3.2.3 был продемонстрирован способ вычисления градиента функции ошибки, специфический для нейронных сетей и необходимый для градиентной минимизации эмпирического риска. Теперь рассмотрим (довольно поверхностно) некоторые общие методы минимизации функций, применимые для обучения нейронных сетей. Специфики от нейронных сетей остается немного:

- минимизируемая функция определена на множестве $\mathcal{W} \subset \mathbb{R}^n$ (наборы параметров ребер и вершин) огромной размерности n;
- значение функции и ее градиент в любой точке вычисляются за время одного порядка (градиент — в константу раз медленнее);
- матрица вторых производных функции неприемлемо велика;
- у функции может быть очень много симметрий (скрытые нейроны одного и того же слоя могут быть одинаковы);
- у функции может быть очень много локальных минимумов (кроме обусловленных симметриями).

В этой ситуации для минимизации функции естественно использовать градиентные методы первого порядка в сочетании с какими-либо методами ускорения сходимости и выталкивания решения из нежелательных локальных минимумов.

Но при применении градиентных методов к многослойным перцептронам есть еще один специфический момент, влияющий на скорость сходимости:

- параметры нейрона обучаются быстро, когда производная активирующей функции велика, т.е. разность активирующей суммы и смещения ограничена по модулю.

Действительно, у аргумента нелинейной функции активации типа сглаженной ступеньки (например, $g(t) = \mathrm{th}(t)$) есть "хороший" диапазон изменения, в котором производная функции достаточно велика, например, отрезок $[-1, 1]$, и два "плохих" диапазона, в которых производная функции очень мала, например, лучи $(-\infty, -10)$ и $(10, \infty)$. Если для почти всех обучающих векторов активирующая сумма нейрона попадает в "плохой" диапазон, то веса этого нейрона и его входных ребер будут обучаться крайне медленно (предложение 7). А если они попадают в один и тот же "плохой" диапазон, то этот нейрон не только плохо обучаем, но еще и бесполезен, поскольку его потенциал практически не зависит от входных данных.

Ниже перечисляется не претендующий на полноту набор рецептов по обучению многослойных перцептронов. Некоторые рецепты очевидны, а за некоторыми скрыты достаточно сложные теоремы, которые, впрочем, могут быть не всегда применимы. Для удобства будем считать, что функция активации скрытых нейронов — гиперболический тангенс, а смещения нейронов превращены в веса дополнительных ребер (см. формулу (112) и пояснение перед ней).

Нормализация признаков

Чтобы все веса многослойного перцептрона можно было считать равноправными (в частности, чтобы применять регуляризацию $\psi(w) = \epsilon\|w\|^2$, или чтобы градиент $\nabla E_T(w)$ имел хоть какой-то геометрический смысл), нужно отнормировать признаки так, чтобы их значения, как и потенциалы скрытых нейронов, попадали на отрезок $[-1, 1]$.

Начальное приближение обучения

Начальные значения $w^{(1)}$ параметров сети нужно брать такими, чтобы при любом возможном входном векторе активирующая сумма любого нелинейного нейрона попала в "хороший" диапазон. Кроме того, чтобы топологически неразличимые нейроны одного слоя не дублировали друг друга, их начальные параметры должны отличаться. Например, в качестве начальных значений весов можно брать случайные величины, как-то распределенные на отрезке $[-\frac{1}{N_e(v)+1}, \frac{1}{N_e(v)+1}]$, где $N_e(v)$ — число настоящих входных ребер нейрона v, а прибавляемая единица — ребро, заменяющее смещение.

Шаг градиентного спуска

В i-м приближении $w^{(i)}$ к точке минимума регуляризованной ошибки $E_T(\cdot)$ (113) в пространстве параметров $\mathcal{W}$ методом обратного распространения ошибки вычисляется градиент $\nabla E_T(w^{(i)})$ и против этого градиента делается шаг:

$$w^{(i+1)} = w^{(i)} - \eta_i \nabla E_T(w^{(i)}) \ . \tag{114}$$

Для сходимости градиентного спуска к локальному минимуму нужно, чтобы параметры $\eta_i > 0$, регулирующие длину шага, были достаточно малы. Например, для не зависящего от номера шага $\eta_i = \eta$ и функции ошибки $E(w) = w^2$ в одномерном пространстве $\mathcal{W} = \mathbb{R}$ ее градиент равен $\nabla E(w) = 2w$ и градиентный спуск $w^{(i+1)} = w^{(i)} - 2\eta w^{(i)} = (1 - 2\eta)w^{(i)}$ сходится при $\eta < 1$ и расходится при $\eta \geq 1$. А для функции ошибки $E(w) = 10w^2$ сходимость будет только при $\eta < 0.1$.

В общем случае достаточно, чтобы значение η_i стремилось к 0 при возрастании i, но тогда градиентный спуск сходится очень медленно. Есть много эвристических алгоритмов подбора η_i в процессе градиентного спуска, ускоряющих сходимость. Более обоснованные оценки оптимальной длины шага можно выразить через вторые производные минимизируемой функции. У нейронных сетей вторых производных слишком много, чтобы вычислять их все и на каждом шаге, однако приближенно оценить вторые производные по отдельным направлениям нетрудно (см. метод линейного поиска, стр. 77).

Градиентный спуск с инерцией

Вместо шага градиентного спуска (114) можно делать шаг

$$w^{(i+1)} = w^{(i)} - \eta_i \nabla E_T(w^{(i)}) + \mu(w^{(i)} - w^{(i-1)}) \tag{115}$$

с небольшим положительным коэффициентом μ. При этом сходимость градиентного спуска к точке минимума, как правило, ускоряется, особенно в "узких оврагах" минимизируемой функции E_T — областях пространства параметров, в которых градиент почти всюду направлен почти поперек "оврага" и вторая производная в направлении градиента очень велика. Но гарантии сходимости при недостаточно малых параметрах шага η_i по-прежнему нет.

Уравнение (115), особенно при $\eta_i = \text{const}$, имеет наглядную механическую интерпретацию. Если его переписать в виде

$$\mu(w^{(i+1)} - 2w^{(i)} + w^{(i-1)}) = -(1 - \mu)(w^{(i+1)} - w^{(i)}) - \eta \nabla E_T(w^{(i)}) \ ,$$

то видно, что это конечно-разностный аналог уравнения движения

$$\mu w'' = -(1 - \mu)w' - \eta \nabla E_T(w)$$

материальной точки w массы μ в пространстве $\mathcal{W}$ в силовом поле с потенциалом $\eta E_T(\cdot)$ и с жидким трением, пропорциональным ее скорости w' с коэффициентом $(1 - \mu)$.

Линейный поиск

Разумное значение параметра η_i i-го шага градиентного спуска (114) можно находить решая, не обязательно точно, одномерную минимизационную задачу

$$f(\eta_i) = E_T(w^{(i)} - \eta_i \nabla E_T(w^{(i)})) \to \min_{\eta_i \in [0, 2\eta_{i-1}]} \tag{116}$$

(коэффициент 2 в условии не слишком быстрого возрастания параметра шага $\eta_i \leq 2\eta_{i-1}$ — это произвол). Например, можно минимизировать не саму функцию f, а ее квадратичное приближение. Значение $f(0) = E_T(w^{(i)})$ уже известно, производная $f'(0) = -\|\nabla E_T(w^{(i)})\|^2$ при уже известном градиенте $\nabla E_T(w^{(i)})$ вычисляется легко, так что для построения квадратичного приближения достаточно вычислить еще, например, $f(2\eta_{i-1})$.

Упражнение. Проверьте выражение для $f'(0)$ и вычислите η_i вышеописанной минимизацией квадратичного приближения f.

Замечание. Значение функции f в точке минимума ее квадратичного приближения, в отличие от значения квадратичного приближения, может оказаться не только далеким от минимального, но даже бóльшим, чем $f(0)$. Тогда можно искать минимум квадратичного приближения f по вдвое меньшему отрезку. И т.д. — количество эвристик не ограничено.

Метод сопряженных градиентов

Градиентный спуск можно несколько обобщить, делая шаги не в точности против градиента, а в несколько других направлениях. Несколько вариантов таких алгоритмов минимизации под общим названием "*метод сопряженных градиентов*" произошли из одноименного метода решения систем линейных уравнений минимизацией квадрата невязки, предложенного в работе [54]. Эти алгоритмы предполагают возможность вычисления в любой точке значения минимизируемой функции и ее градиента — в точности то, что обеспечивает метод обратного распространения ошибки.

В случае поиска минимума квадратичной функции вида

$$E(w) = H(w, w) + bw \tag{117}$$

с положительно определенной матрицей H эти алгоритмы совпадают друг с другом и сходятся за конечное число шагов, не превосходящее размерности пространства. Начиная с любой начальной точки $w^{(1)}$ строится последовательность точек $w^{(i)}$ с уменьшающимися значениями $E(w^{(i)})$, такая что градиенты $g_i = \nabla E(w^{(i)})$ попарно ортогональны, а направления шагов $w^{(i+1)} - w^{(i)}$ попарно сопряжены[26] относительно H. При этом сама матрица H ни в какой момент не вычисляется.

В окрестности точки локального минимума гладкой функции общего вида матрица ее вторых производных в каждой точке близка к некоторой положительно определенной матрице H. Разные варианты метода сопряженных градиентов ведут себя по-разному, за конечное число шагов не сходятся, красивые утверждения про ортогональность градиентов и сопряженность направлений шагов становятся неверны, но на практике алгоритмы, как правило, сходятся довольно быстро. Алгоритм 2 демонстрирует наиболее популярную реализацию метода сопряженных градиентов.

[26] Напоминание: вектора r и s сопряжены относительно квадратичной формы H, если $H(r, s) = 0$ или, в матричной записи, $r^\top Hs = 0$.

Алгоритм 2: Метод сопряженных градиентов (Polak-Ribière)

1. Вход: вычислитель минимизируемой функции $E(\cdot)$ и ее градиента $\nabla E(\cdot)$, начальная точка $w^{(1)}$, критерий остановки TimeToStop().
2. Вычислить в начальной точке градиент $g_1 = \nabla E(w^{(1)})$ и направление поиска $d_1 = -g_1$.
3. Для каждого шага $i = 1, \ldots,$ пока не ($g_i = 0$ или TimeToStop())
   (a) линейным поиском найти точку минимума $w^{(i+1)}$ функции $E(w)$ на луче $w = w^{(i)} + \alpha d_i$, $\alpha \geq 0$;
   (b) вычислить градиент $g_{i+1} = \nabla E(w^{(i+1)})$;
   (c) вычислить новое направление поиска $d_{i+1} = -g_{i+1} + \frac{(g_{i+1}, g_{i+1} - g_i)}{(g_i, g_i)} d_i$.
4. Выход: последняя вычисленная точка $w^{(i)}$.

Доказательство того, что для квадратичной функции (117) этот алгоритм строит последовательность попарно ортогональных градиентов g_i и попарно сопряженных направлений d_i, а значит сходится за конечное число шагов — это очень громоздкое **упражнение** по линейной алгебре (подсказка: каждый линейный поиск минимума гарантирует, что $(g_{i+1}, d_i) = 0$, а из квадратичности функции E следует, что $g_i - g_j = 2H(w^{(i)} - w^{(j)})$). Это доказательство и обсуждение других вариантов алгоритма приводятся в книге [14] и в свободно распространяемом электронном издании [58]. Условия и скорость сходимости алгоритма для функции общего вида — по-прежнему предмет исследований.

Случайные блуждания

Все вышеописанные градиентные методы искали точки локального минимума регуляризованной ошибки (8), а на самом-то деле нужно искать глобальный, хотя бы без гарантии, но с большой вероятностью. При этом можно снова воспользоваться естественно-научной аналогией, но не физиологической (нейронная сеть) и не механической (градиентный спуск с инерцией), а термодинамической.

Материальная точка, движущаяся с трением в потенциальном поле начиная со случайного места, с большей вероятностью останавливается не в точке глобального минимума, а в точках локального минимума с большей областью притяжения, ограниченной сепаратрисами градиентного потока. Наоборот, молекула идеального газа при броуновском движении в силовом поле с потенциалом E с большей вероятностью находится возле глобального минимума E. Плотность вероятности описывается известным с точностью до коэффициента распределением Больцмана

$$p_B(w) \propto e^{-\frac{E(w)}{kT}} , \tag{118}$$

где T — температура газа, а k — постоянная Больцмана. При низких температурах T для молекулы газа вероятность находиться в области с потенциалом, близким к глобальному минимуму, намного больше, чем вероятность находиться вне ее. При стремлении температуры к абсолютному нулю весь идеальный газ конденсируется в точках глобального минимума потенциала.

Основанный на этой идее метод глобальной минимизации функции называется "*имитация отжига*" (*simulated annealing*) вслед за работой [66], в которой

при минимизации использовалась аналогия не со сжижением газа, а с затвердеванием аморфного вещества, в частности, с отжигом железа. Он представляет собой имитацию броуновского движения в потенциальном поле в пространстве параметров при постепенном понижении температуры.

Для имитации случайных блужданий с распределением (118) чаще всего используется алгоритм, предложенный в работе [86] Николаса Метрополиса с соавторами. Пусть пространство параметров $\mathcal{W}$ является подмножеством евклидова пространства $\mathbb{R}^n$ и распределение Больцмана (118) доопределено нулем (сооветственно, потенциал E — бесконечностью) вне $\mathcal{W}$. Пусть ξ_i — $\dim(\mathcal{W})$-мерные независимые случайные величины с любым центрально-симметричным распределением, конечные суммы которых попадают в любой шар с ненулевой вероятностью (это свойство называется *эргодичностью* случайного блуждания). Например, это может быть нормальное распределение с центром в нуле, равномерное распределение в некотором кубе с центром в нуле или одномерное равномерное распределение на объединении $\dim(\mathcal{W})$ единичных отрезков координатных осей, естественно, тоже с центрами в нуле. Алгоритм Метрополиса из начальной точки $w^{(1)} \in \mathcal{W}$ строит следующую случайную последовательность (очевидным образом являющуюся марковской цепью):

$$w^{(i+1)} = \begin{cases} w^{(i)} + \xi_i & \text{с вероятностью } P = \min\left(1, e^{-\frac{E(w^{(i)}+\xi_i)-E(w^{(i)})}{T_i}}\right) \\ w^{(i)} & \text{с вероятностью } 1-P \end{cases} \tag{119}$$

Здесь, в отличие от термодинамики, "энергия", она же ошибка, E и "температура" T предполагаются безразмерными, и постоянная Больцмана равна 1. При фиксированной температуре алгоритм Метрополиса позволяет моделировать случайную выборку из распределения Больцмана.

Теорема 5 *Распределение случайной величины $w^{(i)}$ (119) в алгоритме Метрополиса при $i \to \infty$ сходится к распределению Больцмана (118), независимо от начальной точки $w^{(1)}$ и распределения шагов ξ_i.*

К сожалению, универсальной оценки скорости сходимости нет.

Заметим, что в алгоритме Метрополиса и в теореме 5 пространство $\mathcal{W} \subset \mathbb{R}^n$ может быть любым, например, областью, дискретной решеткой, набором кривых, а распределение случайных шагов ξ_i также может быть непрерывным, дискретным или каким-либо еще, так что для доказательстве теоремы нужна достаточно продвинутая техника. В оригинальной работе [86] вместо доказательства приводятся общие соображения из термодинамики. Ограничимся доказательством (и то, частичным) в простейшем случае, когда $\mathcal{W} = \mathbb{R}^n$, носитель распределения Больцмана компактен, и все распределения, включая распределение начальной точки $w^{(1)}$, абсолютно непрерывны и задаются своими плотностями, к тому же всюду положительными.

Доказательство теоремы 5 (частичное). Обозначим через q плотность вероятности случайных шагов ξ_i, через $M : w^{(i)} \mapsto w^{(i+1)}$ случайное отображение (119), а через M_* — индуцированное им отображение плотностей распределений[27]:

$$\begin{aligned} M_*p(w') &= \int p(M(w) = w'|w)p(w)dw \\ &= \int q(w'-w)\min\left(1, \frac{p_B(w')}{p_B(w)}\right)p(w)dw \\ &+ \left(\int_{\{w:p_B(w')<p_B(w)\}} q(w'-w)\left(1 - \frac{p_B(w')}{p_B(w)}\right)dw\right)p(w') \end{aligned}$$

[27]Вообще-то функции перетаскиваются навстречу отображениям, но распределения, как линейные функционалы на функциях, а значит и их плотности — вдоль отображений.

(хотя больцмановская плотность вероятности 118 точно не известна, сравнивать ее значения в разных точках и вычислять их отношения можно). Обозначим для краткости плотность вероятности прыжка из точки w в точку w'

$$\alpha(w, w') = q(w' - w) \min\left(1, \frac{p_B(w')}{p_B(w)}\right) \tag{120}$$

и вероятность остановки в точке w'

$$\beta(w') = \int_{\{w: p_B(w') < p_B(w)\}} q(w' - w)\left(1 - \frac{p_B(w')}{p_B(w)}\right) dw \ ; \tag{121}$$

тогда

$$M_* p(w') = \int \alpha(w, w') p(w) dw + \beta(w') p(w') \tag{122}$$

и α и β удовлетворяют вероятностной нормировке

$$\int \alpha(w', w) dw + \beta(w') = 1 \ . \tag{123}$$

Из симметричности распределения q и определения (120) немедленно следует условие

$$\alpha(w, w') p_B(w) = \alpha(w', w) p_B(w') \ . \tag{124}$$

Если интерпретировать плотность больцмановского распределения как плотность газа и считать, что молекулы газа движутся по алгоритму Метрополиса, то условие (124) означает отсутствие переноса массы между любыми областями пространства, а следовательно, стабильность распределения.

Дальнейшее доказательство теоремы состоит из трех утверждений:

1. распределение Больцмана (118) является неподвижной точкой преобразования (122);
2. других неподвижных точек нет;
3. для любого распределения последовательные применения к нему преобразования (122) имеют предельную точку (которая автоматически является пределом и совпадает с распределением Больцмана).

Первое утверждение немедленно следует из равенств (124) и (123):

$$\begin{aligned} M_* p_B(w') &= \int \alpha(w, w') p_B(w) dw + \beta(w') p_B(w') \\ &= \int \alpha(w', w) p_B(w') dw + \beta(w') p_B(w') = p_B(w') \ . \end{aligned}$$

Второе утверждение, неформально говоря, следует из принципа возрастания энтропии. А формально, из вогнутости логарифма следует, что для любого распределения с плотностью p

$$\begin{aligned} & \int \ln \frac{M_* p(w')}{p_B(w')} p_B(w') dw' \\ = & \int \ln \left(\int \alpha(w, w') \frac{p(w)}{p_B(w')} dw + \beta(w') \frac{p(w')}{p_B(w')} \right) p_B(w') dw' \\ = & \int \ln \left(\int \alpha(w', w) \frac{p(w)}{p_B(w)} dw + \beta(w') \frac{p(w')}{p_B(w')} \right) p_B(w') dw' \\ \geq & \int \left(\int \alpha(w', w) \ln \frac{p(w)}{p_B(w)} dw + \beta(w') \ln \frac{p(w')}{p_B(w')} \right) p_B(w') dw' \end{aligned}$$

$$
\begin{aligned}
&= \int\int \alpha(w',w)\ln\frac{p(w)}{p_B(w)}p_B(w')dwdw' + \int \beta(w')\ln\frac{p(w')}{p_B(w')}p_B(w')dw' \\
&= \int\int \alpha(w,w')\ln\frac{p(w')}{p_B(w')}p_B(w)dwdw' + \int \beta(w')\ln\frac{p(w')}{p_B(w')}p_B(w')dw' \\
&= \int\int \alpha(w',w)\ln\frac{p(w')}{p_B(w')}p_B(w')dwdw' + \int \beta(w')\ln\frac{p(w')}{p_B(w')}p_B(w')dw' \\
&= \int\left(\int \alpha(w',w)dw + \beta(w')\right)\ln\frac{p(w')}{p_B(w')}p_B(w')dw' \\
&= \int \ln\frac{p(w')}{p_B(w')}p_B(w')dw' \ .
\end{aligned}
$$

Но из эргодичности определяемого алгоритмом Метрополиса блуждания и строгой вогнутости логарифма следует, что при $\frac{p(w)}{p_B(w)} \neq \text{const}$, т.е. при $p \neq p_B$, неравенство в этой цепочке — строгое, а значит p не может быть неподвижной точкой отображения M_*. Подробности оставляются в качестве **упражнения** с подсказкой: из точки максимума отношения $\frac{p(w)}{p_B(w)}$ в точку ее минимума можно с положительной вероятностью попасть за несколько итераций алгоритма Метрополиса.

Третье утверждение следует из соображений компактности в пространстве распределений, и его аккуратное доказательство также оставляется в качестве **упражнения**. □

При применении алгоритма Метрополиса для имитации отжига в качестве потенциала $E(w)$ берется значение ошибки распознавателя (нейронной сети) с вектор-параметром w на фиксированном обучающем наборе. В отличие от условия теоремы 5, алгоритм Метрополиса применяется не при постоянной температуре T, а при температуре T_i, убывающей на каждом шаге. Скорость понижения температуры T_i при имитации отжига, например, степенная $T_i = i^{-\gamma}$ с небольшим положительным γ, по возможности обеспечивающая приемлемую вероятность попадания в приемлемую окрестность глобального минимума за приемлемое количество шагов, к сожалению, определяется эмпирически.

При обучении многослойных нейронных сетей метод имитации отжига применяют, чередуя с каким-либо методом градиентного спуска, только для выталкивания градиентного спуска из локальных минимумов.

Эволюция популяции

Еще одно применение естественных наук к минимизации функций, и в частности, к обучению многослойных нейронных сетей, состоит в том, чтобы обучать сразу несколько сетей (популяцию), в процессе обучения имитируя эволюцию популяции живых особей. Подобные методы оптимизации под названием "генетические алгоритмы" описаны в книге [57]. Про их сходимость мало что доказано, но они очень просто реализуются и на практике бывают полезны.

Для обучения нейронных сетей применение эволюционных идей может выглядеть, например, следующим образом:

Начальная популяция. Несколько нейронных сетей с одинаковыми графами, но разными наборами параметров w.

Наследственность. Каждая нейронная сеть обучается каким-либо градиентным методом.

Изменчивость. Время от времени какая-нибудь сеть переходит на обучение методом имитации отжига или просто ее параметры случайным образом возмущаются.

Естественный отбор. Время от времени сеть с наибольшей ошибкой обучения E_T "умирает" — удаляется из популяции (вариант: сеть из популяции выбирается случайно и умирает с вероятностью, зависящей от ее ошибки обучения, естественно, тем большей, чем больше ошибка).

Вегетативное размножение. Время от времени сеть с наименьшей ошибкой обучения порождает еще одну такую же сеть (вариант: сеть из популяции выбирается случайно и размножается с вероятностью, зависящей от ее ошибки обучения, тем большей, чем меньше ошибка).

Половое размножение. Время от времени какие-то две случайные сети (вариант: выбранные с вероятностью, зависящей от их ошибок обучения тем большими, чем меньше ошибки) порождают новую гибридную сеть. В книге [57] набор параметров w рассматривался как геном, и потомок двух сетей наследовал часть параметров (генов) от одного родителя, а часть — от другого, откуда и пошло название "генетические алгоритмы". Если же параметры сетей вещественны, за вектор параметров сети-потомка можно также взять случайную выпуклую линейную комбинацию векторов параметров родителей.

Венец эволюции. Когда запланированные вычислительные ресурсы исчерпаны или сеть с наименьшей ошибкой стабилизировалась, эта сеть и выдается в качестве ответа.

Можно еще каким-либо способом поддерживать приемлемую численность популяции в процессе эволюции. Применение эволюционной надстройки над любыми методами обучения сетей не очень дорого, легко распараллеливается и дает результат, заведомо не худший, чем обучение одной сети.

Регуляризация, досрочная остановка, связывание параметров

Все описанные выше методы обучения нейронных сетей минимизировали регуляризованную ошибку (113) на обучающем наборе. Оптимальную функцию регуляризации $\psi(w)$ нужно еще подбирать, сравнивая ошибки обученных сетей на валидационном наборе или с помощью кросс-валидации (стр. 1.1.7). Таким образом приходится обучать целое семейство по-разному регуляризованных сетей, что, хотя и легко распараллеливается, довольно ресурсоемко.

Вычислительно более дешевый способ обучения нейронных сетей без риска переобучения состоит в том, чтобы зафиксировать одну достаточно малую функцию регуляризации, лишь бы она не давала набору параметров сети w при обучении уходить на бесконечность, и вместо семейства хорошо обученных (близких к точке локального минимума ошибки) по-разному регуляризованных сетей рассматривать семейство состояний одной и той же сети на разных стадиях обучения. При этом валидация производится одновременно с обучением, и в какой-то момент при увеличении ошибки валидации обучение прекращается.

Другой часто применяющийся для нейронных сетей способ регуляризации, называемый связыванием параметров, состоит в том, что группы нейронов в пределах одного слоя предполагаются одинаковыми и (почти) одинаково ведущими себя при обучении, только связанными с разными подмножествами нейронов предыдущего слоя (или входов). Например, в многослойном перцептроне для анализа растровых изображений, приведенном в качестве примера в разделе 3.2.2, группы нейронов первого скрытого слоя могут иметь не только одинаково устроенные, лишь сдвинутые друг относительно друга "поля зрения", но и принудительно одинаковые (или почти одинаковые) веса ребер, приходящих из соответствующих частей полей зрения (например, из левых нижних углов).

Связывание параметров может быть жесткое или мягкое. Жесткое связывание — это набор условий вида $w_{i_1} = w_{i_2} = \ldots = w_{i_k} = z$; при этом при вычислении градиента ошибки полагается

$$\frac{\partial E_T(w)}{\partial z} = \sum_{j-1}^{k} \frac{\partial E_T(w)}{\partial w_{i_j}} \ ,$$

где производные $\frac{\partial E_T(w)}{\partial w_{i_j}}$ вычисляются методом обратного распространения ошибки. Мягкое связывание — это добавление к функции регуляризации штрафов за отличие связываемых параметров, например, слагаемых вида $\epsilon(w_{i_j} - w_{i_k})^2$.

Стохастический градиентный спуск

Все рассмотренные выше рецепты обучения многослойных сетей были основаны на минимизации регуляризованной суммарной ошибки (113) на некотором обучающем наборе T. Напомним, что минимизация регуляризованной суммарной ошибки — это не самоцель, а лишь суррогат минимизации ожидания ошибки (2). Но ожидание ошибки можно пытаться минимизировать и непосредственно, используя следующий итеративный процесс.

Обучающий набор будем считать бесконечным (на самом деле он может быть просто бесконечной случайной последовательностью векторов из конечного набора). На i-м шаге для сети с набором параметров $w^{(i)}$ и обучающего вектора (x_i, y_i) вычисляется ответ сети $F(w^{(i)}, x_i)$, ошибка $E(F(w^{(i)}, x_i), y_i)$ и — методом обратного распространения — ее градиент $\nabla_w E(F(w, x_i), y_i)|_{w=w^{(i)}}$, после чего делается шаг обучения

$$w^{(i+1)} = w^{(i)} - \eta_i \ \nabla_w E(F(w, x_i), y_i)|_{w=w^{(i)}} \ . \tag{125}$$

Такой метод называется *стохастическим градиентным спуском* в отличие от градиентного спуска (114), который иногда называется *пакетным градиентным спуском*.

К стохастическому градиентному спуску применимы асимптотические теоремы типа *теоремы Роббинса-Монро* ([96]), которая в данной ситуации выглядит так:

Теорема 6 *Если*

- *обучающие векторы* $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$ *— независимые с одинаковым распределением* $\pi(x, y)$,
- *для любого значения* w *случайная величина* $\nabla_w E(F(w, x), y)$, *зависящая от случайного вектора* (x, y) *с тем же распределением* π, *имеет конечное ожидание и дисперсию,*
- $\eta_i > 0$ *при всех* i *и стремится к* 0 *при возрастании* i,
- *ряд* $\sum \eta_i$ *расходится, а*
- *ряд* $\sum \eta_i^2$ *сходится,*

то с вероятностью 1 *стохастический градиентный спуск (125) сходится к локальному минимуму ожидания ошибки (2).*

Доказательство теоремы опустим, только неформально прокомментируем требуемую скорость убывания параметров шага η_i. Очевидно, что расходимость ряда $\sum \eta_i$ необходима для того, чтобы точки локального минимума можно было достичь из любой начальной точки. Менее очевидно, что сходимость ряда

$\sum \eta_i^2$ (следовательно, и стремление η_i к 0) нужна, совместно с конечностью дисперсии градиента ошибки, для того, чтобы оценить квадратичное отклонение случайной последовательности (125) от детерминированной последовательности

$$w^{(i+1)} = w^{(i)} - \eta_i \, \nabla_w E_\pi(w)|_{w=w^{(i)}}$$

минимизации ожидания ошибки

$$E_\pi(w) = \int_{(x,y)\in\mathcal{X}\times\mathcal{Y}} E(F(w,x),y)d\pi(x,y)$$

(с точностью до обозначений это — средняя ошибка (2)), но необходимым условием сходимости обучения не является.

В этой теореме про скорость сходимости ничего не утверждается и формально к последовательности, даже случайной, векторов из конечного обучающего набора она не применима. Требуемая сходимость ряда $\sum \eta_i$ и расходимость $\sum \eta_i^2$ выполняется, например, для параметров шага η_i, пропорциональных $\frac{1}{i^\alpha}$ при $\frac{1}{2} < \alpha \leq 1$. При этом стохастический градиентный спуск сходится, но, как правило, неприемлемо медленно для практического применения . Иногда успешно применяется стохастический градиентный спуск с гораздо медленнее убывающими параметрами η_i.

Стохастический градиентный спуск (125) обычно сходится быстрее пакетного (114), особенно если обучающих векторов достаточно много и среди них много почти одинаковых. Его удобно применять, когда обучающих векторов неограниченно много или они поступают в реальном времени — тогда обучающие вектора можно не хранить в памяти, а сразу забывать. Кроме того, при обучении в реальном времени обучающие данные могут от времени зависеть, и тогда обучаемая стохастическим градиентным спуском нейронная сеть в каждый момент обучается распознавать современные ей данные, постепенно забывая о распознавании позапрошлогодних. Но анализировать процесс обучения стохастическим градиентным спуском и комбинировать его с другими методами обучения труднее, чем пакетное обучение.

Применяется и промежуточный между пакетным и стохастическим вариант градиентного спуска: из огромного обучающего набора или из потока обучающих данных реального времени выбирается не очень большой обучающий набор, на нем производится обучение, после чего он забывается, выбирается следующий не очень большой обучающий набор, обучение продолжается на нем, и т.д.

Компактный обзор применения стохастического градиентного спуска в различных задачах распознавания приведен в лекции [19]. Разнообразные рецепты градиентного обучения нейронных сетей и сравнение условий их применимости и эффективности описаны в статье [74]. Многие методы обучения нейронных сетей приводятся во всех толстых книгах по нейронным сетям, например, [14] или [58]. Одного-единственного самого лучшего универсального метода обучения нейронных сетей нет.

3.3 RBF-сети

В отличие от перцептронов, RBF-сети возникли не в нейрофизиологии, а в теории приближений как формализация приближения экспериментальной функции, заданной таблицей замеренных значений, линейной комбинацией базисных функций (basis functions) заранее определенного класса, к тому же зависящих только от радиус-вектора (т.е. radial) от аргумента до “базовой точки” в пространстве-прообразе (т.е. на более современном языке, от их разности). В

отличие от ранее рассмотренных линейных методов, базисные функции не фиксированы заранее (раздел 2.3.1) и не определены обучающим набором (раздел 2.3.2), а подбираются в процессе обучения.

Простейшая RBF-сеть состоит из двух слоев. Каждый нейрон первого слоя соединен с каждым входом и с каждым нейроном второго слоя. Каждый нейрон v первого (скрытого) слоя вычисляет

$$u(v) = g\left(s_v, \sum_{e \in v_{\text{in}}} (u(e) - w_e)^2\right),$$

где функция g равна $g(s,t) = e^{-t/2s}$, а нетрадиционное обозначение параметра нейрона — s_v вместо традиционного w_v — связано с тем, что вскоре у RBF-нейронов появятся еще параметры. Нейроны второго слоя устроены так же, как выходные нейроны многослойного перцептрона: каждый нейрон v второго слоя вычисляет

$$u(v) = \left(\sum_{e \in v_{\text{in}}} w_e u(e)\right) - w_v \ ,$$

если сеть предназначена для многомерной регрессии, или, как правило,

$$u(v) = \sigma\left(\left(\sum_{e \in v_{\text{in}}} w_e u(e)\right) - w_v\right) \ ,$$

если она предназначена для многоклассовой классификации.

Нейроны первого слоя и вычисляемые ими функции можно интерпретировать следующим образом. Входы каждого нейрона — это в точности входы $x^1, \ldots, x^d$. всей сети, т.е. d-мерный вектор признаков x. Веса входных ребер каждого такого нейрона v также образуют d-мерный вектор μ_v в пространстве признаков, называемый центром нейрона, а нейрон вычисляет функцию

$$u(v) = e^{-\frac{\|x-\mu_v\|^2}{2s_v}} \ ,$$

с точностью до коэффициента $(2\pi s_v)^{-\frac{d}{2}}$ равную значению в x плотности сферического гауссова распределения с центром μ_v и дисперсией s_v.

Появление гауссианов там, где есть большое количество независимых одинаково распределенных случайных величин, а именно, векторов признаков из обучающего множества, вполне естественно. Такое распределение вероятностей получается, когда вектора признаков получаются добавлением суммы мелких независимых помех к одному из конечного числа эталонов, выбранному случайно. А то, что все гауссианы — сферические, — всего лишь модельный пример.

В большинстве практически применяемых сетей RBF каждый нейрон первого слоя v вычисляет свой гауссиан общего вида

$$u(v) = \left((2\pi)^d |A_v|\right)^{-\frac{1}{2}} e^{-\frac{A_v^{-1}(x-\mu_v, x-\mu_v)}{2}}, \tag{126}$$

где A_v — приписанная нейрону симметрическая положительно определенная матрица (матрица ковариации), $|A_v|$ — ее определитель, а $A_v^{-1}(\cdot,\cdot)$ — квадратичная форма с матрицей A_v^{-1}. То есть для каждого нейрона нужно хранить и обучать не только d координат вектора μ_v, но еще и $\frac{d(d+1)}{2}$ различных коэффициентов матрицы A_v. Такая сеть не укладывается в общую схему раздела 3.1, поскольку в производимом нейроном вычислении нельзя выделить суммирование по входам (109). В частности, производные функционала ошибки по параметрам нейронов уже не вычисляются двойственной нейронной сетью. Впрочем, формулы для их вычисления нетрудно выписать явно.

Замечание. Поскольку любая положительно определенная квадратичная форма аффинной заменой координат переводится в сумму квадратов, RBF-сеть с k гауссианами вида (126) можно заменить на эквивалентную ей трехслойную сеть в смысле раздела 3.1, в которой первый слой состоит из линейных нейронов, вычисляющих эти аффинные преобразования, второй слой состоит из k необучающихся нейронов, вычисляющих один и тот же стандартный сферический гауссиан $u(v) = e^{-\frac{\|x\|^2}{2}}$, а третий слой — это бывший второй. Но такое представление неэффективно, поскольку теряется информация о симметричности матриц A_v и, соответственно, почти удваивается число обучаемых параметров.

Замечание. Нейроны первого слоя RBF-сети могут вычислять не только гауссовы, но и любые другие радиальные базисные функции, например, лапласовы. Описываемые далее методы обучения RBF-сетей с бóльшими или меньшими техническими изменениями можно применять к довольно широкому классу базисных функций. Но на практике применяются почти исключительно гауссовы базисные функции.

Замечание. Возможны и RBF-сети с большим числом слоев, в которых первый слой нейронов вычисляет радиальные функции, а последующие слои устроены как в многослойном перцептроне. Применяются они крайне редко.

Далее мы рассматриваем RBF-сети с нейронами, вычисляющими простейший сферический гауссиан, но все утверждения легко переносятся на случай гауссианов общего вида и, несколько менее легко, на многие другие базисные функции и даже на RBF-сети с более чем двумя слоями.

Вычислительные возможности RBF-сетей

В качестве **упражнения** рекомендуется доказать следующее предложение.

Предложение 8 *Любую непрерывную функцию на любом компакте M в пространстве признаков можно равномерно приблизить функциями, вычисляемыми RBF-сетями.*

То есть даже с помощью простейших RBF-сетей можно с любой точностью что угодно вычислить. Плохо только то, что при убывании погрешности приближения ϵ дисперсия s убывает как ϵ^2, а количество скрытых нейронов растет как $\epsilon^{-dim(M)}$, в частности как ϵ^{-d}, если M — куб или шар. Причем оно экспоненциально растет даже если функция не зависит от почти всех признаков (крайний пример — зависящая только от одного признака функция

$$u(x^1, \ldots, x^d) = e^{-\frac{(x^1-\mu^1)^2}{2s}},$$

которая при одномерном пространстве признаков точно вычисляется одним нейроном). Напомним, что в задачах распознавания d бывает порядка десятков или сотен. Применение несферических и разных для разных нейронов гауссианов позволяет уменьшить требуемое число нейронов, но все равно их требуется очень много.

3.3.1 Обучение RBF-сетей: метод максимизации ожидания (*expectation maximization*)

RBF-сети можно обучать методом обратного распространения ошибки и градиентного спуска, как и многослойный перцептрон. Но их можно обучать и другим способом, тоже итеративным, но, как правило, сходящимся быстрее

градиентного спуска. При этом слои сети обучаются раздельно: сначала первый, потом второй, и вообще говоря, на разных обучающих множествах. Обучение второго слоя при обученном первом — это обучение линейной или логистической регрессии, рассмотренное в разделах 2.1 и 2.2.2, соответственно. А обучение первого слоя мы сейчас рассмотрим на простом примере, когда каждый нейрон v первого слоя вычисляет сферический гауссиан с центром μ_v и дисперсией s_v.

Описываемый далее алгоритм обучения является частным случаем алгоритмов *максимизации ожидания* или *EM-алгоритмов* (Expectation-Maximization), в довольно общем виде описанных в статье [33], а в других частных случаях известных под самыми разными названиями, например, алгоритм Баума-Велша [8]. Для краткости алгоритм излагается довольно формально (по книге [14]); при этом остается неясным смысл слова "ожидание" в названии алгоритма.

Будем считать обучающие вектора признаков значениями независимых одинаково распределенных многомерных случайных величин, плотность распределения которых является *гауссовой смесью*, т.е. линейной комбинацией

$$p(x) = \sum_{j=1}^{k} P_j p_j(x) = \sum_{j=1}^{k} P_j \frac{1}{(2\pi s_j)^{\frac{d}{2}}} e^{-\frac{\|x-\mu_j\|^2}{2s_j}} \tag{127}$$

k гауссианов $p_j(x)$ с положительными коэффициентами P_j, удовлетворяющих условию $\sum_{j=1}^{k} P_j = 1$. Плотность $p(x)$ по формуле (127) вычисляется двухслойной RBF-сетью с k скрытыми нейронами и одним выходным. Обучать набор параметров этой сети $\mathbf{W} = (P_1, \mu_1, s_1, \ldots, P_k, \mu_k, s_k)$ будем методом максимизации правдоподобия (стр. 23), т.е. в данной ситуации — максимизации плотности вероятности появления набора векторов признаков $\mathbf{X} = (x_1, \ldots, x_N)$ обучающего набора $T = ((x_1, y_1), \ldots, (x_N, y_N))$.

Такое обучение, хотя оно и не учит непосредственно распознаванию, удобно и полезно вот почему. Для векторов признаков не требуется знать ответы (т.е. обучение проводится без учителя), что облегчает сбор обучающих примеров. А вероятностная модель (127) адекватна следующей, вполне реальной, ситуации. Каждый распознаваемый объект, например, изображение буквы, получается из некоторого эталонного способа написания, которому учат или не учат в школе, добавлением многих мелких независимых ошибок. Количество эталонов k не связано с количеством классов q, если решается задача классификации, и может его намного превышать. Например, в задаче распознавания буквы по траектории ее написания угловатую "рукопечатную" букву E, состоящую из четырех отрезков, можно писать $4! * 2^4 = 384$ топологически разными способами, хотя большинство из них крайне маловероятны. Обучение первого слоя сети — это поиск таких эталонов μ_j и соответствующих им дисперсий s_j, а также их вероятностей P_j. Поэтому в отличие от многослойного перцептрона, в котором каждый скрытый нейрон учится неизвестно чему, за обучением RBF-сети вполне можно следить и понимать, какой скрытый нейрон чему учится.

В вероятностных терминах соображение про эталоны, порождающие распределение (127), выражается так. У каждого обучающего объекта кроме наблюдаемого вектора признаков $x \in \mathbb{R}^d$ и не используемого для обучения ответа y есть еще *ненаблюдаемый признак* $j \in \{1, \ldots, k\}$ (номер эталона). На прямом произведении $\mathbb{R}^d \times \{1, \ldots, k\}$ определено распределение с плотностью

$$p(x, j) = P_j p_j(x)$$

и обучающие пары (x_i, j_i) являются независимыми случайными величинами,

распределенными с плотностью $p(x, j)$. Тогда

$$P\{j\} = \int_{x \in \mathbb{R}^d} p(x, j)dx = P_j$$

— вероятность j-го эталона,

$$p(x|j) = \frac{p(x, j)}{P\{j\}} = p_j(x)$$

— условная плотность вероятности наблюдаемых признаков x при ненаблюдаемых j, $p(x) = \sum_{j=1}^{k} P\{j\}p(x|j)$ — полная плотность вероятности (127), и по формуле Байеса условная вероятность j-го эталона при наблюдении вектора x равна

$$P\{j|x\} = \frac{p(\{j\}, x)}{p(x)} = \frac{P\{j\}p(x|j)}{\sum_{j=1}^{k} P\{j\}p(x|j)} = \frac{P_j p_j(x)}{\sum_{j=1}^{k} P_j p_j(x)} \ . \tag{128}$$

В рассматриваемой нами модели плотность вероятности $p_j(x)$ всюду положительна, вероятность P_j можно считать положительной (иначе j-й эталон можно выбросить), а значит и условная вероятность $P\{j|x\}$ положительна.

Максимизация правдоподобия

$$p(\mathbf{X}|\mathbf{W}) = \prod_{i=1}^{N} \sum_{j=1}^{k} P_j p_j(x_i) \to \max_{\mathbf{W}} \tag{129}$$

эквивалентна минимизации "функции ошибки"

$$E(\mathbf{X}, \mathbf{W}) = -\ln p(\mathbf{X}|\mathbf{W}) = -\sum_{i=1}^{N} \ln p(x_i) = -\sum_{i=1}^{N} \ln \sum_{j=1}^{k} P_j p_j(x_i) \to \min_{\mathbf{W}} \ .$$

Точек минимума, даже глобального, может быть много: в общем случае их не меньше числа перенумераций множества k нейронов, т.е. $k!$, при этом заведомо будут и другие критические точки. Сейчас будет описан итеративный алгоритм локальной минимизации $E(\mathbf{X}, \mathbf{W})$. На каждом шаге функция ошибки мажорируется функцией более простого вида, минимум которой единственен и находится явно.

Зафиксируем любое допустимое (т.е. $P_j > 0$, $\sum_{j=1}^{k} P_j = 1$, $s_j > 0$) значение набора параметров $\mathbf{W}^{(0)}$ и соответствующие ему вероятности и плотности вероятностей будем также обозначать с верхним индексом $^{(0)}$. Тогда

$$\begin{aligned}
E(\mathbf{X}, \mathbf{W}) - E(\mathbf{X}, \mathbf{W}^{(0)}) &= -\sum_{i=1}^{N} \ln \frac{p(x_i)}{p^{(0)}(x_i)} \qquad (130)\\
&= -\sum_{i=1}^{N} \ln \left(\sum_{j=1}^{k} \frac{P_j p_j(x_i)}{p^{(0)}(x_i)} \right) \\
&= -\sum_{i=1}^{N} \ln \left(\sum_{j=1}^{k} P^{(0)}\{j|x_i\} \frac{P_j p_j(x_i)}{P_j^{(0)} p_j^{(0)}(x_i)} \right) \\
&\leq -\sum_{i=1}^{N} \sum_{j=1}^{k} \left(P^{(0)}\{j|x_i\} \ln \frac{P_j p_j(x_i)}{P_j^{(0)} p_j^{(0)}(x_i)} \right) \\
&= -\sum_{i=1}^{N} \sum_{j=1}^{k} P^{(0)}\{j|x_i\} \ln(P_j p_j(x_i)) \\
&\quad + \sum_{i=1}^{N} \sum_{j=1}^{k} P^{(0)}\{j|x_i\} \ln(P_j^{(0)} p_j^{(0)}(x_i)) \ .
\end{aligned}$$

Неравенство в этой цепочке преобразований следует из выпуклости функции $-\ln(\cdot)$ (неравенство Йенсена): если все c_j неотрицательны и $\sum_{j=1}^{k} c_j = 1$, то

$$-\ln\left(\sum_{j=1}^{k} c_j x^j\right) \le -\sum_{j=1}^{k} c_j \ln(x^j) \ .$$

Введем обозначение

$$Q(\mathbf{X}, \mathbf{W}^{(0)}, \mathbf{W}) = -\sum_{i=1}^{N}\sum_{j=1}^{k} P^{(0)}\{j|x_i\} \ln(P_j p_j(x_i)) \ . \tag{131}$$

Тогда полученное неравенство (130) означает, что

$$E(\mathbf{X}, \mathbf{W}) \le Q(\mathbf{X}, \mathbf{W}^{(0)}, \mathbf{W}) - Q(\mathbf{X}, \mathbf{W}^{(0)}, \mathbf{W}^{(0)}) + E(\mathbf{X}, \mathbf{W}^{(0)}) \ .$$

Правая часть мажорирует $E(\mathbf{X}, \mathbf{W})$ и совпадает с ней в точке $\mathbf{W}^{(0)}$. Значит чтобы уменьшить значение ошибки $E(\mathbf{X}, \mathbf{W})$ достаточно найти такую точку $\mathbf{W}$, что $Q(\mathbf{X}, \mathbf{W}^{(0)}, \mathbf{W}) < Q(\mathbf{X}, \mathbf{W}^{(0)}, \mathbf{W}^{(0)})$. Но у функции $Q(\mathbf{X}, \mathbf{W}^{(0)}, \cdot)$ нетрудно найти точку глобального минимума по всей области определения с помощью метода множителей Лагранжа.

Действительно, выпишем лагранжиан

$$L(\mathbf{W}, \lambda) = Q(\mathbf{X}, \mathbf{W}^{(0)}, \mathbf{W}) + \lambda\left(\sum_{j=1}^{k} P_j - 1\right)$$

и систему уравнений

$$0 = \frac{\partial L}{\partial \lambda} = \sum_{j=1}^{k} P_j - 1 \tag{132}$$

$$0 = \frac{\partial L}{\partial P_j} = \lambda - \frac{1}{P_j}\sum_{i=1}^{N} P^{(0)}\{j|x_i\} \tag{133}$$

$$0 = \frac{\partial L}{\partial \mu_j} = \sum_{i=1}^{N} P^{(0)}\{j|x_i\}\frac{\mu_j - x_i}{s_j} \tag{134}$$

$$0 = \frac{\partial L}{\partial s_j} = \sum_{i=1}^{N} P^{(0)}\{j|x_i\}\left(\frac{d}{2s_j} - \frac{\|x_i - \mu_j\|^2}{2s_j^2}\right) \ . \tag{135}$$

Из первых двух строк, т.е. $k+1$ уравнений, совершенно независимо от конкретного вида плотностей вероятности $p_j(\cdot)$ получается, что

$$\lambda = \sum_{j=1}^{k} P_j \lambda = \sum_{j=1}^{k}\sum_{i=1}^{N} P^{(0)}\{j|x_i\} = \sum_{i=1}^{N}\sum_{j=1}^{k} P^{(0)}\{j|x_i\} = \sum_{i=1}^{N} 1 = N$$

Дальнейшее решение системы совсем элементарно. Получается

$$P_j = \frac{1}{N}\sum_{i=1}^{N} P^{(0)}\{j|x_i\} \tag{136}$$

$$\mu_j = \frac{\sum_{i=1}^{N} P^{(0)}\{j|x_i\} x_i}{\sum_{i=1}^{N} P^{(0)}\{j|x_i\}} \tag{137}$$

$$s_j = \frac{1}{d}\frac{\sum_{i=1}^{N} P^{(0)}\{j|x_i\}\|x_i - \mu_j\|^2}{\sum_{i=1}^{N} P^{(0)}\{j|x_i\}} \ , \tag{138}$$

где μ_j в третью строку нужно подставить из второй.

Упражнение. Проверьте, что найденное решение $\mathbf{W}$ попадает в область определения функции $Q(\mathbf{X}, \mathbf{W}^{(0)}, \cdot)$ и является точкой ее минимума (из единственности решения следует, что глобального).

Упражнение. Проверьте, что если $\mathbf{W} = \mathbf{W}^{(0)}$, то $\mathbf{W}^{(0)}$ является точкой локального минимума функции ошибки $E(\mathbf{X}, \cdot)$.

Шаг обучения первого слоя RBF-сети методом максимизации ожидания завершен. Для почти всех обучающих наборов $\mathbf{X}$ и допустимых начальных значений параметров модели ее обучение сходится к локальному минимуму функции $E(\mathbf{X}, \cdot)$, который, в отличие от минимума функции $Q(\mathbf{X}, \mathbf{W}^{(0)}, \cdot)$, совершенно не обязан быть глобальным. Начальные значения $\mathbf{W}$ можно брать следующим образом: $P_j = \frac{1}{k}$, $\mu_j = x_{i_j}$ — k различных векторов признаков из обучающего набора (первых или случайных), $s_j = \frac{1}{Nd} \sum_{i=1}^{N} \|x_i - \mu_j\|^2$.

Напоминаем, что в рассматриваемом модельном примере все распределения p_j — сферические гауссовы. Для остальных параметров гауссианов общего вида (а именно, для коэффициентов матриц ковариации A_j) ответ тоже единственен, элементарно вычислим и даже легко угадывается.

Предложение 9 *Вышеописанным метод максимизации ожидания применим и к гауссианам общего вида*

$$p_j(x) = \left((2\pi)^d |A_j|\right)^{-\frac{1}{2}} e^{-\frac{A_j^{-1}(x-\mu_j, x-\mu_j)}{2}} . \tag{139}$$

В шаге обучения формулы (136) и (137) остаются без изменений, а вместо каждой формулы (138) получается $\frac{d(d+1)}{2}$ *формул*

$$A_j^{lm} = \frac{\sum_{i=1}^{N} P^{(0)}\{j|x_i\}(x_i^l - \mu_j^l)(x_i^m - \mu_j^m)}{\sum_{i=1}^{N} P^{(0)}\{j|x_i\}}, \quad 1 \le l \le m \le d,$$

где отличные от $^{(0)}$ *верхние индексы относятся к координатам в пространстве признаков.*

Доказательство оставляется в качестве **упражнения**.

В завершение следует еще напомнить, что обучение RBF-сети не кончается обучением ее первого слоя. После этого вместо временного второго слоя с одним нейроном, вычислявшего плотность вероятности (127), вставляют и обучают — как линейный распознаватель над базисными функциями, вычисляемыми нейронами первого слоя — настоящий второй слой, вычисляющий требуемую классификацию или регрессию, а после раздельного обучения слоев сети ее обычно еще доучивают как целое каким-либо градиентным методом (раздел 3.2.4). И после всего этого, минимизируя ошибку RBF-сети на независимом от обучающего проверочном наборе, подбирают оптимальное число скрытых нейронов k.

Раздельное обучение слоев RBF-сети замечательно тем, что обучение первого слоя можно проводить на неразмеченном обучающем наборе, состоящем только из векторов признаков без ответов. Это особенно удобно в тех задачах, в которых сбор векторов признаков производится автоматически и их можно набрать сколько угодно, а приписывание соответствующих признакам ответов — вручную, так что объем полноценного обучающего набора довольно-таки ограничен.

Замечание. Если обучающих векторов с ответами достаточно много, то в задачах классификации, особенно с малым числом классов q, возможен альтернативный способ обучения RBF-сети: для каждого класса обучать отдельную

группу нейронов, т.е. искать эталоны векторов признаков для каждого класса независимо. При этом еще потребуется как-то решать, сколько нейронов выделить какому классу. Соответственно, каждому классу (l-му) при обучении первого слоя соответствует свой выходной нейрон, вычисляющий оценку плотности условной вероятности $p(x|y = l)$. Но поскольку задачей классификационной RBF-сети является вычисление условных вероятностей $P\{y|x\}$, а не плотностей $p(x|y)$, после обучения первого слоя сети второй слой нужно полностью переучивать.

4 Линейные распознаватели и ядра: некоторые подробности

В этом разделе рассматриваются довольно сложные примеры успешно применяемых на практике линейных распознавателей, использующих ядра. Изложение не является исчерпывающе полным: подробности теорий, лежащей в их основе, лучше изучать по первоисточникам [116, 117, 114] и авторским обзорам [115], а практика применения и программные реализации описаны в весьма многочисленной литературе. Здесь же основное внимание уделяется пониманию применяемого математического аппарата.

4.1 Ядра

Ядра и их применение для распознавания уже упоминались в разделе 2.3.2. Формальное определение ядра таково. Пусть ϕ — произвольное отображение (не обязательно линейное, не обязательно непрерывное, не обязательно вложение) любого пространства признаков $\mathcal{X}$ (не обязательно евклидова $\mathbb{R}^d$) в евклидово или гильбертово пространство $\mathbf{H}$. Функция $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, вычисляющая скалярное произведение в $\mathbf{H}$ образов пары векторов признаков:

$$K(x_1, x_2) = (\phi(x_1), \phi(x_2)) \tag{140}$$

называется *ядром*. Название "ядро" связано с тем, что при наличии на пространстве признаков $\mathcal{X}$ меры μ, в частности при $\mathcal{X} = \mathbb{R}^d$, можно определить интегральный оператор

$$f \mapsto \hat{K}(f) = \int\limits_{\mathcal{X}} K(\cdot, x) f(x) d\mu(x) \ , \tag{141}$$

имеющий-таки отношение к делу, для которого функция K является ядром. Пространство $\mathbf{H}$ будем называть *спрямляющим пространством* ядра K, а отображение ϕ — соответственно, *спрямляющим отображением*.

Напомним, что отображение ϕ из пространства признаков в пространство большой размерности строится для того, чтобы в пространстве-образе можно было обучать линейные распознаватели независимо от того, был ли прообраз линейным пространством, а если и был, то не были ли в нем обучающие векторы патологически линейно зависимыми. В частности, при двухклассовой классификации линейно неразделимые наборы векторов в пространстве признаков могут перейти в линейно разделимые наборы в новом большом пространстве, т.е. появится возможность безошибочной (на обучающем наборе!) классификации. Но из отсутствия ошибок обучения еще не следует малость средней ошибки.

Несмотря на то, что размерность пространства-образа может быть очень большой и даже бесконечной, линейные распознаватели довольно популярного класса в этом большом пространстве обучаются за время, не зависящее от

размерности. А именно, это распознаватели, обучаемые посредством минимизации суммы некоторой (любой!) функции от ошибок на обучающем наборе и квадрата евклидовой нормы вектора коэффициентов (только коэффициентов, не свободного члена!) распознающей функции. Например, такими являются методы ridge regression (72), logistic regression (86) и soft margin classifier (100). Это следует из следующей простой теоремы.

Теорема 7 *Пусть задано отображение $\phi : \mathcal{X} \to \mathbf{H}$ из пространства признаков в полное пространство (евклидово или гильбертово) со скалярным произведением $(\cdot,\cdot)$ и соответствующей ему нормой $\|\cdot\|$. Пусть линейный распознаватель $f(x) = (w, \phi(x)) + b$, где $w \in \mathbf{H}$ и $b \in \mathbb{R}$, обучается на обучающем наборе $T = ((x_1, y_1), \ldots, (x_N, y_N))$ минимизацией функционала*

$$\epsilon\|w\|^2 + \sum_{i=1}^{N} E(f(x_i), y_i) \to \min_{w,b} \tag{142}$$

при некоторой функции ошибки E. Тогда для любой точки минимума (w, b) вектор w принадлежит линейной оболочке векторов $\phi(x_1), \ldots, \phi(x_N)$.

Замечание. Про свободный член b теорема ничего не утверждает.

Замечание. Эта теорема без каких-либо усилий переносится на обучение с функцией ошибки, не распадающейся на сумму ошибок на отдельных векторах, и/или регуляризатором, являющимся любой строго возрастающей функцией от $\|w\|$, то есть на минимизацию любого функционала вида

$$\psi(\|w\|) + E(f(x_1), y_1, \ldots, f(x_N), y_N) \to \min_{w,b}$$

с возрастающей функцией ψ;

Доказательство. Пусть (w, b) — решение задачи (142), а w_X — ортогональная проекция вектора w на линейную оболочку обучающих векторов $\phi(x_1), \ldots, \phi(x_N)$. Тогда если $w_X \neq w$, то $\|w_X\| < \|w\|$, а при любом i $(w, \phi(x_i))+b = (w_X, \phi(x_i))+b$. Значит в точке (w_X, b) значение минимизируемого функционала (142) меньше. Противоречие. □

Получается, что распознаватель достаточно обучать в не более чем $(N+1)$-мерном пространстве. Подставляя $w = \sum_{i=1}^{N} \beta_i \phi(x_i)$ в выражение (142), получим минимизационную задачу

$$\begin{aligned}
&\epsilon\|w\|^2 + \textstyle\sum_{i=1}^{N} E(f(x_i), y_i) \\
&= \epsilon \left(\sum_{i=1}^{N} \beta_i \phi(x_i), \sum_{j=1}^{N} \beta_j \phi(x_j) \right) + \sum_{i=1}^{N} E\left(\left(\sum_{j=1}^{N} \beta_j \phi(x_j), \phi(x_i) \right) + b, y_i \right) \\
&= \epsilon \sum_{i,j=1}^{N} \beta_i \beta_j (\phi(x_i), \phi(x_j)) + \sum_{i=1}^{N} E\left(\sum_{j=1}^{N} \beta_j (\phi(x_i), \phi(x_j)) + b, y_i \right) \\
&= \epsilon \sum_{i,j=1}^{N} \beta_i \beta_j K(x_i, x_j) + \sum_{i=1}^{N} E\left(\sum_{j=1}^{N} \beta_j K(x_i, x_j) + b, y_i \right) \to \min_{\beta_1,\ldots,\beta_N,b},
\end{aligned} \tag{143}$$

в которой нет никаких упоминаний о пространстве $\mathbf{H}$ и отображении ϕ, а есть только значения ядра на парах обучающих векторов. В зависимости от конкретного вида функции ошибок E, значения ϵ и способа минимизации (в случае неединственного решения) будут получаться разные распознаватели.

Теперь рассмотрим ядра подробнее, доказывая утверждения, анонсированные в разделе 2.3.2.

4.1.1 Свойства ядер Мерсера и теорема о реализации

Ядром (140) может быть не любая функция двух (векторных) переменных. Простое **упражнение**:

Предложение 10

Любое ядро (140) симметрично ($K(x_1, x_2) = K(x_2, x_1)$ для любых x_1 и x_2) и неотрицательно определено (матрица $K_{ij} = K(x_i, x_j)$ неотрицательно определена для любого конечного набора векторов $x_1, \ldots, x_N$). В частности, $K(x_1, x_1)$ 0 и $|K(x_1, x_2)| \leq \sqrt{K(x_1, x_1)K(x_2, x_2)}$.

□

Симметричные неотрицательно определенные ядра называются *ядрами Мерсера*. Поскольку в этом разделе мы рассматриваем в основном их, будем для краткости называть их просто ядрами.

Оказывается, ничего больше от ядра не требуется. Но прежде, чем доказывать это в общем виде, в качестве **упражнений** рассмотрим полезные частные случаи.

Предложение 11 *Для конечного пространства признаков $\mathcal{X}$ любая симметричная и неотрицательно определенная функция $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ является ядром.*

□

Предложение 12 *Любая непрерывная, симметричная и неотрицательно определенная функция $K : I^d \times I^d \to \mathbb{R}$ на квадрате d-мерного куба I^d является ядром.*

Доказать последнее предложение в лоб довольно трудно, но его легко свести к следующей теореме из теории интегральных операторов:

Теорема 8 *(Mercer, 1909 [83]). Если функция $K \in L^2(\mathbb{R}^n \times \mathbb{R}^n)$ симметрична и задаваемый ею интегральный оператор (141) в $L^2(\mathbb{R}^n)$ неотрицательно определен, т.е. для любой функции $f \in L^2(\mathbb{R}^n)$*

$$\int_{x,z \in \mathbb{R}^n} K(x, z) f(x) f(z) dx dz \geq 0 \ ,$$

то функция K представима в виде суммы сходящегося в $L^2(\mathbb{R}^n \times \mathbb{R}^n)$ ряда

$$K(x, z) = \sum_{i=1}^{\infty} \lambda_i \phi_i(x) \phi_i(z), \quad \lambda_i \geq 0, \quad \phi_i \in L^2(\mathbb{R}^n)$$

с ортонормированной системой функций $\{\phi_i\}$ и последовательностью коэффициентов $\lambda \in l^2$.

Заметим, что теорема Мерсера является обобщением на гильбертово пространство $L^2(\mathbb{R}^n)$ и интегральные операторы в нем следующего общеизвестного конечномерного факта: неотрицательно определенный симметрический оператор ортогонально диагонализируем и все его собственные значения неотрицательны.

Теперь докажем полноценное обращение предложения 10.

Теорема 9 *Для любого пространства признаков $\mathcal{X}$ и любой симметричной неотрицательно определенной функции $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ существует гильбертово пространство $\mathbf{H}$ и отображение $\phi : \mathcal{X} \to \mathbf{H}$, удовлетворяющее условию (140).*

Замечание. Пространство $\mathbf{H}$ может случайно оказаться и конечномерным евклидовым.

Доказательство. Рассмотрим в линейном пространстве функций $\mathcal{X} \to \mathbb{R}$ линейное подпространство H_0, порожденное функциями вида $\phi(x) : z \mapsto K(x,z)$. Определим на этом подпространстве билинейную форму $(\cdot,\cdot)$, билинейно продолжив отображение $(\phi(x),\phi(z)) \mapsto (\phi(x))(z) = K(x,z) = (\phi(z))(x)$. При этом автоматически получится, что $(h,\phi(z)) = h(z) = (\phi(z),h)$ для любой функции $h \in H_0$. Поскольку функции вида $\phi(x)$ могут оказаться линейно зависимыми, а значит разложение функций из H_0 по функциям вида $\phi(x)$ — неоднозначным, необходимо проверить корректность такого определения и это оставляется в качестве простого **упражнения**. Из симметричности и неотрицательной определенности функции K сразу следует, что полученная билинейная форма тоже симметрична и неотрицательно определена. Более того, она определена строго положительно, т.е. $(h,h) > 0$ при $h \neq 0$. Действительно, для симметричной неотрицательно определенной формы справедливо неравенство Шварца-Коши $(h,g)^2 \leq (h,h)(g,g)$. Значит если $(h,h) = 0$, то $(h,g) = 0$ для любой функции $g \in H_0$, в частности, $(h,\phi(x)) = h(x) = 0$ для любого $x \in \mathcal{X}$, то есть $h = 0$. А раз форма $(\cdot,\cdot)$ положительно определена, то пространство H_0 можно пополнить по определяемой ею норме. Получится гильбертово пространство $\mathbf{H}$, обещанное в теореме[28]. $\square$

Доказательство теоремы 9 уже закончено, но чтобы не было ощущения обмана, покажем еще, что построенное гильбертово пространство $\mathbf{H}$ естественно вкладывается в пространство функций $\mathcal{X} \to \mathbb{R}$. Действительно, для любых $x \in \mathcal{X}$ и $h \in H_0$

$$|h(x)| = |(h,\phi(x))| \leq \sqrt{(h,h)(\phi(x),\phi(x))} = \|h\|\sqrt{K(x,x)}$$

(опять неравенство Шварца-Коши), а значит для любой фундаментальной последовательности функций из H_0 последовательность ее значений в любой точке $x \in \mathcal{X}$ тоже фундаментальна и последовательность сходится поточечно к некоторой функции.

4.1.2 Построение ядер Мерсера

Очевидно, что

Предложение 13 *Ядрами Мерсера и соответствующими им спрямляющими пространствами и отображениями являются*

скалярное произведение в $\mathbb{R}^d$, соответствующее тождественному отображению $\phi : \mathbb{R}^d \to \mathbb{R}^d$,

произведение вещественнозначных функций $K(x,z) = \phi(x)\phi(z)$, соответствующее отображению ϕ в пространство $\mathbf{H} = \mathbb{R}$,

в частности, неотрицательные константы (отображение в точку)

[28] Связь между ядрами и гильбертовыми пространствами изучалась в первой половине XX века и почти все используемые нами результаты содержатся в работе [4] про воспроизводящие ядра (reproducing kernels). Воспроизводящее ядро для некоторого снабженного скалярным произведением $(\cdot,\cdot)$ пространства $\mathcal{F}$ числовых функций на множестве $\mathcal{X}$ — это такая функция K на $\mathcal{X} \times \mathcal{X}$, что для любой функции $f \in \mathcal{F}$ и для любой точки $x \in \mathcal{X}$ $f(x) = (K(x,\cdot), f(\cdot))$. Хотя в распознавании ядра и гильбертовы пространства имеют совершенно другую интерпретацию, формальные соотношения между ними те же, и термин "reproducing kernel" сохранился. В частности, построенное по ядру пространство $\mathbf{H}$ получило непереводимое название *RKHS (Reproducing Kernel Hilbert Space)*. В других областях математики оно иногда называется расширением Гильберта-Шмидта пространства $\mathcal{X}$ и применяется для построения обобщенных функций, когда пространство $\mathcal{X}$ само состоит из функций.

и индикаторные функции декартова квадрата подмножества (отображение в пару точек $\{0,1\}$).

□

В качестве **упражнений с подсказками** докажите еще пару предложений.

Предложение 14 *Ядра Мерсера можно строить следующими способами:*

сумма конечного числа ядер — ядро;

произведение конечного числа ядер — ядро;

композиция ядра и любого отображения (т.е. $K(\psi(x),\psi(z))$) — ядро;

если отображение $\psi : \mathcal{X}' \to 2^{\mathcal{X}}$ переводит каждый элемент пространства $\mathcal{X}'$ в конечное подмножество пространства $\mathcal{X}$ (возможно, пустое) и K — ядро на $\mathcal{X}$, то

$$K'(x',z') = \sum_{x\in\psi(x')}\sum_{z\in\psi(z')} K(x,z)$$

— ядро;

предел сходящейся последовательности, в частности, сумма сходящегося ряда ядер — ядро.

Подсказки. Симметричность и неотрицательная определенность выдерживают предельный переход (последний способ построения ядер). Для остальных способов можно предъявить спрямляющее пространство и спрямляющее отображение. Это прямое произведение (оно же — тензорная сумма) исходных спрямляющих пространств и отображений, их тензорное произведение, композиция $\phi\circ\psi$ и отображение $x' \mapsto \sum_{x\in\psi(x')}\phi(x)$, соответственно.
□

Следствие 1 *В частности, ядрами являются:*

линейные комбинации ядер с положительными коэффициентами;

интегралы по параметру от зависящих от параметра ядер;

многочлены с положительными коэффициентами от ядер;

экспоненты от ядер.

□

Еще несколько конкретных примеров ядер Мерсера в $\mathbb{R}^d$:

Предложение 15

$K(x,z) = \cos(x-z)$ *при* $x,z\in\mathbb{R}$ *— ядро (вспомните тригонометрию!);*

$K(x,z) = e^{xz}$ *при* $x,z\in\mathbb{R}^d$ *— ядро (следствие 1);*

$K(x,z) = e^{-(x-z)^2}$ *при* $x,z\in\mathbb{R}$ *— ядро (предложения 13 и 14 и предыдущий пункт);*

$K(x,z) = e^{-\|x-z\|^2}$ *при* $x,z\in\mathbb{R}^d$ *— ядро (предложение 14 и предыдущий пункт);*

$K(x,z) = e^{-\left(\frac{\|x-z\|}{\sigma}\right)^2}$ *при* $x,z\in\mathbb{R}^d$, $\sigma>0$ *— ядро (предложение 14 и предыдущий пункт);*

$K(x,z) = e^{-a\|x-z\|}$ *при* $x,z\in\mathbb{R}^d$, $a>0$ *— ядро (предыдущий пункт, вычисление интеграла*

$$\frac{a}{\sqrt{\pi}}\int_0^\infty e^{-\left(\frac{\|x-z\|}{\sigma}\right)^2} e^{-\left(\frac{a\sigma}{2}\right)^2} d\sigma = e^{-a\|x-z\|}$$

с помощью подстановки $\tau = \frac{a\sigma}{2} - \frac{\|x-z\|}{\sigma}$ и следствие 1).

□

Предпоследние три ядра называются гауссовыми, поскольку отличаются от плотности гауссова распределения лишь положительным коэффициентом и растяжением координат. Следовательно и многомерный гауссиан общего вида

$$K(x,z) = (2\pi)^{-\frac{d}{2}}|A|^{-\frac{1}{2}}e^{-\frac{1}{2}A^{-1}(x-z,x-z)}\ , \tag{144}$$

сводящийся к стандартному сферическому гауссиану

$$K(x,z) = (2\pi)^{-\frac{d}{2}}e^{-\frac{\|x-z\|^2}{2}} \tag{145}$$

линейной заменой переменных, тоже является ядром.

Для разнообразия приведем пример ядра, не являющегося неотрицательно определенным, т.е. ядром Мерсера.

Предложение 16 *Ядро на пространстве действительных чисел*

$$K(x,z) = \sigma(xz) = \frac{1}{1+e^{-xz}}$$

не является неотрицательно определенным.

Доказательство. Возьмем $x_1 = \sqrt{\ln 2}$ и $x_2 = 2\sqrt{\ln 2}$. Тогда матрица $K(x_i, x_j)$ равна

$$\begin{pmatrix} \frac{2}{3} & \frac{4}{5} \\ \\ \frac{4}{5} & \frac{16}{17} \end{pmatrix}\ ,$$

ее определитель отрицателен, значит по критерию Сильвестра матрица не неотрицательно определена, а по предложению 10 ядро — тоже. □

4.1.3 Ядра и линейная разделимость

Ядер Мерсера, построенных в разделе 4.1.2, уже вполне достаточно, чтобы гарантировать линейную разделимость любого обучающего набора в $\mathbb{R}^d$ без повторяющихся векторов. Действительно, полиномиальное ядро

$$K(x,z) = ((x,z)+c)^m\ , \quad c > 0$$

реализуется (индуцируется из скалярного произведения) полиномиальным отображением ϕ, переводящим вектор x в набор всех мономов степени $\leq m$ от координат x с некоторыми коэффициентами (**упражнение**: проверьте и посчитайте коэффициенты!), то есть сводит разделимость к полиномиальной и гарантирует разделение любых $\leq m+1$ векторов. А с гауссианом дела обстоят еще лучше: гауссово ядро переводит любые конечные наборы несовпадающих векторов в наборы линейно независимых векторов.

Предложение 17

Гауссово ядро $K(x,z) = e^{-\|x-z\|^2}$ индуцируется из скалярного произведения отображением $\phi : \mathbb{R}^d \to L^2(\mathbb{R}^d)$, переводящим вектор x в функцию

$$\phi(x) : t \mapsto \pi^{-\frac{d}{4}}e^{-\frac{\|t-2x\|^2}{2}}\ .$$

Для любого конечного набора векторов $x_1, \ldots, x_N$ функции $\phi(x_1), \ldots, \phi(x_N)$ линейно независимы.

Доказательство оставляется в качестве **упражнения**. □

Из линейной независимости следует линейная разделимость на любые два класса (предложение 2, стр. 53).

4.1.4 Ядра, расстояние и "похожесть"

Из предложений 10, 13 и 14 немедленно следует

Следствие 2 *Для любого ядра Мерсера K*

$$K_1(x,z) = \lim_{\epsilon \searrow 0} \frac{K(x,z)+\epsilon}{\sqrt{K(x,x)+\epsilon}\sqrt{K(z,z)+\epsilon}}$$
$$= \begin{cases} \frac{K(x,z)}{\sqrt{K(x,x)}\sqrt{K(z,z)}} & \text{при } K(x,x)K(z,z) \neq 0 \\ 0 & \text{при } K(x,x)K(z,z) = 0 \text{ и } K(x,x)+K(z,z) \neq 0 \\ 1 & \text{при } K(x,x) = K(z,z) = 0 \end{cases}$$

также является ядром Мерсера, причем для любых x и z $|K_1(x,z)| \leq 1$ и $K_1(x,x) = 1$.

Назовем такое ядро K_1 нормализованным. Пусть ϕ_1 и $\mathbf{H}_1$ — спрямляющее отображение и спрямляющее пространство ядра K_1. Тогда квадрат расстояния между образами точек x и z в $\mathbf{H}_1$

$$\|\phi_1(x) - \phi_(z)\|^2 = K_1(x,x) - 2K_1(x,z) + K_1(z,z) = 2 - 2K_1(x,z)$$

тем меньше, чем больше значение $K_1(x,z)$. Поэтому нормализованные ядра часто интерпретируют как "меру похожести": чем больше значение ядра на паре точек, тем больше эти точки (строго говоря — не сами точки, а их образы при спрямляющем отображении) друг на друга похожи.

Замечание. Иногда про любое ядро Мерсера K без ограничений на значения $K(x,x)$ утверждают, что оно является мерой похожести. Это — просто неаккуратность, строгий смысл которой придать нельзя. Получилось бы, что на плоскости с обычным скалярным произведением векторы $(1,11)$ и $(4,1)$ более похожи друг на друга (в 15 раз? или в $\sqrt{15}$?), чем любой вектор единичной длины сам на себя.

4.1.5 Сверточные ядра Мерсера

Для ядер, определенных на $\mathbb{R}^d$ или, вообще говоря, на любой коммутативной группе, и зависящих только от разности аргументов, как, например, ядра из предложения 15, есть дополнительные способы построения. Такие ядра называются *RBF-ядрами* или *сверточными ядрами*, поскольку при $K(x,z) = \kappa(x-z)$ интегральный оператор с ядром K (формула (141)) превращается в оператор свертки с функцией κ:

$$f \mapsto \kappa * f = \int_{\mathcal{X}} \kappa(\cdot - z) f(z) dz \ , \tag{146}$$

А свертка, как известно (а если неизвестно, легко проверить) коммутативна, ассоциативна, при преобразовании Фурье переходит в умножение, а оператор свертки с четной функцией симметричен в L^2.

Упражнение. Для функции κ, определяющей сверточное ядро на любой группе, $\kappa(0) \geq 0$ и для любой точки x $|\kappa(x)| \leq \kappa(0)$.

Осознание мелких подробностей, опущенных в следующих двух предложениях и их доказательствах, является **упражнением** по стандартному курсу функционального анализа. В частности, чтобы не было проблем со сходимостью интегралов, можно сначала считать, что все утверждения формулируются для функций из пространства Шварца (гладких функций, любая производная которых убывает на бесконечности быстрее любой степени), а потом, если нужно, обобщать их на бóльшие пространства.

Предложение 18 *Если функция κ четна, то сверточное ядро, заданное ее сверточным квадратом $\kappa * \kappa$ симметрично и неотрицательно определено. И наоборот, любое симметричное и неотрицательно определенное сверточное ядро представимо в виде сверточного квадрата.*

Доказательство. Действительно, симметричность непосредственно следует из четности κ, а неотрицательная определенность доказывается простым вычислением: для любой пробной функции f

$$(\kappa * \kappa * f, f) = (\kappa * f, \check{\kappa} * f) = (\kappa * f, \kappa * f) \geq 0 \ ,$$

где используется обозначение $\check{g}(x) = g(-x)$ и четность функции κ. Обратное утверждение следует из предложения 19. $\square$

В частности, $K(x,z) = \kappa(x-z) = \max(1-|x-z|, 0)$ является ядром на $\mathbb{R}^1$, поскольку функция κ является сверточным квадратом индикаторной функции $\chi_{[-\frac{1}{2},\frac{1}{2}]}$ отрезка $[-\frac{1}{2}, \frac{1}{2}]$.

Следствие 3 *Свертка сверточных ядер — ядро.*

Доказательство. Это немедленно следует из предложения 18 и коммутативности и ассоциативности свертки:

$$(\kappa * \kappa) * (\lambda * \lambda) = (\kappa * \lambda) * (\kappa * \lambda) \ .$$

$\square$

Предложение 19 *Если функция κ четна, то определяемое ею сверточное ядро неотрицательно определено тогда и только тогда, когда ее преобразование Фурье*[29] $\mathrm{F}[\kappa]$ *всюду неотрицательно.*

Доказательство. Действительно, если $\mathrm{F}[\kappa](\xi) \geq 0$ в любой точке ξ, то из $\mathrm{F}[\kappa](\xi)$ можно поточечно извлечь квадратный корень, применить к получившейся функции обратное преобразование Фурье и получить "сверточный квадратный корень" из функции κ (см. предложение 18). Если же неравенство $\mathrm{F}[\kappa] \geq 0$ неверно в окрестности какой-то точки ξ, то найдется такая положительная пробная функция ϕ (например, сферический гауссиан с центром в этой точке и малой дисперсией), что $(\mathrm{F}[\kappa]\phi, \phi) < 0$. Применяя к участвующим в этом неравенстве функциям обратное преобразование Фурье, получаем, что $(\kappa * \mathrm{F}^{-1}[\phi], \mathrm{F}^{-1}[\phi]) < 0$, т.е. оператор сверки с κ, а значит и ядро $\kappa(x-z)$, не являются неотрицательно определенными. Предложение доказано с точностью до подробностей, связанных с тем, что извлечение квадратного корня и преобразование Фурье могут превратить регулярную функцию из L^2 в обобщенную, а также с тем, что функция $\mathrm{F}^{-1}[\phi]$ — не вещественнозначная, и вместо скалярного произведения нужно аккуратно работать с эрмитовым. $\square$

Приведем примеры ядер и неядер, получаемых с помощью предложений 18 и 19.

[29] Есть несколько разных определений преобразования Фурье для функций $\mathbb{R}^n \to \mathbb{C}$, различающихся коэффициентами. Будем считать, что

$$\mathrm{F}[f](\xi) = \int f(x) e^{-i\xi x} dx \ ,$$

тогда обратное преобразование Фурье

$$\mathrm{F}^{-1}[\phi](x) = \frac{1}{(2\pi)^n} \int \phi(\xi) e^{i\xi x} d\xi \ ,$$

т.е. все проблемы с коэффициентами загоним в обратное преобразование.

- Преобразование Фурье от индикаторной функции $\chi_{[-a,a]}$ отрезка $[-a,a]$

$$F[\chi_{[-a,a]}](\xi) = \int_{-a}^{a} e^{-i\xi x} dx = \frac{e^{-ia\xi} - e^{ia\xi}}{-i\xi} = \frac{2\sin a\xi}{\xi}$$

— знакопеременная функция, поэтому четные сверточные степени самой функции $\chi_{[-a,a]}$ являются неотрицательно определенными сверточными ядрами, а нечетные — не являются. А ядро $\frac{\sin a(x-z)}{x-z}$ неотрицательно определено при любом $a > 0$.

- Для плотности распределения Лапласа на $\mathbb{R}^1$ $l(x) = \frac{a}{2} e^{-a|x|}$, $a > 0$ преобразование Фурье равно

$$\begin{aligned} \mathrm{F}[l](\xi) &= \frac{a}{2}\left(\int_{-\infty}^{0} e^{(a-i\xi)x} dx + \int_{0}^{\infty} e^{(-a-i\xi)x} dx\right) \\ &= \frac{a}{2}\left(\frac{1}{a-i\xi} - \frac{1}{-a-i\xi}\right) = \frac{a^2}{a^2+\xi^2} \ . \end{aligned}$$

И l, и $\mathrm{F}[l]$ всюду положительны. Значит и ядра $e^{-a|x-z|}$, и $\frac{1}{a^2+(x-z)^2}$ (положительные коэффициенты отброшены) являются неотрицательно определенными.

- Ядра из предыдущего пункта можно обобщить с одномерного случая на многомерный, причем несколькими способами. В $\mathbb{R}^d$ при $d > 1$ ядро $\frac{1}{a^2+\|x-z\|^2}$ неотрицательно определено при любой размерности d. Доказательство получаются вычислением знака преобразования Фурье (полностью вычислять преобразование Фурье трудно и незачем). Это тяжелое техническое **упражнение!**[30]. Ядро

$$e^{-a\|x-z\|_1} = e^{-a\sum_{j=1}^{d} |x^j - z^j|}$$

неотрицательно определено по предложению 14. Ядро $e^{-a\|x-z\|}$ с обычной евклидовой длиной — тоже неотрицательно определено, и вычисление тоже сложное, проще вспомнить предложение 15.

- К сверточным ядрам можно применять еще и правила построения ядер из предложения 14. В частности, из неотрицательной определенности ядра $K(x,z) = \frac{1}{a^2+\|x-z\|^2}$ следует неотрицательная определенность ядра *KMOD (Kernel with MOderate Decreasing)* [6]

$$KMOD_{a^2,b^2}(x,z) = e^{\frac{b^2}{a^2+\|x-z\|^2}} - 1 \ ,$$

равного сумме степенного ряда с положительными коэффициентами от ядра $\frac{1}{a^2+\|x-z\|^2}$.

[30] Поскольку в пространстве размерности $d > 1$ интеграл от функции $f_a^d(x) = \frac{1}{a^2+\|x\|^2}$ расходится, вычисления нужно проводить в обобщенных функциях или как-то аккуратно следить за сходимостью интегралов. А идея вычисления знака проста. В силу сферической симметрии функции f_a^d достаточно вычислить ее преобразование Фурье в точке $\xi = (\xi_1, 0, \ldots, 0)$ на оси абсцисс. Обозначим через $\hat{x}$ $(d-1)$-мерный вектор $(x^2, \ldots, x^d)$. Тогда, игнорируя проблемы сходимости интегралов,

$$\mathrm{F}[f_a^d](\xi) = \int \frac{e^{-i\xi_1 x^1}}{a^2+\|x\|^2} dx = \int \left(\int \frac{e^{-i\xi_1 x^1}}{(a^2+\|\hat{x}\|^2)+\|x^1\|^2} dx^1\right) d\hat{x}$$

Внутренний интеграл равен уже обсуждавшемуся одномерному преобразованию Фурье $\mathrm{F}[f^1_{\sqrt{a^2+\|\hat{x}\|^2}}](\xi_1)$ и всегда положителен. Значит и весь интеграл положителен.

Кроме "трудного" прямого доказательства неотрицательной определенности есть "легкое" обходное: можно воспользоваться следствием 6, стр. 104.

4.1.6 Условно-неотрицательно определенные ядра

Напомним, что ядра мы изучаем не совсем абстрактно, а для того, чтобы обучать распознаватели, линейные в спрямляющих пространствах. Но некоторые виды линейного распознавания — например, разделение двух классов плоской полосой — инвариантны относительно параллельного переноса (в спрямляющем пространстве), а используемое для обучения ядро, т.е. скалярное произведение — нет. Нет ли какого-нибудь более трансляционно-инвариантного способа обучения? Например, нельзя ли проводить обучение не в терминах скалярного произведения, а в терминах расстояния между образами точек в спрямляющем пространстве? Можно ли при этом получить какую-либо практическую пользу?

Сейчас будет построен некоторый класс ядер (симметричных функций от пары векторов признаков), больше похожих на попарное расстояние, чем на скалярное произведение. Этот класс содержит класс ядер Мерсера, пригоден для обучения линейных распознавателей большинством методов опорных векторов (не всеми, но в точности трансляционно-инвариантными) и содержит практически полезные ядра, не являющиеся ядрами Мерсера. Правда, никаких новых распознавателей по сравнению с распознавателями, обучаемыми с помощью ядер Мерсера, этот класс ядер не дает: скорее он дает дополнительные удобные способы построения ядер Мерсера, иногда неявные.

Ядра Мерсера и CPD-ядра: определения и простые соотношения

Назовем функцию $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ *условно-неотрицательно определенным ядром* (*CPD-kernel*, *conditionally positive definite kernel*), если она симметрична и для любого конечного набора точек $x_1, \ldots, x_n \in \mathcal{X}$ для любого вектора $c \in \mathbb{R}^n$ с нулевой суммой координат

$$\sum_{i=1}^{n} c^i = 0 \tag{147}$$

выполнено неравенство

$$\sum_{i,j=1}^{n} K(x_i, x_j) c^i c^j \geq 0 \ . \tag{148}$$

Для краткости будем называть условно-неотрицательно определенные ядра *CPD-ядрами*.

Очевидно, что всякое ядро Мерсера (т.е. неотрицательно определенное) является CPD-ядром. Примерами CPD-ядер, не являющихся ядрами Мерсера, могут служить функции $K(x, z) = x + z$ и $K(x, z) = -(x - z)^2$ на $\mathbb{R} \times \mathbb{R}$ (**упражнение**!). Между этими классами ядер есть и другие важные соотношения (предложения 22,25).

Из способов построения ядер Мерсера (предложения 13,14 и следствие 1) для CPD-ядер пригодны все, кроме произведения ядер (и следующих из него многочленов с положительными коэффициентами и (пока) экспонент[31]), поскольку при них условие (148) очевидным образом сохраняется.

Предложение 20 *Условно-неотрицательно определенными ядрами (CPD-ядрами) являются:*

неотрицательно определенные ядра (ядра Мерсера);

сумма CPD-ядер;

линейная комбинация CPD-ядер с положительными коэффициентами;

[31] Экспонента от CPD-ядра даже лучше, чем CPD-ядро, см. предложение 25.

предел сходящейся последовательности (в частности, сумма сходящегося ряда) CPD-ядер;

интеграл по параметру от зависящих от параметра CPD-ядер;

композиция CPD-ядра и любого отображения.

□

Доказательство следующего предложения оставляется в качестве **упражнени** для устного счета.

Предложение 21 *Любая функция вида $K(x,z) = f(x) + f(z)$ (в частности, любая константа) является CPD-ядром, причем вместо неравенства (148) при условии (147) выполняется равенство*

$$\sum_{i,j=1}^{n} K(x_i, x_j) c^i c^j = 0 \ . \tag{149}$$

□

Следствие 4 *Для любого CPD-ядра K функция*

$$K_0(x,z) = K(x,z) - \frac{1}{2}(K(x,x) + K(z,z)) \tag{150}$$

— тоже CPD-ядро. Для ядра K_0 во всех точках x и z $K_0(x,x) = 0$ и $K_0(x,z) \leq 0$.

□

Предложение 22 *Пусть $x_0 \in \mathcal{X}$ — произвольная точка. Симметричная функция $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ является CPD-ядром тогда и только тогда, когда*

$$K_+(x,z) = K(x,z) - K(x,x_0) - K(x_0,z) + K(x_0,x_0) \tag{151}$$

является ядром Мерсера.

Доказательство. Симметричность K_+ очевидна. Возьмем любой набор точек $x_1, \ldots, x_n \in \mathcal{X}$, любой вектор $c \in \mathbb{R}^n$ и положим $c^0 = -\sum_{i=1}^n c^i$. Тогда

$$\sum_{i,j=1}^{n} K_+(x_i, x_j) c^i c^j = \sum_{i,j=0}^{n} K(x_i, x_j) c^i c^j \ ,$$

так что если K — CPD-ядро, то K_+ — ядро Мерсера. Наоборот, если K_+ — ядро Мерсера, то оно является и CPD-ядром, функция

$$k_0(x,z) = K(x,x_0) + K(x_0,z) - K(x_0,x_0)$$

по предложению 21 является CPD-ядром при любой функции K, значит их сумма $K(x,z) = K_+(x,z) + k_0(x,z)$ — тоже CPD-ядро. □

Упражнение. Докажите, что ядро на пространстве действительных чисел

$$K(x,z) = \sigma(xz) = \frac{1}{1 + e^{-xz}}$$

не является CPD-ядром (ср. с предложением 16).

Геометрия пространства CPD-ядер

И ядра Мерсера, и CPD-ядра образуют конуса(!)[32] в линейном пространстве функций на $\mathcal{X}\times\mathcal{X}$. Обозначим их $\mathcal{K}_+$ и $\mathcal{K}$, соответственно (более точные, но чересчур громоздкие, обозначения — $\mathcal{K}_+(\mathcal{X})$ и $\mathcal{K}(\mathcal{X})$). Линейное пространство(!) CPD-ядер вида $K(x,z)=f(x)+f(z)$ (предложение 21) обозначим через $\mathcal{K}_=$. Конус(!) ядер с нулевой диагональю (следствие 4) обозначим через $\mathcal{K}_0$, а линейную проекцию(!) $K\mapsto K_0$ (150) — через π_0. Линейное(!) отображение $K\mapsto K_+$ (151) обозначим через π^{x_0}. Очевидно, что для любого x $K_+(x,x_0)=0$. Обозначим через $\mathcal{K}_+^{x_0}$ конус(!) ядер Мерсера, обладающих этим свойством. Следующее предложение суммирует соотношения между конусами $\mathcal{K}$, $\mathcal{K}_+$, $\mathcal{K}_+^x$, $\mathcal{K}_0$, линейным пространством $\mathcal{K}_=$ и отображениями π^x и π_0, большая часть из которых очевидна или уже доказана, а остальные проверяются тривиально.

Предложение 23 *Для любых* $x,z\in\mathcal{X}$

$\mathcal{K}_0\cap\mathcal{K}_==\{0\}$, $\mathcal{K}=\mathcal{K}_0+\mathcal{K}_=$;

$\mathcal{K}_+\cap\mathcal{K}_0=\{0\}$,

$\mathcal{K}_+^x\subset\mathcal{K}_+\subset\mathcal{K}$, $\mathcal{K}_+^x\cap\mathcal{K}_==\{0\}$;

$\pi_0:\mathcal{K}\to\mathcal{K}_0$ — *проекция со слоем* $\mathcal{K}_=$, *ее ограничение* $\pi_0|_{\mathcal{K}_+^x}$ — *биекция;*

$\pi^x:\mathcal{K}\to\mathcal{K}_+^x$ — *проекция со слоем* $\mathcal{K}_=$, *ее ограничение* $\pi^x|_{\mathcal{K}_0}$ — *биекция, обратная к* $\pi_0|_{\mathcal{K}_+^x}$;

ограничение $\pi^x|_{\mathcal{K}_+^z}$ — *биекция* $\mathcal{K}_+^z\to\mathcal{K}_+^x$, *обратная к* $\pi^z|_{\mathcal{K}_+^x}:\mathcal{K}_+^x\to\mathcal{K}_+^z$.

□

По определению CPD-ядер для любых точек $x_1,\dots,x_n\in\mathcal{X}$ каждое CPD-ядро K задает неотрицательно определенную квадратичную форму (148) на $(n-1)$-мерной гиперплоскости (147) n-мерных векторов с нулевой суммой координат.

Следствие 5 *Все отображения* π_0 *и* π^x *сохраняют эту квадратичную форму.*

Доказательство. См. предложения 21 и 23. □

Геометрия спрямляющих пространств CPD-ядер

Пусть K — CPD-ядро, $K_0=\pi_0(K)$, $K_+=\pi^{x_0}(K_0)$ для некоторой точки $x_0\in\mathcal{X}$ и $\phi:\mathcal{X}\to\mathbf{H}$ — спрямляющее отображение ядра Мерсера K_+. Будем называть ϕ и $\mathbf{H}$ спрямляющим отображением и, соответственно, спрямляющим пространством также и для CPD-ядер K и K_0.

Тогда по предложению 23 $K_0=\pi_0(K_+)$, то есть

$$\begin{aligned}K_0(x,z)&=K_+(x,z)-\frac{1}{2}(K_+(x,x)+K_+(z,z))\\&=-\frac{1}{2}\left(\|\phi(x)\|^2-2(\phi(x),\phi(z))+\|\phi(z)\|^2\right)=-\frac{1}{2}\|\phi(x)-\phi(z)\|^2\ ,\end{aligned}$$

т.е. CPD-ядро K_0 с нулевой диагональю равно квадрату евклидова расстояния в спрямляющем пространстве с (отрицательным!) коэффициентом $-\frac{1}{2}$.

Замечание. Про произвольное CPD-ядро K ничего не утверждается.

[32]Восклицательным знаком в скобках здесь и далее обозначаются очень простые утверждения, требующие, однако проверки, и эта проверка оставляется в качестве упражнения.

Построение спрямляющего пространства $\mathbf{H}$ зависело от выбора точки x_0. Пусть $x_0' \in \mathcal{X}$ — другая точка, и $K_+' = \pi^{x_0'}(K_0) = \pi^{x_0'}(K_+)$ (предложение 23) — соответствующее ядро Мерсера. Тогда

$$\begin{aligned} K_+'(x,z) &= K_+(x,z) - K_+(x,x_0') - K_+(x_0',z) + K_+(x_0',x_0') \\ &= (\phi(x),\phi(z)) - (\phi(x),\phi(x_0')) - (\phi(x_0'),\phi(z)) + (\phi(x_0'),\phi(x_0')) \\ &= (\phi(x)-\phi(x_0'),\phi(z)-\phi(x_0'))\ , \end{aligned}$$

т.е. новое ядро K_+' равно скалярному произведению в старом, построенном по ядру K_+, спрямляющем пространстве $\mathbf{H}$, только со сдвинутым в точку $\phi(x_0')$ началом координат. Так что спрямляющие пространства ядер K_+ и K_+' различаются всего лишь сдвигом начала координат.

Подробнее связь CPD-ядер с расстояниями обсуждается в работе [103]. А пока получен фундаментальный пример CPD-ядра.

Предложение 24 *Если пространство $\mathcal{X}$ — евклидово, то для любого $c > 0$ функция $K(x,z) = -c\|x-z\|^2$ является CPD-ядром на $\mathcal{X}$.*

□

Ядра Мерсера и CPD-ядра: продолжение

Предложение 25 *Функция $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ является CPD-ядром тогда и только тогда, когда для любого $t > 0$*

$$K_t(x,z) = e^{tK(x,z)} \tag{152}$$

является ядром Мерсера.

Доказательство. Ядра Мерсера K_t являются CPD-ядрами, константа -1 — тоже, а значит отношения $\frac{K_t-1}{t}$ и предел $\lim_{t\searrow 0}\frac{K_t-1}{t} = K$ — тоже CPD-ядра. В обратную сторону: построим по CPD-ядру K и произвольной точке $x_0 \in \mathcal{X}$ ядро Мерсера K_+ (формула (151)). Тогда при $t > 0$ ядра tK_+ и e^{tK_+} — тоже ядра Мерсера, а значит и

$$\begin{aligned} K_t(x,z) &= e^{tK(x,z)} = e^{t(K_+(x,z)+K(x,x_0)+K(x_0,z)-K(x_0,x_0))} \\ &= e^{tK_+(x,z)}\left(e^{tK(x,x_0)}e^{tK(z,x_0)}\right)e^{-tK(x_0,x_0)} \end{aligned}$$

по предложениям 13,14 — тоже ядро Мерсера. □

С помощью предложений 20 и 25 и технической леммы 3 можно строить новые семейства CPD-ядер и ядер Мерсера.

Лемма 3 *Для любого $x \geq 0$*

$$\begin{aligned} -x^\alpha &= \frac{\alpha}{\Gamma(1-\alpha)}\int_0^\infty (e^{-tx}-1)t^{-(\alpha+1)}dt \ \textit{при}\ \ 0<\alpha<1\ ; \\ -\ln(1+x) &= \int_0^\infty \frac{e^{-t(1+x)}-e^{-t}}{t}dt\ . \end{aligned}$$

Доказательство. При $x = 0$ равенства очевидны. При $x > 0$ и $0 < \alpha < 1$ интегралы в правых частях абсолютно сходятся. Дифференцирование уравнений по x сводит доказательство леммы к проверке тождеств

$$\begin{aligned} \int_0^\infty e^{-tx}t^{-\alpha}dt &= x^{\alpha-1}\Gamma(1-\alpha) \text{ и} \\ \int_0^\infty e^{-t(1+x)}dt &= \frac{1}{1+x}\ , \end{aligned}$$

соответственно. Второе тождество общеизвестно (и верно при любом $x > -1$), а первое эквивалентно определению гамма-функции. □

Предложение 26 *Для любого CPD-ядра K, принимающего только неположительные значения (в частности, для любого CPD-ядра с нулевой диагональю из следствия 4) CPD-ядрами являются также $-(-K)^\alpha$ при $0 < \alpha < 1$ и $-\ln(1-K)$.*

□

Следствие 6 *В евклидовом пространстве при $0 < \alpha \leq 2$ и $c > 0$ функции $K(x,z) = -c\|x-z\|^\alpha$ и $K(x,z) = -\ln(1+c\|x-z\|^\alpha)$ являются CPD-ядрами, а функции $K(x,z) = e^{-c\|x-z\|^\alpha}$ и $K(x,z) = \frac{1}{1+c\|x-z\|^\alpha}$ — ядрами Мерсера.*

Доказательство. См. предложения 24, 25, 26. □

Заметим, что ядра Мерсера $e^{-c\|x-z\|^2}$, $e^{-c\|x-z\|}$ и $\frac{1}{1+\|x-z\|^2}$ уже появлялись в разделах 4.1.2 и 4.1.5, но сейчас построено целое семейство таких ядер. Следствие 6 соблазняет попробовать строить ядра Мерсера как экспоненты от расстояний (естественно, с противоположным знаком) в неевклидовых пространствах. В статье [25] приведены примеры таких ядер, в частности, приведен пример того, что не всякое расстояние годится для построения ядра.

Большая часть приведенных утверждений про CPD-ядра — это переформулировки утверждений из книги [12] про "negative definite kernels", отличающиеся от CPD-ядер противоположным знаком.

4.2 Метод опорных векторов (SVM, SVC, SVR)

Метод опорных векторов или *SVM (Support Vector Machine)*, разработанный в серии работ В.Н.Вапника от [117] до [18, 26], — это метод обучения распознавателей, как для классификации, так и для регрессии, состоящий в решении минимизационных задач типа (143) с кусочно-линейными функциями штрафа. Идейно он вырос из классификатора с разделяющей полосой (раздел 2.2.4) и некоторые хорошие свойства его решений определяются тем, что уверенный правильный (в задаче классификации) или почти правильный (в задаче регрессии) ответ на обучающем векторе не штрафуется совсем. Вычислительно метод сводится к решению задачи квадратичной оптимизации с линейными ограничениями. Поскольку и размерность оптимизационной задачи, и число ограничений слегка превышают число N векторов обучающего набора и могут быть порядка многих тысяч, традиционные методы квадратичного программирования оказались слишком медленными и были разработаны новые.

SVM удобно формулируется для задач двухклассовой классификации и одномерной регрессии и называется *SVC (Support Vector Classification)* и *SVR (Support Vector Regression)*, соответственно. Многоклассовую классификацию и многомерную регрессию обычно сводят к серии двухклассовых классификаций и одномерных регрессий. Прямые многомерные обобщения SVM существуют, но, как правило, оказывается, что обучить один сложный многомерный распознаватель намного дольше, чем много простых одномерных.

Содержимое этого раздела, за исключением мест с явно указанными библиографическими ссылками, основано на работах [31, 22, 106].

4.2.1 Двухклассовая классификация

Рассмотрим двухклассовую классификацию с произвольным пространство признаков $\mathcal{X}$, на котором определено ядро $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, и пространством ответов

$\mathcal{Y} = \{-1, +1\}$. У этого ядра есть спрямляющие пространство $\mathbf{H}$ и отображение $\phi : \mathcal{X} \to \mathbf{H}$. Рассмотрим аналогичное задаче (99) обучение на обучающем наборе $T \in (\mathcal{X} \times \mathcal{Y})^N$ классификатора

$$f(x) = (w, \phi(x)) + b \ , \tag{153}$$

линейного в спрямляющем пространстве, с мягким зазором и функцией штрафа $E(\xi) = \max(0, \xi)$:

$$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N} \max(0, 1 - y_i f(x_i)) \to \min_{w,b} \tag{154}$$

или, что эквивалентно,

$$\begin{aligned} &\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N} \xi_i \to \min_{w,b,\xi} \\ &y_i((w, \phi(x_i)) + b) \geq 1 - \xi_i \text{ при } 1 \leq i \leq N \\ &\xi_i \geq 0 \text{ при } 1 \leq i \leq N \ . \end{aligned} \tag{155}$$

Это один из способов формализовать пожелание, чтобы на положительных обучающих векторах ($y_i = 1$) классификатор давал уверенно положительный ответ ($f(x_i) \geq 1$), а на отрицательных ($y_i = -1$) — уверенно отрицательный ($f(x_i) \leq -1$).

Непосредственно в виде (155) решать задачу минимизации может оказаться нереально, поскольку вектор w живет в очень большом, может быть даже бесконечномерном, пространстве. Но согласно простой общей теореме 7, пространство возможных решений ограничено линейной оболочкой обучающих векторов. Сейчас мы передокажем теорему 7 для случая SVC громоздким вычислением, заодно дающим дополнительную информацию о разложении w по обучающим векторам в этой конкретной задаче.

Поиск точек минимума функции (155) равносилен поиску седловых точек ее лагранжиана

$$L(w, b, \xi, \alpha) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \alpha_i (y_i((w, \phi(x_i)) + b) - 1 + \xi_i) \ , \tag{156}$$

являющихся точками минимума по w, b и ξ и максимума по α, при ограничениях

$$\xi_i \geq 0, \quad \alpha_i \geq 0 \text{ при } 1 \leq i \leq N \ . \tag{157}$$

Дифференцируя лагранжиан, получаем систему уравнений и неравенств, которым удовлетворяют седловые точки:

$$\begin{aligned} 0 = \frac{\partial L(w, b, \xi, \alpha)}{\partial w_i} &= w_i - \sum_{j=1}^{N} \alpha_j y_j (\phi(x_j))_i \\ 0 = \frac{\partial L(w, b, \xi, \alpha)}{\partial b} &= -\sum_{j=1}^{N} \alpha_j y_j \\ 0 = \frac{\partial L(w, b, \xi, \alpha)}{\partial \xi_i} &= C - \alpha_i \text{ при } \xi_i \neq 0 \\ 0 \leq \frac{\partial L(w, b, \xi, \alpha)}{\partial \xi_i} &= C - \alpha_i \text{ при } \xi_i = 0 \\ 0 = \frac{\partial L(w, b, \xi, \alpha)}{\partial \alpha_i} &= 1 - \xi_i - y_i((w, \phi(x_i)) + b) \text{ при } \alpha_i \neq 0 \\ 0 \geq \frac{\partial L(w, b, \xi, \alpha)}{\partial \alpha_i} &= 1 - \xi_i - y_i((w, \phi(x_i)) + b) \text{ при } \alpha_i = 0 \ , \end{aligned}$$

из которой с учетом ограничений (157) следует, что

$$w = \sum_{j=1}^{N} \alpha_j y_j \phi(x_j) \tag{158}$$

$$\sum_{j=1}^{N} \alpha_j y_j = 0 \tag{159}$$

$$(C - \alpha_i)\xi_i = 0 \tag{160}$$

$$0 \le \alpha_i \le C \ . \tag{161}$$

Неравенство (161) совместно с равенством (158) дает больше информации про разложение вектора w по образам обучающих векторов, чем общая теорема 7. Подставляя равенства (158–160) в лагранжиан (156), получаем функцию (двойственный лагранжиан), не зависящую не только от w, но и от b и ξ, и выражающуюся через ядро

$$\begin{aligned}
L^*(\alpha) =& \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i(y_i((w,\phi(x_i)) + b) - 1 + \xi_i) \\
=& \frac{1}{2}\|w\|^2 - \sum_{i=1}^{N}\alpha_i y_i(w,\phi(x_i)) - b\sum_{i=1}^{N}\alpha_i y_i + \sum_{i=1}^{N}\alpha_i + \sum_{i=1}^{N}(C-\alpha_i)\xi_i \\
=& \frac{1}{2}\left(\sum_{j=1}^{N}\alpha_j y_j \phi(x_j), \sum_{j=1}^{N}\alpha_j y_j \phi(x_j)\right) - \sum_{i=1}^{N}\alpha_i y_i \left(\sum_{j=1}^{N}\alpha_j y_j \phi(x_j), \phi(x_i)\right) + \sum_{i=1}^{N}\alpha \\
=& -\frac{1}{2}\sum_{i,j=1}^{N}\alpha_i\alpha_j y_i y_j (\phi(x_i),\phi(x_j)) + \sum_{i=1}^{N}\alpha_i \\
=& -\frac{1}{2}\sum_{i,j=1}^{N}\alpha_i\alpha_j y_i y_j K(x_i,x_j) + \sum_{i=1}^{N}\alpha_i \ .
\end{aligned}$$

Поиск седловой точки лагранжиана $L(w, b, \xi, \alpha)$ при ограничениях (157) почти сводится к поиску точки максимума $L^*(\alpha)$ при ограничениях (159) и (161). Из неотрицательной определенности ядра K и выпуклости ограничений следует, что максимум L^* единственен и достигается на выпуклом множестве коэффициентов α. Если матрица $K(x_i, x_j)$ невырождена, то точка максимума единственна.

"Почти сводится" вот в каком смысле. Метод множителей Лагранжа гарантирует, что α является точкой максимума двойственного лагранжиана L^* тогда и только тогда, когда существуют такие w, b и ξ, что (w, b, ξ, α) является седловой точкой лагранжиана L (при ограничениях (159,161) и (157), соответственно). Но w по α вычисляется непосредственно по формуле (158), а b нужно еще поискать. Если при каком-то i выполнены строгие неравенства $0 < \alpha_i < C$, то $\xi_i = 0$ и $y_i((w, \phi(x_i)) + b) = 1$, откуда b находится однозначно:

$$b = y_i - (w, \phi(x_i)) = y_i - \sum_{j=1}^{N}\alpha_j y_j K(x_j, x_i)) \ . \tag{162}$$

Геометрически это означает, что вектор $\phi(x_i)$ лежит на краю разделяющей полосы. В общем случае при уже вычисленном w штрафы ξ_i зависят от свободного члена b кусочно-линейно

$$\xi_i = \max(0, 1 - y_i((w, \phi(x_i)) + b))$$

и задача (155) сводится к вполне элементарной минимизации выпуклой кусочно-линейной функции с не более, чем N изломами (в точках вида (162)) от одного переменного b. Минимальное значение единственно, но достигаться оно может на целом отрезке. При этом для разных точек отрезка классификаторы (153) будут разными! Для удобства в случае неоднозначности свободного члена b будем минимизировать его абсолютную величину $|b|$. Это соответствует идеологии регуляризации по Тихонову: если регуляризатора $\|w^2\|$ в задаче (155) недостаточно, добавим еще слагаемое ϵb^2, а потом устремим ϵ к нулю. Теперь обучение SVC однозначно определяет классификаторы (153).

Таким образом, обучение классификатора в пространстве неизвестно какой размерности, определяемой ядром K, свелось к решению N-мерной задачи квадратичной минимизации с одним линейным ограничением типа равенства и N парами ограничений-неравенств типа "ящиков"

$$\frac{1}{2}\sum_{i,j=1}^{N}\alpha_i\alpha_j y_i y_j K(x_i,x_j) - \sum_{i=1}^{N}\alpha_i \to \min_{\alpha} \tag{163}$$

$$\sum_{j=1}^{N}\alpha_j y_j = 0 \tag{164}$$

$$0 \le \alpha_i \le C \text{ при } 1 \le i \le N \ ,$$

а сам классификатор имеет вид

$$f(x) = \sum_{j=1}^{N}\alpha_j y_j K(x_j,x) + b \ , \tag{165}$$

где вычисление свободного члена b описано в предыдущем абзаце. Функция f определена однозначно, несмотря на то, что множители Лагранжа α_j, — вообще говоря, неоднозначно. Линейное уравнение (164) гарантирует выпуклость минимизируемой функции (163) для любого условно-неотрицательно определенного ядра (CPD-ядра) K, что позволяет обучать SVC с любыми CPD-ядрами, а не только с ядрами Мерсера. Однако согласно следствию 5 при обучении с CPD-ядром K получится тот же классификатор, что и при обучении с любым полученным из K ядром Мерсера вида K_+ из предложения 22.

4.2.2 Опорные и другие векторы

Судя по названию метода SVM, в нем где-то должны присутствовать опорные векторы. В разделе 2.2.4 было введено понятие *опорного вектора* (стр. 58) для классификатора в пространстве признаков. Для классификаторов с ядром оно естественно возникает в спрямляющем пространстве, а оттуда переносится — просто как название — в пространство признаков, которое может быть не евклидовым, не векторным, и вообще чем угодно. А именно, из равенств и неравенств нулю производных лагранжиана (156), т.е. из бесчисленных формул (157–161), немедленно следует, что для обученного классификатора (165) обучающие "векторы" x_i распадаются на три категории:

Хорошо классифицированные, на которых классификатор дает уверенный правильный ответ "с запасом", т.е. правильного знака и больше 1 по модулю ($y_i f(x_i) > 1$). Соответствующий коэффициент $\alpha_i = 0$, т.е. наличие или отсутствие этого вектора в обучающем наборе не влияет на результат обучения. Классификатор учится не на своих успехах, а на неудачах или угрозах неудач.

Опорные (в строгом смысле), на которых классификатор дает уверенный правильный ответ, но "без запаса", т.е. правильного знака и равный 1 по модулю ($y_i f(x_i) = 1$). Образы этих векторов в спрямляющем пространстве являются опорными для разделяющей полосы $\{z : |(w, z) + b| \leq 1\}$. Этим векторам могут соответствовать любые коэффициенты $0 \leq \alpha_i \leq C$. Случай $\alpha_i = 0$ для опорных векторов нетипичен. Типичные опорные векторы имеют ненулевой коэффициент.

Плохо классифицированные, на которых классификатор дает неуверенный правильный ответ, т.е. правильного знака и меньше 1 по модулю, или даже неправильный, т.е. неправильного знака или 0 (итого $y_i f(x_i) < 1$). Этим векторам соответствуют наибольшие коэффициенты $\alpha_i = C$ и именно от них классификатор зависит сильнее всего.

А дальше происходит терминологическая подмена: плохо классифицированные обучающие векторы тоже называются опорными. То есть этим словом называются все обучающие векторы, кроме правильно классифицированных, в частности все, входящие в распознающую функцию с ненулевыми коэффициентами.

4.2.3 Проблемы обучения SVC

Численные методы оптимизации

Оказалось, что при большом числе N обучающих векторов классические методы решения задач квадратичного программирования сходятся слишком медленно. Быстрее работают методы, основанные на идее "последовательной минимальной оптимизации" *SMO (Sequential Minimal Optimization)*, см. [91]. Коэффициенты α_i обучаются итеративно, причем на каждом шаге итерации зафиксированы все коэффициенты, кроме очень маленького подмножества, разного на разных шагах, по которым ищется минимум функции (163). В силу соотношения (159) минимальное подмножество коэффициентов, которые можно пошевелить, двухэлементно. Минимизация квадратичной функции (163) от двух переменных, удовлетворяющих линейному соотношению (159) и ограничениям (161), т.е. минимизация квадратичной функции на отрезке, — это задача для средней школы. А вот организация перебора двухэлементных подмножеств, обеспечивающая достаточно быструю сходимость к минимуму, является предметом работ [91], [64] и других[33].

Эмпирический подбор параметров

В отличие от коэффициентов α_i и свободного члена b, обучение которых описано в разделе 4.2.1, положительная константа C и ядро K подбираются с помощью валидации или кросс-валидации (раздел 1.1.6). Особенно удобно для обучения SVM применять метод скользящего контроля (LOO). Обучение при методе скользящего контроля можно (и необходимо!) существенно ускорить за счет того, что переучивание методом SMO уже обученного распознавателя при замене всего лишь одного обучающего вектора другим происходит намного быстрее, чем обучение "с нуля". В частном, но очень частом случае, когда правильный (не опорный) вектор заменяется на правильный, переучивания вообще не будет.

[33] В тех случаях, когда по каким-то априорным соображениям можно обойтись классификаторами (153) с фиксированным, например, нулевым свободным членом b, условие (159) в двойственной задаче пропадает и минимизировать квадратичную функцию (163) можно покоординатно. При этом ядро K обязано быть ядром Мерсера, условно-неотрицательной определенности не достаточно.

Оценка ошибки, полученная методом скользящего контроля, дает ограничение снизу на долю опорных векторов распознавателя, обученного на всех N векторах.

Предложение 27 *Пусть f — распознаватель (165), обученный решением задачи минимизации (155) на наборе T из N векторов, f_m — распознаватель, таким же образом и с тем же значением параметра C и тем же ядром K обученный на наборе $T \setminus \{(x_m, y_m)\}$ из $N-1$ вектора. Тогда если $y_m f_m(x_m) \leq 1$ (вектор (x_m, y_m) неправильно или без избытка уверенности классифицируется распознавателем f_m), то $y_m f(x_m) \leq 1$ (вектор (x_m, y_m) является опорным для обученного распознавателя f).*

Доказательство. Предположим, что наоборот $y_m f(x_m) > 1$, т.е. классификатор f классифицирует вектор (x_m, y_m) правильно и с запасом, а значит этот вектор не является опорным. Распознаватели f_m и f задаются по формулам (153, 158) (не обязательно единственным способом) точками $\tilde{f}_m$ и $\tilde{f}$ в пространстве коэффициентов $(\alpha_1, \ldots, \alpha_N, b)$, причем и $\tilde{f}_m$, и $\tilde{f}$ принадлежат выпуклому $N-1$-мерному множеству, задаваемому условиями (159, 161) и равенством $\alpha_m = 0$. $\tilde{f}_m$ и $\tilde{f}$ являются точками минимума на этом множестве выпуклых функций

$$\begin{aligned} L_m((\alpha, b)) &= \frac{1}{2}\|w\|^2 + C \sum_{i \neq m} \max(0, 1 - y_i f(x_i)) \\ &= \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j K(x_i, x_j) + C \sum_{i \neq m} \max(0, 1 - y_i f(x_i)) \end{aligned}$$

и

$$L((\alpha, b)) = \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j K(x_i, x_j) + C \sum_{i=1}^{N} \max(0, 1 - y_i f(x_i)) \ ,$$

соответственно (это результаты подстановки равенства (158) в минимизируемую сумму (154)). На отрезке $[\tilde{f}, \tilde{f}_m] = \{\tilde{f}^t = t\tilde{f}_m + (1-t)\tilde{f}\}$, $0 \leq t \leq 1$ непрерывная функция $1 - y_m f^t(x_m)$ из отрицательной становится неотрицательной. Пусть $\tau = \min\{t \in [0,1] | y_m f^t(x_m) = 1\}$. Тогда $\tau > 0$, на отрезке $[0, \tau]$ выпуклые функции $L(\tilde{f}^t)$ и $L_m(\tilde{f}^t)$ совпадают, но $L(\tilde{f}^t)$ не убывает, так как достигает минимума при $t = 0$, а $L_m(\tilde{f}^t)$ не возрастает, так как достигает минимума при $t = 1$. Значит выпуклая функция $L_m(\tilde{f}^t)$ постоянна на отрезке $[0, \tau]$, а значит и на всем отрезке $[0, 1]$. Следовательно, поскольку L_m строго выпукла по $\|w\|$, а свободный член обученного классификатора его коэффициентами и минимальностью $|b|$ определен однозначно, классификаторы f и f_m совпадают как линейные функции. Но $y_m f_m(x_m) \leq 1 < y_m f(x_m)$. Противоречие. □

Из предложения 27 немедленно следует теорема

Теорема 10 *Если у распознавателя, обученного на N векторах, имеется n опорных векторов, то оценка средней ошибки методом скользящего контроля, не превосходит $\frac{n}{N}$.*

□

Обучение с предписанной долей опорных векторов (ν-SVC)

Теорема 10 наводит на мысль, что вместо экспериментального подбора параметра C, минимизирующего среднюю ошибку валидации, хорошо бы сразу минимизировать количество опорных векторов распознавателя. Такой способ обучения распознавателей есть, он имеет такую же вычислительную сложность,

как и обучение при фиксированном C, но полностью от экспериментирования не избавляет.

Идея обучения состоит в том, чтобы, не нарушая обобщенных условий теоремы 7 (см. замечание после нее), изменить функцию ошибки и регуляризатор так, чтобы минимизируемая при обучении функция явно зависела от количества опорных векторов. Пропуская промежуточные вычисления, приведем сразу результат.

Предложение 28 *Пусть при некотором значении параметра $\nu \in [0,1]$ набор $(w', b', \xi', \rho') \in \mathbf{H} \times \mathbb{R} \times \mathbb{R}^N \times \mathbb{R}$ является решением задачи*

$$\begin{aligned} &\frac{1}{2}\|w\|^2 + \sum_{i=1}^{N}(\xi_i - \nu\rho) \to \min_{w,b,\xi,\rho} \\ &y_i((w, \phi(x_i)) + b) \geq \rho - \xi_i \text{ при } i = 1, \ldots, N \\ &\xi_i \geq 0 \text{ при } i = 1, \ldots, N \\ &\rho \geq 0\ . \end{aligned} \tag{166}$$

Тогда если $\rho' > 0$, то $(\frac{w'}{\rho'}, \frac{b'}{\rho'}, \frac{\xi'}{\rho'})$ является решением задачи (155) при $C = \frac{1}{\rho'}$, и для этого решения доля плохо классифицированных векторов среди всех обучающих векторов не превосходит ν, а доля всех опорных векторов не меньше ν. А если решения с $\rho' > 0$ нет, то при любом $C \geq 0$ доля опорных векторов в решении задачи (155) больше ν.

Доказательство оставляется в качестве **упражнения**. Задача (166) является задачей квадратичной минимизации с линейными ограничениями и решается аналогично задаче (155). Выписывается лагранжиан

$$L(w, b, \xi, \rho, \alpha) = \frac{1}{2}\|w\|^2 + \sum_{i=1}^{N}(\xi_i - \nu\rho) - \sum_{i=1}^{N}\alpha_i(y_i((w, \phi(x_i)) + b) - \rho + \xi_i) \tag{167}$$

(ср. с формулой (156)), приравниваются нулю его производные, выписывается двойственный лагранжиан, получается минимизационная задача для поиска множителей Лагранжа

$$\begin{aligned} &\frac{1}{2}\sum_{i,j=1}^{N}\alpha_i\alpha_j y_i y_j K(x_i, x_j) \to \min_{\alpha} \\ &\sum_{j=1}^{N}\alpha_j y_j = 0 \\ &\sum_{i=1}^{N}\alpha_i \leq \nu N \\ &0 \leq \alpha_i \leq 1 \text{ при } 1 \leq i \leq N \end{aligned} \tag{168}$$

(ср. с задачей (163)), через найденные множители Лагранжа по формуле (158) выражается вектор w, подставляется в задачу (166) и она превращается в задачу линейной оптимизации относительно b, ρ и ξ. Подробности также оставляются в качестве **упражнения**.

Таким образом подбор параметра C, минимизирующего среднюю ошибку валидации иди кросс-валидации, заменился на подбор минимального параметра ν, при котором задача (166) имеет решение с положительным порогом ρ.

Обучение семейств распознавателей

Хотя традиционно параметр C подбирается эмпирически, на самом деле можно обучить сразу все семейство распознавателей при всех C (но фиксированном ядре K). Эта возможность обеспечивается следующей теоремой и экспериментальным наблюдением (см. [48]).

Теорема 11 *Существуют семейства $\alpha(C)$ и $b(C)$ коэффициентов и свободного члена обученного распознавателя (165), кусочно-линейно зависящие от параметра C.*

Замечание. Квантор существования в теореме необходим, поскольку при вырожденной матрице $K(x_i, x_j)$ и коэффициенты α, находятся по C, вообще говоря, неоднозначно. Несмотря на неоднозначность α, распознаватель (165) определен однозначно.

Экспериментальное наблюдение. Количество изломов этих кусочно-линейных функций с ростом числа N обучающих векторов растет не намного быстрее, чем линейно.

Доказательство. У N-мерного куба $0 \le \alpha_i \le C$ имеется 3^N открытых граней разных размерностей. Каждая грань определяется разбиением S множества индексов $\{1, \ldots, N\}$ на три подмножества $S_{=0}$ $S_{=C}$ и $S_{\ne}$ индексов i, в которых $\alpha_i = 0$, $\alpha_i = C$ и $0 < \alpha_i < C$, соответственно. При каждом значении параметра $C > 0$ каждое решение минимизационной задачи (163) принадлежит какой-то грани S, а значит, является решением задачи

$$\frac{1}{2}\sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j K(x_i, x_j) - \sum_{i=1}^{N} \alpha_i \to \min_{\alpha} \tag{169}$$

$$\sum_{j=1}^{N} \alpha_j y_j = 0 \tag{170}$$

$$\alpha_i = 0 \text{ при } i \in S_{=0} \tag{171}$$

$$\alpha_i = C \text{ при } i \in S_{=C} \tag{172}$$

$$0 < \alpha_i < C \text{ при } i \in S_{\ne}\ , \tag{173}$$

ограниченной на грань S. Эти решения (точки минимума квадратичной функции (169) на плоскости, задаваемой уравнениями (170,171,172), удовлетворяющие неравенствам (173)) являются решениями системы линейных уравнений и неравенств относительно α и C, а значит образуют (открытый) выпуклый многогранник какой-то размерности в $(N+1)$-мерном пространстве. Его проекция на координатную прямую C является открытым интервалом I_S, конечным или бесконечным. Выбросим из луча $\{C > 0\}$ все концы всех интервалов I_S, т.е. некоторое конечное множество $C_1 < \ldots < C_m$. Оставшееся открытое множество является объединением конечного числа непересекающихся интервалов $I_1, \ldots, I_{m+1}$ (последний, I_{m+1} — луч).

Минимум выпуклой квадратичной функции (163) в замкнутом кубе непрерывно зависит от C и равен минимуму из ее минимумов по тем открытым граням куба, на которых этот минимум достигается. При каждом C среди таких открытых граней есть одна грань $S(C)$ максимальной размерности, а именно, внутренность выпуклой оболочки всех таких граней. Для всех C из одного и того же интервала I_i эта грань — одна и та же. Действительно, если при $C' \in I_i$ минимум функции (163) достигается внутри грани $S(C')$, а при $C'' \in I_i$ — внутри грани $S(C'')$, то при каком-то промежуточном значении C он достигается и внутри $S(C')$, и внутри $S(C'')$, а значит и внутри их выпуклой оболочки S, имеющей большую размерность, чем по крайней мере одна из граней $S(C')$ и

$S(C'')$. А раз внутри S достигается минимум при каком-то значении $C \in I_i$, то по построению интервала I_i он достигается и при любом C, в частности, при C' и C''. При этом он равен минимуму из минимумов по $S(C')$ и $S(C'')$, т.е. глобальному минимуму. Получаем противоречие с предположением о максимальности $S(C')$ и $S(C'')$. Полученную максимальную грань для интервала I_i обозначим через $S(I_i)$.

Докажем, что для правого конца C_i интервала I_i множество $S(C_i)$ либо совпадает с $S(I_i)$, либо является его гранью (строго говоря, гранью замыкания); то же верно и для левых концов интервалов и так же доказывается. Действительно, из соображений непрерывности при $C = C_i$ в замыкании $S(I_i)$ есть точка (α_i) глобального минимума функции (163). Если минимум достигается еще в какой-то точке α' вне замыкания $S(I_i)$, то функция (163) постоянна на отрезке $t\alpha' + (1-t)\alpha_i$, а значит вектор $\alpha' - \alpha_i$ во-первых, трансверсален к плоскости грани $S(I_i)$, и во-вторых, аннулируется квадратичной формой (163). Но тогда при любом $C \in I_i$ для каждой точки минимума $\alpha \in S(I_i)$ точками минимума будут также все точки пересечения прямой $\alpha + t(\alpha' - \alpha_i)$ с кубом, а значит грань $S(I_i)$ не была максимальной.

Теперь можно строить кусочно-линейную (вектор-)функцию $\alpha(C)$. $\alpha(0) = 0$, $\alpha(C_i) = \alpha_i$ в обозначениях предыдущего абзаца, на каждый интервал I_i при $i \leq m$ она продолжается линейно, двигаясь при этом внутри грани $S(I_i)$, а на луч I_{m+1} — константой.

Для граней S_i положительной размерности свободный член b однозначно и кусочно-линейно зависит от α (формула (162)), а значит и от C. Для граней размерности 0, т.е. вершин куба, свободный член вычисляется по-другому (см. абзац после формулы (162)). Завершение доказательства теоремы в этом случае оставляется в качестве **упражнения**. $\square$

Доказательство было бы намного проще, если бы была гарантирована единственность $\alpha(C)$, т.е. невырожденность матрицы $K(x_i, x_j)$. Если в обучающем наборе допускаются одинаковые векторы признаков, матрица обязательно вырождена. В противном случае при некоторых ядрах, например, при гауссовом, матрица всегда невырождена.

Теорема 11 не говорит ничего хорошего про скорость вычисления кусочно-линейных функций $\alpha(C)$ и $b(C)$, поскольку граней у N-мерного куба экспоненциально много. В работе [48] приводится алгоритм обучения, работающий при условии невырожденности матрицы $K(x_i, x_j)$ и вычисляющий эти при функции "слева направо" (начиная с $C = 0$). Экспериментально замечено, что на практике линейных кусков в этих кусочно-линейных функциях немного, и этот алгоритм по скорости сравним с однократным обучением распознавателя при параметре C общего положения.

4.2.4 Регрессия

Перенесем упомянутое в разделе 2.1.3 обучение линейной регрессии с квадратичной регуляризацией и линейной ϵ-нечувствительной ошибкой на случай ядер и рассмотрим подробнее. Пространство ответов $\mathcal{Y} = \mathbb{R}$, пространство признаков $\mathcal{X}$ произвольно, на нем определено ядро $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, у этого ядра есть спрямляющие пространство $\mathbf{H}$ и отображение $\phi : \mathcal{X} \to \mathbf{H}$. На обучающем наборе $T \in (\mathcal{X} \times \mathcal{Y})^N$ будем обучать распознаватель

$$f(x) = (w, \phi(x)) + b \tag{174}$$

(в точности такой же, как и классификатор (153)), линейный в спрямляющем пространстве. Обучать будем минимизируя взвешенную сумму регуляризатора $\frac{1}{2}\|w\|^2$ и суммарной ϵ-нечувствительной ошибки $E(r, y) = \max(0, |y - r| - \epsilon)$ на

обучающем наборе:

$$\frac{1}{2}\|w\|^2+C\sum_{i=1}^{N}E(f(x_i),y_i)=\frac{1}{2}\|w\|^2+C\sum_{i=1}^{N}\max(0,|(w,\phi(x_i))+b-y_i|-\epsilon)\to\min_{w,b}, \tag{175}$$

а оптимизацию параметров $C>0$ и $\epsilon>0$ обсудим позже. Вводя по две дополнительных переменных ξ_i^+ и ξ_i^- на каждый обучающий вектор, которые у обученного распознавателя должны принимать значения $\max(0,y_i-(w,\phi(x_i))-b-\epsilon)$ и $\max(0,(w,\phi(x_i))+b-y_i-\epsilon)$, соответственно, задачу обучения регрессии (175) можно переформулировать в виде, аналогичном задаче обучения классификации (155):

$$\begin{aligned}&\frac{1}{2}\|w\|^2+C\sum_{i=1}^{N}(\xi_i^++\xi_i^-)\to\min_{w,b,\xi} \\ &-\epsilon-\xi_i^-\le y_i-((w,\phi(x_i))+b)\le\epsilon+\xi_i^+ \text{ при } 1\le i\le N \\ &\xi_i^+\ge 0,\quad \xi_i^-\ge 0 \text{ при } 1\le i\le N,\end{aligned} \tag{176}$$

причем очевидно, что в точках минимума из любой пары переменных ξ_i^+ и ξ_i^- по крайней мере одна переменная равна 0. Из теоремы 7 следует, что у обученного распознавателя вектор w принадлежит линейной оболочке векторов $\phi(x_i)$, но, как и при обучении классификации, применим метод множителей Лагранжа и переход к двойственному лагранжиану, чтобы больше узнать о конкретном виде обученного распознавателя. Повторим соответствующие вычисления из раздела 4.2.1 почти дословно.

Поиск точек минимума функции (176) равносилен поиску седловых точек ее лагранжиана

$$\begin{aligned}L(w,b,\xi,\alpha)=&\frac{1}{2}\|w\|^2+C\sum_{i=1}^{N}(\xi_i^++\xi_i^-)-\sum_{i=1}^{N}\alpha_i^+(\xi_i^+-(y_i-(w,\phi(x_i))-b-\epsilon)) \\ &-\sum_{i=1}^{N}\alpha_i^-(\xi_i^--((w,\phi(x_i))+b-y_i-\epsilon)),\end{aligned} \tag{177}$$

являющихся точками минимума по w, b и ξ и максимума по α, при ограничениях

$$\xi_i^+\ge 0,\quad \xi_i^-\ge 0,\quad \alpha_i^+\ge 0,\quad \alpha_i^-\ge 0 \text{ при } 1\le i\le N. \tag{178}$$

Дифференцируя лагранжиан, получаем систему уравнений и неравенств, которым удовлетворяют седловые точки:

$$\begin{aligned}0=\frac{\partial L(w,b,\xi,\alpha)}{\partial w_i} &= w_i-\sum_{j=1}^{N}(\alpha_j^+-\alpha_j^-)(\phi(x_j))_i \\ 0=\frac{\partial L(w,b,\xi,\alpha)}{\partial b} &= -\sum_{j=1}^{N}(\alpha_j^+-\alpha_j^-) \\ 0=\frac{\partial L(w,b,\xi,\alpha)}{\partial \xi_i^*} &= C-\alpha_i^* \text{ при } \xi_i^*\ne 0 \\ 0\le\frac{\partial L(w,b,\xi,\alpha)}{\partial \xi_i^*} &= C-\alpha_i^* \text{ при } \xi_i^*=0 \\ 0=\frac{\partial L(w,b,\xi,\alpha)}{\partial \alpha_i^+} &= (w,\phi(x_i))+b-y_i-\epsilon-\xi_i^+ \text{ при } \alpha_i^+\ne 0\end{aligned}$$

$$
\begin{aligned}
0 \geq \frac{\partial L(w,b,\xi,\alpha)}{\partial \alpha_i^+} &= (w,\phi(x_i)) + b - y_i - \epsilon - \xi_i^+ \text{ при } \alpha_i^+ = 0 \\
0 = \frac{\partial L(w,b,\xi,\alpha)}{\partial \alpha_i^-} &= y_i - (w,\phi(x_i)) - b - \epsilon - \xi_i^- \text{ при } \alpha_i^- \neq 0 \\
0 \geq \frac{\partial L(w,b,\xi,\alpha)}{\partial \alpha_i^-} &= y_i - (w,\phi(x_i)) - b - \epsilon - \xi_i^- \text{ при } \alpha_i^- = 0 \ ,
\end{aligned}
$$

где верхний индекс "$*$" означает "$+$" или "$-$". Из этой системы с учетом ограничений (178) следует, что

$$
w = \sum_{j=1}^{N} (\alpha_j^+ - \alpha_j^-)\phi(x_j) \tag{179}
$$

$$
\sum_{j=1}^{N} (\alpha_j^+ - \alpha_j^-) = 0 \tag{180}
$$

$$
(C - \alpha_i^*)\xi_i^* = 0 \tag{181}
$$

$$
0 \leq \alpha_i^* \leq C \ . \tag{182}
$$

Подставляя равенства (179–181) в лагранжиан (177), получаем функцию (двойственный лагранжиан), не зависящую не только от w, но и от b и ξ, и выражающуюся через ядро

$$
\begin{aligned}
L^*(\alpha) = & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}(\xi_i^+ + \xi_i^-) - \sum_{i=1}^{N}\alpha_i^+(\xi_i^+ - (y_i - (w,\phi(x_i)) - b - \epsilon)) \\
& - \sum_{i=1}^{N}\alpha_i^-(\xi_i^- - ((w,\phi(x_i)) + b - y_i - \epsilon)) \\
= & \frac{1}{2}\|w\|^2 - \sum_{i=1}^{N}(\alpha_i^+ - \alpha_i^-)(w,\phi(x_i)) - b\sum_{i=1}^{N}(\alpha_i^+ - \alpha_i^-) + \sum_{i=1}^{N}(\alpha_i^+ - \alpha_i^-)y_i \\
& - \epsilon\sum_{i=1}^{N}(\alpha_i^+ + \alpha_i^-) + \sum_{i=1}^{N}((C - \alpha_i^+)\xi_i^+ + (C - \alpha_i^-)\xi_i^-) \\
= & \frac{1}{2}\left(\sum_{j=1}^{N}(\alpha_j^+ - \alpha_j^-)\phi(x_j), \sum_{j=1}^{N}(\alpha_j^+ - \alpha_j^-)\phi(x_j)\right) \\
& - \sum_{i=1}^{N}(\alpha_i^+ - \alpha_i^-)\left(\sum_{j=1}^{N}(\alpha_j^+ - \alpha_j^-)\phi(x_j), \phi(x_i)\right) \\
& + \sum_{i=1}^{N}(\alpha_i^+ - \alpha_i^-)y_i - \epsilon\sum_{i=1}^{N}(\alpha_i^+ + \alpha_i^-) \\
= & -\frac{1}{2}\sum_{i,j=1}^{N}(\alpha_i^+ - \alpha_i^-)(\alpha_j^+ - \alpha_j^-)(\phi(x_i),\phi(x_j)) + \sum_{i=1}^{N}(\alpha_i^+ - \alpha_i^-)y_i - \epsilon\sum_{i=1}^{N}(\alpha_i^+ + \alpha \\
= & -\frac{1}{2}\sum_{i,j=1}^{N}(\alpha_i^+ - \alpha_i^-)(\alpha_j^+ - \alpha_j^-)K(x_i,x_j) + \sum_{i=1}^{N}(\alpha_i^+ - \alpha_i^-)y_i - \epsilon\sum_{i=1}^{N}(\alpha_i^+ + \alpha_i^-)
\end{aligned}
$$

Поиск седловой точки лагранжиана $L(w,b,\xi,\alpha)$ при ограничениях (178) почти сводится к поиску точки максимума $L^*(\alpha)$ при ограничениях (180) и (182). Из линейного условия (180), условно-неотрицательной определенности ядра K и

выпуклости ограничений (182) следует, что максимум L^* единственен и достигается на выпуклом множестве коэффициентов α. Если матрица $K(x_i, x_j)$ невырождена, то точка максимума единственна, несмотря на то, что количество коэффициентов α вдвое превышает размер матрицы (это будет показано ниже, см. абзац с формулой (186)).

"Почти сводится", как и в случае классификации, означает, что вектор w по точке максимума α двойственного лагранжиана L^* вычисляется непосредственно по формуле (179), а свободный член b нужно еще поискать. Если при каком-то i выполнены строгие неравенства $0 < \alpha_i^+ < C$, то $\xi_i^+ = 0$ и $y_i = (w, \phi(x_i)) + b - \epsilon$, откуда b находится однозначно:

$$b = y_i - (w, \phi(x_i)) + \epsilon = y_i - \sum_{j=1}^{N} (\alpha_j^+ - \alpha_j^-) K(x_j, x_i)) + \epsilon \ . \tag{183}$$

Геометрически это означает, что точка $(\phi(x_i), y_i) \in \mathbf{H} \times \mathbb{R}$ лежит на нижнем (относительно сомножителя $\mathbb{R}$) краю полосы $|y - f(x)| \leq \epsilon$. Аналогично рассматривается случай $0 < \alpha_i^- < C$. В общем случае при уже вычисленном w штрафы ξ_i^* зависят от свободного члена b кусочно-линейно и задача (176) сводится к вполне элементарной минимизации выпуклой кусочно-линейной функции с не более, чем $2N$ изломами от одного переменного b. Минимальное значение единственно, но достигаться оно может на целом отрезке. Как и в случае классификации, для определенности можно минимизировать $|b|$.

Таким образом, обучение регрессии в пространстве неизвестно какой размерности, определяемой ядром K, свелось к решению $2N$-мерной задачи квадратичной минимизации с одним линейным ограничением типа равенства и N парами ограничений-неравенств типа "ящиков"

$$\frac{1}{2} \sum_{i,j=1}^{N} (\alpha_i^+ - \alpha_i^-)(\alpha_j^+ - \alpha_j^-) K(x_i, x_j) - \sum_{i=1}^{N} (\alpha_i^+ - \alpha_i^-) y_i + \epsilon \sum_{i=1}^{N} (\alpha_i^+ + \alpha_i^-) \to \min_{\alpha} \tag{184}$$

$$\sum_{j=1}^{N} (\alpha_j^+ - \alpha_j^-) = 0$$
$$0 \leq \alpha_i^+ \leq C, \quad 0 \leq \alpha_i^- \leq C \text{ при } 1 \leq i \leq N \ ,$$

а сама регрессия имеет вид

$$f(x) = \sum_{j=1}^{N} (\alpha_j^+ - \alpha_j^-) K(x_j, x) + b \ , \tag{185}$$

где вычисление свободного члена b описано в предыдущем абзаце.

В точках минимума функции (184) из любой пары переменных α_i^+ и α_i^- по крайней мере одна переменная равна 0 (иначе, вычитая из них по $\min(\alpha_i^+, \alpha_i^-)$, можно уменьшить сумму (184)). Следовательно $\alpha_i^+ + \alpha_i^- = |\alpha_i^+ - \alpha_i^-|$ и, обозначая $\alpha_i^+ - \alpha_i^-$ через β_i, получаем, что задача (184) эквивалентна задаче

$$\frac{1}{2} \sum_{i,j=1}^{N} \beta_i \beta_j K(x_i, x_j) - \sum_{i=1}^{N} \beta_i y_i + \epsilon \sum_{i=1}^{N} |\beta_i| \to \min_{\beta} \tag{186}$$
$$\sum_{j=1}^{N} \beta_j = 0$$
$$-C \leq \beta_i \leq C \text{ при } 1 \leq i \leq N \ ,$$

вдвое меньшей размерности. Решать ее неудобно из-за негладкости функции (186), зато сразу видна единственность решения при невырожденной матрице $K(x_i, x_j)$. Действительно, при этом минимизируется сумма строго выпуклой квадратичной и выпуклой кусочно-линейной функции от N переменных в выпуклом множестве — кубе с центром в нуле и стороной $2C$.

4.2.5 Регрессия и классификация

Технические подробности решения задачи квадратичного программирования (184), за исключением подбора разумного значения допустимой погрешности регрессии ϵ, аналогичны рассмотренным в разделе 4.2.3 для случая классификации. Ниже описаны только некоторые интересные отличия.

Опорные и другие векторы

Геометрию регрессии методом опорных векторов, уже упомянутую при обсуждении формулы (176), удобно описывать в пространстве $\mathbf{H} \times \mathbb{R}$. Она похожа на геометрию классификации, рассмотренную в разделе 4.2.2, но роль разделяющей полосы $|(w, \phi(x)) + b| \leq 1$ в пространстве $\mathbf{H}$ играет полоса допустимой ошибки $|y - ((w, \phi(x)) + b)| \leq \epsilon$ в $\mathbf{H} \times \mathbb{R}$, и немного меняются произносимые слова. Точки $(\phi(x), y)$, или, с некоторой вольностью, их прообразы (x, y), лежащие внутри полосы, считаются правильными, лежащие на краю полосы — опорными (в строгом геометрическом смысле), а лежащие вне полосы — ошибочными (и опорными в широком смысле). Причем опорные точки бывают верхними и нижними. Итого, для обучающих векторов имеется пять вариантов, свойства которых следуют из равенств и неравенств от (178) до (182) и невозможности ошибиться и вверх, и вниз одновременно.

Нижние ошибочные $y_i - f(x_i) < -\epsilon$, $\alpha_i^+ = 0$, $\alpha_i^- = C$.

Нижние опорные (геометрически) $y_i - f(x_i) = -\epsilon$, $0 \leq \alpha_i^- \leq C$, $\alpha_i^+ = 0$.

Правильные $-\epsilon < y_i - f(x_i) < \epsilon$, $\alpha_i^+ = 0$, $\alpha_i^- = 0$.

Верхние опорные (геометрически) $y_i - f(x_i) = \epsilon$, $0 \leq \alpha_i^+ \leq C$, $\alpha_i^- = 0$.

Верхние ошибочные $y_i - f(x_i) > \epsilon$, $\alpha_i^+ = C$, $\alpha_i^- = 0$.

Обучение с предписанной долей опорных векторов (ν-*SVR*)

Если про величину допустимой погрешности нет никаких содержательных соображений, ее можно подбирать (обучать) исходя из желаемого количества опорных векторов. Аналогично предложению 27, оценивающему долю опорных векторов классификатора среди обучающих векторов, при регрессии вместо задачи (176) с фиксированной погрешностью ϵ можно решать задачу

$$\begin{aligned}
&\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}(\xi_i^+ + \xi_i^- + \nu\epsilon) \to \min_{w,b,\epsilon,\xi} \\
&-\epsilon - \xi_i^- \leq y_i - ((w, \phi(x_i)) + b) \leq \epsilon + \xi_i^+ \text{ при } 1 \leq i \leq N \\
&\xi_i^+ \geq 0, \quad \xi_i^- \geq 0 \text{ при } 1 \leq i \leq N \\
&\epsilon \geq 0
\end{aligned} \tag{187}$$

при $\nu \in [0, 1]$.

Замечание. В отличие от классификации, когда задание допустимой доли ошибок ν определяло параметр регуляризации C (предложение 28), при обучении регрессии параметр C придется подбирать.

Предложение 29 *Если набор* $(w', b', \xi', \epsilon') \in \mathbf{H} \times \mathbb{R} \times \mathbb{R}^N \times \mathbb{R}$ *является решением задачи (187), то* (w', b', ξ') *очевидным образом является решением задачи (176) при* $\epsilon = \epsilon'$*, и для этого решения доля ошибочных векторов среди всех обучающих векторов не превосходит* ν*, а доля всех опорных векторов не меньше* ν*.*

Доказательство предложения и исследование задачи (187) повторяют исследование задачи (166) и оставляются в качестве **упражнений**.

Сведение регрессии к классификации

Применение метода опорных векторов к задачам двухклассовой классификации (раздел 4.2.1) и регрессии (раздел 4.2.4) очень похоже, и возникает желание не решать похожие задачи независимо, а научиться сводить одну к другой. Но это не вполне получается так как у классификации и у регрессии существенно разные функции ошибки.

Наивная попытка рассматривать классификацию как частный случай регрессии и прямо на обучающем наборе T классификации с ответами $y_i = \pm 1$ обучать регрессию, предсказывающую эти ответы, проваливается. При $\epsilon \geq 1$ регрессия обучается предсказывать константу (любую) на отрезке $[1-\epsilon, \epsilon-1]$, а при $\epsilon < 1$ обучение регрессии выглядит разумнее, и вроде бы действительно учится классифицировать, но зачем-то штрафует уверенные правильные ответы $(y_i f(x_i) > 1 + \epsilon)$ наряду с неправильными.

Наоборот, регрессию к классификации свести можно, хотя и несколько криво. Зафиксируем число $\epsilon > 0$ и обозначим $y^+ = y - 2\epsilon$ и $y^- = y + 2\epsilon$. Построим по обучающему набору $T = ((x_1, y_1), \ldots, (x_N, y_N))$ вдвое больший обучающий набор $T_\epsilon = (((x_1, y_1^+), +1), \ldots, ((x_N, y_N^+), +1), ((x_1, y_1^-), -1), \ldots, ((x_N, y_N^-), -1))$, а по спрямляющему отображению $\phi : \mathcal{X} \to \mathbf{H}$ ядра K — спрямляющее отображение $\phi \times I_1 : \mathcal{X} \times \mathcal{Y} \to \mathbf{H} \times \mathbb{R}$ ядра $K'((x, y), (x', y')) = K(x, x') + yy'$. Будем обучать на наборе T_ϵ классификатор вида

$$f((x, y)) = (w, \phi(x)) - uy + b$$

в пространстве $\mathcal{X} \times \mathcal{Y}$, решая задачу

$$\begin{aligned} &\frac{1}{2}(\|w\|^2 + u^2) + C\sum_{i=1}^{N}(\xi_i^+ + \xi_i^-) \to \min_{w,u,b,\xi} \\ &((w, \phi(x_i)) - uy^+ + b) \geq 1 - \xi_i^+ \text{ при } 1 \leq i \leq N \\ &-((w, \phi(x_i)) - uy^- + b) \geq 1 - \xi_i^- \text{ при } 1 \leq i \leq N \\ &\xi_i^+ \geq 0, \quad \xi_i^- \geq 0 \text{ при } 1 \leq i \leq N\ , \end{aligned} \tag{188}$$

являющуюся частным случаем задачи (155). Это не совсем то, что обучать регрессию (174) на наборе T, решая задачу (176), но похоже. При $u = 1$ приближение $|(w, \phi(x)) + b - y| \leq \epsilon$ эквивалентно разделению точек y^+ и y^- полосой $|(w, \phi(x)) - uy + b| \leq \epsilon$. Остается избавиться от ϵ в правой части последнего неравенства, что легко сделать заменой переменных, и обеспечить равенство $u = 1$, что честно в рамках метода опорных векторов сделать нельзя. Точное утверждение, проверяемое прямыми вычислениями (**упражнение!**), таково.

Предложение 30 *Если* (w', u', b', ξ') *— решение задачи (188) обучения классификатора на наборе* T_ϵ*, то* $(\frac{\epsilon' w'}{u'}, \frac{\epsilon' b}{u'}, \frac{\epsilon' \xi}{u'})$ *— решение задачи обучения регрессии (176) на наборе* T *при* $C = \epsilon' C'$ *и* $\epsilon = \epsilon'(2 - \frac{1}{u'})$*. Положительность* ϵ *не гарантирована.*

4.2.6 Многоклассовая классификация

Обобщение метода опорных векторов на q-классовую классификацию описано в [121]. Идея немного напоминает многоклассовый дискриминант Фишера (раздел 2.2.1). Для каждого класса строится своя оценочная функция, так чтобы попарные разности этих функция хорошо попарно разделяли классы.

В такой идеологии вместо обычного двухклассового классификатора (153) выписываются оценочные функции

$$f^1(x) = (w^1, \phi(x)) + b^1$$

и

$$f^{-1}(x) = (w^{-1}, \phi(x)) + b^{-1}$$

и дискриминант $f^1 - f^{-1}$, разделяющий классы. Поскольку дискриминант не меняется при одновременном прибавлении к f^1 и f^{-1} одной и той же функции, можно считать, что $f^{-1} = -f^1$. Задача обучения классификатора вместо (155) теперь выглядит так:

$$\begin{aligned}
&\frac{1}{2}(\|w^1\|^2 + \|w^{-1}\|^2) + C\sum_{i=1}^{N}(\xi_i^1 + \xi_i^{-1}) \to \min_{w,b,\xi} \qquad (189\\
&((w^{y_i}, \phi(x_i)) + b^{y_i}) - ((w^j, \phi(x_i)) + b^j) \geq 2 - \xi_i^j \text{ при } 1 \leq i \leq N,\ j \in \{-1, 1\},\ j \neq y\\
&\xi_i^j \geq 0 \qquad\qquad \text{при } 1 \leq i \leq N,\ j \in \{-1, 1\}\\
&w^1 + w^{-1} = 0, \quad b^1 + b^{-1} = 0\ .
\end{aligned}$$

Она эквивалентна задаче (155) при $w = w^1 = -w^{-1}$, $b = b^1 = -b^{-1}$, $\xi_i = \frac{1}{2}\xi_i^{-y_i}$ и $\xi_i^{y_i} = 0$.

Такую задачу уже понятно, как обобщить на много классов. Пусть y_i теперь равно не ± 1, а номеру класса, которому принадлежит обучающий вектор x_i.

$$\begin{aligned}
&\frac{1}{2}\sum_{j=1}^{q}\|w^j\|^2 + C\sum_{j=1}^{q}\sum_{i=1}^{N}\xi_i^j \to \min_{w,b,\xi} \qquad (190)\\
&((w^{y_i}, \phi(x_i)) + b^{y_i}) - ((w^j, \phi(x_i)) + b^j) \geq 2 - \xi_i^j \text{ при } 1 \leq j \leq q,\ 1 \leq i \leq N,\ j \neq y_i\\
&\xi_i^j \geq 0 \qquad\qquad \text{при } 1 \leq j \leq q,\ 1 \leq i \leq N\\
&\sum_{j=1}^{q} w^j = 0, \quad \sum_{j=1}^{q} b^j = 0.
\end{aligned}$$

Задачу (190), как и задачу (155) решают сводя к максимизации двойственного лагранжиана, как и (163) являющейся задачей квадратичной оптимизации в пересечении куба со стороной C с плоскостью. Но эта плоскость, возникающая как условие равенства нулю производных лагранжиана по всем свободным членам b^j, имеет коразмерность уже не 1, а $q-1$. Соответственно, при численном решении методом SMO и его аналогам в качестве элементарных шагов приходится оптимизировать сразу по q переменным. Быстрый и надежный алгоритм угадывания, по каким группам переменных делать очередной шаг оптимизации, пока неизвестен, а количество таких групп, равное $\binom{N}{q}$, при $q > 2$ неприемлемо велико.

4.2.7 Сведение многоклассовой классификации к последовательности двухклассовых

Сведение многоклассовой классификации к решению нескольких задач двухклассовой классификации рассматривается далее не примере метода опорных

векторов. Но оно так же и столь же эффективно применимо для всех методов обучения классификаторов, при которых время обучения растет быстрее квадрата числа обучающих векторов. Во всех этих вариантах удобно считать, что зафиксирован общий набор обучающих векторов всех q классов, хотя на самом деле это не обязательно.

Двоичное кодирование. Будем записывать номер класса в виде k-значного ($k = \lceil \log_2 q \rceil$) двоичного числа. Обучим k классификаторов, каждый из которых распознает один из k разрядов номера класса. По результатам распознавания входного вектора этими k классификаторами однозначно восстанавливается номер класса, к которому он принадлежит.

На самом деле все не так радостно, поскольку любая ошибка любого из этих классификаторов приводит к неправильному ответу, а обучить классификатор разделять произвольные наборы классов довольно трудно, да и получаются они медленными и ненадежными (т.е. с большим количеством опорных векторов и большой средней ошибкой). В простых случаях можно пытаться подобрать способ кодирования так, чтобы каждый разряд кода распознавался относительно легко. Немного уменьшить ошибку распознавания можно применяя для номера класса двоичные коды с коррекцией ошибок и обучая разные классификаторы на разных наборах обучающих векторов. Но чаще идут совсем другим путем, и вместо очень малого количества очень сложных классификаторов обучают и применяют большое количество более простых.

Каждый против всех. Для каждого класса обучим классификатор f^j, считающий векторы j-го класса положительными, а всех остальных классов — отрицательными. Для каждого распознаваемого вектора x вычислим все q значений $f^1(x), \ldots, f^q(x)$ и выберем из них наибольшее. Соответствующий класс, для которого получена максимальная оценка, и будет ответом распознавателя.

Заметим, что если такие классификаторы $f^j(x) = (w, x) + b$ и штрафы обучающих векторов ξ_i^j получены решением стандартной минимизационной задачи (155) для двухклассовой SVC, то можно построить следующее субоптимальное решение задачи многоклассовой классификации (190)

$$\tilde{f}^j(x) = f^j(x) - \frac{1}{q}\sum_{k=1}^{q} f^k(x)$$
$$\tilde{\xi}_i^j = \xi_i^j + \xi_i^{y_i} \text{ при } 1 \le j \le q,\ 1 \le i \le N,\ j \ne y_i$$
$$\tilde{\xi}_i^{y_i} = 0 \text{ при } 1 \le i \le N\ .$$

Функции $\tilde{f}^j$ определяют ту же многоклассовую классификацию, что функции f^j, все ограничения задачи (190) для них и штрафов $\tilde{\xi}_i^j$ выполнены, минимизируемая функция не превосходит суммы функций, (155) минимизированных при обучении классификаторов f^j, но не обязательно достигает своего минимума.

Каждый против каждого. Вместо q "больших" классификаторов, отличающих каждый класс от всех остальных, обучим $\frac{q(q-1)}{2}$ "маленьких" классификаторов, различающих пары классов. Каждый из них, естественно, обучается только на векторах, принадлежащих двум интересующим его классам, поэтому время обучения и количество опорных векторов получаются-таки меньше, чем у классификатора типа "каждый против всех". Для каждого распознаваемого вектора x вычислим все $\frac{q(q-1)}{2}$ значений $f^{ij}(x) = -f^{ji}(x)$ классифицирующих функций (дискриминантов),

отделяющих i-й класс ("положительный") от j-го ("отрицательного"), затем вычислим q сумм $f^i(\psi, x) = \sum_{j \neq i} \psi(f^{ij}(x))$, где ψ — некоторая монотонно неубывающая функция, например, тождественная, логистическая (сглаженная ступенька) или θ-функция Хевисайда (ступенька), и выберем из них наибольшую. Соответствующий класс, для которого получена максимальная оценка, и будет ответом распознавателя.

Турнир на выбывание. $\frac{q(q-1)}{2}$ "маленьких" классификаторов из предыдущего пункта можно использовать более экономным способом. Для каждого распознаваемого вектора x устроим среди q классов турнир за право обладания этим вектором. Игра между двумя классами состоит в применении к вектору x различающего эти классы классификатора и победитель естественно определяется знаком классифицирующей функции. Проигравший класс выбывает, а победитель играет дальше. После $q-1$ игр, т.е. работы $q-1$ "маленьких" классификаторов, все классы, кроме "чемпиона" выбывают, и именно к нему приписывается вектор x.

Дихотомия. Разобьем множество всех q классов на два подмножества — "надкласса" — и обучим классификатор, определяющий принадлежность обучающего вектора надклассу[34]. Повторим процедуру для каждого надкласса. И т.д. В конце концов получим двоичное дерево, $q-1$ вершинам ветвления которого соответствуют классификаторы, а q листьям — классы. Каждый распознаваемый вектор x сперва подадим на классификацию классификатора верхнего уровня, затем, в зависимости от его ответа, одному из не более двух классификаторов следующего уровня и т.д. После не более $q-1$ шагов (а для сбалансированного дерева — всего лишь $\lceil \log_2 q \rceil$ шагов) получим ответ.

Поскольку время работы каждого классификатора пропорционально количеству опорных векторов (очевидно), а время обучения растет немного быстрее квадрата числа обучающих векторов (не очевидно, но для общеизвестных методов обучения SVC с ядрами верно), непосредственно из конструкции видно, что турнирный и дихотомический методы должны обучаться и работать быстрее, чем метод "каждый против всех", а дихотомический со сбалансированным деревом — еще и быстрее двоичного кодирования, см. таблицу 1. На практике так оно и получается, даже если модифицировать их так, чтобы они выдавали в качестве ответа не только номер класса-победителя, но и классы, занявшие несколько следующих мест, и их количественные оценки. Эксперименты показывают неочевидный факт, что иногда у турнирного метода еще и ошибок меньше, чем у методов "каждый против всех" и "каждый против каждого". А вот у дихотомического метода при неудачном делении на надклассы ошибок может быть неприемлемо много. Подробное описание экспериментов по сравнению наиболее популярных методов многоклассовой классификации с помощью SVM приведено в работе [59].

Одноклассовый классификатор

Для того, чтобы научиться хорошо узнавать рукописную букву "A" ребенку нужно увидеть много разных вариантов написания буквы "A", но совершенно не нужно видеть, как пишется буква "ϒ" (а вы когда-нибудь видели? это

[34] Разбиение нужно не выбирать произвольно, а подбирать так, чтобы распознавать надклассы было относительно легко. Например, рукописные буквы разумно делить на буквы с хвостом вниз и без него и неразумно делить на гласные и согласные. Перепробовать все возможные разбиения и выбрать лучшее — слишком дорого. Общепринятых универсальных быстрых способов разбиения на надклассы пока нет.

метод	количество классификаторов	время обучения	время распознавания
многоклассовый	1	$\gg \upsilon N^{2+\epsilon}$	ηN
двоичное кодирование	$\lceil \log_2 q \rceil$	$\upsilon \lceil \log_2 q \rceil N^{2+\epsilon}$	$\eta \lceil \log_2 q \rceil N$
каждый против всех	q	$\upsilon q N^{2+\epsilon}$	$\eta q N$
каждый против каждого	$\frac{q(q-1)}{2}$	$\upsilon \frac{2^{1+\epsilon}(q-1)}{q^{1+\epsilon}} N^{2+\epsilon}$	$\eta(q-1)N$
турнир	$\frac{q(q-1)}{2}$	$\upsilon \frac{2^{1+\epsilon}(q-1)}{q^{1+\epsilon}} N^{2+\epsilon}$	$\eta \frac{2(q-1)}{q} N$
дихотомия	$q-1$	$< 2\upsilon N^{2+\epsilon}$	$< 2\eta N$
каждый за себя	q	$\upsilon \frac{1}{q^{1+\epsilon}} N^{2+\epsilon}$	ηN

Таблица 1: Оценка зависимости времени обучения и работы разных вариантов многоклассовой классификации, базирующихся на методе опорных векторов, от числа классов q и количества обучающих векторов N. Оптимистично предполагается, что классы сбалансированы, деревья (где есть) — тоже, время обучения одного двухклассового классификатора на N обучающих векторах пропорционально $N^{2+\epsilon}$, $0 < \epsilon < 1$ с коэффициентом υ, постоянным для подзадач одной многоклассовой задачи, а время классификации пропорционально N с коэффициентом η (т.е. доля опорных векторов считается постоянной). Предположения довольно сомнительные, равно как и ценность всей таблицы. Она лишь иллюстрирует идею, что методом опорных векторов легче обучить много простых распознавателей, чем мало сложных.

заглавная греческая буква ипсилон). Эту идею можно применить и к обучению классификатора, в частности для ускорения обучения многоклассовому распознаванию.

Рассмотрим сначала распознавание с гауссовым ядром. Соответствующее ему спрямляющее отображение $\phi : \mathbb{R}^d \to \mathbf{H} = L^2(\mathbb{R}^d)$ (предложение 17) переводит все пространство-прообраз в небольшой колпачок, высеченный на сфере с центром в $0 \in \mathbf{H}$ (т.е. в функции, тождественно равной нулю) конусом всюду положительных функций. Возьмем в $\mathbf{H}$ образы обучающих векторов одного и того же класса ("положительного"), объявим 0 "отрицательным" классом и обучим классификатор, различающий эти два класса, т.е. отделяющий некоторое конечное множество на сфере от центра сферы широкой полосой. Но отделить образы обучающих векторов от нуля полосой — это все равно, что построить проходящую через ноль параллельную ей (и содержащую ее) полосу вдвое большей ширины, относительно которой все эти образы находятся с одной стороны, т.е. построить линейную в спрямляющем пространстве функцию без свободного члена, принимающую положительные значения на образах обучающих векторов. Из предложения 17 следует, что в качестве разделяющей функции годится скалярное произведение с любой положительной функцией в $L^2(\mathbb{R}^d)$, но из разделяющих функций мы ищем самую лучшую.

В общем случае годится любое ядро Мерсера (желательно все же, чтобы прообраз нуля при спрямляющем отображении ϕ был пуст или хотя бы не содержал ничего разумно классифицируемого, например, в случае распознавания букв был пустым белым полем), а вместо строгого разделения используется идеологию мягкого зазора. Формально обучение сводится к оптимизационной задаче

$$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N} \xi_i \to \min_{w,\xi} \qquad (191)$$
$$(w, \phi(x_i)) \geq 1 - \xi_i \text{ при } 1 \leq i \leq N$$

$$\xi_i \geq 0 \text{ при } 1 \leq i \leq N \ ,$$

которая решается аналогично задаче двухклассового обучения (155), и даже несколько проще.

Или можно ставить задачу как в предложении 28:

$$\begin{aligned} &\frac{1}{2}\|w\|^2 + \sum_{i=1}^{N}(\xi_i - \nu\rho) \to \min_{w,b,\xi,\rho} \\ &(w, \phi(x_i)) \geq \rho - \xi_i \text{ при } i = 1, \ldots, N \\ &\xi_i \geq 0 \text{ при } i = 1, \ldots, N \ \ , \end{aligned} \tag{192}$$

причем зазор ρ не обязан быть положительным. В таком виде одноклассовые классификаторы появились в работе [102] для оценки плотности распределения и обнаружения выбросов.

Упражнение. Сформулируйте точное утверждение о соответствии оптимизационных задач (191) и (192), выпишите двойственные им задачи и убедитесь, что все обучение и последующее распознавание выразимо через ядро $K(x, z) = (\phi(x), \phi(z))$.

Из одноклассовых классификаторов аналогично методу "каждый против всех" (стр. 119) можно собрать очень быстрый — и по времени обучения, и по скорости работы —, хотя и не самый точный многоклассовый классификатор:

Каждый за себя. Для каждого класса обучим одноклассовый классификатор f^j. Для каждого распознаваемого вектора x вычислим все q значений $f^1(x), \ldots, f^q(x)$ и выберем из них наибольшее[35]. Соответствующий класс, для которого получена максимальная оценка, и будет ответом распознавателя.

В отличие от всех ранее описанных многоклассовых классификаторов, такой классификатор в случае двух классов не совпадает с обычным двухклассовым классификатором.

5 Линейные комбинации распознавателей

В этом разделе подробнее рассматриваются некоторые способы объединения распознавателей, анонсированные в разделе 2.3.3: как из многих не очень хороших распознавателей сделать один хороший. Изложение, в основном, следует статье [42]. Для удобства сравнения разных методов друг с другом, они единообразно выписаны в виде упрощенных "алгоритмов".

На протяжении всего раздела при рассмотрении двухклассовой классификации предполагается пространство ответов $\mathcal{Y} = \{-1, 1\}$, а при представлении классификатора f в виде знака какой-то функции $f(x) = \text{sign}(g(x))$ возможный ответ $f(x) = 0$ не рассматривается, см. примечание 20 на странице 53.

5.1 Общие идеи: голосование распознавателей и распознаватели над распознавателями

Предположим, что по какой-то причине удалось обучить несколько (возможно, много) распознавателей, но ни один из них не является достаточно хорошим, т.е. у каждого недостаточно мала средняя ошибка на независимых тестах. Таких причин может быть много:

[35] Это не вся правда. Еще можно учитывать априорные вероятности разных классов.

- обучение проводится каким-то итеративным методом, который сходится к разным распознавателям в зависимости от начального состояния и параметров обучения (например, такое происходит с обучением нейронных сетей);

- обучение проводится каким-то итеративным методом, который сходится к очень разным распознавателям в зависимости от обучающего набора (например, такое происходит с обучением деревьев);

- время обучения настолько быстро растет с ростом обучающего набора, что большой обучающий набор не удается использовать весь сразу, но можно использовать по частям (например, в методе опорных векторов);

- время обучения настолько быстро растет с ростом числа признаков, что их не удается использовать все сразу (например, в методах гребневой (ridge) и логистической (logistic) регрессии);

- перепробовали много методов обучения и ни один из них не удовлетворил.

Вместо того, чтобы из получившихся "слабых" распознавателей выбирать наименее плохой, можно попробовать их "усилить", ища самый лучший распознаватель не только среди имеющихся распознавателей, но и среди функций от них. Или, что то же самое, используя ответы "слабых" распознавателей в качестве новых признаков, построить над ними новый "сильный" распознаватель. Класс допустимых функций от распознавателей должен быть не слишком большим, чтобы избежать переобучения. Чаще всего используют линейные комбинации распознавателей, и даже не все линейные комбинации, а только выпуклые — с неотрицательными коэффициентами с единичной суммой, т.е. устраивают взвешенное голосование "слабых" распознавателей. Этот подход одинаково применим и к регрессии, и к классификации, особенно к двухклассовой классификации, при которой он превращается в голосование в обычном бытовом смысле.

5.1.1 Голосование независимо обучаемых распознавателей

Сначала рассмотрим задачу двухклассовой классификации и наивную идею голосования в некотором смысле независимых классификаторов простым большинством.

Предложение 31 *Пусть есть $2n+1$ двухклассовых классификаторов, ошибки которых не коррелированы и вероятность ошибки каждого не превосходит некоторого числа $\nu < \frac{1}{2}$. Тогда вероятность ошибки "сильного" классификатора, дающего ответ простым голосованием "слабых", оценивается сверху через n и ν и стремится к нулю при $n \to \infty$.*

Доказательство. 0-1-ошибка $e : \mathcal{X} \times \mathcal{Y} \to \{0, 1\}$ "слабого" классификатора в точке произведения пространства признаков на пространство ответов является случайной величиной, принимающей значения 0 или 1. Ожидание ошибки не превосходит $\nu < \frac{1}{2}$, значит дисперсия не превосходит $\nu(1-\nu)$ (**упражнение!**). Рассмотрим среднее арифметическое $2n+1$ случайных величин — ошибок классификаторов. Его ожидание равно среднему арифметическому ожиданий и не превосходит ν, а дисперсия равна среднему арифметическому дисперсий деленному на $2n+1$, и не превосходит $\frac{\nu(1-\nu)}{2n+1}$, поскольку ошибки распознавателей предполагаются некоррелированными. Значит вероятность ошибки "сильного" распознавателя, т.е. вероятность того, что неправильных ответов — большинство, что равносильно тому, что среднее арифметическое ошибок больше $\frac{1}{2}$, по

неравенству Чебышёва[36] не превосходит $\frac{\nu(1-\nu)}{(2n+1)(\frac{1}{2}-\nu)^2}$ и стремится к нулю при возрастании n. $\square$

Для малых n оценка предложения 31 вполне применима. Для больших n вместо неравенства Чебышёва можно применить более сильное неравенство Чернова. Но все бесполезно, поскольку, во-первых, результат предложения о стремлении ошибки к нулю неверен, если вероятность ошибки байесовского классификатора (13) — минимально возможная — положительна, во-вторых, предположение о том, что на любом (достаточно длинном) обучающем наборе можно обучить не совсем плохой "слабый" распознаватель, невыполнимо: не совсем на любом, а в третьих, даже независимость обучения классификаторов не гарантирует некоррелированности их ошибок. Таким образом, большое количество классификаторов с некоррелированными ошибками взять негде.

Тем не менее, идея независимого обучения многих распознавателей, даже без гарантии независимости или некоррелированности их ошибок, и их последующего голосования (в задачах классификации) или усреднения (в задачах регрессии) очень соблазнительна возможностью распараллелить обучение. Среди ее успешных реализаций можно отметить случайные леса (*random forests*) [21]. Случайный лес — это большое количество классификационных или регрессионных деревьев с малым количеством вершин, обученных на одном и том же обучающем наборе одним и тем же алгоритмом, но на различных случайных подмножествах (или выборках с повторением из) обучающего набора и/или на различных случайных подмножествах признаков. В случайных лесах также используется специфика деревьев в качестве "слабых" распознавателей: на различных шагах построения дерева можно брать в качестве кандидатов для очередного разбиения различные случайные подмножества признаков. Эксперименты показали, что в задачах с очень большим количеством признаков случайные леса успешно конкурируют с другими методами распознавания.

5.1.2 Предыстория и история бустинга

Исследования объединения классификаторов путем голосования начались в самом начале 1990-х годов в работах [39, 101] с изучения того, насколько сильный классификатор может получиться объединением слабых. Тогда же появился термин boosting ("усиление", "поддержка", точного русского эквивалента нет, используется транслитерация "*бустинг*") для "усиления" распознавателей, т.е. их последовательного обучения с учетом ошибок ранее обученных распознавателей.

В работе [101] предъявлен способ обучения "слабых" распознавателей на не просто непересекающихся, а специальным образом прореженных обучающих наборах, для которого аналог предложения 31 справедлив при более реалистических предположениях. Но на практике этот способ обучения почти бесполезен.

В середине 1990-х годов в работах [38, 37] был предложен удачный итеративный алгоритм *AdaBoost* (adaptive boosting), обучающий слабые распознаватели постепенно, с учетом результатов голосования уже обученных распознавателей. AdaBoost и его многочисленные модификации, в том числе и еще более удачные, обучали все слабые распознаватели на одном и том же обучающем наборе, не слишком усердствовали в оптимизации голосования на каждом шаге обучения, но на удивление не были склонны к переобучению, и по загадочным причинам всегда сходились.

[36] Неравенство Чебышёва состоит в том, что для случайной величины ξ с ожиданием m и дисперсией D вероятность большого отклонения от ожидания можно оценить сверху: $P(|\xi - m| > \eta) \leq \frac{D}{\eta^2}$.

В алгоритме AdaBoost используется следующая идея: при обучении "слабого" классификатора можно разным обучающим векторам (точнее, ошибкам распознавателя на них) назначать разные веса[37]. Для классификатора, обучаемого минимизацией эмпирической ошибки, с регуляризацией

$$\psi(f) + \sum_{i=1}^{N} E(f(x_i), y_i) \to \min_{f \in \mathcal{F}} \tag{193}$$

или без нее (ср. с формулами (4) и (8)), это реализуется домножением в минимизируемом функционале значения функции ошибки $E(f(x_i), y_i)$ на *вес i-го обучающего вектора* α_i:

$$\psi(f) + \sum_{i=1}^{N} \alpha_i E(f(x_i), y_i) \to \min_{f \in \mathcal{F}} \ . \tag{194}$$

Все веса α_i неотрицательны. Бóльшие веса означают, что нужно обратить бóльшее внимание на соответствующие обучающие вектора. Удобно считать, что набор весов α удовлетворяет нормировке $\sum_{i=1}^{n} \alpha_i = 1$ и может интерпретироваться как некоторое распределение вероятностей на обучающем наборе. Для обучения с регуляризацией (194) нормировка весов необходима, для обучения без регуляризации — не обязательна.

Алгоритм 3 реализует идею *AdaBoost* в простейшем варианте — с фиксированным числом шагов — для некоторого способа выбора весов α, который пока кажется взятым с потолка, но в разделе 5.2.3 будет обоснован.

Алгоритм 3: AdaBoost

1. Вход: обучающий набор $T = ((x_1, y_1), \ldots, (x_N, y_N))$ длины N, число шагов M.
2. Назначить начальные веса $\alpha_i = \frac{1}{N}, i = 1, \ldots, N$.
3. Для каждого шага $j = 1, \ldots, M$
   (a) обучить "слабый" классификатор h^j на наборе T с весами α;
   (b) вычислить его взвешенную 0-1-ошибку $E = \sum_{i|h^j(x_i) \neq y_i} \alpha_i$;
   (c) если $E = 0$, то выйти: $f(\cdot) = h^j(\cdot)$ — безошибочный классификатор;
   (d) вычислить коэффициент $c_j = \ln \frac{1-E}{E}$;
   (e) для каждого ошибочно классифицированного вектора $(h^j(x_i) \neq y_i)$ домножить его вес α_i на $e^{c_j} = \frac{1-E}{E}$;
   (f) отнормировать пересчитанные веса, чтобы их сумма $\sum_{i=1}^{N} \alpha_i$ равнялась 1.
4. Выход: классификатор $f(\cdot) = \operatorname{sign} \sum_{j=1}^{M} c_j h^j(\cdot)$.

---

[37] Чтобы не путать веса обучающих векторов с весами распознавателей в голосовании, первые будем называть весами и обозначать буквами α_*, а последние будем называть коэффициентами и обозначать буквами c_*.

Модификации алгоритма AdaBoost отличаются способами пересчета весов и коэффициентов, другими критериями остановки (например, зависящими от ошибки на отличном от обучающего оценочном наборе) а также распространением с двухклассовой классификации на оценки вероятности классов, регрессию и многоклассовую классификацию. Через некоторое время была обнаружена связь алгоритма AdaBoost и его модификаций с классификаторами с разделяющей полосой [100] и с логистической регрессией [41]. К концу 1990-х годов, появилось понимание того, почему AdaBoost не переобучается и сходится, а также как его правильно модифицировать — как обучать очередной слабый распознаватель и как подбирать веса при очередном голосовании, см. [42, 24]. Как и в случае метода опорных векторов, на практике методы бустинга давали лучшее распознавание, чем гарантировала теория.

5.1.3 "Сила слабости"

Несмотря на наличие теоретического обоснования бустинга, в современной практике его применяются различные эвристики, теоретически не обоснованные, но существенно ускоряющие обучение без потери качества результата. А именно,

- "слабые" распознаватели делают как можно проще, лишь бы они быстро обучались; например, используют распознающие деревья с очень малым количеством вершин, вплоть до пней (stump), т.е. деревьев, состоящих из корня и двух листьев;
- обучают их без регуляризации (в формуле (194) $\psi = 0$);
- часто их обучают не на всем обучающем наборе, отбрасывая большую часть (на каждом шаге — свою) обучающих векторов и/или признаков;
- веса при голосовании не очень-то оптимизируют.

Зато таких дешевых шагов обучения делают много. Сравните название этого раздела с названием работы [101].

В следующих разделах мы рассмотрим некоторые варианты бустинга подробнее.

5.2 Градиентный спуск в пространстве распознавателей

5.2.1 Регрессия

Пусть есть обучающий набор для задачи регрессии и недорогой алгоритм обучения "слабых" распознавателей в пространстве, содержащем 0. Алгоритм 4 (по мотивам работы [41]) демонстрирует, как можно обучить линейную комбинацию таких распознавателей.

Алгоритм 4: Разностный бустинг

1. Вход: обучающий набор $((x_1, y_1), \ldots, (x_N, y_N))$ длины N, функция ошибки $E(\cdot, \cdot)$, число шагов M, "длина" шага $c \in (0, 1]$.
2. Скопировать ответы обучающего набора y_i в невязки z_i, $i = 1, \ldots, N$.
3. Для каждого шага $j = 1, \ldots, M$
   (a) обучить "слабый" распознаватель h^j на наборе $((x_1, z_1), \ldots, (x_N, z_N))$ минимизацией суммы ошибок $\sum_{i=1}^{n} E(h^j(x_i), z_i)$;
   (b) вычесть из каждой невязки z_i умноженный на c прогноз $h^j(x_i)$;
4. Выход: регрессия $f(\cdot) = \sum_{j=1}^{M} ch^j(\cdot) = c\sum_{j=1}^{M} h^j(\cdot)$.

В полученной линейной комбинации все коэффициенты равны c и считается, что дешевле обучить очередной распознаватель, чем оптимизировать коэффициенты при имеющихся. Противоположный подход описан в разделе 5.3.

Если используемая при обучении "слабых" распознавателей (193) функция ошибки $E(\cdot, \cdot)$ зависит только от разности своих аргументов, а ошибка обученного "слабого" распознавателя меньше, чем ошибка функции, тождественно равной 0 (на практике эти условия почти всегда выполнены), то очевидно, что при $c = 1$ каждый шаг обучения уменьшает суммарную ошибку

$$\sum_{i=1}^{N} E\left(f^m(x_i), y_i\right)$$

для частичной суммы $f^m(\cdot) = \sum_{j=1}^{m} h^j(\cdot)$. Если же ошибка как функция от разности аргументов выпукла, то каждый шаг обучения уменьшает суммарную ошибку и при любом $c \in (0, 1]$. Чем меньше параметр c, тем дольше обучается "сильный" распознаватель (очевидно), но тем меньше он склонен к переобучению (экспериментальный факт). Таким образом, c является параметром регуляризации, аналогичным параметру C в задаче (99) обучения классификатора с разделяющей полосой.

Тем самым, алгоритм бустинга для регрессии похож на градиентный спуск в линейной оболочке пространства "слабых" распознавателей: на каждом шаге приблизительно вычисляется направление, в котором уменьшается суммарная ошибка, т.е. обучается распознаватель, а затем в этом направлении делается шаг длины c. Но раз уж все равно выгодно ходить мелкими шагами, можно двигаться не в направлении минимума суммарной ошибки, а против ее градиента. С другой стороны, можно длину шага брать не произвольно, а, например, решая одномерную задачу минимизации ошибки по длине шага. Можно также применять метод Ньютона второго порядка: пример приведен в разделе 5.2.2 для логистической регрессии.

Упражнение. Для квадратичной ошибки $E(r, y) = (r - y)^2$ градиентный спуск методом Ньютона совпадает с вышеописанным разностным бустингом при $c = 1$.

Собственно градиентный спуск требует, чтобы функция ошибки $E(\cdot, \cdot)$ была дифференцируемой, но не требует, чтобы она зависела только от разности

аргументов. Впрочем, не злостно недифференцируемую функцию ошибки, например, ϵ-нечувствительную, тоже можно применять, как-то доопределив ее производную. Алгоритм 5 реализует фиксированное число шагов градиентного спуска.

Алгоритм 5: Градиентный бустинг

1. Вход: обучающий набор $((x_1, y_1), \ldots, (x_N, y_N))$ длины N, функция ошибки $E(\cdot, \cdot)$, число шагов M.
2. Построить начальное приближение: обучить "слабый" распознаватель f^0 на наборе $((x_1, y_1), \ldots, (x_N, y_N))$ минимизацией суммы ошибок $\sum_{i=1}^{n} E(f^0(x_i), y_i)$, а если это слишком трудно, задать его произвольно, например, $f^0 = 0$.
3. Для каждого шага $j = 1, \ldots, M$
   (a) вычислить производные $z_i = -\left.\frac{\partial E(r, y_i)}{\partial r}\right|_{r=f^{j-1}(x_i)}$, $i = 1, \ldots, N$;
   (b) обучить "слабый" распознаватель h^j на наборе $((x_1, z_1), \ldots, (x_N, z_N))$ минимизацией суммы квадратичных ошибок $\sum_{i=1}^{n} (h^j(x_i) - z_i)^2$;
   (c) как-нибудь, хоть приблизительно, решить одномерную минимизационную задачу $\sum_{i=1}^{n} E((f^{j-1} + c_j h^j)(x_i), y_i) \to \min_{c_j \in \mathbb{R}_+}$
   (d) обозначить $f^j(\cdot) = f^{j-1}(\cdot) + c_j h^j(\cdot)$.
4. Выход: регрессия $f^M(\cdot) = f^0(\cdot) + \sum_{j=1}^{M} c_j h^j(\cdot)$.

Использование в алгоритме 5 при обучении распознавателя, приближающего производные, квадратичной ошибки — это некоторый произвол, имеющий целью ускорение обучения. В остальных местах алгоритма используется настоящая ошибка $E(\cdot, \cdot)$. Причем для обучения полноценного распознавателя она используется не более одного раза, а в остальных случаях она минимизируется только на некоторой прямой. Этот алгоритм можно модифицировать в духе предыдущего, введя параметр $c^* < 1$ и делая на каждой очередной итерации уменьшенный шаг $f^j(\cdot) = f^{j-1}(\cdot) + c^* c_j h^j(x_i)(\cdot)$: алгоритм будет сходиться при большем числе шагов, но будет устойчивее к переобучению.

5.2.2 Логистическая регрессия

Аналогичные идеи градиентного спуска применимы и к оценке вероятностей классов при двухклассовой классификации методом логистической регрессии (раздел 2.2.2). Но в этом случае функция ошибки сложнее, чем зависимость только от разности аргументов, и алгоритм обучения получается сложнее.

Напомним, что логистической регрессией для двухклассовой классификации называется распознаватель (регрессия, функция) $f : \mathcal{X} \to \mathbb{R}$, предсказыва-

ющая логарифм отношения условных вероятностей возможных ответов

$$\ln \frac{P(y=1|x)}{P(y=-1|x)} \ ,$$

называемый *log-odds* и, как правило, обучаемый максимизацией правдоподобия или максимизацией апостериорной вероятности, см. разделы 1.2.8 и 2.2.2. Оценки вероятностей тогда выражаются по формулам

$$\begin{aligned} p_f(x) &= \frac{1}{1+e^{-f(x)}} \text{ для } P(y=1|x) \ , \\ 1-p_f(x) &= \frac{1}{1+e^{f(x)}} \text{ для } P(y=-1|x) \ , \end{aligned}$$

т.е. в принятых обозначениях для любого ответа $y = \pm 1$ его вероятность оценивается как $\frac{1}{1+e^{-yf(x)}}$, и минус логарифм правдоподобия, принимаемый за функцию ошибки при обучении, равен

$$E(f(x), y) = \ln(1 + e^{-yf(x)}) \ . \tag{195}$$

Производные такой ошибки по прогнозу распознавателя легко вычисляются:

$$\left.\frac{\partial E(r,y)}{\partial r}\right|_{r=f(x)} = \left.\frac{\partial \ln(1+e^{-yr})}{\partial r}\right|_{r=f(x)} = \frac{-ye^{-yf(x)}}{1+e^{-yf(x)}} = \frac{-y}{1+e^{yf(x)}} \tag{196}$$

$$\begin{aligned} \left.\frac{\partial^2 E(r,y)}{(\partial r)^2}\right|_{r=f(x)} &= \left.\frac{\partial^2 \ln(1+e^{-yr})}{(\partial r)^2}\right|_{r=f(x)} = \frac{y^2 e^{yf(x)}}{(1+e^{yf(x)})^2} \\ &= \frac{1}{(1+e^{-yf(x)})(1+e^{yf(x)})} = \frac{1}{(1+e^{-f(x)})(1+e^{f(x)})} \\ &= p_f(x)(1-p_f(x)) \end{aligned} \tag{197}$$

(использовалось тождество $y^2 = 1$). Из равенства (197) видно, что вторая производная ошибки положительна, а значит суммарная ошибка распознавателя f на обучающем наборе T

$$E(f,T) = \sum_{i=1}^{N} E(f(x_i), y_i)$$

выпукла. Заодно из равенства (196) видно, что уравнение $\frac{\partial E(f,T)}{\partial f} = 0$ трансцендентно, вряд ли решается аналитически и ошибку нужно минимизировать численно.

Можно применить метод градиентного спуска из раздела 5.2.1, но мы опробуем другой путь. Применим метод Ньютона второго порядка, т.е. итеративную минимизацию квадратичного приближения минимизируемой ошибки в окрестности текущей точки. Квадратичное приближение ошибки в окрестности точки f (f — это функция! На самом деле, все определяется ее значениями на обучающих векторах) в силу равенств (196) и (197) задается многочленом Тейлора

$$E(f+h,T) \approx E(f,T) - \sum_{i=1}^{N} \frac{y_i}{1+e^{y_i f(x_i)}} h(x_i) + \frac{1}{2} \sum_{i=1}^{N} p_f(x_i)(1-p_f(x_i))(h(x_i))^2 \tag{198}$$

от значений $h_i = h(x_i)$ требуемой поправки в точках обучающего набора. Точка минимума суммы N квадратных трехчленов от N независимых переменных[38]

[38] В маловероятном случае совпадения некоторых обучающих векторов не все N переменных h_i независимы, но вычислить точку минимума суммы квадратных трехчленов все равно не представляет труда (**упражнение!**) . Аналогичные уточнения нужны к каждому варианту обучения из этого раздела.

h_i выписывается явно:

$$h_i = \frac{\frac{y_i}{1+e^{y_i f(x_i)}}}{p_f(x_i)(1-p_f(x_i))} = y_i(1+e^{-y_i f(x_i)}) = \frac{y_i}{P(y_i|f(x_i))}\ , \qquad (199)$$

так что

$$E(f+h,T) \approx \text{const} + \frac{1}{2}\sum_{i=1}^{N} p_f(x_i)(1-p_f(x_i))\left(h(x_i) - \frac{y_i}{P(y_i|f(x_i))}\right)^2\ . \qquad (200)$$

Получается, что целью обучения поправки h к распознавателю f в точке x_i является правильный ответ y_i, деленный на уверенность распознавателя f в этом ответе, а штраф в обучении $h(x_i)$ квадратичен с весом (ненормированным) $\frac{1}{2}p_f(x_i)(1-p_f(x_i))$. Алгоритм 6 демонстрирует такое обучение с фиксированным числом шагов.

Алгоритм 6: LogitBoost

1. Вход: обучающий набор $((x_1,y_1),\ldots,(x_N,y_N))$ длины N, число шагов M.
2. Построить начальное приближение: обучить "слабый" распознаватель f^0 на наборе $((x_1,y_1),\ldots,(x_N,y_N))$ минимизацией суммы ошибок $\sum_{i=1}^{n} \ln\left(1+e^{-y_i f^0(x_i)}\right)$, а если это слишком трудно, задать его произвольно, например, $f^0 = 0$.
3. Для каждого шага $j = 1,\ldots,M$
   (a) вычислить прогнозы предыдущего распознавателя $p_i = \frac{1}{1+e^{-f^{j-1}(x_i)}}$, веса $\alpha_i = p_i(1-p_i)$ и цели обучения $z_i = y_i\left(1+e^{-y_i f^{j-1}(x_i)}\right)$, $i = 1,\ldots,N$;
   (b) обучить "слабый" распознаватель h^j на наборе $((x_1,z_1),\ldots,(x_N,z_N))$ минимизацией суммы квадратичных ошибок с весами α, если нужно, нормированными: $\sum_{i-1}^{n} \alpha_i (h^j(x_i)-z_i)^2$;
   (c) обозначить $f^j(\cdot) = f^{j-1}(\cdot) + h^j(x_i)(\cdot)$.
4. Выход: логистическая регрессия $f^M(\cdot) = f^0(\cdot) + \sum_{j=1}^{M} h^j(\cdot)$ и ее оценка вероятности единицы $p_{f^M}(\cdot) = \frac{1}{1+e^{-f^M(\cdot)}}$, а если нужно, то и классификатор $\text{sign}(f^M(\cdot))$.

Алгоритм 6 (но не только он) называется *LogitBoost*. Поскольку шаг метода Ньютона, вообще говоря, не всегда уменьшает минимизируемую функцию, о его сходимости нужно заботиться. Кроме того возможны проблемы при обучении распознавателей h^j, вызванные тем, что их цели обучения z_i могут быть экспоненциально велики, а веса α_i — исчезающе малы. Авторы алгоритма в статье [42] предлагали просто обрезать цели и веса по модулю сверху и, соответственно, снизу по довольно произвольным порогам. Кроме того для устойчивости к ошибкам в разметке ответов y_i в обучающем наборе лучше обучать

распознаватели h^j не с квадратичной функцией ошибки, а с любой функцией ошибки, растущей не быстрее, чем линейно (см. раздел 2.1.3), поскольку тогда независимо от величины z_i цена такой ошибки разметки не будет превосходить константы (**упражнение!**). Кроме того, поскольку нельзя полагаться на то, что распознаватели h^j будут обучены идеально, можно вместо h^j на каждом шаге бустинга добавлять $c^j h^j$ с коэффициентом c^j, вычисляемым, например, тем же методом Ньютона

$$c^j = \frac{\sum\limits_{i=1}^{N} \frac{y_i h^j(x_i)}{1 + e^{y_i f^{j-1}(x_i)}}}{\sum\limits_{i=1}^{N} \frac{(h^j(x_i))^2}{(1 + e^{-f^{j-1}(x)})(1 + e^{f^{j-1}(x)})}}$$

(**упражнение!**), да еще и с любыми проверками и усовершенствованиями.

После всех этих технических усовершенствований получается более громоздкий алгоритм 7.

Алгоритм 7: LogitBoost с системой сдержек и противовесов

1. Вход: обучающий набор $((x_1, y_1), \ldots, (x_N, y_N))$ длины N, число шагов M, пороги $\epsilon > 0$, $\Omega \gg 1$.

2. Построить начальное приближение: обучить "слабый" распознаватель f^0 на наборе $((x_1, y_1), \ldots, (x_N, y_N))$ минимизацией суммы ошибок $\sum_{i=1}^{n} \ln\left(1 + e^{-y_i f^0(x_i)}\right)$, а если это слишком трудно, задать его произвольно, например, $f^0 = 0$.

3. Вычислить суммарную логистическую ошибку начального приближения $E^0 = \sum_{i=1}^{N} \ln\left(1 + e^{-y_i f^0(x_i)}\right)$.

4. Для каждого шага $j = 1, \ldots, M$

   (a) вычислить прогнозы предыдущего распознавателя $p_i = \frac{1}{1+e^{-f^{j-1}(x_i)}}$, веса $\alpha_i = \max(\epsilon, p_i(1-p_i))$ и цели обучения $z_i = y_i \min\left(\Omega, 1 + e^{-y_i f^{j-1}(x_i)}\right)$, $i = 1, \ldots, N$;

   (b) обучить "слабый" распознаватель h^j на наборе $((x_1, z_1), \ldots, (x_N, z_N))$ минимизацией суммы медленно растущих ошибок с весами α, если нужно, нормированными: $\sum_{i-1}^{n} \alpha_i (h^j(x_i) - z_i)^2$;

   (c) вычислить коэффициент $c^j = \dfrac{\sum_{i=1}^{N} \alpha_i z_i h^j(x_i)}{\sum_{i=1}^{N} \alpha_i (h^j(x_i))^2}$

   (d) обозначить $f^j(\cdot) = f^{j-1}(\cdot) + c^j h^j(x_i)(\cdot)$;

   (e) пока $c^j \neq 0$ и суммарная логистическая ошибка не уменьшилась, т.е. пока $\left(E^j = \sum_{i=1}^{N} \ln\left(1 + e^{-y_i f^j(x_i)}\right)\right) \geq E^{j-1}$, пополамить коэффициент $c^j = c^j/2$

   (f) если $c^j = 0$, прекратить обучение досрочно; все равно $f^M(\cdot) = f^{j-1}(\cdot)$.

5. Выход: логистическая регрессия $f^M(\cdot) = f^0(\cdot) + \sum_{j=1}^{M} c^j h^j(\cdot)$ и ее оценка вероятности единицы $p_{f^M}(\cdot) = \frac{1}{1+e^{-f^M(\cdot)}}$.

5.2.3 Двухклассовая классификация; AdaBoost как минимизация экспоненциальной ошибки

Методы бустинга двухклассовых классификаторов аналогичны методам бустинга регрессии, особенно логистической, за исключением того, что поскольку классификатор принимает только значения ± 1, при каждом поправочном клас-

сификаторе обязательно нужно обучать коэффициент, а мономы четных степеней от классификатора равны единице. В результате вычислительные формулы получаются другими.

Например, для метода логистической регрессии квадратичное приближение ошибки вместо формулы (198) будет задаваться формулой

$$\begin{aligned} E(f+ch,T) \approx & E(f,T) - \sum_{i=1}^{N} \frac{y_i}{1+e^{y_i f(x_i)}} ch(x_i) + \frac{1}{2} \sum_{i=1}^{N} p_f(x_i)(1-p_f(x_i)) c^2 (h(x_i))^2 \\ = & E(f,T) - \sum_{i=1}^{N} \frac{y_i}{1+e^{y_i f(x_i)}} ch(x_i) + \frac{1}{2} \sum_{i=1}^{N} p_f(x_i)(1-p_f(x_i)) c^2 \ , \end{aligned}$$

в которой квадратичный член не зависит от классификатора h. А минимизация линейного по $ch(\cdot)$ члена ошибки обучения по $h(\cdot)$ при положительном коэффициенте c от значения c не зависит. То есть, как и при шаге градиентного бустинга, сначала можно обучить классификатор h, а потом подобрать при нем коэффициент.

Минимум ошибки достигается при $h(x_i) = y_i$, т.е. каждый очередной поправочный классификатор нужно обучать одним и тем же ответам[39]. Но цена ошибки поправочного классификатора в точке x_i равна $\frac{2}{1+e^{y_i f(x_i)}}$, т.е. поправочный классификатор нужно обучать с весами, пропорциональными $\frac{1}{1+e^{y_i f(x_i)}} = P(-y_i|f(x_i))$ — правдоподобию ошибки предыдущего классификатора. После обучения поправочного классификатора квадратичная минимизация по коэффициенту c выписывается явно:

$$c = \frac{\sum\limits_{i=1}^{N} y_i h(x_i) P(-y_i|f(x_i))}{\sum\limits_{i=1}^{N} p_f(x_i)(1-p_f(x_i))} = \frac{\sum\limits_{i=1}^{N} y_i h(x_i) P(-y_i|f(x_i))}{\sum\limits_{i=1}^{N} P(-y_i|f(x_i))(1-P(-y_i|f(x_i)))} \ .$$

Алгоритм 8 представляет собой вариант алгоритма LogitBoost, обучающийся логистической регрессии (и, как следствие, классификации) усилением слабых классификаторов.

[39] См. примечание 38 на стр. 129

Алгоритм 8: LogitBoost над дискретнозначными классификаторами

1. Вход: обучающий набор $T = ((x_1, y_1), \ldots, (x_N, y_N))$ длины N, число шагов M.
2. Построить начальное приближение: распознаватель $f^0(\cdot) = 0$.
3. Для каждого шага $j = 1, \ldots, M$
   (a) вычислить веса $\alpha_i = \frac{1}{1+e^{y_i f^{j-1}(x_i)}}$, $i = 1, \ldots, N$;
   (b) обучить "слабый" классификатор h^j на наборе T с отнормированными (если нужно) весами α;
   (c) вычислить коэффициент $c_j = \dfrac{\sum_{i=1}^{N} \alpha_i y_i h^j(x_i)}{\sum_{i=1}^{N} \alpha_i (1 - \alpha_i)}$;
   (d) обозначить $f^j(\cdot) = f^{j-1}(\cdot) + c_j h^j(x_i)(\cdot)$.
4. Выход: логистическая регрессия $f^M(\cdot) = \sum_{j=1}^{M} c_j h^j(\cdot)$ и ее оценка вероятности единичного ответа $p_{f^M}(\cdot) = \frac{1}{1+e^{-f^M(\cdot)}}$, а если нужно, то и классификатор $\operatorname{sign}(f^M(\cdot))$.

Ошибка, используемая в алгоритме 8 для обучения "слабых" классификаторов, может не совпадать с логистической. С другой стороны, аналогичное обучение можно организовать для любой функции ошибки регрессии, дискретизацией которой является "сильный" классификатор, не обязательно для ошибки (195), но при этом получится-таки только классификатор без оценки вероятностей классов.

Очень простой алгоритм бустинга получается при *экспоненциальной ошибке* $E(r, y) = e^{-\frac{yr}{2}}$. Он даже не использует квадратичное приближение ошибки, а честно минимизирует ошибку в направлении против ее градиента (точнее, против приближения градиента обученным классификатором).

А именно, линейное приближение ошибки обучения в точке f

$$E(f + ch, T) \quad \approx \quad E(f, T) + \frac{1}{2} \sum_{i=1}^{N} e^{-\frac{y_i f(x_i)}{2}} y_i c h(x_i)$$

показывает, что при $c > 0$ классификатор h в точке x_i нужно обучать ответу y_i с ненормированным весом $e^{-\frac{y_i f(x_i)}{2}}$. После обучения классификатора h минимум ошибки по c находится приравниванием нулю производной:

$$0 = \frac{\partial}{\partial c} E(f + ch, T) = \frac{\partial}{\partial c} \sum_{i=1}^{N} e^{-\frac{y_i (f(x_i) + c h(x_i))}{2}} = \sum_{i=1}^{N} -y_i h(x_i) e^{-\frac{y_i (f(x_i) + c h(x_i))}{2}}$$
$$= e^{\frac{c}{2}} \sum_{i | h(x_i) \neq y_i} e^{-\frac{y_i f(x_i)}{2}} - e^{-\frac{c}{2}} \sum_{i | h(x_i) = y_i} e^{-\frac{y_i f(x_i)}{2}},$$

значит

$$c = \ln \frac{\sum\limits_{i|h(x_i)=y_i} e^{-\frac{y_i f(x_i))}{2}}}{\sum\limits_{i|h(x_i)\neq y_i} e^{-\frac{y_i f(x_i))}{2}}} ,$$

если только знаменатель под логарифмом не равен нулю.

Упражнение. Что делать, если знаменатель $\sum_{i|h(x_i)\neq y_i} e^{-\frac{y_i f(x_i))}{2}} = 0$?

Из вышеприведенных вычислений и явной нормировки весов получается алгоритм 9.

Алгоритм 9: Бустинг с экспоненциальной ошибкой над классификаторами

1. Вход: обучающий набор $T = ((x_1, y_1), \ldots, (x_N, y_N))$ длины N, число шагов M.
2. Определить начальный классификатор $f^0(\cdot) = 0$.
3. Для каждого шага $j = 1, \ldots, M$
   (a) вычислить веса $\alpha_i = e^{-\frac{y_i f^{j-1}(x_i)}{2}}$, $i = 1, \ldots, N$;
   (b) отнормировать веса, чтобы их сумма $\sum\limits_{i=1}^{N} \alpha_i$ равнялась 1;
   (c) обучить "слабый" классификатор h^j на наборе T с весами α;
   (d) вычислить его взвешенную 0-1-ошибку $E = \sum\limits_{i|h^j(x_i)\neq y_i} \alpha_i$;
   (e) если $E = 0$, выход: $f(\cdot) = h^j(\cdot)$ — безошибочный классификатор;
   (f) вычислить коэффициент $c_j = \ln \frac{1-E}{E}$;
   (g) обозначить $f^j(\cdot) = f^{j-1}(\cdot) + c_j h^j(x_i)(\cdot)$.
4. Выход: классификатор $f(\cdot) = \operatorname{sign} \sum\limits_{j=1}^{M} c_j h^j(\cdot)$.

Упражнение. Докажите, что алгоритм 9 пошагово совпадает с алгоритмом 3 (AdaBoost), только по-другому вычисляет те же веса.

Сама по себе экспоненциальная функция ошибки не очень хороша для обучения. На правильно распознаваемых примерах, у которых $yf(x) > 0$, экспоненциальная ошибка $E(f(x), y) = e^{-\frac{yf(x)}{2}}$, как и логистическая ошибка $E(f(x), y) = \ln(1 + e^{-yf(x)})$, экспоненциально убывает с ростом $|yf(x)|$, зато неправильные примеры она штрафует очень сильно, экспоненциально по $|yf(x)|$, а не приблизительно линейно, как логистическая. Из-за этого один случайный ошибочный ответ y_i в обучающем наборе может испортить все обучение. Действительно, по мере того, как обучаемые классификаторы h^j будут, в основном давать на этом обучающем векторе разумный, но формально неверный, ответ $h^j(x_i) = -y_i$, величина $-y_i f^j(x_i)$ будет расти, ошибка $e^{-\frac{y_i f^j(x_i)}{2}}$ будет расти экспоненциально быстро по j, и постепенно основной целью обучения классификаторов h^j станет

не уменьшение количества ошибок на обучающем наборе, а уменьшение величины неустранимой ошибки на одном только i-м обучающем векторе. Несмотря на простоту алгоритма AdaBoost, лучше применять другие варианты бустинга.

При бустинге довольно часто почти весь суммарный вес сосредотачивается на весьма малой части обучающего набора, и можно сильно ускорить обучение очередного "слабого" классификатора, обучая его только на этой небольшой части. Однако навсегда выкидывать из обучающего набора остальные вектора нельзя.

5.2.4 Многоклассовая классификация

Методы бустинга многоклассовых классификаторов также аналогичны методам бустинга логистической регрессии и двухклассовой классификации, за исключением того, что для q-классовой классификации нужно обучать q или хотя бы $q-1$ функцию. Например, можно независимо запустить q двухклассовых логистических бустингов для оценки вероятностей каждого из q классов. Можно обучать их и совместно: единственное существенное отличие от двухклассового случая состоит в том, что для шага метода Ньютона минимизации ошибки, зависящей от q функций, нужно будет в каждом векторе признаков вычислять и обращать матрицу вторых производных ошибки по значениям q функций размером $q \times q$. См., например, статью [42], где вычисление всей матрицы подменено вычислением ее диагонали (в соответствии с идеологией градиентного бустинга, что каждый шаг обучения должен быть не оптимальным, а быстрым).

В качестве примера приведем многоклассовый вариант логистического бустинга из работы [24]. Идейно он ближе не к LogitBoost, а к AdaBoost (алгоритм 9), но честно обучается минимизации логистической ошибки и дает полноценный вероятностный ответ, а в качестве "слабых" распознавателей может переваривать и вероятностные, и дискретнозначные классификаторы.

Напомним, что q-классовая логистическая регрессия определяется q-мерной вектор-функцией $f : \mathbf{X} \to \mathbb{R}^{\mathbf{Y}}$, к которой затем применяется логистическое преобразование $(u^1, \ldots, u^q) \mapsto \left(\frac{e^{u^1}}{\sum_{j=1}^q e^{u^j}}, \ldots, \frac{e^{u^q}}{\sum_{j=1}^q e^{u^j}} \right)$. Обучение на наборе $T = ((x_1, y_1), \ldots, (x_N, y_N))$ с ответами $y_i \in \{1, \ldots, q\}$ состоит в попытке минимизации минус суммы логарифмов правдоподобия

$$-\sum_{i=1}^N \ln p(y_i|x_i) = -\sum_{i=1}^N \ln \frac{e^{f^{y_i}(x_i)}}{\sum\limits_{j=1}^q e^{f^j(x_i)}} = \sum_{i=1}^N \ln \left(\sum_{j=1}^q e^{(f^j(x_i) - f^{y_i}(x_i))} \right) .$$

Алгоритм 10: Многоклассовый логистический бустинг

1. Вход: обучающий набор $T = ((x_1, y_1), \ldots, (x_N, y_N))$ длины N, число шагов M.
2. Определить $f_0(\cdot) = 0$.
3. Для каждого шага $m = 1, \ldots, M$
   (a) вычислить веса $\alpha_i^j = \frac{e^{f^j_{m-1}(x_i)}}{\sum\limits_{j'=1}^{q} e^{f^{j'}_{m-1}(x_i)}}$, $(i,j) \in S_T$;
   (b) обучить "слабый" классификатор на наборе T с весами α следующим странным способом: $\left|\sum\limits_{(i,j)\in S_T} \alpha_i^j h^j(x_i)\right| \to \max\limits_{h||h^j(x_i)|\le 1}$;
   (c) вычислить его целевую функцию $R = \sum\limits_{(i,j)\in S_T} \alpha_i^j h_m^j(x_i)$ и сумму весов $Z = \sum\limits_{(i,j\in S_T)} \alpha_i^j$;
   (d) если $R = 0$, выход: дальше учиться бесполезно;
   (e) вычислить коэффициент $c_m = \frac{1}{2}\ln\frac{Z-R}{Z+R}$ (мнемоника: если h_m на предлагаемые ему *неверные* предложения $(x_i, j \ne y_i)$ чаще отвечает "да", т.е. $h_m^j(x_i) > 0$, то $R > 0$, а коэффициент $c_m < 0$, и этот ошибочный "слабый" классификатор будет не прибавляться, а вычитаться);
   (f) обозначить $f_m j(\cdot) = f_{m-1}(\cdot) + c_m h_m(x_i)(\cdot)$.
4. Выход: логистическая регрессия

$$p(y|x) = \frac{\exp\left(\sum\limits_{m=1}^{M} c_m h_m^y(x)\right)}{\sum\limits_{j=1}^{q}\exp\left(\sum\limits_{m=1}^{M} c_m h_m^j(x)\right)}\ ,\ y = 1, \ldots, q\ .$$

Для обучения распознавателя алгоритм 10 поддерживает систему из $N(q-1)$ весов α_i^j с $j \ne y_i$, т.е. по одному весу на каждый возможный неправильный ответ. Каждый вес в точности равен прогнозируемой вероятности такого ответа. Обозначим множество пар индексов $\{(i,j)|j \ne y_i\}$ через S_T.

Обучаемый на m-ом шаге "слабый" распознаватель $h_m = (h_m^1, \ldots, h_m^1)$ должен (по крайней мере на всем обучающем наборе) принимать значения в кубе $[-1,1]^q$. За это работа [24] гарантирует сходимость алгоритма[40] к распознавателю, минимизирующему ошибку обучения. Несколько неожиданна целевая функция обучения "слабого" распознавателя. В частном случае дискретнозначных классификаторов, принимающих значения ± 1, она почти совпадает с це-

[40] В работе [24] предъявлен алгоритм, казалось бы отличный от алгоритма 10. Тем не менее алгоритм 10 получается из него систематическим применением двойственности Лежандра. Ну, и изменением обозначений, конечно.

левой функцией в алгоритме 9: на всех предъявленных (неверных!) вариантах ответа нужно давать один и тот же прогноз, а $+1$ или -1 — неважно, это учтется коэффициентом c_m. Конкретный способ вычисления коэффициента c_m имеет ту же природу, что и в алгоритме 9, и получается при доказательстве сходимости.

5.3 Оптимизация голосования распознавателей

Более основательный подход к бустингу — на каждом шаге не только добавлять к голосованию новый распознаватель с каким-то коэффициентом, но и оптимизировать коэффициенты старых распознавателей — в 1990-е годы не использовался, поскольку считался слишком ресурсоемким. Но в 2000-м году в работе [11] было замечено, что метод column generation (порождение столбцов), применяемый для решения задач линейного программирования особо крупных размеров, позволяет оптимизировать коэффициенты при бустинге за время, сравнимое с временем обучения "слабых" распознавателей (подробности приведены в работе [32]). При этом полученный в результате бустинга распознаватель фактически зависит только от небольшой части обученных в процессе бустинга "слабых" распознавателей аналогично тому, как распознаватель, обученный методом lasso (стр. 43), зависит только от небольшой части переменных. Такая разреженность существенно ускоряет распознавание. Этот метод бустинга называется *LPBoost* (*linear programming boosting*).

Продемонстрируем метод LPBoost для двухклассовой классификации с ответами ± 1; о его применении для регрессии и многоклассовой классификации см. [32]. На обучающем наборе $(x_1, y_1), \ldots, (x_N, y_N)$ длины N можно обучить $M \leq 2^N$ различных "слабых" классификаторов $h^1, \ldots, h^M$. Как было анонсировано в разделе 2.3.3, в пространстве векторов-столбцов $h(x) = (h^1(x), \ldots, h^M(x))$ будем обучать линейный распознаватель на обучающем наборе $(h(x_1), y_1), \ldots, (h(x$ Предварительно обучить огромное количество M "слабых" распознавателей было бы нереально, но к счастью, с огромной вероятностью их понадобится совсем немного.

Пусть $h_i^j = h^j(x_i)$ — значение "слабого" классификатора на обучающем векторе, $h_i = h(x_i)$ — M-мерный вектор значений "слабых" классификаторов, $c = (c_1, \ldots, c_M)$ — M-мерный (ко-)вектор коэффициентов, определяющий линейную функцию $h \mapsto \sum_{j=1}^M c_j h^j$ на пространстве M-мерных векторов значений "слабых" классификаторов. Именно эту функцию и нужно обучать.

Задача обучения коэффициентов при "слабых" классификаторах ставится аналогично задаче (99) как минимизация суммарного штрафа за нарушение зазора, только с линейным, как в методе лассо (73), а не квадратичным регуляризирующим слагаемым. Но поскольку речь идет именно о голосовании классификаторов, то в распознавателе нет свободного члена, а все коэффициенты неотрицательны. В отличие от задачи (99) удобно регуляризирующее слагаемое, равное теперь сумме коэффициентов, зафиксировать равным 1, а зазор, наоборот, максимизировать. Получается линейная оптимизационная задача:

$$\rho - D \sum_{i=1}^N \xi_i \to \max_{c,\rho,\xi} \tag{201}$$

$$y_i \sum_{j=1}^M c_j h_i^j \geq \rho - \xi_i \text{ при } 1 \leq i \leq N \tag{202}$$

$$\sum_{j=1}^M c_j = 1 \tag{203}$$

$$c_j \geq 0 \text{ при } 1 \leq j \leq M \tag{204}$$
$$\xi_i \geq 0 \text{ при } 1 \leq i \leq N. \tag{205}$$

Решение задачи (201) зависит от параметра $D > 0$, определяющего, сколь сильно штрафовать несоблюдение желаемого зазора ρ. В отличие от параметра C в задаче (99), значение D зависит от длины N обучающего набора. Чтобы понять смысл параметра D, полезно записать его в виде $D = \frac{1}{\nu N}$ и переписать оптимизацию в задаче (201) в виде

$$\sum_{i=1}^{N}(\xi_i - \nu\rho) \to \min_{c,\rho,\xi}$$

(ср. с ν-SVC (166)).

Упражнение. Докажите, что при $D = \frac{1}{\nu N}$ задача (201) имеет конечные решения только при $\frac{1}{N} \leq \nu \leq 1$, т.е. $\frac{1}{N} \leq D \leq 1$. При этом не более $\frac{1}{D} = \nu N$ "обучающих векторов" h_i штрафуются ($\xi_i > 0$), и не менее νN имеют зазор не более ρ.

Если зазор ρ получился положительным, то имеются теоретические оценки (они приводятся в статье [32]) того, насколько средняя ошибка полученного классификатора может отличаться от ошибки на обучающем наборе, не превосходящей ν. Если зазор получился отрицательным, таких оценок нет.

Чтобы решать задачу (201) в пространстве признаков огромной размерности M нужно применить метод множителей Лагранжа: ввести N неотрицательных множителей α_i для неравенств (202) и множитель β для равенства (203)[41] и перейти к двойственной задаче в пространстве умеренной размерности N. Поиск точки максимума в (201) равносилен поиску седловых точек лагранжиана

$$L(c,\rho,\xi;\alpha,\beta) = \rho - D\sum_{i=1}^{N}\xi_i + \sum_{i=1}^{N}\alpha_i\left(y_i\sum_{j=1}^{M}c_j h_i^j - \rho + \xi_i\right) + \beta\left(1 - \sum_{j=1}^{M}c_j\right), \tag{206}$$

являющихся точками максимума по c, ρ, ξ и минимума по α, β, при ограничениях (202–205) и дополнительных ограничениях

$$\alpha_i \geq 0 \text{ при } 1 \leq i \leq N\ .$$

Приравнивая нулю производные L по c, ρ и ξ внутри их области определения и выписывая неположительность производных по внутренней нормали на границе, в частности, приравнивая нулю линейные по этим переменным слагаемые, получаем двойственную задачу:

$$\beta \to \min_{\alpha,\beta} \tag{207}$$
$$\sum_{i=1}^{N}\alpha_i y_i h_i^j \leq \beta \text{ при } 1 \leq j \leq M \tag{208}$$
$$\sum_{i=1}^{N}\alpha_i = 1$$
$$0 \leq \alpha_i \leq D \text{ при } 1 \leq i \leq N.$$

Упражнение. Просчитайте вывод двойственной задачи (207).

[41] Общая теория требует также вводить множители Лагранжа для всех неравенств (204) и (205), но в решении этой конкретной задачи лишние множители Лагранжа добавляют писанины, но не добавляют понимания.

Упражнение. Пусть (α^*, β^*) — решение двойственной задачи (207). Покажите, что в решении основной задачи (201) коэффициенты c_j являются множителями Лагранжа при неравенствах (208) в точке (α^*, β^*), в отличие от задач квадратичной оптимизации, где они линейно разлагаются по весам α_i (как, например, в равенстве (158)). В частности, $c_j = 0$ при $\sum_{i=1}^N \alpha_i y_i h_i^j < \beta$. А как найти зазор ρ и штрафы ξ_i?

Идея метода column generation (в используемых обозначениях порождаются не столбцы, а строки матрицы $\left(h_i^j\right)$) состоит в том, чтобы решать не сразу огромную задачу (201), а последовательность небольших ослабленных двойственных задач (207), в которых наложено лишь небольшое число ограничений (208). К началу m-ого шага ослабленная двойственная задача решена при $m-1$ ограничениях (208), т.е. она зависит от $m-1$ "слабых" классификаторов $h^{j_1}, \ldots, h^{j_{m-1}}$, а остальных "слабых" классификаторов хоть бы и вообще не было. На m-м шаге добавляется (т.е. обучается) еще один классификатор h^{j_m}, причем такой, для которого левая часть неравенства (208) — максимально возможная. Если неравенство (208) для h^{j_m} при текущих значениях α_i и β нарушено, новую ослабленную задачу (207) нужно решать снова, а если не нарушено, то обучение закончено: имеющимися средствами обучить классификатор, позволяющий увеличить "мягкий зазор" (201) невозможно. Напоминаем, что еще остается подбор параметра регуляризации D или, что то же самое, доли "опорных векторов" ν.

Поскольку и обучающие ответы y_i, и значения "слабых" классификаторов $h^j(x)$ равны ± 1, из вида левой части неравенства (208) следует, что на каждом шаге нужно обучать классификатор правильным ответам, минимизируя взвешенную суммарную ошибку по обучающему набору, что эквивалентно максимизации

$$\sum_{i=1}^N \alpha_i y_i h(x_i) = \sum_{i|h(x_i)=y_i} \alpha_i - \sum_{i|h(x_i)\neq y_i} \alpha_i = 1 - 2 \sum_{i|h(x_i)\neq y_i} \alpha_i \to \max_h \ .$$

Начальные веса α_i естественно брать равными $\frac{1}{N}$, т.е. первый "слабый" классификатор обучать минимизации настоящей ошибки обучения.

В результате получается алгоритм 11 (*LPBoost*).

Алгоритм 11: LPBoost

1. Вход: обучающий набор $T = ((x_1, y_1), \ldots, (x_N, y_N))$ длины N.
2. Назначить начальные веса $\alpha_i = \frac{1}{N}, i = 1, \ldots, N$.
3. Положить начальную верхнюю оценка зазора $\beta = 0$.
4. В цикле по m, начиная с $m = 1$
   (a) обучить "слабый" классификатор h^m на наборе T с весами α;
   (b) если взвешенная с весами α ошибка E классификатора h^m на наборе T с весами α достаточно велика, так что,
   $$1 - 2E = \sum_{i=1}^{N} \alpha_i y_i h^m(x_i) \leq \beta \ ,$$
   то выйти из цикла;
   (c) запомнить $h_i^m = h^m(x_i)$;
   (d) решить задачу линейной оптимизации
   $$\beta \to \min_{\alpha,\beta} \tag{209}$$
   $$\sum_{i=1}^{N} \alpha_i y_i h_i^j \leq \beta \text{ при } 1 \leq j \leq m \tag{210}$$
   $$\sum_{i=1}^{N} \alpha_i = 1$$
   $$0 \leq \alpha_i \leq D \text{ при } 1 \leq i \leq N \ ,$$
   используя предыдущий набор весов α и $\beta = \sum_{i=1}^{N} \alpha_i y_i h_i^m$ в качестве начального приближения.
5. Выход: классификатор $f(\cdot) = \operatorname{sign} \sum_{j=1}^{m-1} c_j h^j(\cdot)$, где $c_j \geq 0$ — множитель Лагранжа при j-м неравенстве (210) последней решенной задачи (209).

Хотя алгоритм 11 всегда сходится за конечное число шагов, в практических реализациях в него добавляют какие-либо дополнительные критерии остановки.

Напоминаем, что хотя на каждом шаге обучения нужно решать (например, симплекс-методом) вспомогательную задачу линейной оптимизации, ее нужно решать не с нуля, а начиная с некоторого довольно хорошего приближения, так что обычно решается она за небольшое число шагов. В ее решении набор весов α обычно получается разреженным, так что и обучение очередного "слабого" классификатора происходит не на всем обучающем наборе, а тем самым, быстрее, чем в ситуации общего положения.

Метод LPBoost обучает "слабые" классификаторы и оптимизирует их коэффициенты в голосовании, максимизируя мягкий зазор (201). Имеются и методы бустинга, оптимизирующие коэффициенты, непосредственно минимизируя выпуклую ошибку обучения (в частности, логистическую или экспоненциальную), и гарантирующие сходимость алгоритмов, см., например, [24].

5.4 Введение случайностей в обучение

Любой из описанных в предыдущих разделах методов бустинга является детерминированным процессом, если только обучение "слабых" распознавателей само по себе детерминировано. Но в обучение "слабых" распознавателей можно ввести случайности так же, как и при обучении независимых распознавателей в разделе 5.1.1: обучать их на случайных подмножествах обучающего набора и/или случайных подмножествах признаков. Эксперименты показывают (см., например, [40]), что при этом обучение самих "слабых" распознавателей существенно ускоряется (очевидно), а требуемое количество шагов бустинга почти не меняется.

6 Предварительное заключение

Плюнь тому в глаза, кто скажет, что можно обнять необъятное!
К.П.Прутков

В предыдущих разделах была в нескольких вариантах сформулирована постановка задач статистического обучения, и рассмотрены некоторые методы решения одной из них (задачи распознавания), как классические, представляющие только исторический и педагогический интерес, так и широко применяемые в настоящее время. Многие не менее интересные и полезные методы распознавания, а также методы решения других задач статистического обучения, не рассматривались исключительно ради краткости. 100- и даже 200-страничный вводный курс имеет некоторые преимущества перед 1000-страничным.

В этом разделе напоминаются рассмотренные в предыдущих разделах методы распознавания с описанием области их применимости, а также перечисляются некоторые не рассмотренные методы и задачи, заслуживающие рассмотрения в более подробном курсе статистического обучения.

6.1 Сравнительный обзор рассмотренных методов распознавания

Методы, про которые не сказано, классификация это или регрессия, применимы и для того, и для другого. Классификация, если не указано явно, может быть с любым числом классов. Признаки, если не описаны явно, предполагаются числовыми.

Метод ближайших соседей (раздел 1.1.3). Применим в пространствах малой размерности. Две творческих проблемы: как определить метрику, чтобы похожие с точки зрения решаемой задачи объекты были близкими, а пространство с такой метрикой было похоже на маломерное, и как быстро искать ближайшего соседа для выбранной, вообще говоря, нестандартной, метрики. Решение первой проблемы требует умения содержательно интерпретировать признаки.

Распознающие деревья (раздел 1.1.5). Обучаются быстро, распознают очень быстро, инвариантны относительно монотонных преобразований признаков, процесс распознавания легко интерпретировать содержательно. Но получающийся распознаватель сильно зависит от обучающего набора и, как правило, распознает не слишком хорошо. Не рассматривавшиеся выше более сложные методы обучения деревьев, такие как CART [20] и C4.5 [92], обучаются медленнее, но дают лучший распознаватель.

Чаще применяется не одно дерево, а результат голосования многих деревьев, см. ниже про случайные леса и бустинг.

Наивный байесовский классификатор (раздел 1.2.5). Применяется для классификации с дискретнозначными признаками, про которые либо есть соображения об их условной независимости, либо нет никаких соображений. Очень быстро обучается и быстро распознает. Может обучаться и распознавать при отсутствии некоторых значений признаков (*missing data*). При очень большом количестве признаков и очень большом обучающем наборе успешно конкурирует с другими методами распознавания.

Гребневая регрессия (раздел 2.1.2). Является регуляризованным вариантом классического метода наименьших квадратов. Допускает применения ядер Мерсера. Обучение распознавателя сводится к простым матричным вычислениям, включающем, однако, обращение матрицы большого (или очень большого) размера, равного либо размерности пространства признаков (+1), либо длине обучающего набора. Получающаяся линейная (в случае применения ядер — линейная в спрямляющем пространстве) регрессия зависит от всех признаков и от всех обучающих векторов. Применима, когда либо количество признаков, либо количество обучающих векторов, не слишком велико (порядка тысячи).

Лассо и другие виды регрессии с разреживающей регуляризацией (ра
2.1.2). В отличие от гребневой регрессии, не допускает применения ядер, зато строит линейную регрессию, зависящую не от всех признаков, а от некоторой доли "наиболее существенных". Позволяет обучаться и распознавать при очень большом количестве признаков (сотни тысяч), хотя при небольшом количестве признаков уступает гребневой регрессии.

Дискриминант Фишера (раздел 2.2.1). Линейный классификатор, предполагающий гауссовость условных распределений признаков для каждого ответа, и при этом предположении являющийся оптимальным. Как и для гребневой регрессии, обучение сводится к простым матричным вычислениям. Дает не только ответ классификации, но и количественную оценку вероятностей классов. На практике применяется крайне редко, только когда на логистическую регрессию (см. ниже) не хватает вычислительных мощностей.

Линейная логистическая регрессия (раздел 2.2.2). Применима для классификации при тех же условиях и обладает теми же свойствами, что дискриминант Фишера, только не требует никаких предположений о распределениях признаков. Обучается не прямым матричным вычислением, а итеративным процессом, каждый шаг которого по объему вычислений сравним с гребневой регрессией.

Перцептрон Розенблатта (раздел 2.2.3). Очень простой итеративный алгоритм обучения двухклассовой классификации, не всегда сходящийся. На практике не применяется.

Многослойные перцептроны (раздел 3.2). Позволяют обучить очень быстрые (при небольшом числе признаков) и вполне хорошие распознаватели. Но сам процесс обучения, хотя и растет с ростом длины обучающего набора не быстрее, чем линейно, весьма длителен и требует творчества на всех этапах: и конфигурацию сети, и начальные веса, и способ регуляризации, и последовательность, вообще говоря, разнотипных, шагов обучения нужно подбирать.

RBF-сети (раздел 3.3). Применяются аналогично многослойным перцептронам. Процесс обучения, тоже длительный, требует гораздо меньше твор-

чества. Хуже, чем перцептроны, обучаются при наличии шумовых признаков.

Метод опорных векторов (разделы 2.1.3, 2.2.4 и 4.2). Ориентирован на применение ядер Мерсера. Подбор подходящего ядра не автоматизируется. Время обучения довольно быстро растет с ростом длины обучающего набора. Обученный линейный (в спрямляющем пространстве) распознаватель зависит только от некоторой (если повезет, то небольшой) доли обучающих векторов. Применим при не слишком большой длине обучающего набора (десятки тысяч), а при малой его длине (десятки) превосходит прочие методы распознавания.

Классификация методом опорных векторов не оценивает вероятность классов. Обучение многоклассовой классификации заметно труднее, чем обучение двухклассовой.

Голосование независимых распознавателей, случайные леса (раздел 5.1. Недорогой, хотя и не гарантированный, способ из большого количества плохих базовых распознавателей получить один лучший. Применим для быстро обучающихся базовых распознавателей любых типов, обучаемых на разных обучающих наборах (или случайных поднаборах одного и того же) и/или разных случайных подмножествах признаков. При применении деревьев в качестве базовых распознавателей, наследует их инвариантность относительно монотонных преобразований признаков и способность работать с дискретными нечисловыми признаками. И обучение, и распознавание хорошо распараллеливаются.

Бустинг (разделы 2.3.3 и 5 кроме 5.1.1). Аналогичен случайным лесам (и вообще, голосованию), но обучение базовых распознавателей проводится не параллельно, а последовательно на одном и том же обучающем наборе, и базовые распознаватели должны уметь обучаться минимизировать ошибки с весами. Позволяет обучать весьма хорошие распознаватели.

Обучение многоклассовой классификации требует некоторых ухищрений.

6.2 Не рассмотренные постановки задач распознавания и методы обучения распознавателей

Вот несколько примеров методов и задач статистического обучения, идейно дополняющих методы, рассмотренные выше, иллюстрирующих байесовское обучение или максимизацию апостериорной вероятности, полезных для общего образования и широко применяемых на практике:

Максимизация ожидания Метод максимизации ожидания [33] был описан в урезанном виде и только для обучения RBF-сетей (раздел 3.3.1). На самом деле он применяется для обучения многих вероятностных моделей, распознавания с пропущенными признаками (*missing data*), *цензурированными данными*[42] (*censored data*), кластеризации и других задач и заслуживает более подробного рассмотрения.

Марковские модели Порождающие вероятностные модели были рассмотрены только в двух очень простых случаях: в "наивном Байесовском" предположении об условной независимости признаков при данных ответах

[42] Данные называются цензурированными, если про некоторые признаки или, чаще, ответы могут быть известны не их точные значения, а только какие-то ограничения на значения. Основной пример — время жизни больного, которое либо известно точно, либо только ограничено снизу периодом наблюдений.

(раздел 1.2.5) и в предположении о гауссовости условного распределения признаков при данных ответах (раздел 2.2.1). Для многих задач более адекватными являются несколько более сложные модели, в которых на множестве признаков определен граф зависимостей — ориентированный или неориентированный — и признаки, не соединенные ребрами, считаются условно независимыми при условии признаков, от которых они зависят явно (из которых в них идут ребра в ориентированном случае или с которыми они соединены ребрами в неориентированном). В частности, такие модели естественно применять когда признаки являются значениями какой-то величины, возможно, векторной, измеренными в разные моменты времени или в разных узлах решетки в двух-, трех- или более -мерном пространстве, например, при распознавании звуковых сигналов или изображений. Эти модели называются *марковскими цепями* в одномерном случае с ориентированным графом зависимостей, *марковскими полями* в многомерном с неориентированным графом или *байесовскими сетями* в общем случае, когда граф зависимостей ориентирован и нерегулярен. Обзор этих моделей приводится, например, в книге [15].

Но и марковские модели, оказываются слишком наивными. Довольно хорошего распознавания при удовлетворительной скорости обучения удается достичь, применяя более сложные *скрытые марковские модели* (*HMM*, *Hidden Markov Model*), в которых требования условной независимости накладываются на ненаблюдаемые величины — "состояния модели" — от которых наблюдаемые признаки зависят случайным образом. Обучение и применение скрытых марковских моделей описано, например, в обзорах [94, 10].

Наряду с порождающими марковскими моделями для распознавания применяются аналогичные им дискриминантные модели. Например, для распознавания изображений применяются аналоги марковских полей, называемые *условными случайными полями* (*CRF*, *Conditional Random Field*) [71], в которых условия условной независимости накладываются не не распределения наблюдаемых признаков (марковские модели) или скрытых состояний (скрытые марковские модели), а на условные распределения скрытых состояний относительно наблюдаемых признаков.

Байесовское обучение Байесовское обучение вероятностных моделей было продемонстрировано только на одном простом примере наивной байесовской модели (раздел 1.2.5). В более сложных случаях полноценное байесовское обучение, как правило, невозможно из-за трудоемкости многомерного интегрирования, но имеются методы, основанные на идеологии байесовского обучения, в которых интегрирование по пространству моделей успешно подменяется каким-либо его приближением.

Кроме того, сравнивая интегралы апостериорной вероятности обучающего набора по разным пространствам моделей (или по одному и тому же пространству, но с разными априорными распределениями), можно сравнивать "правдоподобность"[43] пространств моделей (соответственно, априорных распределений) и выбирать лучшие. Такой способ выбора априорного распределения равносилен автоматическому — без кросс-валидации или какой-либо другой экспериментальной проверки — подбору регуляризации, (ср. с переходом от максимизации (44) к минимизации (46)). Этот подход был разработан разными авторами к концу 80-х годов и хорошо изложен и проиллюстрирован в статье [80]. Более современной реализацией этого подхода является метод RVM.

[43] Формально используется термин *evidence*.

RVM *Метод уместных (релевантных) векторов* (*RVM*, *relevance vector machine* [112, 16, 111] разработан в нескольких вариантах М.Типпингом и соавторами около 2000 года и является удачной попыткой применить теорию байесовского обучения и автоматический подбор регуляризации для линейной регрессии и классификации. Этот метод, начиная с его названия, стилизован под метод опорных векторов (SVM) с ядром и применим в тех же задачах, что SVM, но, в отличие от SVM, для RVM требуется не обязательно ядро Мерсера, а любой набор базисных функций. На самом деле RVM является обобщением гребневой и логистической регрессии с автоматически подбираемой анизотропной квадратичной регуляризацией. Обученный распознаватель обычно получается еще более разреженным, чем SVM.

Некоторые из этих методов и задач рассмотрены в нижеследующих приложениях к основному тексту.

A Пропущенные данные и метод максимизации ожидания

A.1 Пропущенные данные

При сборе реальных данных для последующего обучения распознаванию и непосредственно для распознавания значения каких-то признаков могут оказаться неизвестными. Например,

1. значение признака, измеряемого каким-то прибором, может оказаться вне диапазона чувствительности этого прибора;

2. в социологических анкетах какие-то графы могут оказаться незаполненными по неизвестным причинам;

3. какие-то стандартные медицинские анализы больному не делаются по указанию врача (потому ли, что некогда, потому ли, что врач счел их несущественными для диагностики этого больного либо не показанными при его состоянии — причин может быть много);

4. точное значение признака в момент измерения принципиально не может быть известно (в частности, время жизни тестируемого электроприбора не известно, пока он не сгорит), хотя что-то про него известно (а именно, что он еще не сгорел).

Неизвестные значения из пунктов 1–3 обычно называются *пропущенными данными* (*missing data*), а частично известные (основной пример, как в пункте 4 — время жизни чего-либо) — цензурированными данными. Пропущенные данные подразделяются в соответствие с тем, как зависит или не зависит факт пропуска значений признаков от неизвестных значений этих признаков и от значений других признаков.

Если пропуск значения признака не зависит от значений пропущенных признаков, то этот признак называется *пропущенным случайно* (*missing at random* (*MAR*)), а если вообще не зависит от значений всех признаков, то *пропущенным вполне случайно* (*missing completely at random* (*MCAR*))[44]. В частности, в

[44] Формальное определение таково. Пусть r_j — случайная величина, зависящая от вектора признаков и равная 1, если j-й признак пропущен и 0, если он не пропущен; $\overset{+}{z}$ — вектор наблюдаемых (observed) признаков и $\overset{-}{z}$ — вектор пропущенных (missing). Тогда условие MAR

пункте 1 был приведен пример, не удовлетворяющий условию MAR, а в пункте 3 — пример, не удовлетворяющий условию MCAR, хотя, возможно, удовлетворяющий MAR.

Далее будет рассматриваться преимущественно случай вполне случайно пропущенных данных (MCAR), хотя некоторые общие соображения применимы и в других случаях. Распознавание с цензурированными данными — весьма важное в задачах прогноза выживаемости и весьма интересное с точки зрения применяемого математического аппарата — будет рассматриваться в разделе B. Подробности можно прочитать в книгах [77] (пропущенные данные) и [29] (цензурированные данные).

A.2 Распознавание и обучение с пропущенными данными

При наличии пропущенных данных в обучающем наборе и пропущенных признаков у распознаваемых объектов можно действовать различными способами, от кустарных до высоконаучных:

1. выбросить из обучающего набора примеры с пропущенными признаками и обучать распознаватель на остальных, а также обучать сразу целое семейство распознавателей, использующих разные подмножества признаков;

2. применять методы обучения распознавателей, способные игнорировать пропущенные данные;

3. как-то доопределять (*impute*) пропущенные данные в обучающем наборе до обучения и доопределять пропущенные признаки при распознавании;

4. наличие или отсутствие каждого признака также использовать как признак со значениями 0 и 1;

5. строить и обучать порождающую вероятностную модель.

Вариант 1 приемлем только когда пропущенные признаки встречаются очень редко. И то, когда, хотя бы и редко, у распознаваемого объекта есть пропущенные признаки, придется применять не распознаватель, использующий все признаки, а какой-то более слабый распознаватель, использующий только те признаки, которые известны. Кроме того, если условие MCAR не выполнено, то обучение производится на "статистически перекошенных" наборах.

Вариант 2 от варианта 1 отличается не принципиально, а только тем, какую долю могут составлять пропущенные признаки. К наличию пропущенных данных более устойчивы распознаватели, получающиеся в результате голосования большого семейства "маленьких" распознавателей, каждый из которых обучается на очень небольшом подмножестве признаков. Из ранее описанных распознавателей такими являются случайные леса (раздел 5.1) и, в предположении условной независимости признаков, аддитивные распознаватели, например, наивные байесовский классификаторы. Каждый "маленький" распознаватель обучается на подмножестве обучающего набора, для элементов которого какое-то небольшое подмножество признаков (для дерева случайного леса) или один признак (для наивного байесовского классификатора) определены. При распознавании в голосовании участвуют только те "маленькие" распознаватели, для которых определены все признаки.

состоит в равенстве условных вероятностей $P(r_j|\ \bar{z}, \overset{+}{z}) = P(r_j|\ \overset{+}{z})$, а условие MCAR — в равенстве условной вероятности безусловной: $P(r_j|\ \bar{z}, \overset{+}{z}) = P(r_j)$.

Применение наивного байесовского метода, модифицированного для обучения с пропущенными данными, будет подробно обсуждаться в разделе А.2.1.

Вариант 3 хорош тем, что после доопределения (*imputation*) пропущенных данных можно применять любые методы обучения распознавателей. Способов доопределения много и они существенно зависят от имеющейся априорной информации о возможных значениях пропущенных данных и о взаимозависмости признаков. Иногда даже удается обосновать оптимальность какого-либо способа доопределения при каких-либо предположениях о структуре данных и о допустимых алгоритмах распознавания. Приведем несколько простейших примеров.

При выполнении условия MCAR пропущенные значения дискретного признака (или ответа) доопределяются самым частым его значением, встречающимся в обучающем наборе (*модой*), а пропущенные значения непрерывного — *средним арифметическим* его значений, встречающихся в обучающем наборе (среднее арифметическое не инвариантно относительно нелинейных преобразований и обосновано только для применения линейных распознавателей), или *медианой* (медиана годится и для упорядоченных дискретных признаков).

В предположении выполнения условия MAR, но не MCAR, естественно было бы доопределять пропущенные значения условными средними, медианами или самыми частыми относительно присутствующих значений. Это практически осуществимо только если все признаки и ответы дискретны и количество реально встречающихся сочетаний значений всех признаков и ответов намного меньше длины обучающего набора.

При невыполнении условия MAR может оказаться полезнее заменять пропущенные значения признака не его "наиболее типичным" значением в обучающем наборе, как в предыдущих примерах, а наоборот, абсолютно нетипичным, в обучающем наборе не встречающемся, например, выходящим за диапазон чувствительности прибора, измеряющего этот признак. Подробнее эта идея развита в следующем варианте.

Вариант 4 хорош тем, что не требует никаких предположений о независимости пропуска признаков от значений, как пропущенных, так и присутствующих. При нем не теряется никакая информация и не вносятся никакие статистические искажения в обучающий набор. Пропущенные значения признаков нужно как-то доопределить, но как именно — не очень существенно, например, каким-нибудь фиксированным значением[45]. Добавленные признаки наличия признаков можно рассматривать и как дискретные, и как вещественнозначные, так что применимы любые методы обучения распознавателей. Есть только одна техническая проблема: поскольку признаков стало вдвое больше, для успешного обучения распознавателя может требоваться искать его в большем пространстве, а также, во избежание переобучения, потребуется больший обучающий набор, чем при обучении без пропущенных признаков. Например, даже если при отсутствии пропущенных признаков можно хорошо обучить линейный распознаватель, при их наличии хорошего линейного (в пространстве вдвое большей размерности) распознавателя может не найтись.

Вариант 5 обычно применяется в предположении выполнения условия MCAR; в противном случае нужно строить модель, описывающую совместное распределение не только признаков и ответов, но и пропусков их значений. Подробно он рассматривается ниже.

[45] Если это фиксированное значение не встречается как значение признака реального объекта, то без парного к этому признаку признака наличия можно обойтись.

A.2.1 Пример: наивное байесовское обучение с пропущенными данными

В наивной байесовской модели вероятности $p_\pi(x,y)$ представимы в виде произведения

$$p_\pi(x,y) = p_\pi(y)\prod_{j=1}^{d} p_\pi(x^j|y) = \pi_y \prod_{j=1}^{d} \pi^j_{x^j y}\ .$$

(см. формулу (30) и вводимые в следующем за ней абзаце обозначения π_k и π^j_{mk}). Пусть теперь у вектора признаков x некоторые признаки пропущены, $\overset{+}{D}_x \subset \{1,\ldots,d\}$ — множество индексов присутствующих признаков, $\overset{-}{D}_x = \{1,\ldots,d\}\setminus \overset{+}{D}_x$ — множество индексов пропущенных признаков, а $x = \overset{+}{x} \oplus \overset{-}{x}$ и $\mathcal{X} = \overset{+}{\mathcal{X}}_x \oplus \overset{-}{\mathcal{X}}_x$ — разложения вектора признаков x и всего пространства признаков на присутствующие и пропущенные в x признаки. Тогда вероятность того, что значения присутствующих признаков и ответа такие, какие есть, а значения пропущенных признаков — любые, равна

$$p_\pi(x,y) = p_\pi(\overset{+}{x},y) = \sum_{\overset{-}{x}\in\overset{-}{\mathcal{X}}_x} \left(\pi_y \prod_{j=1}^{d} \pi^j_{x^j y}\right) = \pi_y \prod_{j\in\overset{+}{D}_x} \pi^j_{x^j y}\ . \tag{211}$$

Аналогично, если пропущен еще и ответ y, то вероятность того, что значения присутствующих признаков такие, какие есть, т.е. $\overset{+}{x}$ и y, равна

$$p_\pi(x) = p_\pi(\overset{+}{x}) = \sum_{y\in\mathcal{Y}} \left(\pi_y \prod_{j\in\overset{+}{D}_x} \pi^j_{x^j y}\right)\ . \tag{212}$$

Рассмотрим сначала простейшую задачу двухклассовой классификации (пространство ответов $\mathcal{Y} = \{1,2\}$) по одному-единственному двузначному признаку (пространство признаков $\mathcal{X} = \{1,2\}$), и просчитаем несколько примеров. Наивные байесовские модели в этой ситуации задаются шестью числами π_1, π_2, π_{11}, π_{21}, π_{12} и π_{22} (верхний индекс, всегда равный 1, опущен), удовлетворяющими соотношениям $\pi_1 + \pi_2 = 1$, $\pi_{11} + \pi_{21} = 1$ и $\pi_{12} + \pi_{22} = 1$ и образующим трехмерный единичный куб $(\pi_1, \pi_{11}, \pi_{12}) \in [0,1]^3$. Априорную вероятность модели, как обычно, будем считать равномерной на этом кубе. Вместо байесовского обучения наивного байесовского классификатора (раздел 1.2.5), являющегося в случае единственного признака также и оптимальным байесовским (раздел 1.2.1), будем применять эквивалентную обучению формулу (26).

Пример. При обучающем наборе $T_1 = ((1,1),(2,2))$ отношение условных вероятностей ответов для "вектора" признаков $x = 1$ равно

$$\begin{aligned}\frac{p_{T_1}(y=1|x=1)}{p_{T_1}(y=2|x=1)} &= \frac{\int_\pi \pi_1\pi_{11}\cdot\pi_1\pi_{11}\cdot(1-\pi_1)(1-\pi_{12})d\pi}{\int_\pi (1-\pi_1)\pi_{12}\cdot\pi_1\pi_{11}\cdot(1-\pi_1)(1-\pi_{12})d\pi} \\ &= \frac{\int_0^1(\pi_1)^2(1-\pi_1)d\pi_1\cdot\int_0^1(\pi_{11})^2 d\pi_{11}\cdot\int_0^1(1-\pi_{12})d\pi_{12}}{\int_0^1\pi_1(1-\pi_1)^2 d\pi_1\cdot\int_0^1\pi_{11}d\pi_{11}\cdot\int_0^1\pi_{12}(1-\pi_{12})d\pi_{12}} \\ &= \frac{\frac{1}{12}\cdot\frac{1}{3}\cdot\frac{1}{2}}{\frac{1}{12}\cdot\frac{1}{2}\cdot\frac{1}{6}} = 2\end{aligned}$$

и аналогично

$$\frac{p_{T_1}(y=1|x=2)}{p_{T_1}(y=2|x=2)} = \frac{1}{2}\ .$$

При добавлении обучающих объектов с известным ответом, но неизвестным признаком, прогноз распознавателя смещается довольно естественным образом:

Пример. При обучающем наборе $T_2 = ((1,1),(2,2),(,2),(,2))$ отношение условных вероятностей ответов для "вектора" признаков $x = 1$ равно

$$\begin{aligned}\frac{p_{T_2}(y=1|x=1)}{p_{T_2}(y=2|x=1)} &= \frac{\int_\pi \pi_1\pi_{11}\cdot\pi_1\pi_{11}\cdot(1-\pi_1)(1-\pi_{12})\cdot(1-\pi_1)^2 d\pi}{\int_\pi (1-\pi_1)\pi_{12}\cdot\pi_1\pi_{11}\cdot(1-\pi_1)(1-\pi_{12})\cdot(1-\pi_1)^2 d\pi} \\ &= \frac{\int_0^1(\pi_1)^2(1-\pi_1)^3 d\pi_1 \cdot \int_0^1(\pi_{11})^2 d\pi_{11}\cdot\int_0^1(1-\pi_{12})d\pi_{12}}{\int_0^1 \pi_1(1-\pi_1)^4 d\pi_1\cdot\int_0^1\pi_{11}d\pi_{11}\cdot\int_0^1\pi_{12}(1-\pi_{12})d\pi_{12}} \\ &= \frac{\frac{1}{60}\cdot\frac{1}{3}\cdot\frac{1}{2}}{\frac{1}{30}\cdot\frac{1}{2}\cdot\frac{1}{6}} = 1\end{aligned}$$

и аналогично

$$\frac{p_{T_2}(y=1|x=2)}{p_{T_2}(y=2|x=2)} = \frac{1}{4}\ .$$

Менее очевидно изменяется прогноз распознавателя (оптимального! лучше не бывает) при добавлении обучающего объекта с неизвестным ответом.

Пример. При обучающем наборе $T_3 = ((1,1),(2,2),(,2),(,2),(1,))$

$$\begin{aligned}p_{T_3}(y=1|x=1) &= \int_\pi \pi_1\pi_{11}\cdot\pi_1\pi_{11}\cdot(1-\pi_1)(1-\pi_{12})\cdot(1-\pi_1)^2\cdot\pi_1\pi_{11}d\pi \\ &\quad + \int_\pi \pi_1\pi_{11}\cdot\pi_1\pi_{11}\cdot(1-\pi_1)(1-\pi_{12})\cdot(1-\pi_1)^2\cdot(1-\pi_1)\pi_{12}d\pi \\ &= \frac{3!3!}{7!}\frac{3!0!}{4!}\frac{0!1!}{2!} + \frac{2!4!}{7!}\frac{2!0!}{3!}\frac{1!1!}{3!} = \frac{1}{7!}\left(\frac{9}{2}+\frac{8}{3}\right)\ ,\end{aligned}$$

а

$$\begin{aligned}p_{T_3}(y=2|x=1) &= \int_\pi (1-\pi_1)\pi_{12}\cdot\pi_1\pi_{11}\cdot(1-\pi_1)(1-\pi_{12})\cdot(1-\pi_1)^2\cdot\pi_1\pi_{11}d\pi \\ &\quad + \int_\pi (1-\pi_1)\pi_{12}\cdot\pi_1\pi_{11}\cdot(1-\pi_1)(1-\pi_{12})\cdot(1-\pi_1)^2\cdot(1-\pi_1)\pi_1 \\ &= \frac{2!4!}{7!}\frac{2!0!}{3!}\frac{1!1!}{3!} + \frac{1!5!}{7!}\frac{1!0!}{2!}\frac{2!1!}{4!} = \frac{1}{7!}\left(\frac{8}{3}+5\right)\ ,\end{aligned}$$

так что отношение условных вероятностей ответов для "вектора" признаков $x = 1$ равно

$$\frac{p_{T_3}(y=1|x=1)}{p_{T_3}(y=2|x=1)} = \frac{\frac{9}{2}+\frac{8}{3}}{\frac{8}{3}+5} = \frac{43}{46} < 1$$

(ср. с обучением на наборе T_2).

Упражнение. Вычислить $\frac{p_{T_3}(y=1|x=2)}{p_{T_3}(y=2|x=2)}$.

В общем случае байесовское обучение и применение наивной байесовской модели, допускающее пропущенные признаки (но не пропущенные ответы в обучающем наборе: с пропущенными ответами все более громоздко) выглядят следующим образом. Для обучающего набора $T = ((x_1,y_1),\ldots,(x_N,y_N))$, в котором могут встречаться пропущенные признаки, вычисляются числа N_k (формула (31)), N_{mk}^j (формула (32)) и $N_k^j = \sum_{m=1}^{M^j} N_{mk}^j \le N_k$ — количество обучающих объектов k-го класса с присутствующим j-м признаком. В этом и состоит

обучение. Для распознаваемого вектора x с множеством присутствующих признаков $\overset{+}{D}_x$ и ответа y прогноз их совместной вероятности равен

$$P_T(x,y) = \frac{N_y + 1}{N + q} \prod_{j \in \overset{+}{D}_x}^{d} \frac{N^j_{x^j y} + 1}{N^j_y + M^j} \tag{213}$$

(ср. с формулой (35)).

Упражнение. Докажите равенство (213)

Упражнение. Выпишите дискриминант (37) для наивного байесовского распознавателя, допускающего пропущенные признаки.

A.2.2 Порождающие модели и пропущенные данные

Напомним идеологию применения параметрических моделирующих методов для распознавания, аналогичную минимизации эмпирического риска:

- построить конечномерное семейство моделей совместного распределения всех признаков и ответов, и, по возможности, определить на нем априорное распределение вероятностей моделей;
- постараться выбрать в нем модель с наибольшей апостериорной вероятностью или правдоподобием ("обучить модель");
- с помощью этой обученной модели по совместному распределению посчитать условное распределение ответов при условии признаков и именно его использовать в качестве распознавателя.

Последний пункт применим и в ситуации пропущенных данных, когда известны значения не всех признаков. Только условное распределение ответов при условии всех признаков можно вычислить однократно после обучения модели, а условные распределения ответов при условии части признаков придется вычислять при распознавании, вообще говоря, многократно, в зависимости от того, значения каких признаков распознаваемого объекта известны.

Упражнение. Дискриминант Фишера (раздел 2.2.1) является порождающей моделью, обучаемой методом наибольшего правдоподобия. Как его обучать и применять при наличии пропущенных данных?

В следующем разделе описывается еще один метод обучения порождающих моделей, допускающий и обучение, и распознавание с пропущенным данными. Более того, он позволяет обучать и применять модели с фиктивными признаками, значения которых всегда пропущены и при обучении, и при распознавании, но предположение о существовании которых позволяет строить более адекватные модели.

A.3 Метод максимизации ожидания

A.3.1 Обозначения

Перенесем обозначения раздела 1.1.4 на случай допустимости пропуска данных и введем дополнительные обозначения:

Для одного объекта (i-го):

x_i — вектор признаков (d-мерный, часть значений признаков может быть пропущена);

y_i — ответ (q-мерный вектор, часть ответов может быть пропущена);

$z_i = (x_i, y_i)$ — пара признаки-ответ(ы);

v_i — фиктивные признаки (или векторы признаков) (*latent variables*), придуманные для упрощения модели.

Те же самые векторы, компоненты которых перегруппированы в соответствии с тем, известно ли их значение:

$\overset{+}{x}_i$ — вектор присутствующих признаков;

$\overset{+}{y}_i$ — присутствующие ответы;

$\overset{+}{z}_i = (\overset{+}{x}_i, \overset{+}{y}_i)$ — все наблюдаемые параметры (присутствующие признаки и ответы);

$\bar{x}_i$, $\bar{y}_i$ и $\bar{z}_i$ — пропущенные признаки, ответы и все пропущенные параметры;

$u_i = (v_i, \bar{z}_i)$ — все ненаблюдаемые параметры, т.е. и фиктивные признаки, и пропущенные параметры.

Для обучающего набора:

$T = ((x_1, y_1), \ldots, (x_N, y_N))$ — обучающий набор, в котором значения каких-то признаков и ответов могут быть пропущены;

$\mathbf{v} = (v_1, \ldots, v_N)$ — все фиктивные признаки,

$\overset{+}{\mathbf{z}} = (\overset{+}{z}_1, \ldots, \overset{+}{z}_N)$ — все наблюдаемые параметры (присутствующие признаки и ответы) всего обучающего набора;

$\bar{\mathbf{z}} = (\bar{z}_1, \ldots, \bar{z}_N)$ — все пропущенные параметры всего обучающего набора;

$\mathbf{u} = (u_1, \ldots, u_N) = (\mathbf{v}, \bar{\mathbf{z}})$ — все ненаблюдаемые параметры (признаки и ответы) всего обучающего набора; таким образом, $(\overset{+}{\mathbf{z}}, \mathbf{u})$ — это все параметры обучающего набора — и признаки (наблюдаемые, ненаблюдаемые, фиктивные), и ответы (наблюдаемые, ненаблюдаемые);

ω — распределение вероятностей на пространстве всех ненаблюдаемых параметров.

Для вероятностной модели одного объекта:

f — модель распределения вероятностей $p(x_i, y_i, v_i)$ для каждого объекта, снабженного фиктивными признаками, принадлежащая некоторому пространству моделей $\mathcal{F}$.

f считается случайной величиной с априорным распределением $p_0(f)$ на $\mathcal{F}$, так что распределение $p(x_i, y_i, v_i)$ при конкретном значении f является условным и, соответственно, обозначается $p(x_i, y_i, v_i|f)$; по нему вычисляются распределения $p(x_i, y_i|f) = \int_{v_i} p(x_i, y_i, v_i|f)dv_i$ и $p(\overset{+}{z}_i, u_i|f)$ (то же распределение $p(x_i, y_i, v_i|f)$, но с переупорядоченными аргументами).

Для вероятностной модели всего обучающего набора:

$p(T, \mathbf{v}|f)$ или $p(\overset{+}{\mathbf{z}}, \mathbf{u}|f)$ (это то же самое распределение, но с переупорядоченными аргументами); как обычно, элементы обучающего набора будут предполагаться независимыми, так что $p(T, \mathbf{v}|f) = \prod_{i=1}^{N} p(x_i, y_i, v_i|f)$ или $p(\overset{+}{\mathbf{z}}, \mathbf{u}|f) = \prod_{i=1}^{N} p(\overset{+}{z}_i, u_i|f)$.

В примерах будут также использоваться обозначения $N^d_{(\mu,A)}$ для плотности d-мерного гауссова распределения с центром μ и матрицей ковариаций A

$$N^d_{(\mu,A)}(x) = (2\pi)^{-\frac{d}{2}}|A|^{-\frac{1}{2}}e^{-\frac{1}{2}A^{-1}(x-\mu,x-\mu)} \ , \tag{214}$$

а также $N^d_{(\mu,\alpha)}$ для сферического гауссова распределения с центром μ и диагональной матрицей ковариаций αI^d.

$$N^d_{(\mu,\alpha)}(x) = (2\pi\alpha)^{-\frac{d}{2}}e^{-\frac{\|x-\mu\|^2}{2\alpha}} \ . \tag{215}$$

В тех случаях, когда размерность распределения видна из контекста, будем писать просто $N_{(\mu,A)}$.

Далее, за исключением нескольких конкретных примеров, со всеми переменными и распределениями будем обращаться как с непрерывными и, соответственно, писать их плотности и интегралы по ним. На дискретный случай все естественным образом тоже переносится. В частности, ожидание $\mathbf{E}_{\omega;\mathbf{u}}g(\cdot,\mathbf{u})$ зависящей от $\mathbf{u}$ (и еще чего-то) величины g по распределению $\mathbf{u}$ с плотностью ω будем для наглядности расписывать через интеграл:

$$\mathbf{E}_{\omega;\mathbf{u}}g(\cdot,\mathbf{u}) = \int g(\cdot,\mathbf{u})\omega(\mathbf{u})d\mathbf{u} \ ,$$

хотя в дискретном случае нужно писать сумму

$$\mathbf{E}_{\omega;\mathbf{u}}g(\cdot,\mathbf{u}) = \sum_j g(\cdot,u_j)\omega(u_j) \ .$$

Пространство моделей $\mathcal{F}$, как правило, параметризовано областью евклидова пространства, хотя может быть и дискретным, и еще каким-нибудь. Распределения $p(x,y,v|f)$ будем считать непрерывными функциями от всех своих параметров, включая f. Тогда непрерывными будут и вычисляемые по ним условные распределения, их ожидания, логарифмы и т.п.

Последующие рассуждения применимы и к весьма вырожденному, но широко распространенному, частному случаю, когда ответ "у" всегда принимает одно и то же фиксированное значение или, что эквивалентно, всегда отсутствует. Естественно, получающиеся модели распределения признаков не могут использоваться непосредственно для распознавания, но могут применяться в качестве составных частей распознавателей (см., например, раздел 3.3.1).

Цель обучения модели — максимизация плотности апостериорной вероятности $p(f|\overset{+}{\mathbf{z}})$ модели f при условии наблюдаемых параметров $\overset{+}{\mathbf{z}}$ обучающего набора, что равносильно максимизации плотности вероятности

$$p(\overset{+}{\mathbf{z}},f) = p_0(f)p(\overset{+}{\mathbf{z}}|f) = p_0(f)\int p(\overset{+}{\mathbf{z}},\mathbf{u}|f)d\mathbf{u} \to \max_f \ , \tag{216}$$

или, что то же самое, ее логарифма

$$L(f;\overset{+}{\mathbf{z}}) = \ln p_0(f) + \ln p(\overset{+}{\mathbf{z}}|f) = \ln p_0(f) + \ln\int p(\overset{+}{\mathbf{z}},\mathbf{u}|f)d\mathbf{u} \to \max_f \ . \tag{217}$$

Далее все плотности вероятностей предполагаются всюду положительными, чтобы логарифмы были всюду определены, хотя на самом деле это требование не принципиально. Во всех рассуждениях параметр $\overset{+}{\mathbf{z}}$ (все имеющееся знание про обучающий набор) будет фиксирован и функции будут рассматриваться как функции от остальных аргументов.

Пример. Гауссова смесь (раздел 3.3) и гребневая регрессия (раздел 2.1.2).

- Признаки обучающих векторов $x_i \in \mathcal{X} = \mathbb{R}^d$, $i = 1, \ldots, N$, часть из которых неизвестна.

- Ответы обучающих векторов $y_i \in \mathcal{Y} = \mathbb{R}$, $i = 1, \ldots, N$, часть из которых неизвестна.

- Совокупность признаков и ответов обучающего набора длины N — это $(d+1)N$ чисел, часть из которых известны, и вместе образуют вектор $\overset{+}{\mathbf{z}}$ некоторой размерности $\overset{+}{N}$, а часть — неизвестны и вместе образуют случайный вектор $\bar{\mathbf{z}}$ размерности $\bar{N} = (d+1)N - \overset{+}{N}$.

- Кроме того каждому обучающему вектору (x_i, y_i) приписан ненаблюдаемый признак v_i, с конечным множеством значений $\{1, \ldots, k\}$.

- Модель f состоит из линейной функции $l : \mathcal{X} \to \mathcal{Y}$, $l(x) = wx + b$, положительного числа β, k точек $\mu_j \in \mathcal{X}$ и k положительно определенных квадратичных форм A_j на $\mathcal{X}$, и определяет условную вероятность $p(x, y, v|f)$ как произведение не зависящего от v одномерного гауссиана в $\mathcal{Y}$, зависящего от v гауссиана в $\mathcal{X}$ и зависящего только от v положительного коэффициента:

$$\begin{aligned} p(x, y, v|f) &= p(y|x, f)\, p(x|v, f)\, P_v \\ &= N^1_{(0,\beta)}(y - l(x))\, N^d_{(\mu_v, A_v)}(x)\, P_v \ . \end{aligned} \tag{218}$$

  Обучающие векторы считаются независимыми, так что

$$p((x_1, y_1, v_1), \ldots, (x_N, y_N, v_N)|f) = \prod_{i=1}^{N} p(x_i, y_i, v_i|f) \ . \tag{219}$$

- Априорное распределение вероятностей модели f зависит только от w и является d-мерным сферическим гауссианом с центром в 0:

$$p_0(f) = N^d_{(0,\alpha)}(w) \ , \tag{220}$$

  где $\alpha > 0$ выполняет роль параметра регуляризации и подбирается эмпирически.

Гауссианы $p(y|x, f)$ и $p_0(f)$ в этом примере — в точности те, которые приводят к методу гребневой регрессии(раздел 2.1.2), т.е. минимизации квадратичной ошибки с квадратичной регуляризацией. А "проинтегрированная" (просуммированная) по v гауссова смесь

$$p(x|f) = \sum_{v=1}^{k} p(x, v|f) = \sum_{v=1}^{k} p(x|v, f) p(v)$$

описывает плотность распределения векторов признаков. Вся же модель описывает распределение пар (x, y) (признаки-ответ) и позволяет вычислять условные распределения вида $p(y|\overset{+}{x})$, т.е. распознавать даже когда не все признаки известны.

A.3.2 Алгоритм EM для обучения порождающей модели

Даже при несложных распределениях $p(\overset{+}{\mathbf{z}}, \mathbf{u}|f)$ искать максимум логарифма апостериорной вероятности функции $L(f; \overset{+}{\mathbf{z}})$ аналитически, дифференцируя L

по f под логарифмом интеграла в формуле (217) и приравнивая производные нулю — дело, как правило, безнадежное. Но функцию L можно оценить снизу более простой функцией L_ω, зависящей еще от распределения ω ненаблюдаемых параметров $\mathbf{u}$, предполагаемого всюду положительным:

$$L(f;\overset{+}{\mathbf{z}}) = \ln p_0(f) + \ln \int p(\overset{+}{\mathbf{z}},\mathbf{u}|f)d\mathbf{u} = \ln \int \frac{p_0(f)p(\overset{+}{\mathbf{z}},\mathbf{u}|f)}{\omega(\mathbf{u})}\omega(\mathbf{u})d\mathbf{u}$$
$$= \ln \mathbf{E}_{\omega;\mathbf{u}} \frac{p_0(f)p(\overset{+}{\mathbf{z}},\mathbf{u}|f)}{\omega(\mathbf{u})} \geq \mathbf{E}_{\omega;\mathbf{u}} \ln \frac{p_0(f)p(\overset{+}{\mathbf{z}},\mathbf{u}|f)}{\omega(\mathbf{u})} = L_\omega(f;\overset{+}{\mathbf{z}}) \quad (221)$$

(неравенство следует из вогнутости логарифма). Оценивающая функция L_ω представима в виде разности

$$L_\omega(f;\overset{+}{\mathbf{z}}) = \mathbf{E}_{\omega;\mathbf{u}} \ln p_0(f) + \mathbf{E}_{\omega;\mathbf{u}} \ln p(\overset{+}{\mathbf{z}},\mathbf{u}|f) - \mathbf{E}_{\omega;\mathbf{u}} \ln \omega(\mathbf{u})$$
$$= \ln p_0(f) + \mathbf{E}_{\omega;\mathbf{u}} \ln p(\overset{+}{\mathbf{z}},\mathbf{u}|f) + H(\omega)\ , \quad (222)$$

где априорная вероятность модели f от распределения ω не зависит и ее логарифм из-под ожидания вынесен, а последнее слагаемое — зависящая только от распределения ω его *энтропия* $H(\omega) = -\mathbf{E}_{\omega;\mathbf{u}} \ln \omega(\mathbf{u})$. Хотя максимизация

$$L_\omega(f;\overset{+}{\mathbf{z}}) \to \max_{f,\omega}\ ,$$

на первый взгляд кажется еще сложнее, чем максимизация $L(f;\overset{+}{\mathbf{z}})$ (217), сейчас будет предъявлен итеративный алгоритм, достаточно простой при просто устроенных логарифмах плотности $\ln p(\overset{+}{\mathbf{z}},\mathbf{u}|f)$ и почти всегда сходящийся к решениям задачи (217).

Этот итеративный алгоритм называется *алгоритмом максимизации ожидания*, например, потому, что максимизируемая функция $L_\omega(f;\overset{+}{\mathbf{z}})$ определяется как ожидание некоторой величины[46]. Идея алгоритма состоит в чередовании двух увеличивающих значение функции $L_\omega(f;\overset{+}{\mathbf{z}})$ шагов, которые в англоязычной традиции называются половинками названия алгоритма — *expectation maximization* или *EM*:

E(xpectation) максимизация (*ожидания*) $L_\omega(f;\overset{+}{\mathbf{z}})$ по распределению ω при фиксированной модели f;

M(aximization) *максимизация* (ожидания) $L_\omega(f;\overset{+}{\mathbf{z}})$ по модели f при фиксированном распределении ω.

[46] Исторически слово “ожидание” в названии алгоритма появилось в несколько другом смысле, см. ссылку 47 на стр. 159

Алгоритм 12: Максимизация ожидания (EM); современная

1. Вход: обучающий набор $T = ((x_1, y_1), \ldots, (x_N, y_N))$, в котором некоторые признаки и/или ответы могут быть пропущены: $\overset{+}{\mathbf{z}}$ — вектор всех присутствующих значений.

2. Выбрать пространство фиктивных признаков v, пространство моделей распределения вероятностей троек (x, y, v), априорное распределение p_0 на пространстве моделей и начальную модель $f = f^{(0)}$.

3. В цикле

   E максимизировать ожидание $L_\omega(f; \overset{+}{\mathbf{z}}) = \mathbf{E}_{\omega;\mathbf{u}} \ln \frac{p_0(f)p(\overset{+}{\mathbf{z}},\mathbf{u}|f)}{\omega(\mathbf{u})}$ по распределению ω всех ненаблюдаемых величин (фиктивных и пропущенных) при фиксированной модели f;

   M максимизировать $L_\omega(f; \overset{+}{\mathbf{z}})$ по f при фиксированном распределении ω;

   и выйти из цикла, если модель f не изменилась.

4. Выход: порождающая модель f.

Для шага **E** точка максимума не зависит от априорного распределения $p_0(f)$ и предъявляется явно:

Предложение 32 *Условное распределение ненаблюдаемых параметров* $\mathbf{u}$ *при наблюдаемых параметрах* $\overset{+}{\mathbf{z}}$ *и модели* f

$$\omega_f(\mathbf{u}) = p(\mathbf{u}|\overset{+}{\mathbf{z}}, f) = \frac{p(\overset{+}{\mathbf{z}}, \mathbf{u}|f)}{p(\overset{+}{\mathbf{z}}|f)} \tag{223}$$

является единственной точкой глобального максимума $L_\omega(f; \overset{+}{\mathbf{z}})$ *по* ω*, и в ней достигается равенство* $L_{\omega_f}(f; \overset{+}{\mathbf{z}}) = L(f; \overset{+}{\mathbf{z}})$.

Доказательство. Точка глобального максимума $L_\omega(f; \overset{+}{\mathbf{z}})$ по ω является решением вариационной задачи

$$\int \ln \frac{p(\overset{+}{\mathbf{z}}, \mathbf{u}|f)}{\omega(\mathbf{u})} \omega(\mathbf{u}) d\mathbf{u} \to \max_{\omega | \int \omega(\mathbf{u}) d\mathbf{u} = 1} .$$

Выпишем лагранжиан и приравняем нулю его вариационную производную:

$$\begin{aligned} 0 &= \frac{\delta}{\delta\omega} \left(\int \ln \frac{p(\overset{+}{\mathbf{z}}, \mathbf{u}|f)}{\omega(\mathbf{u})} \omega(\mathbf{u}) d\mathbf{u} - \lambda \left(\int \omega(\mathbf{u}) d\mathbf{u} - 1 \right) \right) \\ &= \int \left(\ln \frac{p(\overset{+}{\mathbf{z}}, \mathbf{u}|f)}{\omega(\mathbf{u})} - \lambda - 1 \right) \delta\omega(\mathbf{u}) d\mathbf{u} . \end{aligned}$$

Поскольку вариация $\delta\omega$ произвольна, при любом ненаблюдаемом $\mathbf{u}$

$$\ln \frac{p(\overset{+}{\mathbf{z}}, \mathbf{u}|f)}{\omega(\mathbf{u})} - \lambda - 1 = 0 ,$$

т.е.

$$\omega(\mathbf{u}) = \frac{p(\overset{+}{\mathbf{z}}, \mathbf{u}|f)}{e^{\lambda+1}} \ .$$

Интегрируя это равенство по пространству ненаблюдаемых параметров, получаем

$$1 = \int \omega(\mathbf{u}) d\mathbf{u} = \int \frac{p(\overset{+}{\mathbf{z}}, \mathbf{u}|f)}{e^{\lambda+1}} d\mathbf{u} = \frac{p(\overset{+}{\mathbf{z}}\,|f)}{e^{\lambda+1}} \ ,$$

значит

$$\omega(\mathbf{u}) = \frac{p(\overset{+}{\mathbf{z}}, \mathbf{u}|f)}{p(\overset{+}{\mathbf{z}}\,|f)} = p(\mathbf{u}|\overset{+}{\mathbf{z}}, f)$$

и это распределение ω является единственной экстремальной точкой функции $L_\omega(f; \overset{+}{\mathbf{z}})$. То, что оно является точкой глобального максимума, проверяется непосредственно:

$$\begin{aligned} L_{\omega_f}(f; \overset{+}{\mathbf{z}}) &= \ln p_0(f) + \mathbf{E}_{\omega_f;\mathbf{u}} \ln \frac{p(\overset{+}{\mathbf{z}}, \mathbf{u}|f)}{p(\mathbf{u}|\overset{+}{\mathbf{z}}, f)} = \ln p_0(f) + \mathbf{E}_{\omega_f;\mathbf{u}} \ln p(\overset{+}{\mathbf{z}}\,|f) \\ &= \ln p_0(f) + \ln p(\overset{+}{\mathbf{z}}\,|f) = L(f; \overset{+}{\mathbf{z}}) \end{aligned}$$

и глобальность максимума следует из неравенства (221). □

Следствие 7 *На каждой итерации алгоритма 12 плотность апостериорной вероятности $p(f|\overset{+}{\mathbf{z}})$ модели f, равно как и значение функции $L(f; \overset{+}{\mathbf{z}})$ (217), строго возрастает.*

Для шага **M** универсального способа поиска точки максимума нет, но максимизируемое выражение можно несколько упростить:

Предложение 33 *Любая точка f, максимизирующая $L_\omega(f; \overset{+}{\mathbf{z}})$, является точкой максимума выражения*

$$\ln p_0(f) + \mathbf{E}_{\omega;\mathbf{u}} \ln p(\overset{+}{\mathbf{z}}, \mathbf{u}|f) = L_\omega(f; \overset{+}{\mathbf{z}}) - H(\omega) \tag{224}$$

□

В отличие от шага **E**, во-первых, на шаге **M** точек максимума может быть много, и во-вторых, в них выполнение равенства $L_\omega(f; \overset{+}{\mathbf{z}}) = L(f; \overset{+}{\mathbf{z}})$ не гарантировано.

Шаг **M** особенно удобно формулировать для регулярных экспоненциальных семейств распределений. Напомним, что регулярным экспоненциальным семейством распределений называется зависящее от параметра $\theta \in \Theta \subset \mathbb{R}^n$ семейство распределений величины $x \in \mathcal{X}$ с плотностями (для непрерывной величины) или вероятностями (для дискретной) вида

$$p(x|\theta) = \frac{h(x)}{g(\theta)} e^{(\phi(x),\theta)} \ ,$$

где $\phi : \mathcal{X} \to \mathbb{R}^n$ — некоторое отображение в евклидово пространство $\mathbb{R}^n$, а $h : \mathcal{X} \to \mathbb{R}$ и $g : \Theta \to \mathbb{R}$ — числовые функции. Функция g однозначно определена нормировочным условием $\int p(x|\theta) dx = 1$:

$$g(\theta) = \int h(x) e^{(\phi(x),\theta)} dx \ .$$

При обучении порождающих моделей для распознавания речь идет о семействах распределений на произведении $\mathcal{X} \times \mathcal{Y}$, а не просто на пространстве признаков $\mathcal{X}$. Для регулярных экспоненциальных семейств необходимым условием максимума выражения (224) является равенство

$$0 = \frac{\partial}{\partial f}\left(\ln p_0(f) + \mathbf{E}_{\omega;\mathbf{u}} \ln p(\overset{+}{\mathbf{z}}, \mathbf{u}|f)\right) = \frac{\partial}{\partial f} \ln \frac{p_0}{g^N}(f) + \mathbf{E}_{\omega;\mathbf{u}}\left(\sum_{i=1}^{N} \phi(\overset{+}{z}_i, u_i)\right) ,$$

которое, в зависимости от конкретного вида функций p_0 и g, может оказаться легко и однозначно разрешимым относительно f, а может и не оказаться.

Упражнение. Покажите, что семейство гауссовых распределений (214), параметризованное парой (μ, A), является регулярным экспоненциальным с параметром $\theta = (-\frac{1}{2}A^{-1}, A^{-1}\mu) \in \mathbb{R}^{d^2+d}$ и отображением $\phi(x) = ((x^j x^k, 1 \le j, k \le d), x)$ (на самом деле из-за симметричности матрицы A размерность пространства моделей равна не $d^2 + d$, а $\frac{d(d+1)}{2} + d$). Вычислите функции h и g.

Предложения 32 и 33 позволяют переформулировать алгоритм максимизации ожидания 12, заменив в шаге **E** максимизацию готовым ответом (223) и несколько упростив шаг **M**. Получается алгоритм 13, в котором, в соответствии с исторической традицией, предварительные действия из шага **M** перенесены в шаг **E**.

Алгоритм 13: Максимизация ожидания (EM); классическая

1. Вход: обучающий набор $T = ((x_1, y_1), \ldots, (x_N, y_N))$, в котором некоторые признаки и/или ответы могут быть пропущены: $\overset{+}{\mathbf{z}}$ — вектор всех присутствующих значений.

2. Выбрать пространство фиктивных признаков v, пространство моделей распределения вероятностей троек (x, y, v), априорное распределение p_0 на нем и начальную модель $f = f^{(0)}$.

3. В цикле

   E по f вычислить условное распределение ω_f (223) вектора $\mathbf{u}$ всех ненаблюдаемых параметров

   $$\omega_f(\mathbf{u}) = p(\mathbf{u}|\overset{+}{\mathbf{z}}, f) ;$$

   и определить (вычислять значения в точках пока не надо) зависящую от этого распределения функцию от модели f'

   $$Q(f', f; \overset{+}{\mathbf{z}}) = \ln p_0(f') + \mathbf{E}_{\omega_f;\mathbf{u}} \ln p(\overset{+}{\mathbf{z}}, \mathbf{u}|f') ; \qquad (225)$$

   M максимизировать $Q(f', f; \overset{+}{\mathbf{z}})$ по модели f' при фиксированной модели f

   если $f' \ne f$, положить $f = f'$, иначе выйти из цикла.

4. Выход: порождающая модель f.

Замечание. В большинстве описаний метода максимизации ожидания в литературе максимизируется не апостериорная вероятность $p(f|\overset{+}{\mathbf{z}})$, что равно-

сильно максимизации $p(\overset{+}{\mathbf{z}}, f)$ (216), а правдоподобие $p(\overset{+}{\mathbf{z}}\,|f)$. При очень большом обучающем наборе разница между максимизацией $p(\overset{+}{\mathbf{z}}\,|f)$ и $p(\overset{+}{\mathbf{z}}, f) = p_0(f)p(\overset{+}{\mathbf{z}}\,|f)$ обычно пренебрежима, а при небольшом — может оказаться весьма существенной.

Упражнение. Нетривиальное априорное распределение вероятностей может быть определено не только для параметров модели f, но и для признаков. Например, оно есть при доучивании уже когда-то обученной модели на новом обучающем наборе. Сформулируйте алгоритм максимизации ожидания как итеративную максимизацию апостериорной вероятности в таком общем случае.

Конечно, не при всяком пространстве моделей и не при всяком априорном распределении алгоритм максимизации ожидания можно эффективно реализовать.

Историко-литературное замечание. Алгоритмы, похожие на алгоритм максимизации ожидания, появлялись под разными названиями в решении разных задач построения оптимальных вероятностных моделей в 1950–60-ые годы. Например, алгоритм Баума-Велша [8] для обучения марковских моделей является частным случаем алгоритма 13. В статье [33] было введено название "максимизация ожидания" [47], разделение алгоритма на шаги **E** и **M** и продемонстрирована применимость алгоритма к любым пропущенным данным, а не только к вероятностным моделям специального вида. Удобная для исследования и обобщения формулировка алгоритма 12 появилась в работах 1980-х годов, а взята из статьи [88].

A.3.3 Примеры обучения с помощью алгоритма EM

Самый популярный пример применения алгоритма 13 — оценка параметров гауссовой смеси. Этот пример уже был рассмотрен в разделе 3.3.1 в несколько специфическом контексте и несколько других обозначениях. При этом моделируется только распределение вектора признаков $x \in \mathbb{R}^d$, без ответов, к которым добавляется всегда неизвестный, фиктивный признак v с целочисленными значениями от 1 до k. Модель f (в разделе 3.3.1 — $\mathbf{W}$) имеет вид

$$p(x, v|f) = P_v N^d_{(\mu_v, s_v)}(x), \quad v = 1, \ldots, k\ . \tag{226}$$

(ср. с формулой (127)) Она определяется $2k$ положительными числами P_v и s_v с соотношением $\sum_{v=1}^k P_v = 1$ и k d-мерными векторами μ_v. Априорная плотность вероятности $p_0(f)$ модели f полагается равной 1, т.е. вместо максимизации апостериорной вероятности модели производится максимизация правдоподобия.

Если, как в в разделе 3.3.1, все векторы признаков x_i полностью наблюдаемы, то шаг **E** алгоритма 13 состоит в вычислении по модели f распределения ω_f всех ненаблюдаемых параметров, в данном случае всех фиктивных признаков v_i векторов обучающего набора. Поскольку векторы обучающего набора независимы, логарифм условной вероятности аддитивен

$$\ln p(\overset{+}{\mathbf{z}}, \mathbf{u}|f) = \sum_{i=1}^N \ln p(x_i, v_i|f)$$

и его ожидание по N-мерному распределению $\omega(\mathbf{v})$ равно сумме ожиданий по одномерным (marginal) распределениям $\omega^i(v_i)$:

$$\mathbf{E}_{\omega;\mathbf{u}} \ln p(\overset{+}{\mathbf{z}}, \mathbf{u}|f) = \sum_{\mathbf{v}} \omega(\mathbf{v}) \sum_{i=1}^N \ln p(x_i, v_i|f) = \sum_{i=1}^N \sum_{v_i=1}^k \omega^i(v_i) \ln p(x_i, v_i|f)\ .$$

[47] Под *ожиданием* понималось ожидание по старой модели логарифма правдоподобия новой (второе слагаемое формулы (225)), а первое полагалось равным нулю.

Энтропия распределения ω с фиксированными одномерными распределениями ω^i максимальна(!) при $\omega(\mathbf{v}) = \prod_{i=1}^N \omega^i(v_i)$, поэтому в алгоритме максимизации ожидания будут получаться только такие распределения и его шаг **E** сводится к вычислению условных вероятностей $P\{j|x_i, f\}$ (ср. с формулой (128)) и выписывании остающегося от формулы (225) при $p_0 = 1$ ожидания логарифма правдоподобия

$$Q(f', f; \overset{+}{\mathbf{z}}) = \mathbf{E}_{\omega_f;\mathbf{u}} \ln p(\overset{+}{\mathbf{z}}, \mathbf{u}|f') = \sum_{i=1}^N \sum_{j=1}^k P\{j|x_i, f\} \ln p(x_i, j|f')$$

(ср. с формулой (131), в которой не только использованы другие обозначения, но и изменен порядок аргументов, а также знак функции Q). Шаг **M** состоит в максимизации $Q(f', f; \overset{+}{\mathbf{z}})$ по f'(в разделе 3.3.1 — минимизации).

Этот пример легко обобщается на случай наличия пропущенных признаков в обучающем наборе. Чтобы избежать громоздких обозначений введем компактные, но нестандартные. Каждый вектор признаков x определяет разложение пространства признаков $\mathcal{X} = \overset{+}{\mathcal{X}}_x \oplus \overset{-}{\mathcal{X}}_x$ в сумму пространства признаков, представленных в x_i и пропущенных в нем. Это разложение естественно порождает разложение любого вектора $\mu \in \mathcal{X}$ в прямую сумму $\mu = \overset{+}{\mu}_x \oplus \overset{-}{\mu}_x$, где $\overset{+}{\mu}_x \in \overset{+}{\mathcal{X}}_x$ и $\overset{-i}{\mu}_x \in \overset{-}{\mathcal{X}}_x$. Обозначим также через $\overset{+}{d}_x$ и $\overset{-}{d}_x$ количество признаков, присутствующих и, соответственно, пропущенных в векторе x, а через $x[\mu] = \overset{+}{x} \oplus \overset{-}{\mu}_x$ — вектор x, в котором пропущенные компоненты заменены на соответствующие компоненты вектора μ.

Следующее предложение описывает основные этапы вычислений.

Предложение 34 *При применении алгоритма 13 для моделирования гауссовой смеси (226) с пропущенными признаками в обучающем наборе*

на шаге ***E***

$$P\{v_i|\overset{+}{x}_i, f\} = \frac{P\{\overset{+}{x}_i, v_i|f\}}{\sum_{j=1}^k P\{\overset{+}{x}_i, j|f\}} = \frac{P_{v_i} N^{\overset{+}{d}_{x_i}}_{((\mu_{v_i})_{x_i}, s_{v_i})}(\overset{+}{x}_i)}{\sum_{j=1}^k P_j N^{\overset{+}{d}_{x_i}}_{((\mu_j)_{x_i}, s_j)}(\overset{+}{x}_i)},$$

$$\begin{aligned} p(\overset{-}{x}_i, v_i|\overset{+}{x}_i, f) &= P\{v_i|\overset{+}{x}_i, f\} p(\overset{-}{x}_i \mid \overset{+}{x}_i, v_i, f) \\ &= P\{v_i|\overset{+}{x}_i, f\} N^{\overset{-}{d}_{x_i}}_{((\mu_{v_i})_{x_i}, s_{v_i})}(\overset{-}{x}_i) \end{aligned}$$

и

$$\begin{aligned} Q(f', f; \overset{+}{\mathbf{z}}) &= \sum_{i=1}^N \sum_{j=1}^k \int p(\overset{-}{x}_i, j|\overset{+}{x}_i, f) \ln p(x_i, j|f') d\overset{-}{x}_i \\ &= \sum_{i=1}^N \sum_{j=1}^k P\{j|\overset{+}{x}_i, f\} \left(\ln P'_j - \frac{d}{2} \ln(2\pi s'_j) - \frac{\|x_i[\mu_j] - \mu'_j\|^2 + \overset{-}{d}_{x_i} s_j}{2s'_j} \right) \end{aligned}$$

на шаге ***M*** *точка максимума Q по $f' = (P'_1, \ldots, P'_k, \mu'_1, \ldots, \mu'_k, s'_1, \ldots, s'_k)$ задается равенствами*

$$
\begin{aligned}
P'_j &= \frac{1}{N}\sum_{i=1}^{N} P\{j|\overset{+}{x}_i, f\} \\
\mu'_j &= \frac{\sum_{i=1}^{N} P\{j|\overset{+}{x}_i, f\} x_i[\mu_j]}{NP'_j} \\
s'_j &= \frac{\sum_{i=1}^{N} P\{j|\overset{+}{x}_i, f\}\left(\|x_i[\mu_j]-\mu'_j\|^2 + \bar{d}_{x_i}\, s_j\right)}{dNP'_j}
\end{aligned}
$$

(ср. с формулами (136)–(138)).

Доказательство повторяет вычисления раздела 3.3.1 и оставляется в качестве **упражнения**.

Аналогично применяется алгоритм 13 и в более общем примере со стр. 153 с распределением (218)–(220), в котором представлено все: и признаки, и ответы, в том числе пропущенные, и априорное распределение на пространстве распознавателей. Вычисление условных вероятностей $P\{v_i|\ \overset{+}{x}_i, y_i, f\}$ (или $P\{v_i|\ \overset{+}{x}_i, f\}$, если ответ y_i не определен) и функции $Q(f', f; \overset{+}{\mathbf{z}})$ требует интегрирования по пространствам $\bar{\mathcal{X}}_{x_i}$ гауссовых плотностей общего вида, умноженных (для функции Q) на квадратичные функции от x_i, что легко делается аналитически. Максимизация функции $Q(f', f; \overset{+}{\mathbf{z}})$ по f' сводится к решению систем уравнений, получающихся дифференцированием по параметрам модели f' лагранжиана апостериорной вероятности (217) и аналогичных (132)–(135) и (65). Точка экстремума $Q(f', f; \overset{+}{\mathbf{z}})$ единственна, является точкой глобального максимума и находится явно на нескольких последующих страницах.

Чтобы выписать вычисления в хоть сколько-то обозримом виде, введем сокращенные обозначения. В общепринятых обозначениях любое разложение линейного пространства Z в прямую сумму $Z = X \oplus Y$ порождает соответствующие разложения для векторов ($\mu = \mu|_X \oplus \mu|_Y$, где $\mu \in Z$, $\mu|_X \in X$ и $\mu|_Y \in Y$), линейных форм ($w = w|_X \oplus w|_Y$, где для любого $\mu \in Z$ $w\mu = w|_X\mu|_X + w|_Y\mu|_Y$) и квадратичных форм ($A = A|_{XX} \oplus A|_{XY} \oplus A|_{YX} \oplus A|_{YY}$, где для любых $\mu, \nu \in Z$ $A(\mu,\nu) = A|_{XX}(\mu|_X, \nu|_X) + A|_{XY}(\mu|_X, \nu|_Y) + A|_{YX}(\mu|_Y, \nu|_X) + A|_{YY}(\mu|_Y, \nu|_Y)$). Для разложений пространства признаков $\mathcal{X} = \overset{+}{\mathcal{X}}_{x_i} \oplus \bar{\mathcal{X}}_{x_i}$ в соответствующих им разложениях векторов и форм индексы будем писать наверху и сокращенно: $\mu = \overset{+i}{\mu} \oplus \bar{\mu}$, $w = \overset{+i}{w} \oplus \bar{w}$ и $A = \overset{++i}{A} \oplus \overset{+-i}{A} \oplus \overset{-+i}{A} \oplus \overset{--i}{A}$. Поднятие $M \oplus 0$ квадратичной формы M (например, $\overset{--i}{A}$) с прямого слагаемого (соответственно, $\bar{\mathcal{X}}_{x_i}$) на все пространство, т.е. дополнение матрицы M строками и столбцами нулей, будем для краткости обозначать через $M^{\uparrow}$.

Такие же обозначения без индекса i будут относиться к разложениям пространства признаков $\mathcal{X}$, порожденным произвольным вектором признаков x. Кроме того такие же обозначения будем применять для "расширенных" обучающих векторов $z_i = (x_i, y_i)$ и прочих векторов и матриц в пространстве $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, а также, для единообразия, в одномерном пространстве ответов $\mathcal{Y}$.

Сперва приведем результаты вспомогательных вычислений для гауссовых плотностей, проверяемые непосредственно.

Лемма 4 *Для гауссова распределения $N_{(\mu,A)}$ на пространстве $\mathcal{X} = \overset{+}{\mathcal{X}} \oplus \bar{\mathcal{X}}$ гауссовыми являются также плотности граничного распределения*

$$
p(\overset{+}{x}) = \int N_{(\mu,A)}(x)\, d\,\bar{x} = N_{(\overset{+}{\mu},\overset{+}{A})}(\overset{+}{x})
$$

и условных распределений

$$p(\bar{x} \mid \overset{+}{x}) = \frac{p(x)}{p(\overset{+}{x})} = N_{\bar{(\mu}(\mu,A,\overset{+}{x}),\bar{A})}(\bar{x}) \ ,$$

где $\overset{+}{A}= \left(\overset{++}{A^{-1}} - \overset{+-}{A^{-1}}\overset{--}{A}\overset{-+}{A^{-1}}\right)^{-1}$, $\bar{\mu}\ (\mu, A, \overset{+}{x}) =\bar{\mu} - \overset{--}{A}\overset{-+}{A^{-1}}\ (\overset{+}{x} - \overset{+}{\mu})$ *и* $\bar{A}= \left(\overset{--}{A^{-1}}\right)^{-1}$ *(обозначения* $\overset{+}{A}$ *и* $\bar{A}$ *неравноправны!).*

□

Лемма 5 *Ожидание по гауссову распределению* $N_{(\mu,A)}(x)$ *линейной функции* $l(x)$ *равно* $l(\mu)$*, а ожидание квадратичной функции* $R(x-\nu, x-\nu)$

$$\int R(x-\nu, x-\nu) N_{(\mu,A)}(x)dx = \mathrm{tr}(RA^{-1}) + R(\nu-\mu, \nu-\mu) \ .$$

□

Напомним вероятностную модель (218)–(220):

$$p(x,y,v|f) = P_v\, N_{(\mu_v,A_v)}(x)\, N^1_{(0,\beta)}(wx+b-y))$$
$$p_0(f) = N^d_{(0,\alpha)}(w)$$

Конкретно для нее сформулируем частный случай общего утверждения о произведении гауссианов

Лемма 6

$$N^d_{(\mu,A)}(x)\, N^1_{(0,\beta)}(wx+b-y) = N^{d+1}_{(\nu,B)}(z) \ ,$$

где $B = \left(A^{-1} \oplus 0 + \frac{1}{\beta}(w \oplus (-1))^\top (w \oplus (-1))\right)^{-1}$, $\nu = \mu \oplus (w\mu + b)$ *и (напоминаем обозначения)* $z = x \oplus y$*. При этом справедливо равенство* $|B| = \beta|A|$*.*

□

В дальнейшем для модели f будем использовать обозначения ν_j и B_j ($(d+1)$-мерные) наряду с μ_j и A_j (d-мерными) — где как удобнее. Введем также дополнительные сокращенные обозначения для зафиксированных модели и обучающего набора:

$$\begin{aligned}
\bar{\nu}_{ij} &= \bar{\mu}^i\ (\nu_j, B_j, \overset{+}{z_i}) \text{ (применение леммы 4)},\\
P_{ij} &= P\{j|\ \overset{+}{z_i}, f\} = P\{j|\ \overset{+}{x_i}, \overset{+}{y_i}, f\} \ ,\\
x_{ij} &= x_i[\mu_j] = \overset{+}{x_i} \oplus \bar{\mu}^i_j \ ,\\
y_{ij} &= y_i[wx_i[\mu_j] + b] \ ,\\
z_{ij} &= x_{ij} \oplus y_{ij} = (x_{ij}, y_{ij}) \ ,\\
w_{-1} &= w \oplus (-1) = (w, -1) \ .
\end{aligned}$$

Предложение 35 *При применении шага алгоритма 13 для гребневой регрессии на гауссовой смеси с пропущенными данными в обучающем наборе на шаге* ***E***

$$P_{iv} = P\{v|\ \overset{+}{z_i}, f\} = \frac{P_v N_{(\overset{+i}{\nu_v}, \overset{+i}{B_v})}(\overset{+}{z_i})}{\sum_{j=1}^k P_j N_{(\overset{+i}{\nu_j}, \overset{+i}{B_j})}(\overset{+}{z_i})}$$

$$p(\bar{z}_i, v|\overset{+}{z}_i, f) = P_{iv} p(\bar{z}_i \mid \overset{+}{z}_i, v, f) = P_{iv} N_{(\bar{\nu}_{iv}, \bar{\bar{B}}^{i}_{v})}(\bar{z}_i)$$

$$Q(f', f; \overset{+}{z}) = \ln p_0(f') + \sum_{i=1}^{N}\sum_{j=1}^{k} \int p(\bar{z}_i, j|\overset{+}{z}_i, f) \ln p(z_i, j|f') d\,\bar{z}_i$$

$$= -\frac{1}{2}(\ln(2\pi\alpha) + \|w\|^2) + \sum_{i=1}^{N}\sum_{j=1}^{k} P_{ij} I_{ij} \ ,$$

где

$$I_{ij} = \int p(\bar{z}_i \mid \overset{+}{z}_i, j, f) \ln p(z_i, j|f') d\,\bar{z}_i$$

$$= \ln P'_j - \frac{d}{2}\ln(2\pi) - \frac{\ln|A'_j| + \ln\beta'}{2} - \frac{1}{2}\mathrm{tr}(\bar{\bar{B}}^{i}_{j}(\bar{\bar{B}}'^{-i}_{j})^{-1})$$

$$- \frac{1}{2}\,(\bar{\bar{A}}'^{-i}_{j})^{-1}\,(x_{ij} - \mu'_j, x_{ij} - \mu'_j) - \frac{1}{2\beta'}(w' x_{ij} + b' - y_{ij})^2$$

на шаге $\boldsymbol{M}$ *точка максимума* Q *по* $f' = (P'_1, \ldots, P'_k, \mu'_1, \ldots, \mu'_k, A'_1, \ldots, A'_k, w', b', \beta')$ *единственна и вычисляется явно:* w' *и* b' *получаются решением невырожденной системы линейных уравнений*

$$w'\left(\frac{\beta}{\alpha} I^d + \sum_{i=1}^{N}\sum_{j=1}^{k} P_{ij}\left(\bar{\bar{B}}^{i\uparrow}_{j}\Big|_{\mathcal{X}\mathcal{X}} + x_{ij}(x_{ij})^\top\right)\right) + b'\sum_{i=1}^{N}\sum_{j=1}^{k} P_{ij}(x_{ij})^\top$$

$$= \sum_{i=1}^{N}\sum_{j=1}^{k} P_{ij}\left(\bar{\bar{B}}^{i\uparrow}_{j}\Big|_{\mathcal{X}\mathcal{Y}} + y_{ij}(x_{ij})^\top\right)$$

$$w'\sum_{i=1}^{N}\sum_{j=1}^{k} P_{ij} x_{ij} + b'\sum_{i=1}^{N}\sum_{j=1}^{k} P_{ij} = \sum_{i=1}^{N}\sum_{j=1}^{k} P_{ij} y_{ij} \ ,$$

а значения остальных параметров модели f' *задаются равенствами*

$$\begin{aligned}
P'_j &= \frac{1}{N}\sum_{i=1}^{N} P_{ij} \\
\mu'_j &= \frac{\sum_{i=1}^{N} P_{ij}\left(x_{ij} + w'^\top y_{ij}\right)}{N P'_j} \\
A'_j &= \frac{\sum_{i=1}^{N} P_{ij}\left((x_{ij} - \mu'_j)(x_{ij} - \mu'_j)^\top + \bar{\bar{B}}^{i\uparrow}_{j}\Big|_{\mathcal{X}\mathcal{X}}\right)}{d N P'_j} \\
\beta' &= \frac{1}{N}\mathrm{tr}\left((w_{-1})^\top w_{-1}\sum_{i=1}^{N}\sum_{j=1}^{k} P_{ij}\left((z_{ij} - \nu'_j)(z_{ij} - \nu'_j)^\top + \bar{\bar{B}}^{i\uparrow}_{j}\right)\right)
\end{aligned}$$

Доказательство (и исправление опечаток, которые наверняка есть), как и доказательство предложения 34, сводится к прямым вычислениям с помощью лемм 4–6 и оставляется в качестве **упражнения**.

Напоминаем, что параметр α априорного распределения и число k методом максимизации ожидания не оцениваются. Их нужно подбирать внешним образом. например, с помощью кросс-валидации.

При распознавании вектора x, в котором нет пропущенных признаков, с помощью обученной модели f, от модели нужны только параметры w и b: ответ

$y = wx + b$, как и положено при линейной регрессии. Но распознавать можно и при пропусках признаков: тогда наиболее вероятный ответ

$$y = \mathbf{E}_{p(\bar{x}|\overset{+}{x},f)}(wx+b) = b + w\mathbf{E}_{p(\bar{x}|\overset{+}{x},f)}x = b + w\sum_{j=1}^{k} P_j \int xp(\bar{x}|\overset{+}{x},j,f)d\,\bar{x}$$

$$= b + w\sum_{j=1}^{k} P_j \int xN_{(\bar{\mu}(\mu_j,A_j,\overset{+}{x}),\bar{A}_j)}(\bar{x})d\,\bar{x} = b + w\sum_{j=1}^{k} P_j \left(\overset{+}{x} \oplus \bar{\mu}\,(\mu_j, A_j, \overset{+}{x})\right)$$

$$= b + \overset{+}{w}\overset{+}{x} + \bar{w}\sum_{j=1}^{k} P_j \left(\bar{\mu_j} - \overline{\bar{A}_j}\overline{A_j^{-1}}\,(\overset{+}{x} - \overset{+}{\mu_j})\right)$$

(применялись леммы 4 и 5). Параметр β для распознавания не применяется; он оценивает среднеквадратичную ошибку ответа.

A.3.4 Сходимость алгоритма EM

Алгоритмы 12 и 13, вообще говоря, не останавливаются, так что на практике количество итераций в них нужно ограничивать принудительно. Как? Как ведут себя последовательность моделей $f^{(m)}$, получаемых на m-ом шаге итерации, последовательность логарифмов (плотностей) их апостериорных вероятностей $L(f^{(m)}; \overset{+}{\mathbf{z}})$?

По следствию 7 последовательность $L(f^{(m)}; \overset{+}{\mathbf{z}})$ строго возрастает. Тогда при естественном предположении компактности множеств вида

$$\{f | L(f; \overset{+}{\mathbf{z}}) \geq \text{const}\} \tag{227}$$

что обычно обеспечивается стремлением к нулю плотности априорной вероятности $p_0(f)$ на бесконечности, функция $L(\cdot; \overset{+}{\mathbf{z}})$ имеет глобальный максимум, а последовательность $L(f^{(m)}; \overset{+}{\mathbf{z}})$ сходится (**упражнение!**). Но она может сходиться не к глобальному максимуму, а к локальному или даже к значению в какой-либо седловой точке функции $L(\cdot; \overset{+}{\mathbf{z}})$. Множество троек

(функция $L(f; \overset{+}{\mathbf{z}})$, наблюдаемые параметры $\overset{+}{\mathbf{z}}$, начальная модель $f^{(0)}$),

при которых $L(f^{(m)}; \overset{+}{\mathbf{z}})$ сходится не к (локальному) максимуму, пренебрежимо мало (аккуратная формулировка и доказательство соответствующего утверждения весьма громоздки), а вот возможность сходимости к неглобальному максимуму — не пренебрежима.

Сама последовательность моделей $f^{(m)}$, даже в тех же предположениях о компактности множеств (227) может и не сходиться. В статье [122] приводятся несколько вариантов дополнительных условий, при которых последовательность $f^{(m)}$ сходится. В частности, достаточно, чтобы у функции $L(\cdot; \overset{+}{\mathbf{z}})$ не было седловых точек, точка максимума была единственна, а функция (225) — непрерывно дифференцируема по первому аргументу [122, Corollary 1].

Если последовательность моделей сходится к точке локального максимума функции $L(\cdot; \overset{+}{\mathbf{z}})$, то при некоторых дополнительных условиях можно оценить и скорость сходимости. Например, в статье [33] приводится оценка для достаточно общего случая, когда функция (225) дважды непрерывно дифференцируема по первому аргументу и ее гессиан в этой точке максимума отрицательно определен. Но наличие оценки скорости сходимости к точке максимума не гарантирует сходимости именно к этой точке, а равномерную оценку скорости сходимости по всем точкам максимума, если их много, получить трудно.

С практической точки зрения не так важно то, чтобы алгоритм EM сошелся, как то, чтобы он не застрял в окрестности локального максимума или седловой точки функции $L(\cdot;\overset{+}{\mathbf{z}})$ со значением, заметно меньшим глобального максимума. Поэтому при отсутствии гарантий единственности максимума имеет смысл запускать параллельно несколько алгоритмов EM с разными начальными моделями, останавливаться по каким-либо внешним оценкам моделей, например, при ухудшении распознавания тестового набора, независимого от обучающего, и выбирать из полученных моделей лучшую.

Алгоритм EM обладает всеми недостатками градиентного спуска, кроме двух: в нем не требуется вычислять производные оптимизируемой функции и не требуется подбирать длину шага.

A.3.5 Обобщения алгоритма EM

Алгоритм максимизации ожидания в формулировке последовательной максимизации (алгоритм 12) можно обобщить следующим образом: на каждом шаге **E** и/или **M** не максимизировать значение функции $L_\omega(f;\overset{+}{\mathbf{z}})$ по ω и по f, соответственно, а только увеличивать ее значения. Такие *обобщенные алгоритмы максимизации ожидания* (*generalized expectation maximization* или *GEM*) не обязательно на каждой паре шагов увеличивают целевую функцию $L(f;\overset{+}{\mathbf{z}})$, но могут обучать модели даже быстрее "честного" алгоритма EM за счет более быстрых шагов обучения ([88]).

Недостижение максимума $L_\omega(f;\overset{+}{\mathbf{z}})$ по f на шаге **M** на справедливость предложения 32 и следствия 7 и на рассуждения о сходимости алгоритма в разделе A.3.4 вообще не влияет (оно влияет только на оценки скорости сходимости, все равно не приведенные). А недостижение максимума $L_\omega(f;\overset{+}{\mathbf{z}})$ по ω на шаге **E** делает предложение 32 неприменимым, так что значение $L(f;\overset{+}{\mathbf{z}})$ не обязано возрастать на каждом шаге. Но $L_\omega(f;\overset{+}{\mathbf{z}})$ на каждом шаге возрастает, и алгоритм почти всегда сходится к локальному максимуму $L_\omega(f;\overset{+}{\mathbf{z}})$, а в точке локального максимума (ω', f') предложение 32 применимо, так что $L_{\omega'}(f';\overset{+}{\mathbf{z}}) = L(f';\overset{+}{\mathbf{z}})$.

Предложение 36 *Если (ω', f') — точка локального (глобального) максимума $L_\omega(f;\overset{+}{\mathbf{z}})$, то f' является точкой локального (соответственно, глобального) максимума $L(f;\overset{+}{\mathbf{z}})$*

Доказательство. Из условия непрерывности моделей $p(x, y, v|f)$ по совокупности всех аргументов следует непрерывность отображения $f \mapsto \omega_f$ (формула (223)). Значит f' является точкой локального максимума функции $f \mapsto L_{\omega_f}(f;\overset{+}{\mathbf{z}})$, но по предложению 32 $L_{\omega_f}(f;\overset{+}{\mathbf{z}}) = L(f;\overset{+}{\mathbf{z}})$. Для глобальных максимумов утверждение следует из предложения 32 автоматически, без каких либо условий типа непрерывности. □

Таким образом, обобщенные алгоритмы максимизации ожидания, как и необобщенный, почти всегда сходятся к точке локального максимума апостериорной вероятности модели, но, увы, только локального.

В статье [88] предложено несколько вариантов обобщенного шага **E**, которые, в предположении, что (обобщенный) шаг **M** вычисляется быстрее шага **E**, ускоряют сходимость алгоритма. Опишем один из них.

Поскольку обучающие векторы предполагаются независимыми, вычисляе-

мое на каждом шаге **E** условное распределение (223)

$$\begin{aligned}\omega_f(\mathbf{u}) &= p(\mathbf{u}|\overset{+}{\mathbf{z}}, f) = \frac{p(\overset{+}{\mathbf{z}}, \mathbf{u}|f)}{p(\overset{+}{\mathbf{z}}|f)} = \frac{p(\overset{+}{\mathbf{z}}, \mathbf{u}|f)}{\int_{\mathbf{u}} p(\overset{+}{\mathbf{z}}, \mathbf{u}|f)d\mathbf{u}} \\ &= \frac{\prod_{i=1}^N p(\overset{+}{z_i}, u_i|f)}{\int_{u_1 \dots u_N} \prod_{i=1}^N p(\overset{+}{z_i}, u_i|f)du_1 \dots du_N} = \prod_{i=1}^N \frac{p(\overset{+}{z_i}, u_i|f)}{\int_{u_i} p(\overset{+}{z_i}, u_i|f)du_i} \\ &= \prod_{i=1}^N p(u_i|\overset{+}{z_i}, f)\end{aligned}$$

распадается в произведение распределений $\omega_f^i(u_i) = p(u_i|\overset{+}{z_i}, f)$, то максимизацию $L_\omega(f; \overset{+}{\mathbf{z}})$ можно проводить только по распределениям вида $\omega(\mathbf{u}) = \prod_{i=1}^N \omega^i(u_i)$. Но для таких распределений максимизируемое ожидание

$$\begin{aligned}L_\omega(f; \overset{+}{\mathbf{z}}) &= \mathbf{E}_{\omega;\mathbf{u}} \ln \frac{p_0(f)p(\overset{+}{\mathbf{z}}, \mathbf{u}|f)}{\omega(\mathbf{u})} = \mathbf{E}_{\omega;\mathbf{u}} \left(\ln p_0(f) + \sum_{i=1}^N \ln \frac{p(\overset{+}{z_i}, u_i|f)}{\omega^i(u_i)} \right) \\ &= \ln p_0(f) + \sum_{i=1}^N \mathbf{E}_{\omega^i;u_i} \ln \frac{p(\overset{+}{z_i}, u_i|f)}{\omega^i(u_i)}\end{aligned}$$

распадается в сумму, в которой каждое слагаемое зависит только от одного сомножителя ω^i распределения ω (кроме первого слагаемого, от ω вообще не зависящего), слагаемые можно максимизировать независимо и максимум достигается в точке $\omega_f^i(u_i) = p(u_i|\overset{+}{z_i}, f)$ (ср. с предложением 32). Максимизация по маломерному распределению ω^i намного дешевле, чем максимизация по всему многомерному распределению ω^i. Эта максимизация, т.е. вычисление условного распределения ненаблюдаемых параметров i-го обучающего объекта, и будет обобщенным шагом **E**, а простейший способ организации последовательности шагов приведен в алгоритме 14.

Алгоритм 14: Обобщенная максимизация ожидания (GEM); пример

1. Вход: обучающий набор $T = ((x_1, y_1), \ldots, (x_N, y_N))$, в котором некоторые признаки и/или ответы могут быть пропущены: $\overset{+}{\mathbf{z}}$ — вектор всех присутствующих значений.

2. Выбрать пространство фиктивных признаков v, пространство моделей распределения вероятностей троек (x, y, v), априорное распределение p_0 на пространстве моделей и начальную модель $f = f^{(0)}$.

3. В цикле

   (a) для i от 1 до N

   E максимизировать (или хотя бы просто увеличить) $L_\omega(f; \overset{+}{\mathbf{z}})$ по распределению ω^i вектора u_i всех ненаблюдаемых параметров i-го обучающего объекта при фиксированной модели f и распределении остальных ненаблюдаемых параметров;

   M максимизировать (или хотя бы просто увеличить) $L_\omega(f; \overset{+}{\mathbf{z}})$ по f при фиксированном распределении ω;

   (b) выйти из цикла, если модель f не изменилась.

4. Выход: порождающая модель f.

Алгоритм 14 соотносится с алгоритмом 12 примерно как стохастический градиентный спуск с пакетным (стр. 83).

B Цензурированные данные и анализ выживаемости

В этом разделе приводится еще один пример применения вероятностных моделей, в частности, семипараметрических (параметрических по одним переменным и непараметрических по другим), для распознавания в сомнительных ситуациях с не вполне определенными данными. Он не задумывался как полноценное введение в анализ выживаемости, про который написано много толстых книг, например, [67].

B.1 Постановка задачи анализа выживаемости и специфика терминологии

Анализ выживаемости — это распознавание, в котором пространство ответов $\mathcal{Y}$ имеет смысл времени, прошедшего после какого-то начального момента. Соответственно, ответ y в элементе (x, y) обучающего набора — это время жизни какого-либо объекта с вектором признаков x после какого-то момента (например, после измерения признаков). Цель распознавания состоит в том, чтобы научиться по вектору признаков вычислять типичное время жизни объекта с такими признаками, а еще лучше — научиться вычислять распределение условных вероятностей $p(y|x)$. Реальные примеры: объектами могут быть атомы или изотопы, электрические лампочки или раковые больные.

Время может быть непрерывным ($\mathcal{Y} = \mathbb{R}_+$, время жизни измеряют точно) или дискретным ($\mathcal{Y} = \mathbb{Z}_+$, время жизни измеряют в днях); в обоих случаях

к нему может быть еще добавлена точка ∞, хотя, слегка упрощая некоторые формулировки, такое обозначение усложняет другие.

Для случайной величины η ("времени жизни чего-либо") со значениями в $\mathcal{Y}$ определена *функция распределения*

$$F(t) = P\{\eta < t\} \ . \tag{228}$$

Для задач с непрерывным временем мы пока ограничимся дифференцируемыми функциями распределения, для которых определена плотность

$$p(t) = \frac{d}{dt}F(t) \ .$$

Для задач с дискретным временем всегда определены вероятности

$$p(t) = F(t+1) - F(t) \ .$$

Возможность бесконечного времени жизни соответствует тому, что $F(\infty)$ может быть меньше 1.

Вместо функции распределения F и плотности p при анализе выживаемости обычно применяют *функцию выживаемости* (*survival function*)

$$S(t) = 1 - F(t) = P\{\eta \geq t\} \ , \tag{229}$$

равную вероятности выжить к моменту t, и *риск* (*hazard*)

$$h(t) = \frac{p(t)}{S(t)} = \frac{p(\eta = t)}{P\{\eta \geq t\}} = p(\eta = t | \eta \geq t) \ , \tag{230}$$

равный (плотности) вероятности погибнуть в момент t при условии, что до него дожили. Слово "плотность" в скобках — это единственное различие между случаями непрерывного и дискретного распределения[48]. Легко видеть, что в непрерывном случае

$$h(t) = \frac{-\frac{d}{dt}S(t)}{S(t)} = -\frac{d}{dt}\ln S(t)$$

и

$$S(t) = e^{-\int_0^t h(\tau)d\tau} \ , \tag{231}$$

а при дискретном времени

$$h(t) = \frac{S(t) - S(t+1)}{S(t)} = 1 - \frac{S(t+1)}{S(t)} \ ,$$

$S(0) = 1$ и

$$S(t) = \prod_{i=0}^{t-1}(1 - h(i)) \ , \tag{232}$$

так что по любой из функций $p(\cdot)$, $F(\cdot)$, $S(\cdot)$ и $h(\cdot)$ остальные три однозначно восстанавливаются. В частности, риск $h(\cdot)$ может быть любой неотрицательной функцией, в случае дискретного времени не превосходящей 1, выживаемость $S(\cdot)$ — неотрицательной, начинающейся со значения 1 и монотонно невозрастающей, а условие $S(\infty) = 0$ равносильно расходимости интеграла $\int^\infty h(t)dt$ при непрерывном времени или ряда $\sum^\infty h(n)$ при дискретном.

[48] В общем случае непрерывного времени, но не обязательно абсолютно непрерывного распределения, риск определяется как односторонний предел

$$h(t) = \lim_{\tau \searrow 0} \frac{P\{t \leq \eta < t + \tau\}}{\tau P\{\eta \geq t\}} = \lim_{\tau \searrow 0} \frac{S(t) - S(t+\tau)}{\tau S(t)}$$

в обобщенных функциях.

Замечание. В принятых обозначения, традиционных для русскоязычной математики, функции распределения (228) и выживаемости (229) непрерывны слева. Англоязычная традиция предпочитает непрерывные справа функции распределения $F(t) = P\{\eta \le t\}$ и выживаемости $S(t) = P\{\eta > t\}$, при прежней функции риска (230). Некоторые вычисления в такой правонепрерывной системе определений получаются изящнее. Не переходя на нее полностью, будем пользоваться "функцией переживаемости" $S^{>}(t) = P\{\eta > t\}$ там, где это удобнее.

Для множества из n объектов с разными функциями выживаемости $\{S_1(t), \dots,$ можно определить усредненную функцию выживаемости, $S(t)$ равную математическому ожиданию доли объектов, выживших к моменту t:

$$S(t) = \frac{1}{n}\sum_{i=1}^{n} S_i(t)\ . \tag{233}$$

Плотность $p(t)$ и функция распределения $F(t)$ получившегося "усредненного" объекта также равны средним арифметическим соответствующих функций индивидуальных объектов, а вот риск $h(t)$ — нет.

Примеры согласованных наборов функций $p(\cdot)$, $F(\cdot)$, $S(\cdot)$ и $h(\cdot)$:

постоянный риск $h(t) = h$. Тогда $S(t) = e^{-ht}$, $F(t) = 1 - e^{-ht}$ и $p(t) = he^{-ht}$. Такое бывает при радиоактивном распаде в отсутствии внешнего излучения, выживаемость экспоненциально убывает и $t_{\frac{1}{2}} = \frac{\ln 2}{h}$ — небезызвестный период полураспада. В отличие от периода полураспада, среднее время жизни равно $\int_0^\infty tp(t)dt = \frac{1}{h}$.

положительный ограниченный риск $S(t) = p_1 e^{-h_1 t} + p_2 e^{-h_2 t}$ (смесь двух нестабильных изотопов в пропорции $p_1 : p_2$, $p_1 + p_2 = 1$). Тогда риск $h(t) = \frac{p_1 h_1 e^{-h_1 t} + p_2 h_2 e^{-h_2 t}}{p_1 e^{-h_1 t} + p_2 e^{-h_2 t}}$ убывает от взвешенной суммы рисков $p_1 h_1 + p_2 h_2$ при $t = 0$ до минимума $\min(h_1, h_2)$ при $t \to \infty$.

медленно убывающий риск $h(t) = \frac{h}{t+1}$. Тогда $S(t) = \frac{1}{(t+1)^h}$, $F(t) = 1 - \frac{1}{(t+1)^h}$ и $p(t) = \frac{h}{(t+1)^{h+1}}$. То есть, при любом степенном убывании плотности или функции выживаемости по времени риск убывает всего лишь обратно пропорционально времени.

быстро убывающий риск $S(t) = pe^{-ht} + (1-p)$ (смесь нестабильного и стабильного изотопов в пропорции $p : (1-p)$. Тогда риск $h(t) = \frac{phe^{-ht}}{pe^{-ht} + (1-p)} = h\sigma(\ln \frac{p}{1-p} - ht)$ экспоненциально убывает при $t \to \infty$. А $S(\infty) = 1 - p > 0$.

возрастающий риск $h(t) = ht$. Тогда $S(t) = e^{-\frac{ht^2}{2}}$, $F(t) = 1 - e^{-\frac{ht^2}{2}}$ и $p(t) = hte^{-\frac{ht^2}{2}}$. Такое бывает при старении.

Замечание. Первый и последний примеры являются частными случаями *распределений Вейбулла*, часто применяемых в анализе выживаемости — двухпараметрического семейства распределений с параметрами $k, \lambda > 0$ и $h_{k\lambda}(t) = \frac{kt^{k-1}}{\lambda^k}$, $S_{k\lambda}(t) = e^{-\left(\frac{t}{\lambda}\right)^k}$, $F_{k\lambda}(t) = 1 - e^{-\left(\frac{t}{\lambda}\right)^k}$, $p_{k\lambda}(t) = \frac{kt^{k-1}}{\lambda^k} e^{-\left(\frac{t}{\lambda}\right)^k}$.

Замечание. В примерах со смесями изотопов риск на самом деле зависит не от времени, а от признаков объектов (например, атомарных весов), если только они известны. . . .

B.2 Цензурированные данные

Обучающие данные для анализа выживаемости, как правило, являются неполноценными с точки зрения статистического обучения: для многих объектов значение ответа — то самое, которое нужно научиться предсказывать — неизвестно, а известны только некоторые соображения про этот ответ. Как правило, эти соображения задаются неравенствами. Такие данные называются *цензурированными* (*censored data*). При анализе выживаемости чаще всего встречается *цензурирование справа*, то есть знание, что время жизни объекта не менее чего-то. Например, лампочка не сгорела до конца всех испытаний или больной выписался из больницы и уехал, а что с ними было дальше — неизвестно. Далее речь будет идти только про цензурирование справа[49].

Формализуется цензурирование обучающего набора следующим образом. Обучающий набор $T = ((x_1, t_1, \theta_1), \ldots, (x_N, t_N, \theta_N))$ состоит из троек следующего вида: вектор признаков $x_i \in \mathcal{X}$, время $t_i \in \mathcal{Y}$ и признак цензурированности $\theta_i \in \{0, 1\}$, объясняющий, что это за время. При $\theta_i = 0$ (цензурирования нет) t_i — это время жизни y_i i-го объекта, а при $\theta_i = 1$ (цензурирование есть) t_i — это время наблюдения за i-м объектом, а время его жизни y_i больше t_i. Аналогично устроен и тестовый набор. При $\theta_i = 0$ говорят, что в момент t_i произошло событие, а именно, смерть i-го объекта, а при $\theta_i = 1$ никакого события с i-м объектом в момент t_i не происходит.

Практически всегда предполагается, хотя обычно про это забывают напомнить, что цензурирование θ не зависит от времени жизни y при каждом фиксированном x. Формально это означает, что каждый объект имеет вектор признаков x, время жизни y и время наблюдения за ним z (возможно, бесконечное), причем тройки $(x, y, z) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{Y}$ считаются реализациями независимых случайных величин с одинаковым (неизвестным) распределением, таким, что условные распределения $p(y|x)$ и $p(z|x)$ независимы при всех x. Но из двух времен y и z наблюдается только минимум $t = \min(y, z)$ и признак θ, того, что время цензурирования z строго меньше времени жизни y. Неформально, чтобы обеспечить независимость цензурирования от времени жизни, нельзя прекращать испытание живучести электрической лампочки потому, что из нее идет (или не идет) дым, но можно прекращать из-за независимых событий, например, протечки потолка. В медицине же часто поступают наоборот: наблюдение за пациентом прекращается при заметном улучшении его состояния.

Далее везде предполагается независимость цензурирования от времени жизни.

Кроме специфического для анализа выживаемости цензурирования времени жизни объектов возможно еще и неполное знание их признаков. Про какие-то из признаков объекта могут быть известны не значения, а только ограничения на их возможные значения, или даже вообще ничего. Такие данные называются не цензурированными, а пропущенными (*missing data*). Они были подробно рассмотрены в разделе A, а при дальнейшем обсуждении анализа выживаемости рассматриваться не будут.

B.3 Параметрические оценки выживаемости

Предположим, что имеется конечномерное семейство $\mathcal{P}$ распределений (моделей) π, зависящих от векторов признаков, и априорное распределение вероятностей на нем. Тогда, в принципе, на обучающем наборе с цензурированными данными можно обучать модель всеми описанными в разделе 1.2.4

[49] Кроме цензурирования бывают еще *усечение* (*truncated data*) — невозможность измерить признаки и даже определить наличие или отсутствие объектов со слишком большим или слишком маленьким временем жизни.

методами: байесовским, максимизацией апостериорной вероятности и максимизацией правдоподобия. Отличие цензурированного случая состоит в том, что правдоподобие модели, т.е. условная вероятность обучающего набора $T = ((x_1, t_1, \theta_1), \ldots, (x_N, t_N, \theta_N))$ при этой модели, выражается не простой формулой (18), в которую входят только значения плотностей распределения в точках обучающего набора, а более сложной

$$p(T|\pi) = \prod_{i|\theta_i=0} p_\pi(t_i|x_i) \prod_{i|\theta_i=1} S^>_\pi(t_i|x_i) \ . \tag{234}$$

Если моделируемая функция выживаемости непрерывна, то $S^>_\pi(t|x) = S_\pi(t|x)$ и правдоподобие удобнее выражается через функции выживаемости (229) и риска (230)

$$p(T|\pi) = \prod_{i|\theta_i=0} p_\pi(t_i|x_i) \prod_{i|\theta_i=1} S_\pi(t_i|x_i) = \prod_{i|\theta_i=0} h_\pi(t_i|x_i) \prod_{i=1}^{N} S_\pi(t_i|x_i) \ . \tag{235}$$

Наличие цензурирования приводит к тому, что при сколько-нибудь сложных моделях не только байесовское обучение, но и максимизация правдоподобия становятся слишком сложны вычислительно. Действительно, функция выживаемости S связана соотношениями (231) или (232) со значениями функции риска h на интервалах времени, и если она не вычисляется аналитически, максимизировать произведение (235) нелегко.

Приведем простой пример, когда функция выживаемости вычисляется аналитически и обучение максимизацией правдоподобия или апостериорной вероятности вполне возможно. Рассмотрим пространство моделей с риском $h_\pi(t|x) = r_\pi(x)$, зависящем от признаков, но не от времени. Для таких моделей $S(t|x) = e^{-r_\pi(x)t}$ и правдоподобие

$$p(T|\pi) = \prod_{i|\theta_i=0} h_\pi(t_i|x_i) \prod_{i=1}^{N} S_\pi(t_i|x_i) = \left(\prod_{i|\theta_i=0} r_\pi(x_i) \right) e^{-\sum_{i=1}^{N} r_\pi(x_i) t_i} \ .$$

Для примера ограничимся моделями с евклидовым пространством признаков, риском, равным экспоненте от линейной функции $r_{w,b}(x) = e^{wx+b}$, и априорным распределением с плотностью вида $p_0(w) = e^{-\psi(w)}$, например, гауссовой или лапласовой. Тогда эквивалентная максимизации апостериорной вероятности минимизация минус логарифма совместной вероятности $p(T, w, b)$ выглядит так:

$$-\ln p(T, w, b) = \psi(w) + \sum_{i=1}^{N} \big((\theta_i - 1)(wx_i + b) + t_i e^{wx_i+b} \big) \to \min_{w,b} \ . \tag{236}$$

Как и при обучении логистической регрессии (раздел 2.2.2, формула (86) и следующий за ней абзац), зависящие от обучающего набора слагаемые дифференцируемы и выпуклы по параметрам модели (w, b), и обучение можно проводить всеми теми же методами, что и для логистической регрессии. После того, как модель (w, b) обучена, прогнозируемая вероятность дожить до момента t для объекта с вектором признаков x равна

$$S(t|x) = e^{-te^{wx+b}} \ .$$

Но в реальных задачах построить семейство простых (аналитически интегрируемых по времени) и адекватных моделей удается редко. В анализе выживаемости более распространены непараметрические и семипараметрические модели, в которых про зависимость выживаемости от времени ничего не предполагается. Такие модели рассматриваются в последующих разделах.

B.4 Оценка Каплана-Майера

В этом разделе речь идет не о распознавании выживаемости индивидуальных объектов, а только об оценке усредненной выживаемости (233). Соответственно, в обучающем наборе T векторы признаков x_i полностью игнорируются. Сначала для разминки рассматривается оценка выживаемости по нецензурированным данным, после чего она обобщается на цензурированные.

В отсутствии цензурирования задача, как оценить функцию распределения вещественнозначной случайной величины τ по набору T из n ее независимых реализаций $t_1, \ldots, t_n$, была решена в 1930-ые годы. Ответ: функция распределения $F(t)$ приближается кусочно-постоянной функцией $\hat{F}_n(t) = \frac{1}{n}\#\{i|t_i < t\}$. Казалось бы, это получается очевидным применением закона больших чисел (сходимость частоты к вероятности) ко всем событиям вида $\tau < t$ одновременно. Но для бесконечного числа возможных событий буквальное применение закона больших чисел некорректно. Точное утверждение (*теорема Гливенко-Кантелли* [45, 23], доказанная авторами независимо и опубликованная в одном и том же номере журнала) состоит в том, что последовательность $\hat{F}_n$ при $n \to \infty$ сходится к F по вероятности[50].

Соответственно, и функция выживаемости $S(t)$ приближается ступенчатой функцией $\hat{S}_n(t) = \frac{1}{n}\#\{i|t_i \geq t\}$. Скачки этой функции происходят только в моменты событий t_i. Функцию выживаемости можно вычислять не только суммированием по обучающему набору, но и перемножением слева направо:

$$\hat{S}(0) = 1 \tag{237}$$

$$\hat{S}(t+\epsilon) = \hat{S}(t)\left(1 - \frac{\hat{S}(t) - \hat{S}(t+\epsilon)}{\hat{S}(t)}\right) = \hat{S}(t)\left(1 - \frac{\#\{i|t \leq t_i < t+\epsilon\}}{\#\{i|t_i \geq t\}}\right) . \tag{238}$$

Формула (238) верна при любом положительном ϵ, но интересна только при малых ϵ и моментах времени t, в которые что-то случалось, т.е. равных t_i при некотором i.

Пусть $t_{(1)} < \ldots < t_{(j)} < \ldots < t_{(m)}$ — упорядоченные по возрастанию времена событий $\{t_i,\ i = 1, \ldots, n\}$, $\nu_j = \#\{i|t_i = t_{(j)}\}$ — их кратности и $\rho_j = \#\{i|t_i \geq t_{(j)}\}$. Тогда оценка выживаемости $\hat{S}$ представима в виде произведения

$$\hat{S}(t) = \prod_{j|t_{(j)}<t}\left(1 - \frac{\nu_j}{\rho_j}\right) . \tag{239}$$

При наличии цензурирования оценка функции выживаемости по набору $T = ((t_1, \theta_1), \ldots, (t_n, \theta_n))$ цензурированных значений случайной величины не столь очевидна, как для нецензурированных. Оценка, предложенная Эдвардом Капланом и Полом Майером в их совместной работе [63] и с тех пор широко используемая[51], состоит в небольшой модификации формул пересчета (237, 238). А именно функция выживаемости тоже оценивается невозрастающей ступенчатой функцией, равной 1 в 0 и имеющей скачки только в те моменты, в которые

[50] То есть,

$$\forall \epsilon > 0 \quad \lim_{n\to\infty} P\{\sup_{t\in\mathbb{R}} |\hat{F}_n(t) - F(t)| > \epsilon\} = 0 .$$

[51] В статье [63] рассматривалось несколько разных оценок выживаемости. Широкую известность и имя Каплана-Майера получила оценка, называемая *product limit* (предел произведений), в качестве первоисточника которой указывалась малоизвестная работа [17] 1912 года.

происходит нецензурированное(!) событие (одно или несколько одновременно):

$$\hat{S}(0) = 1 \tag{240}$$

$$\hat{S}(t+\epsilon) = \hat{S}(t)\left(1 - \frac{\#\{i|\theta_i = 0 \ \& \ t \le t_i < t+\epsilon\}}{\#\{i|t_i \ge t\}}\right) . \tag{241}$$

В отличие от нецензурированного случая, в формуле (241) число ϵ должно быть столь мало, чтобы на открытом интервале времени $(t, t+\epsilon)$ либо не было случаев цензурирования, либо не было нецензурированных событий (и тогда $S(t+\epsilon) = S(t)$).

Знаки строгих и нестрогих неравенств подобраны так, что на случай дискретного времени оценки переносятся без малейших изменений (годится $\epsilon = 1$).

Для цензурированного набора T обозначим через $t_{(1)} < \ldots < t_{(j)} < \ldots < t_{(m)}$ упорядоченные по возрастанию моменты событий $\{t_i|\theta_i = 0\}$, через $\nu_j = \#\{i|\theta_i = 0,\ t_i = t_{(j)}\}$ — кратности событий, но, как и в нецензурированном случае, через $\rho_j = \#\{i|t_i \ge t_{(j)}\}$ — количество всех наблюдаемых (включая умирающих) в момент $t_{(j)}$ объектов. В таких обозначениях оценка Каплана-Майера задается той же самой формулой

$$\hat{S}(t) = \prod_{j|t_{(j)}<t}\left(1 - \frac{\nu_j}{\rho_j}\right) , \tag{242}$$

что и оценка выживаемости (239) в отсутствии цензурирования.

В статье [63] утверждается сходимость по вероятности оценок (242) с цензурированными данными к настоящей функции выживаемости (обобщение теоремы Гливенко-Кантелли), но доказательство не приводится. Зато оценка (242) довольно легко выводится из соображений максимизации правдоподобия.

Теорема 12 *Для цензурированного обучающего набора T распределение с выживаемостью Каплана-Майера (242) имеет наибольшее правдоподобие в классе всех вероятностных распределений.*

Замечание. Единственность наиболее правдоподобного распределения в теореме не утверждается.

Доказательство. Нужно было бы уточнить, что такое правдоподобие любого распределения при непрерывном времени. Действительно, выше рассматривались только либо распределения с дискретным временем, и правдоподобие было равно вероятности увидеть то, что увидели (т.е. обучающий набор), либо абсолютно непрерывные распределения, и правдоподобие (234), (235) равнялось плотности вероятности обучающего набора. Но вместо занудного честного обобщения правдоподобия на любые распределения мы дискретизируем непрерывное время с малым шагом ϵ, настолько малым, чтобы несовпадающие по времени точки обучающего набора не совпали, максимизируем получающееся при дискретизации *ϵ-правдоподобие*, докажем теорему, а затем устремим ϵ к нулю.

Для распределения с функцией выживаемости S вероятность цензурированного объекта $(t_i, 1)$ равна $S^{>}(t_i)$, а вероятность нецензурированного объекта $(t_i, 0)$ равна $S(t_i) - S^{>}(t_i)$. Значит *ϵ-правдоподобие* распределения, равное вероятности всего обучающего набора T,

$$L_{T,\epsilon}(S) = P_\epsilon\{T|S\} = \prod_{i|\theta_i=0}\left(S(t_i) - S^{>}(t_i)\right) \prod_{i|\theta_i=1} S^{>}(t_i) . \tag{243}$$

Функцией выживаемости S может быть любая (при непрерывном времени — непрерывная слева, но сейчас время дискретно) неотрицательная монотонно

невозрастающая функция, равная 1 в нуле. Пусть $s_j = S^>(t_{(j)}) = S(t_{(j)} + \epsilon)$ — значение функции выживаемости в момент времени, следующий за j-м по порядку событием, а $s_0 = 1$. Тогда при фиксированных s_j увеличение значений $S(t)$ в остальные моменты времени может только увеличить $L_{T,\epsilon}(S)$. Единственное ограничение на возможность увеличения функции выживаемости — сохранение ее монотонности, значит максимум правдоподобия можно искать только среди ступенчатых функций вида

$$S(t) = \begin{cases} s_0 = 1 & \text{при} \quad t \le t_{(1)} \\ s_j & \text{при} \quad t_{(j)} < t \le t_{(j+1)},\ j < m \\ s_m & \text{при} \quad t_{(m)} < t \end{cases} . \tag{244}$$

Для таких функций выживаемости ϵ-правдоподобие (243) удобно переписать в виде произведения "слева направо" по времени

$$L_{T,\epsilon}(S) = \prod_{j=1}^{m} (s_{j-1} - s_j)^{\nu_j} s_j^{\rho_j - \nu_j - \rho_{j+1}} = \prod_{j=1}^{m} \left(1 - \frac{s_j}{s_{j-1}}\right)^{\nu_j} \left(\frac{s_j}{s_{j-1}}\right)^{\rho_j - \nu_j} , \tag{245}$$

где для удобства введено обозначение $\rho_{m+1} = 0$. В последнем произведении каждый сомножитель можно максимизировать по отношению $\frac{s_j}{s_{j-1}}$ независимо. В точке максимума $\hat{s}$ получается $\frac{\hat{s}_j}{\hat{s}_{j-1}} = \frac{\rho_j - \nu_j}{\rho_j}$ (**упражнение**), а значит при $k \le m$,

$$\hat{s}_k = \prod_{j=1}^{k} \left(1 - \frac{\nu_j}{\rho_j}\right) . \tag{246}$$

Для дискретного (или искусственно дискретизированного) времени получена оценка выживаемости (244, 246), совпадающая с оценкой Каплана-Майера (242). Для непрерывного времени нужно еще устремить ϵ к нулю, что не требует никаких усилий, поскольку буква ϵ была предусмотрительно потеряна в обозначениях, а значения $\hat{s}_k$ от ϵ не зависят. □

Замечание. Неединственность наиболее правдоподобного распределения имеет место при $s_m > 0$, т.е. когда после последней зарегистрированной смерти еще остаются выжившие объекты. Тогда при $t > t_{(m)}$ выживаемость $S(t)$ может быть и меньше s_m — ϵ-правдоподобие (243) от этого не изменится.

Замечание. Как обычно бывает с наиболее правдоподобными оценками, оценка Каплана-Майера, обладая замечательными асимптотическими свойствами при росте обучающих наборов, далека от идеала при малых обучающих наборах: она объявляет невозможным то, чему ее не учили. Например, из нее следует, что вероятность $P\{0 \le t < t_{(1)}\}$ смерти тестового объекта до момента первой смерти в обучающем наборе равна 0.

B.5 Модель Кокса

Вернемся к задаче распознавания выживаемости, когда нужно оценить время жизни y объекта по вектору x его признаков. Если имеется нецензурированный обучающий набор, то для любой гипотезы о плотности распределения $p(y|x)$ из какого-то пространства допустимых гипотез (или, что эквивалентно, любой гипотезы о функции распределения $F(y|x)$, выживаемости $S(y|x)$ или риске $h(y|x)$) можно (теоретически...) посчитать ее правдоподобие и обучить распознаватель методом максимизиции правдоподобия, максимизации апостериорной вероятности или байесовским. Если обучающий набор цензурирован[52],

[52]Здесь и далее предполагается независимость цензурирования не только от времени жизни y, как в разделе B.2, но от пары (x, y). В реальных же задачах цензурирование может оказаться зависящим от вектора признаков x.

то можно обучать распознаватель точно теми же методами, только вычисления становятся еще намного сложнее. Например, правдоподобие вероятностной модели, т.е. плотность вероятности наблюдаемых ответов на обучающем наборе T при условии признаков $\mathbf{X}$, выраженное через функции риска, равно (см. (235))

$$\begin{aligned} p(T|\mathbf{X}) &= \prod_{i|\theta_i=1} S(t_i|x_i) \prod_{i|\theta_i=0} p(t_i|x_i) \\ &= \prod_{i=1}^{n} S(t_i|x_i) \prod_{i|\theta_i=0} h(t_i|x_i) \\ &= \exp\left(-\sum_{i=1}^{n} \int_0^{t_i} h(\tau|x_i)d\tau\right) \prod_{i|\theta_i=0} h(t_i|x_i) \end{aligned}$$

(все обучающие векторы и цензурирование независимы). Но в процессе обучения распознавателя придется решать интегральные уравнения, что трудоемко.

В работе Дэвида Кокса [30] предложено некоторое ограничение на допустимые гипотезы (модель пропорционального риска) и некоторая замена правдоподобия (частичное правдоподобие), при которых обучение распознавателя на цензурированных данных происходит не намного сложнее, чем на нецензурированных. Этот подход стал так популярен, что модель пропорционального риска называют *моделью Кокса*, а метод обучения — *регрессией Кокса*.

B.5.1 Пропорциональный риск

Модель Кокса (*модель пропорционального риска*, *proportional hazard model*) состоит в предположении, что риск $h(t|x)$ распадается в произведение

$$h(t|x) = q(t)r(x) \ . \tag{247}$$

Разложение (247) неоднозначно: функцию q можно поделить на любую константу, а функцию r умножить на нее же. В терминах функции выживаемости модель Кокса равносильна существованию разложения

$$S(t|x) = (Q(t))^{r(x)} \ , \tag{248}$$

причем $Q(t)$ — это в точности функция выживаемости, соответствующая риску $q(t)$:

$$Q(t) = e^{-\int_0^t q(\tau)d\tau} \ .$$

В терминах плотности или функции распределения модель Кокса выглядит менее изящно.

Замечание. В работе Кокса [30] предлагается чуть-чуть другая модель. Вместо риска $h(t) = \frac{p(\eta=t)}{P\{\eta\geq t\}}$ (формула (230)) берется функция $h'(t) = \frac{p(\eta=t)}{P\{\eta>t\}}$ и для нее предполагается существование разложения, аналогичного (247):

$$h'(t|x) = q'(t)r(x) \ . \tag{249}$$

Для непрерывных распределений с непрерывным временем модели (247) и (249) совпадают, а для дискретного времени модель (249) превращается в

$$\frac{h(t|x)}{1-h(t|x)} = \frac{q(t)}{1-q(t)}r(x) \ . \tag{250}$$

B.5.2 Частичное правдоподобие

Идея Кокса состоит в том, чтобы обучать зависящую от признаков функцию r, ничего не предполагая про зависящую от времени q, которую потом оценивать непараметрически аналогично оценке Каплана-Майера. Для этого при обучении модели вместо правдоподобия (плотности вероятности увидеть то, что увидели, с точки зрения данной модели) можно использовать "частичное правдоподобие" (*partial likelihood*): (плотность) вероятности наблюдать именно такие события с точки зрения данной модели при условии, что времена и количества событий (смертей) такие, какие они есть. Информация о том, что происходит в остальное время, например, какие цензурированные объекты сколь долго наблюдались между событиями или после последнего события, частичным правдоподобием не учитывается. Впрочем, как было показано в доказательстве теоремы 12, наиболее правдоподобно, что в остальное время ничего не происходит.

К введенным на стр. 173 обозначениям $t_{(j)}$, ν_j и ρ_j добавим обозначения $N_j{=}\{i|\theta_i = 0,\ t_i = t_{(j)}\}$ и $R_j{=}\{i|t_i \geq t_{(j)}\}$ для множеств объектов, умерших в момент $t_{(j)}$ и не умерших до него, соответственно, и $R_0 = \{1,\ldots,n\}$. Естественно, $\nu_j = \#N_j$ и $\rho_j = \#R_j$. Обозначим через $\mathbf{X} = (x_1,\ldots,x_n)$ набор всех векторов признаков, через $\mathbf{X}_A$ набор векторов признаков подмножества $A \subset \{1,\ldots,n\}$, а через $\mathbf{Y} = ((t_{(1)},\nu_1),\ldots,(t_{(m)},\nu_m))$ набор времен и количеств смертей в обучающем наборе T. В обозначениях $\mathbf{X}$ и $\mathbf{Y}$ преднамеренно отсутствует имеющееся в T знание о времени смерти или цензурирования каждого объекта.

Тогда в стандартных предположениях независимости обучающих векторов и независимости цензурирования от ответов и признаков частичное правдоподобие p' распознавателя f (модели условных распределений $p(t|x)$, а значит и функций выживаемости $S(t|x)$, рисков $h(t|x)$ и всего прочего) равно

$$\begin{aligned}
p'(T|\mathbf{X},\mathbf{Y},f) &= \prod_{j=1}^{m} p(N_j|R_j,\mathbf{X},\mathbf{Y},f) \\
&= \prod_{j=1}^{m} p(N_j|R_j,\mathbf{X}_{R_j},t_{(j)},\nu_j,f) \\
&= \prod_{j=1}^{m} \frac{\prod\limits_{i\in N_j} p(t_i|x_i) \prod\limits_{i\in R_j\setminus N_j} S(t_i|x_i)}{\sum\limits_{M\subset R_j\ |\ \#M=\nu_j} \prod\limits_{i\in M} p(t_i|x_i) \prod\limits_{i\in R_j\setminus M} S(t_i|x_i)} \\
&= \prod_{j=1}^{m} \frac{\prod\limits_{i\in N_j} h(t_i|x_i)}{\sum\limits_{M\subset R_j\ |\ \#M=\nu_j} \prod\limits_{i\in M} h(t_i|x_i)}\ ,
\end{aligned}$$

т.е. удобно выражается именно через риски. Для модели пропорционального риска (247) все зависящие от времени сомножители $q(y_i)$ сокращаются, и частичное правдоподобие равно

$$p'(T|\mathbf{X},\mathbf{Y},r) = \prod_{j=1}^{m} \frac{\prod\limits_{i\in N_j} r(x_i)}{\sum\limits_{M\subset R_j\ |\ \#M=\nu_j} \prod\limits_{i\in M} r(x_i)}\ . \tag{251}$$

Если в обучающем наборе одновременных смертей нет, то вычисление ча-

стичного правдоподобия (251) упрощается до

$$p'(T|\mathbf{X},\mathbf{Y},r) = \prod_{j|\theta_j=0} \frac{r(x_j)}{\sum\limits_{i|t_i\geq t_j} r(x_i)} \ . \tag{252}$$

Общественная практика состоит в том, чтобы в задачах с непрерывным временем всегда вычислять частичное правдоподобие именно по формуле общего положения (252), даже тогда, когда она противоречит точной формуле (251).

B.5.3 Обучение зависимости от признаков

Для обучения модели Кокса нужно определить более конкретное пространство допустимых (всюду неотрицательных) функций риска r и, если нужно, априорное распределение вероятностей на нем. Чаще всего ограничиваются экспонентами от линейных однородных функций от векторов признаков[53]

$$r(x) = e^{wx} \tag{253}$$

не имеющими, заметим, ничего общего с экспоненциальным же убыванием выживаемости по времени при постоянном риске. Получающаяся функция частичного правдоподобия

$$p'(T|\mathbf{X},\mathbf{Y},w) = \prod_{j|\theta_j=0} \frac{e^{wx_j}}{\sum\limits_{i|t_i\geq t_j} e^{wx_i}} \tag{254}$$

(для непрерывного времени) уже вполне приемлема для максимизации и прочих вычислений, оптимизирующих параметр w. Удобнее, как обычно, проводить вычисления в терминах минус логарифма частичного правдоподобия. Тогда, например, максимизация апостериорной "частичной" вероятности будет записываться в виде

$$L(w|T) = \psi(w) + \sum_{j|\theta_j=0} \left(\ln \sum_{i|t_i\geq t_j} e^{wx_i} - wx_j \right) \to \min_w \tag{255}$$

где $\psi(w) = -\log p(w)$ — минус логарифм плотности априорной вероятности параметра w, а $L(w|T)$ — минус логарифм плотности апостериорной вероятности.

От традиционного обучения распознавателей обучение модели Кокса отличается тем, что сумма в формуле (255) не распадается в сумму слагаемых (ошибок) по отдельным обучающим векторам. Но многим численным методам оптимизации это безразлично. Зато при выпуклой функции ψ минимизационная задача (255) строго выпукла по w, так что минимум единственен и глобален.

Упражнение. Докажите, что вторые производные суммы

$$\sum_{j|\theta_j=0} \left(\ln \sum_{i|t_i\geq t_j} e^{wx_i} - wx_j \right)$$

по любому направлению w положительны.

В частности, для обучения модели Кокса успешно применяется метод лассо, т.е. $\psi(w) = \epsilon\|w\|_1$, и получающиеся решения разрежены по признакам, см. [109].

[53] Совершенно аналогично можно обучать модель Кокса в пространстве экспонент от линейных функций не от самих векторов признаков x, а от наборов базисных функций от признаков, например, задаваемых ядрами. Свободный член в линейной функции не нужен из-за инвариантности модели пропорционального риска (247) относительно одновременного умножения сомножителя r и деления q на положительное число.

B.5.4 Обучение зависимости от времени

После того, как зависимость риска в модели Кокса (247) от признаков $r(x)$ как-то оценена, его зависимость от времени $q(t)$ можно оценивать, например, максимизацией правдоподобия, (уже настоящего, а не частичного). Получение этой оценки очень похоже на вывод оценки Каплана-Майера (доказательство теоремы 12), поэтому совпадающие технические подробности будем опускать.

Как и в доказательстве теоремы 12 "базовая" функция выживаемости Q получается непрерывной слева монотонно невозрастающей ступенчатой функцией со скачками в моменты событий $t_{(1)} < \ldots < t_{(m)}$, а при ее оценке время дискретизируется с шагом ϵ, впоследствии устремляемом к нулю. Введем обозначения $Q_0 = Q(0) = 1$, $Q_j = Q(t_{(j)} + \epsilon)$. Тогда, перегруппировав обучающие объекты по возрастанию времени их наблюдения (до смерти или цензурирования), получаем правдоподобие

$$\begin{aligned} P(T|r,Q) &= \prod_{j=1}^{m} \left(\prod_{i \in N_j} \big(S(t_{(j)}|x_i) - S(t_{(j)} + \epsilon|x_i) \big) \prod_{i \in R_j \setminus (R_{j+1} \cup N_j)} S(t_{(j)} + \epsilon|x_i) \right) \\ &= \prod_{j=1}^{m} \left(\prod_{i \in N_j} \left((Q_{j-1})^{r(x_i)} - (Q_j)^{r(x_i)} \right) \prod_{i \in R_j \setminus (R_{j+1} \cup N_j)} (Q_j)^{r(x_i)} \right) \\ &= \prod_{j=1}^{m} \left(\prod_{i \in N_j} \left(1 - \left(\frac{Q_j}{Q_{j-1}} \right)^{r(x_i)} \right) \prod_{i \in R_j \setminus N_j} \left(\frac{Q_j}{Q_{j-1}} \right)^{r(x_i)} \right) . \end{aligned}$$

Его максимизация равносильна независимой максимизации по отношению $q_j = \frac{Q_j}{Q_{j-1}} \in [0,1]$ каждого (j-го) сомножителя:

$$\prod_{i \in N_j} \left(1 - q_j^{r(x_i)} \right) \prod_{i \in R_j \setminus N_j} q_j^{r(x_i)} \to \max_{q_j \in [0,1]} . \tag{256}$$

Приравнивая нулю производную этого сомножителя по q_j и игнорируя $q_j = 0$, заведомо не являющееся точкой максимума (256), получаем уравнение

$$\sum_{i \in N_j} \frac{r(x_i)}{1 - q_j^{r(x_i)}} = \sum_{i \in R_j} r(x_i) . \tag{257}$$

Упражнение. Проверьте, что решение уравнения (257) в точности является решением оптимизационной задачи (256).

Уравнение (257), вообще говоря, неразрешимо аналитически, но его левая часть строго монотонна по q_j, так что оно имеет единственное решение и легко решается численно. После того, как все m отношений q_j вычислены, вычисляются и высоты ступенек $Q_j = \prod_{k=1}^{j} q_k$ базовой функции выживаемости Q.

Если предположить, что все события некратны, т.е. все множества N_j одноэлементны, $N_j = \{i_j\}$, то уравнение (257) решается явно:

$$q_j = \left(1 - \frac{r\left(x_{i_j}\right)}{\sum_{i \in R_j} r(x_i)} \right)^{\frac{1}{r\left(x_{i_j}\right)}} , \tag{258}$$

и базовая функция выживаемости Q обученной модели Кокса тоже выписывается явно:

$$Q(t) = \prod_{i | \theta_i = 0 \ \& \ t_i < t} \left(1 - \frac{r(x_i)}{\sum\limits_{k | t_k \geq t_i} r(x_k)} \right)^{\frac{1}{r(x_i)}} . \tag{259}$$

Часто вместо точных формул (258) и (259) пользуются их приближениями:

$$q_j = e^{\frac{1}{r\left(x_{i_j}\right)} \ln\left(1 - \frac{r\left(x_{i_j}\right)}{\sum_{i \in R_j} r(x_i)}\right)} \approx e^{-\frac{1}{\sum_{i \in R_j} r(x_i)}}, \tag{260}$$

верным при $r\left(x_{i_j}\right) \ll \sum_{i \in R_j} r(x_i)$, т.е. далеко не всегда, и

$$Q(t) \approx \exp\left(-\sum_{i|\theta_i=0 \ \&\ t_i<t} \frac{1}{\sum_{k|t_k \geq t_i} r(x_k)}\right), \tag{261}$$

соответственно.

B.5.5 Прогнозирование

Суммируем результаты, полученные для регрессии Кокса. После того как обучена зависимость модели (247, 253) от признаков (раздел B.5.3), т.е. оценена линейная функция w, для прогнозирования выживаемости $S(t|x)$ объекта с вектором признаков x нужно в формулу (248) подставить результат обучения зависимости риска от признаков (253) и более или менее точную оценку базовой выживаемости (раздел B.5.4). Например, для ситуации общего положения при непрерывном времени с самой грубой приближенной оценкой (261) получится

$$S(t|x) \approx \exp\left(-e^{wx} \sum_{i|\theta_i=0 \ \&\ t_i<t} \frac{1}{\sum_{k|t_k \geq t_i} e^{wx_k}}\right),$$

а с более точной оценкой (259) —

$$S(t|x) = \prod_{i|\theta_i=0 \ \&\ t_i<t} \left(1 - \frac{1}{\sum_{k|t_k \geq t_i} e^{w(x_k - x_i)}}\right)^{e^{w(x-x_i)}}.$$

Замечание. Можно обучать зависимость модели Кокса от признаков (вектор w) и оценивать зависимость риска от времени на разных обучающих наборах. Это имеет смысл, когда природа изучаемых объектов не меняется со временем, а базовая выживаемость Q зависит от меняющихся со временем факторов, например, от погоды.

Замечание. Применяется и параметрический вариант модели Кокса, когда базовая функция выживаемости принадлежит фиксированному классу распределений (например, классу распределений Вейбулла). Такие базовые функции выживаемости тоже можно обучать максимизацией правдоподобия после того, как зависимость риска от признаков оценена. В зависимости от выбранного класса распределений такое обучение может оказаться и очень простым (например, для распределений Вейбулла с фиксированной степенью k), и очень сложным.

C Анализ последовательностей; марковские модели

Во всех предыдущих разделах статистическое обучение (чаще всего, обучение распознаванию, изредка — кластеризации, поиску выбросов и др.) рассматривалось в предположении, что признаки объектов принадлежат фиксированному пространству $\mathcal{X}$, обычно конечномерному евклидову, т.е. признаки — это

векторы фиксированной размерности. Однако во многих реальных задачах и даже в большинстве примеров задач, приведенных во введении (стр. 5) фиксация размерности пространства признаков либо не вполне естественна, как при узнавании картинок и распознавании букв или голосовых команд, либо вообще нереальна, как при распознавании рукописных текстов или речи.

Более естественным для таких задач является представление изучаемых объектов в виде структурированного набора, состоящего из не фиксированного заранее числа "элементарных" объектов, которые в свою очередь уже кодируются конечномерными векторами. Например,

- изображения можно представлять в виде матрицы точек, каждая из которых кодируется своим цветом, например, тремя числами для цветных изображений или одним битом для черно-белых;
- изображения можно также представлять в виде матрицы квадратиков фиксированного размера, каждый из которых имеет фиксированный набор как-то вычисленных характеристик (средний цвет, текстура и т.п.) или результат кластеризации или классификации по этим характеристикам (лес, вода, трава, асфальт и т.п. на аэрофотоснимке);
- звуковой сигнал можно представлять в виде последовательности векторов спектральных характеристик, замеренных в последовательных временны́х окнах фиксированной длительности;
- рукописный текст можно представлять в виде последовательности букв или палочек и кружочков, составляющих буквы;
- рукописный текст можно также представлять в виде последовательности слов, являющихся последовательностями букв, или последовательности строк, являющихся последовательностями слов или букв;
- данные для типичной задачи экономического или финансового прогнозирования — это фиксированный набор величин, измеряемых регулярно (ежедневно, ежемесячно, и т.п.) на протяжении некоторого не фиксированного времени.

Далее в этом разделе рассматривается анализ данных, представимых в виде последовательностей чего-то более простого: конечнозначных величин, конечномерных векторов или чего-то более сложного, но уже поддающегося классификации или кластеризации. Анализ изображений рассматривается (в очень ограниченном объеме) в разделе D.

Будем обозначать через $X_{[m,n]}$ последовательность векторов признаков $(x_m, ..$ через $Y_{[m,n]}$ последовательность ответов $(y_m, \ldots, y_n)$ и т.п. Как правило, индексы элементов последовательности будут начинаться с 1. В качестве обучающего набора обычно будет использоваться последовательность последовательностей вида $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_N) = (\mathbf{X}_{1[1,n_1]}, \ldots, \mathbf{X}_{N[1,n_N]})$, возможно, совместно с некоторым набором ответов $\mathbf{Y}$. j-й элемент i-й последовательности будем обозначать $x_{ij} = \left(\mathbf{X}_{i[1,n_i]}\right)_j$.

C.1 Задачи статистического анализа последовательностей

Напомним, что в общей постановке задачи распознавания (раздел 1.1.4) был задан обучающий набор $T = ((x_1, y_1), \ldots, (x_N, y_N))$ пар, состоящих из векторов признаков x_i и ответов y_i и нужно было научиться по заданному вектору признаков x, не принадлежащему обучающему набору, разумно предсказывать ответ y, а еще лучше — оценивать условные вероятности $p(y|x)$. При этом пары признаки-ответ предполагались независимыми реализациями одной и той

же случайной величины с неизвестным распределением. Кроме того в разделе 1.3 упоминались задачи кластеризации, оценки плотности и обнаружения выбросов, в которых ответы отсутствовали, а вся деятельность происходила в пространстве признаков.

При анализе последовательностей можно ставить и решать все те же задачи, а также еще некоторые, специфичные для последовательностей. В частности,

Прогнозирование. По последовательности $X_{[1,k]}$ предсказать следующий элемент x_{k+1} или оценить условное распределение $p(x_{k+1}|x_1,\ldots,x_k)$. В зависимости от пространства значений признаков x_i — дискретного или векторного — получается задача классификации (в случае оценки распределения ответов — логистической регрессии) или (многомерной) регрессии, соответственно. Например, последовательность может состоять из нулей и единиц и $x_i = 1$ означает, что в i-ю минуту (или час, или день) от начала наблюдений некоторый гейзер извергался, а $x_i = 0$ — что молчал.

Восстановление порождающей последовательности той же длины. Имеется обучающий набор последовательностей векторов признаков $\mathbf{X} = (\mathbf{X}_{1[1,k_1]}$ и набор последовательностей соответствующих им ответов $\mathbf{Y} = (\mathbf{Y}_{1[1,k_1]},\ldots,\mathbf{Y}_{N[1}$ Нужно научиться по любой последовательности векторов признаков $X_{[1,k]}$ находить разумную последовательность ответов $Y_{[1,k]}$. От распознавания отдельных элементов последовательности эта задача отличается тем, что пары (x_i, y_i) и в обучающих, и в распознаваемых последовательностях не предполагаются независимыми. Например, при распознавании рукописного текста, вписанного посимвольно в клеточки разграфленного бланка, x_i — это как-то закодированное изображение в i-й клеточке, а y_i — код соответствующего символа.

Восстановление порождающей последовательности несколько другой длины. Задача и обучающие данные такие же, как и в предыдущем пункте, только длины последовательностей ответов могут отличаться от длин соответствующих им последовательностей векторов признаков. Например, так устроено распознавание печатного текста, литеры которого в клеточки не вписаны, а разделяются программой сегментации, которая, может ошибаться, объединяя лигатуру “ffi” в один символ или разделяя букву “ы” на два.

Классификация последовательностей. Имеется обучающий набор последовательностей векторов признаков $\mathbf{X} = (\mathbf{X}_{1[1,k_1]},\ldots,\mathbf{X}_{N[1,k_N]})$ и набор соответствующих им ответов $\mathbf{Y} = (y_1,\ldots,y_N)$, не являющихся последовательностями или являющихся, но не рассматриваемых как таковые. Нужно научиться по любой последовательности векторов признаков $X_{[1,k]}$ находить наиболее подходящий ответ из присутствовавших в обучающем наборе (т.е. классифицировать последовательность). Например, при распознавании рукописного ввода непосредственно в компьютер с помощью пера (графического планшета), векторы признаков могут кодировать последовательные сплайны траектории пера, а ответы — либо отдельные буквы, либо слитно написанные слова. Или при распознавании речи векторы признаков могут кодировать спектр звукового сигнала в последовательных временны́х окнах, а ответы — произносимые слова.

Сегментация с классификацией сегментов. Обучающие данные такие же, как и в предыдущем пункте, т.е. “короткие” последовательности признаков, соответствующие ответам из фиксированного набора. Нужно научиться

любую "длинную" последовательность векторов признаков $X_{[1,k]}$ разумным образом разбивать на сегменты $X_{[1,k_1]}, X_{[k_1+1,k_2]}, \ldots, X_{[k_{l-1}+1,k]}$ и находить соответствующую последовательность ответов $Y_{[1,l]}$. Например, таковы задачи распознавания слитной речи при обучении на отдельных словах или словосочетаниях и распознавания слитного рукописного текста при обучении на отдельных буквах.

Приведенные примеры задач сразу наводят на следующие соображения.

- В задаче про прогноз извержений гейзера чего-то важного не хватает. Гейзер — это некоторая система сообщающихся сосудов с подогревом и извергается ли гейзер зависит не непосредственно от того, извергался ли он минуту или две минуты назад, а от того, где и сколько воды кипит под землей сию минуту. А вот это уже зависит от того, где сколько кипело под землей минуту (и две, и три, и более) назад, и довольно слабо зависит от того, что происходило на поверхности. То есть, наряду с простыми наблюдаемыми величинами на самом деле имеются неизвестно как устроенные ненаблюдаемые.

- В задачах про восстановление последовательности ответов по последовательности векторов признаков (почти) такой же длины возникает желание сначала обучить поэлементный распознаватель (в приведенном примере — распознаватель отдельных символов), восстанавливающий отдельный ответ по отдельному вектору признаков, а потом надстраивать над ним нечто, учитывающее взаимозависимость элементов.

Далее в разделе С.3 будет построена вероятностная модель последовательностей, пригодная в качестве основы для решения всех вышеописанных задач, и описаны способы ее обучения и применения, для разных задач разные. В разделе С.4, наоборот, будут приведены примеры задач и методов их решения, обходящихся без вероятностных моделей. А пока посмотрим, как вообще можно строить вероятностные модели последовательностей.

С.2 Вероятностные модели последовательностей, применяемые для статистического обучения

Вероятностная модель последовательностей неограниченной длины в пространстве $\mathcal{X}$ — это такая последовательность совместных распределений вероятности $p^k(x_1, \ldots, x_k)$, $x_i \in \mathcal{X}$, $k = 1, 2, \ldots$, что выполняется условие согласованности

$$p^k\left(x_1, \ldots, x_l, \underbrace{\mathcal{X}, \ldots, \mathcal{X}}_{k-l}\right) = p^l(x_1, \ldots, x_l) \text{ при } l < k \ .$$

Даже при самом маленьком нетривиальном пространстве $\mathcal{X}$ — двухэлементном — пространство моделей бесконечномерно и необозримо, а значит попытки статистического обучения модели бесперспективны. Чтобы строить обозримые модели приходится делать какие-то предположения о распределениях.

Безо всяких предположений, просто по определению условной вероятности, распределение $p^k(x_1, \ldots, x_k)$ представимо в виде произведения

$$\begin{aligned} p^k(x_1, \ldots, x_k) &= p_{k|k-1}(x_k|x_1, \ldots, x_{k-1}) p^{k-1}(x_1, \ldots, x_{k-1}) \\ &\ldots \\ &= p_{k|k-1}(x_k|x_1, \ldots, x_{k-1}) \ldots p_{2|1}(x_2|x_1) p^1(x_1) \ . \end{aligned} \tag{262}$$

Естественно, аналогичное произведение выписывается и для любой из $k!$ перестановок на множестве из k индексов. Но, если интерпретировать значение

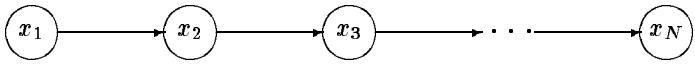

Рис. 6: Схематическое изображение зависимостей случайных величин (признаков) в марковской модели первого порядка.

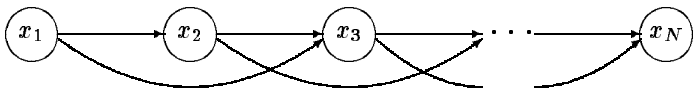

Рис. 7: То же для марковской модели второго порядка.

индекса как дискретное время, только в произведении (262) каждый сомножитель описывает условную вероятность некоторого "настоящего" относительно его "прошлого" (величин с меньшими индексами). Это соответствует желанию строить вероятностную модель не чисто формально, а "в соответствии с природой вещей". Кроме того, это очень удобно, если модель применяется для предсказания будущего по настоящему и прошлому.

Самое распространенное упрощающее предположение состоит в том, что условные вероятности $p_{k|k-1}(x_k|x_1,\dots,x_{k-1})$ в произведении (262) зависят не от всех переменных, а только от небольшого фиксированного числа m переменных с наибольшими индексами и, возможно, от самого индекса k, т.е. $p_{k|k-1}(x_k|x_1,\dots,x_{k-1}) = p_{k,m}(x_k|x_{k-m},\dots,x_{k-1})$. Такая модель называется *марковской моделью* (*марковской цепью*, *цепью Маркова*) порядка m. Если же условное распределение $p_{k,m}$ не зависит от k, то марковская модель называется *однородной*.

Легко видеть, что любая (однородная) марковская модель порядка m на пространстве $\mathcal{X}$ представима в виде (однородной, соответственно) марковской модели порядка 1 на пространстве $\mathcal{X}^k$ последовательностей длины m (тривиальное **упражнение**), так что обычно ограничиваются изучением марковских моделей первого порядка.

Однородная марковская модель первого порядка вполне обозрима: она определяется начальным распределением $p^1(x_1)$ и условным распределением перехода $p_{\cdot,1}(x_{i+1}|x_i)$. Разложение совместной вероятности (262) для нее выглядит так:

$$p^k(x_1,\dots,x_k) = p_{\cdot,1}(x_k|x_{k-1})\dots p_{\cdot,1}(x_2|x_1)p^1(x_1)\ . \tag{263}$$

Если пространство $\mathcal{X}$ конечно, то пространство всех таких моделей конечномерно (**упражнение**: посчитать его размерность), а если бесконечно, то приходится ограничиваться какими-то его конечномерными подпространствами.

Для наглядности разложение совместной вероятности случайных величин $a, b, c, d, \dots$ (например, элементов последовательности $x_1,\dots,x_k,\dots$) в произведение нескольких условных вероятностей вида $p(\cdot|\cdots)$ (до черты-условия стоит одна переменная, а после — сколько угодно) часто изображают следующим образом[54]. Берется ориентированный граф, вершины которого соответствуют случайными величинам, и для каждого сомножителя вида $p(a|b,c,\dots)$ проводятся ребра из вершин $b, c, \dots$ в вершину a. Например, марковская модель первого порядка изображена на рис. 6, а марковская модель второго порядка — на рис. 7. Однородны модели или нет, на рисунках не обозначено.

[54]Сомножители, являющиеся абсолютными, а не условными вероятностями, почему-то изображать не принято, хотя это совсем не трудно.

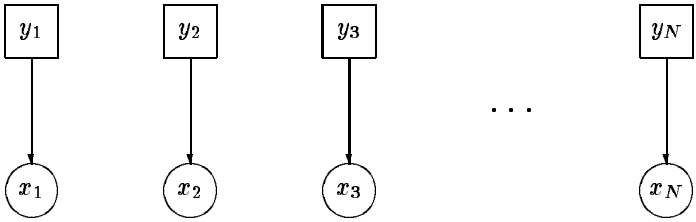

Рис. 8: Схематическое изображение зависимостей при распознавании независимых объектов: зависимость признаков x_i от ответов (классов) y_i ...

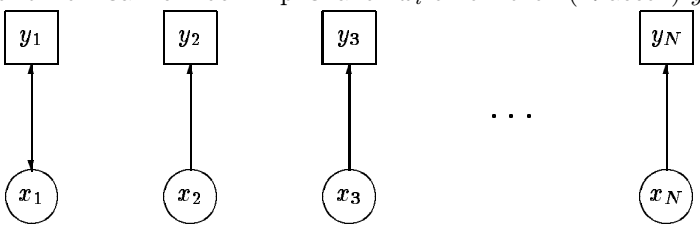

Рис. 9: ...и зависимость ответов y_i от признаков x_i.

Более разнообразны модели последовательностей в пространствах, имеющих дополнительную структуру. Например, в задачах распознавания последовательностей эти пространства состоят из пар (x_i, y_i), т.е. (вектор признаков, ответ), а в задачах прогнозирования (см. рассуждения про гейзеры на стр. 182) — из пар (наблюдаемые признаки, ненаблюдаемые переменные).

Даже в простом вырожденном случае, когда элементы последовательности независимы, есть по крайней мере два разумных типа моделей. В соответствии с двумя вариантами разложения совместной вероятности для каждого элемента последовательности $p(x_i, y_i) = p(y_i)p(x_i|y_i)$ и $p(x_i, y_i) = p(x_i)p(y_i|x_i)$, получаются два разных несвязных графа, изображенные на рис. 8 и 9. При этом модель, изображенная на рис. 8 соответствует, по крайней мере в задаче классификации, "природе вещей" (признаки объекта зависят от того, какому классу он принадлежит). Это традиционный способ построения порождающих моделей (см. раздел 1.2.3); в частности, такие модели применяются в наивном байесовским методе (раздел 1.2.5) и дискриминанте Фишера (раздел 2.2.1). Зато модель с рис. 9 (дискриминантная) непосредственно по признакам предсказывает ответ; такие модели применяются, например, в методе логистической регрессии (раздел 2.2.2).

В содержательном случае, когда элементы последовательности фактически зависимы, есть много типов моделей, различающихся упрощающими предположениями от том, что от чего не зависит. Наиболее распространенная и при этом одна из самых простых моделей, точно соответствующая соображениям про устройство гейзеров (стр. 182), получается следующим образом. Совместное распределение[55] $p(x_1, y_1, \ldots, x_k, y_k)$ всегда представимо в виде произведения

$$p(x_1, y_1, \ldots, x_k, y_k) = p(x_k|x_1, y_1, \ldots, x_{k-1}, y_{k-1}, y_k)p(y_k|x_1, y_1, \ldots, x_{k-1}, y_{k-1}) \cdot p(x_1, y_1 \ldots, x_{k-1}, y_{k-1})$$

...

[55] В этом абзаце все распределения обозначаются одной буквой p без индексов, поскольку аккуратное выписывание индексов слишком громоздко. Следите за индексами аргументов!

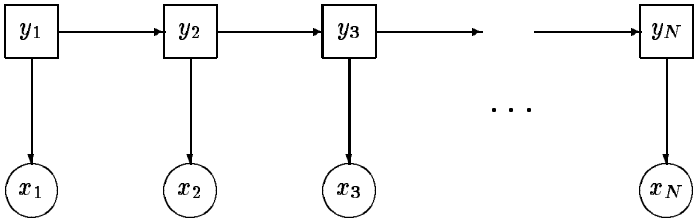

Рис. 10: Схематическое изображение зависимостей признаков (наблюдаемых состояний, наблюдений) x_i и ответов (классов, скрытых состояний) y_i в скрытой марковской модели.

$$= p(x_k|x_1, y_1, \ldots, x_{k-1}, y_{k-1}, y_k)p(y_k|x_1, y_1, \ldots, x_{k-1}, y_{k-1})$$
$$\cdots p(x_2|x_1, y_1, y_2)p(y_2|x_1, y_1)p(x_1|y_1)p(y_1) \ . \qquad (264)$$

Предположим, что условное распределение $p(x_i|\ldots)$ i-го вектора признаков зависит только от ответа (в задачах классификации ответы — это классы, которым принадлежат элементы последовательности, в задаче про гейзер — его ненаблюдаемые состояния) y_i с тем же индексом ("в тот же момент времени"), т.е.

$$p(x_i|x_1, y_1, \ldots, x_{i-1}, y_{i-1}, y_i) = p(x_i|y_i) \ ,$$

а распределение $p(y_i|\ldots)$ i-го ответа — только от предыдущего ответов (классов, состояний), т.е.

$$p(y_i|x_1, y_1, \ldots, x_{i-1}, y_{i-1}) = p(y_i|y_1, \ldots, y_{i-1}) \ .$$

Более того, как и в марковской модели первого порядка (263) предположим, что $p(y_i|\ldots)$ зависит только от предыдущего ответа y_{i-1}:

$$p(y_i|y_1, \ldots, y_{i-1}) = p(y_i|y_{i-1}) \ .$$

Получившаяся модель называется *скрытой марковской моделью* (*HMM*, *hidden Markov model*) и изображена на рис. 10. Векторы признаков x_i традиционно называются *наблюдаемыми состояниями* модели, а ответы y_i — *скрытыми состояниями*. Если же еще предположить, что модель *однородна*, т.е. что получившиеся условные распределения $p(x_i|y_i)$ и $p(y_i|y_{i-1})$ не зависят от индекса ("времени") i, то как и в однородной марковской модели (263), совместное распределение является произведением немногих маломерных распределений (а именно, трех: $p(x_i|y_i)$, $p(y_i|y_{i-1})$ и $p(y_1)$), хотя и в большом количестве:

$$p(x_1, y_1, \ldots, x_k, y_k) = p(x_k|y_k)p(y_k|y_{k-1}) \cdots p(x_2|y_2)p(y_2|y_1)p(x_1|y_1)p(y_1) \ . \quad (265)$$

Именно такие модели и их модификации будут рассматриваться в следующих разделах. То, что при построении модели из многих возможных вариантов разложения полной вероятности систематически выбирались варианты, "соответствующие природе вещей" облегчает введение в модель априорных предположений и объединение моделей друг с другом.

C.3 Скрытая марковская модель (HMM, Hidden Markov Model)

В этом разделе подробно рассматривается построение, обучение и применение простейших скрытых марковских моделей и кратко упоминаются более сложные скрытые марковские модели и их аналоги. Постоянно подчеркивается, что

для разных типов задач описанных на стр. 181–181 — прогнозирования, классификации, восстановления последовательности и др. — одинаковые на вид модели обучаются и применяются по-разному.

С.3.1 Скрытая марковская модель с дискретным временем, конечным пространством состояний и конечным пространством наблюдаемых

Распишем подробно, уже не обозначая все вероятностные распределения одной и той же буквой "p", однородную скрытую марковскую модель (HMM) (265), рис. 10, с конечным пространством (скрытых) состояний, в некоторых задачах имеющих также смысл ответов, $\mathcal{Y} = \{s_1, \ldots, s_n\}$ и конечным пространством наблюдаемых признаков $\mathcal{X} = \{o_1, \ldots, o_m\}$. Кроме количества скрытых состояний n и наблюдаемых состояний m она определяется конечным набором действительных параметров:

- n-мерным вектором-строкой Π начальных вероятностей скрытых состояний: $\pi_i = P\{y_1 = s_i\}$,
- $n \times n$-*матрицей перехода* (*transition matrix*) A: $a_{ij} = P\{y_{l+1} = s_j | y_l = s_i\}$ и
- $n \times m$-*матрицей эмиссии* (*emission matrix*) B: $b_{ij} = P\{x_l = o_j | y_l = s_i\}$,

удовлетворяющих стандартным вероятностным нормировкам (условию стохастичности): все элементы матриц и вектора неотрицательны и сумма элементов каждой строки равна 1. Итого получается $((n-1)+n(n-1)+n(m-1))$-мерное пространство параметров. Через $\mathbf{W}= (\Pi, A, B)$ будем обозначать весь набор параметров. В дальнейшем для краткости будем отождествлять и скрытые состояния, и наблюдаемые, с их номерами.

Подбор модели (одной или многих) для каждой задачи условно разбивается на два этапа. Сначала модель "строится": выбираются количество скрытых состояний n (если только, как в задаче распознавания последовательности символов, стр. 181, оно не определено самой задачей) и, иногда, количество значений признаков m (если на самом деле признаки непрерывны и их нужно дискретизировать искусственно). Кроме того используются имеющиеся априорные знания или гипотезы о природе задачи, чтобы сократить размерность пространства непрерывных параметров. Как правило, сокращение размерности состоит в разреживании вектора Π и матриц A и B, т.е. принудительном приравнивании большинства их элементов нулю. После этого модель "обучается", т.е. при сравнении модели с обучающим набором подбираются оптимальные значения непрерывных параметров.

Построение конкретных HMM

При построении скрытых марковских моделей наибольшее внимание уделяется подбору матрицы перехода A, поскольку, как будет показано далее, разреженность этой матрицы существенно ускоряет все вычисления с HMM. Разреженность матрицы перехода удобно изображать в виде ориентированного графа, вершины которого соответствуют скрытым состояниям s_i, и из i-й вершины в j-ю *не* идет ребро тогда и только тогда, когда вероятность перехода a_{ij} принудительно равна нулю. На рис. 11 изображен совершенно неинтересный граф переходов для HMM с тремя состояниями, в которой возможны все 9 переходов.

На рис. 12 изображен граф переходов для HMM с пятью состояниями, моделирующей циклический случайный процесс: все состояния проходятся по кругу,

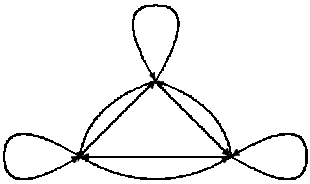

Рис. 11: Граф переходов универсальной (скрытой) марковской модели с тремя состояниями: допустимы все переходы.

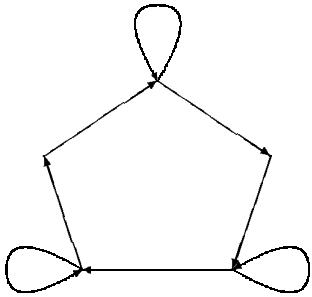

Рис. 12: Граф переходов циклической (скрытой) марковской модели с пятью состояниями: в трех из пяти состояний допустимы задержки.

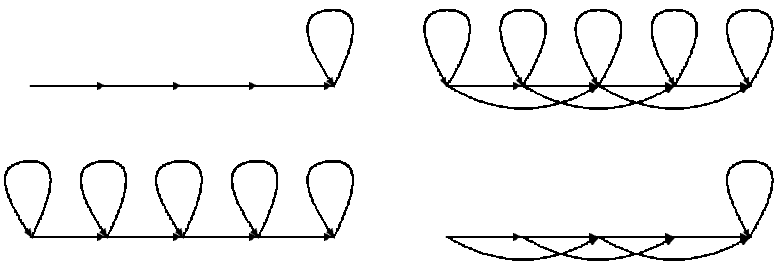

Рис. 13: Четыре примера графа переходов для LR-моделей с пятью состояниями. Левая верхняя модель на самом деле детерминирована — в ней вероятности всех переходов равны 1. В остальных моделях разрешены задержки во всех состояниях и/или пропуски отдельных состояний. Из-за условия стохастичности $\sum_j a_{ij} = 1$ в последнем состоянии любая LR-модель обязательно задерживается.
Левый верхний и правый нижний графы очень похожи на графы зависимостей для марковских моделей первого (рис. 6) и второго (рис. 7) порядков. Практического применения они не имеют и приведены только для напоминания о том, что не следует путать графы переходов с графами зависимостей.

но в некоторых состояниях с некоторой вероятностью процесс задерживается. Матрица перехода для этой модели уже разрежена: в каждой строке не более двух ненулевых элементов. Подобные модели применимы, например, в задаче про прогнозирование извержений гейзеров (стр. 181).

При моделировании последовательностей, имеющих явное начало и конец, например, при распознавании печатных, рукописных или произносимых слов, чаще применяются модели, в которых переход из любого состояния возможен только в состояние с не меньшим (и с не намного большим) номером, т.е. матрица перехода является верхней треугольной. Такие модели называются *лево-правыми* или *LR-моделями* (*left-right model*). Примеры лево-правых моделей изображены на рис. 13. Как правило, в LR-моделях вектор начальных вероятностей фиксирован: $\Pi = (1, 0, \ldots, 0)$.

LR-модели очень удобны для построения сложных моделей из небольших фрагментов. Предположим, что имеется марковская (не скрытая!) модель речи как последовательности слов, т.е. ориентированный граф, вершины которого

соответствуют словам, а ребра — последовательному произнесению пары слов. Тогда вместо каждой вершины можно подставить LR-модель (уже скрытую!) соответствующего слова, входившие в эту вершину ребра прицепив к начальному состоянию LR-модели, а выходившие прицепив к конечному состоянию вместо петли. Получится скрытая марковская модель устной речи. А если вместо вершин подставлять LR-модели рукописных слов, получится модель рукописного текста.

В задачах классификации последовательностей (см. стр. 181) для каждого класса нужно строить отдельную модель, причем размеры этих моделей могут оказаться существенно разными.

Отложим обучение скрытых марковских моделей до стр. 192 и займемся их применением.

Вычисления вероятностей с помощью HMM

Зафиксируем модель $\mathbf{W} = (\Pi, A, B)$ и проведем несколько простых, но полезных для дальнейшего, вычислений и оценим их трудоемкость. Вероятность того, что последовательность $Y_{[1,L]}$ скрытых состояний порождает последовательность $X_{[1,L]}$ наблюдений равна

$$P\{X_{[1,L]}|Y_{[1,L]}\} = \prod_{l=1}^{L} b_{y_l x_l}, \tag{266}$$

а вероятность появления самой последовательности $Y_{[1,L]}$ равна

$$P\{Y_{[1,L]}\} = \pi_{y_1} \prod_{l=1}^{L-1} a_{y_l y_{l+1}} \tag{267}$$

(обе эти равенства следуют непосредственно из формулы (265)). Значит вероятность наблюдения последовательности $X_{[1,L]}$, порожденной последовательностью $Y_{[1,L]}$, равна

$$P\{X_{[1,L]}, Y_{[1,L]}\} = P\{Y_{[1,L]}\}P\{X_{[1,L]}|Y_{[1,L]}\} = \pi_{y_1} \prod_{l=1}^{L-1} a_{y_l y_{l+1}} \prod_{l=1}^{L} b_{y_l x_l} \tag{268}$$

и вычисляется за $2L-1$ умножений, а вероятность наблюдения последовательности $X_{[1,L]}$, порожденной любой последовательностью состояний, равна

$$P\{X_{[1,L]}\} = \sum_{Y_{[1,L]}} P\{X_{[1,L]}, Y_{[1,L]}\} = \sum_{y_1,\ldots,y_L} \left(\pi_{y_1} \prod_{l=1}^{L-1} a_{y_l y_{l+1}} \prod_{l=1}^{L} b_{y_l x_l} \right) . \tag{269}$$

Но по этой формуле вероятность вычисляется катастрофически медленно: за порядка $2Ln^L$ арифметических операций. То есть для модели с всего лишь 10 скрытыми состояниями нереально посчитать вероятность последовательности наблюдений длины 100.

К счастью эту вероятность можно считать существенно быстрее с помощью алгоритма прямого и обратного хода или *"вперед-назад"* (*forward-backward algorithm*). Введем сокращенное обозначение $X = X_{[1,L]}$ и определим $\alpha_l(i, X)$

$$\alpha_l(i, X) = P\{X_{[1,l]}, y_l = i\} = \sum_{Y_{[1,l]}|y_l=i} P\{X_{[1,l]}, Y_{[1,l]}\} . \tag{270}$$

Тогда эти вероятности можно вычислять рекурсивно по l:

$$\alpha_1(i, X) \quad = \quad P\{x_1, y_1 = i\}$$

$$
\begin{aligned}
&= \pi_i b_{ix_1} \\
\alpha_{l+1}(i,X) &= \sum_{j=1}^{n} P\{X_{[1,l]}, y_l = j\} P\{x_{l+1}, y_{l+1} = i | X_{[1,l]}, y_l = j\} \\
&= \sum_{j=1}^{n} \alpha_l(j,X) a_{ji} b_{ix_{l+1}} \quad \text{при } 1 \le l < L \ ,
\end{aligned}
$$

а

$$
P\{X\} = \sum_{i=1}^{n} \alpha_L(i,X) \ . \tag{271}
$$

Таким способом $P\{X\}$, а заодно и все $\alpha_l(i,X)$ при $1 \le l \le L$ и $1 \le i \le n$ можно посчитать за порядка $3Ln^2$ операций. Более того, множитель n^2 — это количество элементов матрицы перехода. Если матрица перехода разреженная и в ней только n' элементов могут отличаться от нуля, то вероятность $P\{X\}$ можно посчитать за $3Ln'$ операций. На практике обычно строят модели с довольно большим числом n скрытых состояний, но не очень большим, не превышающим m', количеством допустимых (т.е. вероятность которых может быть ненулевой) переходов из каждого состояния, т.е. $n' \le m'n$. Странное обозначение m' намекает на то, что обычно $m' \approx m$ (количество наблюдаемых значений).

Вот еще несколько простых упражнений на осознание и быстрое вычисление условных вероятностей в скрытой марковской модели, результаты которых понадобятся при ее обучении.

Упражнение. Для последовательности наблюдений $X = X_{[1,L]}$ все условные вероятности $\beta_l(i,X)$

$$
\beta_l(i,X) = P\{X_{[l+1,L]} | y_l = i\} \tag{272}
$$

при $1 \le l \le L$ и $1 \le i \le n$ можно посчитать за порядка $3Ln'$ операций:

$$
\begin{aligned}
\beta_L(i,X) &= 1 \\
\beta_l(i,X) &= \sum_{j=1}^{n} a_{ij} b_{jx_{l+1}} \beta_{l+1}(j,X) \quad \text{при } l = L-1, \ldots, 1 \ .
\end{aligned}
$$

Упражнение. При любом $1 \le l \le L$ выполнены равенства

$$
\sum_{i=1}^{n} \alpha_l(i,X) \beta_l(i,X) = P\{X\} \ .
$$

Упражнение. Сколько нужно операций для вычисления всех условных вероятностей $\gamma_l(i,X)$ и $\xi_l(i,j,X)$, где

$$
\gamma_l(i,X) = P\{y_l = i | X\} = \frac{\alpha_l(i,X)\beta_l(i,X)}{P\{X\}} \tag{273}
$$

$$
\xi_l(i,j,X) = P\{y_l = i, y_{l+1} = j | X\} = \frac{\alpha_l(i,X) a_{ij} b_{jx_{l+1}} \beta_{l+1}(j,X)}{P\{X\}} \ ? \tag{274}
$$

Мораль этих упражнений такова: для последовательности наблюдений X длины L и HMM с n состояниями и n' ненулевыми элементами матрицы перехода ($n \le n' \le n^2$) за $O(Ln')$ операций можно посчитать все Ln вероятностей $\alpha_l(i,X)$ и условных вероятностей $\beta_l(i,X)$, после чего еще быстрее вычисляются вероятность последовательности X (она же — правдоподобие модели) (271) и разнообразные условные вероятности.

К сожалению, для вычислений вероятностей последовательностей длиной порядка 100 и более результаты этих упражнений неприменимы, поскольку

вероятность $P\{X\}$ каждой отдельной последовательности наблюдений исчезающе мала, меньше минимальных аппаратно реализуемых чисел типичных компьютеров. Уже при вычислении вероятностей $\alpha_l(i,X)$ и $\beta_l(i,X)$ из-за перемножения огромного числа вероятностей, т.е. чисел, меньших единицы, произойдет потеря точности. При этом условные вероятности $\gamma_l(i,X)$, $\xi_l(i,j,X)$ и т.п. совсем не обязательно малы. С потерей точности можно бороться чисто техническими средствами, а можно немного (всего лишь в константу раз) усложнить вычисления.

Вместо совместных вероятностей $\alpha_l(i,X) = P\{y_1 = i, X_{[1,l]}\}$ будем рекурсивно по l вычислять условные вероятности $\alpha'_l(i,X) = P\{y_1 = i | X_{[1,l]}\}$, $\eta_l(i,X) = P\{y_1 = i, x_l | X_{[1,l-1]}\}$ и $\eta_l(X) = P\{X_{[1,l]} | X_{[1,l-1]}\}$:

$$\left.\begin{aligned}
\eta_1(i,X) \quad (=P\{y_1=i,x_1\}) \quad &= \pi_i b_{ix_1} \\
\eta_1(X) \quad (=P\{x_1\}) \quad &= \sum_{i=1}^{n} \eta_1(i,X) \\
\alpha'_1(i,X) &= \frac{\eta_1(i,X)}{\eta_1(X)} \\
\eta_{l+1}(i,X) &= \sum_{j=1}^{n} \alpha'_l(i,X) a_{ij} b_{jx_{l+1}} \\
\eta_{l+1}(X) &= \sum_{i=1}^{n} \eta_{l+1}(i,X) \\
\alpha'_{l+1}(i,X) &= \frac{\eta_{l+1}(i,X)}{\eta_{l+1}(X)}
\end{aligned}\right\} \quad \text{при } 1 \le l < L \ .$$

Затем вместо условных вероятностей $\beta_l(i,X) = P\{X_{[l+1,L]} | y_l = i\}$ вычислим отношения условных вероятностей $\beta'_l(i,X) = \dfrac{P\{X_{[l+1,L]} | y_l = i\}}{P\{X_{[l+1,L]} | X_{[1,l]}\}}$:

$$\begin{aligned}
\beta'_L(i,X) &= 1 \\
\beta'_l(i,X) &= \frac{\sum_{j=1}^{n} a_{ij} b_{jx_{l+1}} \beta'_{l+1}(j,X)}{\eta_l(X)} \quad \text{при } l = L-1, \ldots, 1 \ .
\end{aligned}$$

В процессе этих вычислений никакого неустранимого стремления величин к нулю нет. Остальные условные вероятности легко вычисляются через $\alpha'_l(i,X)$, $\beta'_l(i,X)$ и $\eta_l(X)$:

$$\begin{aligned}
\gamma_l(i,X) &= \alpha'_l(i,X) \beta'_l(i,X) \\
\xi_l(i,j,X) &= \frac{\alpha'_l(i,X) a_{ij} b_{jx_{l+1}} \beta'_{l+1}(j,X)}{\eta_{l+1}(X)} \\
\zeta_l(k,X) &= \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha'_l(i,X) a_{ij} b_{jk}}{\sum_{i=1}^{n} \alpha'_l(i,X)} \ .
\end{aligned}$$

А вместо исчезающе малой вероятности $P\{X\}$ легко вычислить ее логарифм:

$$\ln P\{X\} = \sum_{l=1}^{L} \ln \eta_l(X) \ . \tag{275}$$

Упражнение. Проверьте вычисления.

Для градиентных методов обучения HMM нужно уметь быстро вычислять не только вероятность $P\{X\}$ или ее логарифм, но и градиент (удобнее — градиент логарифма) по параметрам модели $\mathbf{W}$.

Предложение 37

$$\frac{\partial \ln P\{X\}}{\partial \pi_i} = \frac{\gamma_1(i,X)}{\pi_i} \tag{276}$$

$$\frac{\partial \ln P\{X\}}{\partial a_{ij}} = \frac{\sum_{l=1}^{L-1} \xi_l(i,j,X)}{a_{ij}} \tag{277}$$

$$\frac{\partial \ln P\{X\}}{\partial b_{ik}} = \frac{\sum_{l:x_l=k} \gamma_l(i,X)}{b_{ik}} \tag{278}$$

Доказательство, чисто вычислительное, оставляется в качестве **упражнения**.

Применение HMM

Предположим, что модель или (в задачах классификации) несколько моделей уже обучены. Тогда решение задачи прогнозирования сводится к тому, чтобы по последовательности наблюдений $X = X_{[1,L]}$ вычислить вероятности

$$\begin{aligned} P\{x_{L+1}=k|X\} &= \sum_{i=1}^{n}\sum_{j=1}^{n} P\{y_L=i, y_{L+1}=j, x_{L+1}=k|X\} \\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} P\{y_L=i|X\}P\{y_{L+1}=j|y_L=i\}P\{x_{L+1}=k|y_{L+1}=j\} \\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} \gamma_L(i,X)a_{ij}b_{jk}\ . \end{aligned}$$

Решение задачи классификации начинается с того, что для модели каждого класса по формуле (271) оценивается ее правдоподобие или по формуле (275) его логарифм. Затем, если модели прилично обучены и известны априорные вероятности классов, по формуле Байеса вычисляются апостериорные вероятности и в качестве ответа выдается либо само апостериорное распределение, либо один или несколько наиболее вероятных классов. В экзотическом случае, когда либо не удается обучить модели, либо нет никаких соображений об априорных вероятностях (даже соображения, что все они равны), можно вычисляемые правдоподобия моделей считать набором базисных функций и над ними строить классификатор общими методами (см., например, раздел 2.3.1).

Интереснее всего решение задачи восстановления по имеющейся последовательности наблюдений $X = X_{[1,L]}$ последовательности скрытых состояний $Y^* = Y^*_{[1,L]}$, порождающей эти наблюдения с наибольшей вероятностью (см. стр. 181). Длины последовательностей X и Y^* должны совпадать; для задач восстановления последовательности состояний длины, отличной от длины последовательности наблюдений, стандартная HMM неприменима. Искать максимум вероятности (268) перебором по всем последовательностям состояний — неприемлемо долго (n^L вычислений формулы (268)), но можно воспользоваться методом динамического программирования, который для данного случая был впервые применен Эндрю Витерби в работе [118]. Выигрыш в скорости получается такой же, как при использовании формулы (271) и непосредственно предшествующих ей вместо (269).

А именно, аналогично формуле (270), определим $\epsilon_l(i,X)$

$$\epsilon_l(i,X) = \max_{Y_{[1,l]}|y_l=i} P\{X_{[1,l]}, Y_{[1,l]}\} \tag{279}$$

и будем вычислять $\epsilon_l(i,X)$ рекурсивно по l при всех i одновременно, запоминая для каждой пары (l,i) при $l>1$ предыдущее состояние $y^*_{l-1} = \upsilon_l(i,X)$ последовательности Y^*, максимизирующей правую часть равенства (279). Вот эти вычисления (с пояснениями в скобках):

$$\begin{aligned}
\epsilon_1(i,X) \quad (\quad &= \quad P\{x_1, y_1 = i\} \quad) \quad = \pi_i b_{ix_1} \\
\epsilon_{l+1}(i,X) \quad \Big(\quad &= \quad \max_{j|1\le j\le n} P\{X_{[1,l]}, y_l = j\} P\{x_{l+1}, y_{l+1} = i | X_{[1,l]}, y_l = j\} \quad \Big) \\
&= \quad b_{ix_l} \max_{j|1\le j\le n} \epsilon_l(j,X) a_{ji} \quad \text{при } 1 \le l < L \\
\upsilon_{l+1}(i,X) \quad &= \quad \arg \max_{j|1\le j\le n} \epsilon_l(j,X) a_{ji} \quad \text{при } 1 \le l < L \ .
\end{aligned}$$

Затем вычисляют максимальную совместную вероятность $P^*(X)$ последовательности наблюдений $X = X_{[1,L]}$ и порождающей ее последовательности скрытых состояний $Y^* = Y^*_{[1,L]}$ и — справа налево — саму максимизирующую последовательность Y^*:

$$\begin{aligned}
P^*(X) &= \max_{j|1\le j\le n} \epsilon_L(j,X) \qquad (280) \\
y^*_L(X) &= \arg \max_{j|1\le j\le n} \epsilon_L(j,X) \\
y^*_l(X_{[1,L]}) &= \upsilon_{l+1}(y^*_{l+1}(X_{[1,L]}), X_{[1,L]}) \quad \text{при } l = L-1, \ldots, 1 \ .
\end{aligned}$$

Как и при вычислении вероятности $P\{X_{[1,L]}\}$ последовательности наблюдений по формуле (271), для поиска наиболее вероятной последовательности состояний требуется порядка $3Ln'$ арифметических операций.

На практике, чтобы избежать возможной потери точности из-за того, что при больших l все $\epsilon_l(i,X)$ очень близки к нулю, в вышеописанном алгоритме Витерби вычисление $\epsilon_l(i,X)$ и $P^*(X)$ заменяется на вычисление $\ln \epsilon_l(i,X)$ $\ln P^*(X)$, соответственно:

$$\begin{aligned}
\ln \epsilon_1(i,X) &= \ln \pi_i + \ln b_{ix_1} \\
\ln \epsilon_{l+1}(i,X) &= \ln b_{ix_l} + \max_{j|1\le j\le n} (\ln \epsilon_l(j,X) + \ln a_{ji}) \quad \text{при } 1 \le l < L \\
\ln P^*(X) &= \max_{j|1\le j\le n} \ln \epsilon_L(j,X) \ ,
\end{aligned}$$

а вычисления последовательностей состояний остаются без изменений.

Обучение HMM

Обучение скрытой марковской модели состоит в подборе ее параметров $\mathbf{W}$, чтобы модель как можно лучше решала поставленную задачу на обучающем наборе последовательностей. Но задачи и обучающие наборы бывают разными, и соответственно, способы обучения HMM тоже бывают разными. Продолжим рассмотрение нескольких примеров задач и сформулируем соответствующие задачам цели обучения, а затем рассмотрим соответствующие методы обучения.

Прогнозирование (стр. 181). Обучающим набором является либо одна уникальная последовательность наблюдений X, как в задаче про гейзер, либо набор $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_N)$ таких последовательностей. Целью обучения обычно является построение наиболее вероятной модели при условии наблюдаемого набора[56], т.е. максимизация апостериорной вероятности. Но практически всегда все имеющиеся априорные предположения о вероятностях моделей заложены в конструкцию пространства моделей — сколько в моделях скрытых состояний и какие вероятности начальных состояний, переходов и эмиссий заведомо равны нулю. Тогда априорное распределение вероятностей на пространстве

[56] Байесовское обучение было бы еще лучше, но оно безнадежно сложно вычислительно.

допустимых моделей считается равномерным, и максимизация апостериорной вероятности $P\{\mathbf{W}|\mathbf{X}\}$ эквивалентна максимизации правдоподобия

$$P\{\mathbf{X}|\mathbf{W}\} \to \max_{\mathbf{W}} \ . \tag{281}$$

Классификация (стр. 181). Обучающим набором является набор последовательностей $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_N)$ (например, последовательностей звуков) и соответствующий ему набор ответов $\mathbf{Y} = (y_1, \ldots, y_N)$ (соответственно, произносимых слов), являющихся числами от 1 до q (то, что слова можно рассматривать как последовательности букв или слогов не учитывается). Обучается сразу q моделей $\mathbf{W}^1, \ldots, \mathbf{W}^q$, по одной на каждый ответ. В отличие от априорных вероятностей параметров моделей, априорные вероятности ответов $P^1, \ldots, P^q$, как правило, известны. Для любой последовательности наблюдений X q-мерный вектор

$$\big(P^1(X, \mathbf{W}), \ldots, P^q(X, \mathbf{W})\big) = \left(\frac{P^1 P\{X|\mathbf{W}^1\}}{\sum_{j=1}^q P^j P\{X|\mathbf{W}^j\}}, \ldots, \frac{P^q P\{X|\mathbf{W}^q\}}{\sum_{j=1}^q P^j P\{X|\mathbf{W}^j\}} \right)$$

должен оценивать условное распределение $(P\{y = 1|X\}, \ldots, P\{y = q|X\})$. Целью обучения моделей может быть, например, минимизация ожидания на обучающем наборе $(\mathbf{X}, \mathbf{Y})$ какой-либо функции штрафа E (см. раздел 1.2.6, формулу (46))

$$\sum_{i=1}^N \sum_{j=1}^q P^j(\mathbf{X}_i, \mathbf{W}) E(j, y_i) \to \min_{\mathbf{W}} \ , \tag{282}$$

в частности, минимизация ожидания числа ошибок классификации

$$\sum_{i=1}^N (1 - P^{y_i}(\mathbf{X}_i, \mathbf{W})) \to \min_{\mathbf{W}} \ ,$$

или максимизация правдоподобия

$$P\{\mathbf{Y}|\mathbf{X}, \mathbf{W}\} = \prod_{i=1}^N P^{y_i}(\mathbf{X}_i, \mathbf{W}) \to \max_{\mathbf{W}} \ , \tag{283}$$

эквивалентная минимизации взаимной энтропии (см. раздел 1.2.8)

$$-\ln P\{\mathbf{Y}|\mathbf{X}, \mathbf{W}\} = \sum_{i=1}^N -\ln P^{y_i}(\mathbf{X}_i, \mathbf{W}) \to \min_{\mathbf{W}} \ . \tag{284}$$

Такое обучение набора моделей называется *дискриминантным*.

Восстановление последовательности ответов (стр. 181). Обучающим набором является набор последовательностей $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_N)$ (например, последовательностей рукописных каракулей в клеточках бланка) и соответствующий ему набор последовательностей ответов $\mathbf{Y} = (\mathbf{Y}_1, \ldots, \mathbf{Y}_N)$ (соответственно, кодов написанных символов). Целью обучения моделей может быть, как и в случае прогнозирования, максимизация правдоподобия модели[57]

$$P\{\mathbf{X}, \mathbf{Y}|\mathbf{W}\} = \prod_{i=1}^N P\{\mathbf{X}_i, \mathbf{Y}_i|\mathbf{W}\} \to \max_{\mathbf{W}} \ , \tag{285}$$

[57] В отличие от обучения для прогнозирования, в этом случае байесовское обучение с равномерным априорным распределением технически не сложно (интегрирование монома по прямому произведению симплексов) и очень похоже на байесовское обучение наивной байесовской модели. Но его результат непонятно как применить: для этого пришлось бы проинтегрировать по полученному апостериорному распределению алгоритм Витерби.

где совместные вероятности $P\{\mathbf{X}_i, \mathbf{Y}_i|\mathbf{W}\}$ вычисляются явно (формула (268)), в отличие от вероятностей $P\{\mathbf{X}_i|\mathbf{W}\}$ в предыдущих случаях.

Сегментация с классификацией сегментов (стр. 181). Как и в задаче классификации, обучающим набором является набор последовательностей $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_N)$ (например, акустических записей произносимых) и соответствующий ему набор ответов $\mathbf{Y} = (\mathbf{Y}_1, \ldots, \mathbf{Y}_N)$ (соответственно, номеров произносимых слов в каком-то словаре) и обучается сразу q моделей $\mathbf{W}^1, \ldots, \mathbf{W}^q$, по одной на каждый ответ (слово). Но целью обучения является не самая лучшая классификация отдельных слов q моделями, а хорошее распознавание слитной речи как последовательности слов с помощью одной объединенной модели. Для восстановления последовательности слов в объединенной модели применяется алгоритм Витерби или какой-либо его аналог, поэтому, в отличие от классификации отдельных слов, не так важно, чтобы набор моделей слов давал правильную оценку апостериорных вероятностей слов, как то, чтобы выдаваемая алгоритмом Витерби наиболее вероятная последовательность скрытых состояний имела наибольшую вероятность в правильной модели (среди всех q моделей). Для простоты уточнение в скобках ("среди всех q моделей") обычно игнорируют. Тогда каждую из q моделей (j-ую) можно обучать независимо на подмножестве обучающего набора $\mathbf{X}^j = \{\mathbf{X}_i|\mathbf{Y}_i = j\}$ (т.е. на примерах произнесения только j-го слова), максимизируя произведение выдаваемых алгоритмом Витерби максимальных совместных вероятностей P^* (280) наблюдаемых и скрытых последовательностей:

$$P^*\{\mathbf{X}^j|\mathbf{W}^j\} = \prod_{i:\mathbf{Y}_i=j} P^*\{\mathbf{X}_i|\mathbf{W}^j\} \to \max_{\mathbf{W}^j} . \tag{286}$$

Далее подробнее рассматриваются методы решения оптимизационных задач (281)–(286).

Для обучения HMM максимизацией правдоподобия последовательностей наблюдений (281) можно применять градиентные и стохастические методы, как и для нейронных сетей (раздел 3.2.4), поскольку способ вычисления градиента правдоподобия предъявлен в формулах (276)–(278) и предшествующих. Но чаще применяют другой метод, использующий то, что модель — вероятностная, и тоже похожий на обучение нейронных сетей, а именно обучение RBF-сетей в разделе 3.3.1. Этот метод, называющийся алгоритмом Баума-Велша, был разработан в серии работ Леонарда Баума с соавторами в конце 1960-х годов, например, [7] или [8], в которых формально Ллойд Велш соавтором не является (см. лекцию Велша [120]). Алгоритм Баума-Велша, как и метод обучения RBF-сетей максимизацией правдоподобия, является частным случаем метода максимизации ожидания (раздел А.3.2). Он является итеративным и сходится, вообще говоря, не к глобальному максимуму правдоподобия, а к локальному.

Поскольку параметры модели теперь будут меняться, будем явно указывать их в формулах. Зафиксируем обучающий набор $\mathbf{X}$ из N последовательностей наблюдений $\mathbf{X}_i = \mathbf{X}_{i[1,L_i]}$ длин L_i и начальный набор параметров модели $\mathbf{W}^{(0)}$ (например, случайный) и будем пытаться увеличить правдоподобие $P\{\mathbf{X}|\mathbf{W}\}$ или, что то же самое, уменьшить $E\{\mathbf{X}|\mathbf{W}\} = -\ln P\{\mathbf{X}|\mathbf{W}\}$. Из формулы (269) следует, что

$$P\{\mathbf{X}|\mathbf{W}\} \quad = \quad \prod_{i=1}^{N} P\{\mathbf{X}_i|\mathbf{W}\} = \prod_{i=1}^{N} \sum_{Y_{[1,L_i]}} P\{\mathbf{X}_{i[1,L_i]}, Y_{[1,L_i]}|\mathbf{W}\}$$

$$= \prod_{i=1}^{N} \sum_{y_1,\ldots,y_{L_i}} \left(\pi_{y_1} \prod_{l=1}^{L_i-1} a_{y_l y_{l+1}} \prod_{l=1}^{L_i} b_{y_l x_l^i} \right),$$

т.е. мы пытаемся максимизировать многочлен степени $2\sum_{i=1}^{N} L_i$ с неотрицательными коэффициентами от не более чем $n+n^2+mn$ переменных (их может быть меньше из-за разреженности матрицы перехода) в пересечении положительного ортанта с набором гиперплоскостей вида

$$\sum_{j=1}^{n} \pi_j = 1,\ \sum_{k=1}^{n} a_{jk} = 1 \text{ и } \sum_{k=1}^{m} b_{jk} = 1. \tag{287}$$

Для этой задачи легко выписывается лагранжиан, но уравнения Лагранжа получаются полиномиальными весьма высокой степени и аналитически не решаются. Поэтому как и при обучении RBF-сетей (стр. 88) мы пойдем другим путем.

$$\begin{aligned}
E(\mathbf{X},\mathbf{W}) - E(\mathbf{X},\mathbf{W}^{(0)}) &= -\sum_{i=1}^{N} \ln \frac{P\{\mathbf{X}_i|\mathbf{W}\}}{P\{\mathbf{X}_i|\mathbf{W}^{(0)}\}} \\
&= -\sum_{i=1}^{N} \ln \left(\sum_{Y_{[1,L_i]}} \frac{P\{\mathbf{X}_i, Y_{[1,L_i]}|\mathbf{W}\}}{P\{\mathbf{X}_i|\mathbf{W}^{(0)}\}} \right) \\
&= -\sum_{i=1}^{N} \ln \left(\sum_{Y_{[1,L_i]}} P\{Y_{[1,L_i]}|\mathbf{X}_i,\mathbf{W}^{(0)}\} \frac{P\{\mathbf{X}_i, Y_{[1,L_i]}|\mathbf{W}\}}{P\{Y_{[1,L_i]}|\mathbf{X}_i,\mathbf{W}^{(0)}\}P\{\mathbf{X}_i|\mathbf{W}^{(0)}\}} \right) \\
&= -\sum_{i=1}^{N} \ln \left(\sum_{Y_{[1,L_i]}} P\{Y_{[1,L_i]}|\mathbf{X}_i,\mathbf{W}^{(0)}\} \frac{P\{\mathbf{X}_i, Y_{[1,L_i]}|\mathbf{W}\}}{P\{\mathbf{X}_i, Y_{[1,L_i]}|\mathbf{W}^{(0)}\}} \right) \\
&\le -\sum_{i=1}^{N} \sum_{Y_{[1,L_i]}} P\{Y_{[1,L_i]}|\mathbf{X}_i,\mathbf{W}^{(0)}\} \ln \left(\frac{P\{\mathbf{X}_i, Y_{[1,L_i]}|\mathbf{W}\}}{P\{\mathbf{X}_i, Y_{[1,L_i]}|\mathbf{W}^{(0)}\}} \right) \\
&= -\sum_{i=1}^{N} \sum_{Y_{[1,L_i]}} P\{Y_{[1,L_i]}|\mathbf{X}_i,\mathbf{W}^{(0)}\} \ln P\{\mathbf{X}_i, Y_{[1,L_i]}|\mathbf{W}\} \\
&\quad + \sum_{i=1}^{N} \sum_{Y_{[1,L_i]}} P\{Y_{[1,L_i]}|\mathbf{X}_i,\mathbf{W}^{(0)}\} \ln P\{\mathbf{X}_i, Y_{[1,L_i]}|\mathbf{W}^{(0)}\}
\end{aligned}$$

Неравенство в этой цепочке преобразований следует из выпуклости функции $-\ln(\cdot)$ и того, что условные вероятности $P\{Y_{[1,L_i]}|\mathbf{X}_i,\mathbf{W}^{(0)}\}$ являются вероятностями, т.е. неотрицательны и их сумма по всем возможным последовательностям Y длины L_i равна 1.

Введем обозначение

$$Q(\mathbf{X},\mathbf{W}^{(0)},\mathbf{W}) = -\sum_{i=1}^{N} \sum_{Y_{[1,L_i]}} P\{Y_{[1,L_i]}|\mathbf{X}_i,\mathbf{W}^{(0)}\} \ln P\{\mathbf{X}_i, Y_{[1,L_i]}|\mathbf{W}\}\ .$$

Тогда полученное неравенство означает, что

$$E(\mathbf{X},\mathbf{W}) \le Q(\mathbf{X},\mathbf{W}^{(0)},\mathbf{W}) - Q(\mathbf{X},\mathbf{W}^{(0)},\mathbf{W}^{(0)}) + E(\mathbf{X},\mathbf{W}^{(0)})\ .$$

Правая часть мажорирует $E(\mathbf{X},\mathbf{W})$ и совпадает с ней в точке $\mathbf{W}^{(0)}$. Значит чтобы уменьшить значение $E(\mathbf{X},\mathbf{W})$ достаточно найти такую точку $\mathbf{W}$, что

$Q(\mathbf{X}, \mathbf{W}^{(0)}, \mathbf{W}) < Q(\mathbf{X}, \mathbf{W}^{(0)}, \mathbf{W}^{(0)})$. Но у функции $Q(\mathbf{X}, \mathbf{W}^{(0)}, \cdot)$ нетрудно найти точки глобального минимума по всей области определения с помощью метода множителей Лагранжа.

Заметьте, что предыдущий абзац скопирован из описания алгоритма обучения RBF-сетей (стр. 89) почти дословно. Однако для марковских моделей вычисления несколько более громоздки (не помещаются в строку) и мы их оставим в качестве технического **упражнения**, ограничившись предъявлением и обсуждением ответа.

Предложение 38

Минимум функции $Q(\mathbf{X}, \mathbf{W}^{(0)}, \cdot)$ достигается в точке $\mathbf{W}^*$ с координатами

$$\pi_j^* = \frac{1}{N}\sum_{i=1}^{N}\gamma_1(j, \mathbf{X}_i) \tag{288}$$

$$a_{jk}^* = \frac{\sum_{i=1}^{N}\sum_{l=1}^{L_i-1}\xi_l(j,k,\mathbf{X}_i)}{\sum_{i=1}^{N}\sum_{l=1}^{L_i-1}\gamma_l(j,\mathbf{X}_i)} \tag{289}$$

$$b_{jk}^* = \frac{\sum_{i=1}^{N}\sum_{l|x_l^i=k}\gamma_l(j,\mathbf{X}_i)}{\sum_{i=1}^{N}\sum_{l=1}^{L_i}\gamma_l(j,\mathbf{X}_i)}\ . \tag{290}$$

Здесь $\gamma_l(\cdot,\cdot)$ и $\xi_l(\cdot,\cdot,\cdot)$ определены равенствами (270), (272), (273) и (274) для параметров "старой" модели $\mathbf{W}^{(0)}$. Если при каком-то значении j знаменатель правой части уравнения (289) или (290) равен 0, то числитель тоже равен 0 и $Q(\mathbf{X}, \mathbf{W}^{(0)}, (\Pi, A, B))$ не зависит от j-й строки матрицы A (или, соответственно, B), т.е. точка минимума $\mathbf{W}^*$ не единственна. В этом случае для однозначности положим $a_{jk}^* = a_{jk}^{(0)}$ (соответственно, $b_{jk}^* = b_{jk}^{(0)}$) при всех k.

Если $\mathbf{W}^* \neq \mathbf{W}^{(0)}$, то $Q(\mathbf{X}, \mathbf{W}^{(0)}, \mathbf{W}^*) < Q(\mathbf{X}, \mathbf{W}^{(0)}, \mathbf{W}^{(0)})$.

Алгоритм Баума-Велша состоит в итеративном пересчете параметров марковской модели по формулам (288)–(290), пока модель не стабилизируется или пока не исчерпается лимит времени. Он замечателен во многих отношениях.

Упражнение. Проверьте следующие утверждения:

- Шаг алгоритма можно менее формально сформулировать следующим образом [58]:

$$\pi_j^* = \text{вероятность } j\text{-го скрытого состояния в качестве начального}$$

$$a_{jk}^* = \frac{\text{ожидание числа переходов из } j\text{-го скрытого состояния в } k\text{-е}}{\text{ожидание числа выходов из } j\text{-го скрытого состояния}}$$

$$b_{jk}^* = \frac{\text{ожидание числа пар } (j\text{-е скрытое, } k\text{-е наблюдаемое})}{\text{ожидание числа появлений } j\text{-го скрытого состояния}}$$

  Вероятности и ожидания вычисляются при "старых" значениях параметров модели $\mathbf{W}^{(0)}$ и фиксированном наборе обучающих последовательностей наблюдений $\mathbf{X}$. Если знаменатель правой части уравнений обращается в 0, т.е. набор обучающих последовательностей недостаточно представителен, чтобы все скрытые состояния модели могли реализоваться, то соответствующие нереализуемым состояниям строки матриц перехода и эмиссии не обучаются.

[58] В своей лекции [120] Велш рассказывал, что он догадался применить это преобразование для оценки параметров HMM и сколько-то раз экспериментально проверил, что правдоподобие моделей при нем увеличивалось, но доказать, что оно увеличивается всегда, не смог.

- При пересчете автоматически сохраняется неотрицательность параметров и вероятностные нормировки (287).

- При пересчете автоматически сохраняется разреженность матриц перехода A и эмиссии B и начального распределения Π: если какой-то элемент был равен нулю, он и останется нулем.

- Пересчет довольно быстр — порядка $C(n'+mn)\sum_{i=1}^{N} L_i$ арифметических операций, где C — небольшая константа, см. упражнения на стр. 189. Т.е. время пересчета пропорционально произведению количества обучаемых параметров на объем обучающего материала.

Начальные значения ненулевых элементов матрицы перехода A и вектора начальных вероятностей Π можно задавать произвольно, уважая вероятностные нормировки. Начальные значения матрицы эмиссии B можно пытаться задавать осмысленно, если есть соображения, какие скрытые состояния могут порождать какие наблюдаемые.

Алгоритм Баума-Велша всегда сходится и при этом почти всегда — к точке локального максимума правдоподобия $P\{\mathbf{X}|\mathbf{W}\}$, поскольку он на каждом шаге увеличивает значение непрерывной функции на компактном пространстве параметров. Однако остаются вопросы, на которые нет универсального ответа: с какой скоростью он сходится и сходится ли он к точке глобального максимума.

Для дискриминантного обучения HMM можно применять градиентные методы, выразив градиенты входящих в оптимизируемые функционалы (например, (282) или (284)) апостериорных вероятностей $P^j(\mathbf{X}_i, \mathbf{W})$ через градиенты логарифмов правдоподобий $P\{X|\mathbf{W}^j\}$, вычисляемые с помощью предложения 37,

$$\frac{\partial P^j(\mathbf{X}_i, \mathbf{W})}{\partial \mathbf{W}^k} = P^j(\mathbf{X}_i, \mathbf{W}) \left(\delta_k^j \frac{\partial \ln P\{X|\mathbf{W}^j\}}{\partial \mathbf{W}^j} - P^k(\mathbf{X}_i, \mathbf{W}) \frac{\partial \ln P\{X|\mathbf{W}^k\}}{\partial \mathbf{W}^k} \right)$$

и спроектировав градиент на пересечение плоскостей (287) для всех q моделей. Альтернативой градиентному спуску является применение обобщения алгоритма Баума-Велша [46], позволяющего оптимизировать функции от правдоподобий, которое мы сейчас опишем.

Сравнивая шаг алгоритма Баума-Велша (288)–(290) с вычислением градиента логарифма правдоподобия (276)–(278), немедленно получаем

Предложение 39 *Шаг алгоритма Баума-Велша эквивалентен пересчету*[59]

$$\begin{aligned}
\pi_j^* &= \frac{\pi_j \sum_{i=1}^{N} \frac{\partial P\{\mathbf{X}_i\}}{\partial \pi_j}}{\sum_{j=1}^{n} \pi_j \sum_{i=1}^{N} \frac{\partial P\{\mathbf{X}_i\}}{\partial \pi_j}} \\
a_{jk}^* &= \frac{a_{jk} \sum_{i=1}^{N} \frac{\partial P\{\mathbf{X}_i\}}{\partial a_{jk}}}{\sum_{k=1}^{n} a_{jk} \sum_{i=1}^{N} \frac{\partial P\{\mathbf{X}_i\}}{\partial a_{jk}}} \\
b_{jk}^* &= \frac{b_{jk} \sum_{i=1}^{N} \frac{\partial P\{\mathbf{X}_i\}}{\partial b_{jk}}}{\sum_{k=1}^{n} b_{jk} \sum_{i=1}^{N} \frac{\partial P\{\mathbf{X}_i\}}{\partial b_{jk}}} \ .
\end{aligned}$$

□

[59] Именно так формулировался алгоритм в первых работах Баума, например, [7]

Обобщенный алгоритм Баума-Велша, предложенный в работе [46] для поиска максимума по $\mathbf{W}$ некоторой функции $R(\mathbf{X},\mathbf{W})$ от обучающего набора $\mathbf{X}$ и параметров HMM $\mathbf{W}$, это последовательное применение следующих итераций:

$$\begin{aligned}
\pi_j^* &= \frac{\pi_j \sum_{i=1}^N \left(\frac{\partial R(\mathbf{X}_i,\mathbf{W})}{\partial \pi_j} + C\right)}{\sum_{j=1}^n \pi_j \sum_{i=1}^N \left(\frac{\partial R(\mathbf{X}_i,\mathbf{W})}{\partial \pi_j} + C\right)} \\
a_{jk}^* &= \frac{a_{jk} \sum_{i=1}^N \left(\frac{\partial R(\mathbf{X}_i,\mathbf{W})}{\partial a_{jk}} + C\right)}{\sum_{k=1}^n a_{jk} \sum_{i=1}^N \left(\frac{\partial R(\mathbf{X}_i,\mathbf{W})}{\partial a_{jk}} + C\right)} \\
b_{jk}^* &= \frac{b_{jk} \sum_{i=1}^N \left(\frac{\partial R(\mathbf{X}_i,\mathbf{W})}{\partial b_{jk}} + C\right)}{\sum_{k=1}^n b_{jk} \sum_{i=1}^N \left(\frac{\partial R(\mathbf{X}_i,\mathbf{W})}{\partial b_{jk}} + C\right)} ,
\end{aligned}$$

где $C > 0$. В работе [46] показано, что для любой полиномиальной или рациональной по $\mathbf{W}$ функции R (в частности, для функции (283) или (282) с противоположным знаком) при достаточно больших, зависящих от R, значениях параметра C алгоритм сходится к точке (локального!) максимума функции.

На практике при значениях C, гарантирующих сходимость к максимуму, сходимость неприемлемо медленная, но при меньших значениях алгоритм, как правило, сходится быстро и, как правило, куда надо. Опустим все подробности и сошлемся в качестве примера на статью [95], одну из очень многих подобных, в которой сообщается об успешном практическом применении двух вариантов дискриминантного обучения и обобщенного алгоритма Баума-Велша, а также приводятся литературные ссылки.

Обучение HMM максимизацией совместного правдоподобия последовательностей наблюдений и состояний — самое простое. В нем не требуются итеративные алгоритмы, а явно предъявляется ответ.

Предложение 40 *Максимум совместного правдоподобия (285) достигается в точке*

$$\begin{aligned}
\pi_j &= \frac{\sum_{i=1}^N \delta_j^{y_{i1}}}{N} \\
a_{jk}^* &= \frac{\sum_{i=1}^N \sum_{l=1}^{L_i-1} \delta_j^{y_{il}} \delta_k^{y_{i(l+1)}}}{\sum_{i=1}^N \sum_{l=1}^{L_i-1} \delta_j^{y_{il}}} \\
b_{jk}^* &= \frac{\sum_{i=1}^N \sum_{l|x_l^i=k} \delta_j^{y_{il}} \delta_k^{x_{il}}}{\sum_{i=1}^N \sum_{l=1}^{L_i} \delta_j^{y_{il}}} .
\end{aligned}$$

Доказательство. Правдоподобие (285) — это моном от параметров модели, которые пробегают прямое произведение $2n+1$ симплексов, задаваемых вероятностными нормировками параметров. См. лемму 2. □

Обучение HMM максимизацией оптимального правдоподобия (286) производится итеративно. Для модели с какими-то, например, случайными, начальными значениями параметров и обучающего набора $\mathbf{X}$ из N последовательностей наблюдений с помощью алгоритма Витерби восстанавливается набор $\mathbf{Y}$ из N последовательностей скрытых состояний, после чего параметры модели переоцениваются согласно предложению 40. И так далее. Поскольку количество возможных последовательностей скрытых состояний конечно,

алгоритм сходится (параметры модели перестают меняться) за конечное число шагов, хотя, возможно, слишком большое. На каждом шаге правдоподобие (286) возрастает.

Неформально говоря, этот алгоритм соотносится с алгоритмом Баума-Велша так же, как алгоритм кластеризации k-means (стр. 36) с оценкой параметров гауссовой смеси (стр. 86), поэтому его называют *segmental k-means*, см. статьи [93] и [62], хотя ни параметра k, ни каких-либо средних в нем нет.

С.3.2 Обобщения скрытых марковских моделей и объединение их с нейронными сетями и другими распознавателями

В предыдущем разделе была описана самая простая скрытая марковская модель. Для распознавания и моделирования более сложных последовательностей более сложных объектов применяются более сложные модели более общего вида. Обобщения можно условно подразделить на "научные" и "технические". В первом случае строится честная вероятностная модель нескольких взаимодействующих явных или скрытых случайных процессов в разных пространствах. Во втором случае модель преобразуется каким-либо способом, вообще говоря нарушающим вероятностные мотивировки и нормировки, но сохраняющим возможность применения быстрых алгоритмов распознавания и обучения, похожих на алгоритмы Витерби и Баума-Велша. Целью преобразования является уменьшение размеров модели и, следовательно, надежда на ускорение ее обучения и избежание переобучения.

Приведем примеры и мотивации возможных "научных" обобщений скрытой марковской модели.

Непрерывное пространство наблюдаемых состояний. Это обобщение необходимо почти во всех приложениях. Например, чтобы приспособить стандартную HMM к распознаванию последовательности раздельно написанных символов, на стр. 182 было предложено предварительно классифицировать написанные символы независимым распознавателем. Если этот распознаватель допускает градиентное обучение, то и объединенную распознающую систему можно обучать градиентными методами. Чаще всего марковские модели интегрируют таким образом друг с другом или с нейронными сетями.

Альтернативный и более популярный подход состоит в том, чтобы к каждому скрытому состоянию марковской модели привязать вероятностную модель распределения признаков, обучаемую совместно с самой HMM алгоритмами типа Баума-Велша. Наблюдаемое состояние модели — это последовательность X случайных векторов $x_1, \ldots, x_L$ в некотором пространстве (как правило, в евклидовом пространстве $\mathbb{R}^d$) с плотностью распределения

$$p(x_l|x_1, \ldots, x_{l-1}, x_{l+1}, \ldots, s_{j_1}, \ldots, s_{j_l} \ldots) = p(x_l|s_{j_l}) = b_{j_l}(x_l),$$

не зависящей от $l, j_1, \ldots, j_{l-1}, j_{l+1}, \ldots$. Эти плотности, обозначенные $b_j(x)$, принадлежат какому-то заранее выбранному m-мерному параметризованному семейству, т.е. b_j задается вещественными параметрами $b_{j1}, \ldots, b_{jm}$. Матрица параметров $B = (b_{ij})$ играет ту же роль, что матрица эмиссии в случае дискретного пространства состояний.

Пример. Пусть b_j — гауссова смесь (127)

$$b_j(x) = \sum_{k=1}^{m} P_{jk} p_{jk}(x) = \sum_{k=1}^{m} P_{jk} \frac{1}{(2\pi s_{jk})^{\frac{d}{2}}} e^{-\frac{\|x-\mu_{jk}\|^2}{2s_{jk}}} .$$

Тогда для восстановления последовательности состояний применим алгоритм Витерби с заменой b_{ix_l} на $b_i(x_l)$ при вычислении $\epsilon_l(i, X)$, а для обучения такой модели максимизацией правдоподобия на наборе последовательностей $\mathbf{X} =$

$\{\mathbf{X}_1, \ldots, \mathbf{X}_N\}$ применим алгоритм EM (Баума-Велша) в следующем виде. В формулах для вычисления $\alpha_l(i, X)$ и $\beta_l(i, X)$ на стр. 189 и далее нужно заменить b_{ix_l} на $b_i(x_l)$, а последующие вычисления $\gamma_l(i, X)$, $\xi_l(i, j, X)$ и т.п. оставить без изменения. Тогда вектор начального распределения Π и матрица перехода A пересчитываются по формулам (288) и (289), соответственно, а параметры распределений b_j — по формулам (ср. с формулами (136)–(138))

$$\begin{aligned} P^*_{jk} &= \frac{\sum_{i=1}^{N}\sum_{l=1}^{L_i}\Gamma_l(j,k,\mathbf{X}_i)}{\sum_{i=1}^{N}\sum_{l=1}^{L_i}\gamma_l(j,\mathbf{X}_i)} \\ \mu^*_{jk} &= \frac{\sum_{i=1}^{N}\sum_{l=1}^{L_i}\Gamma_l(j,k,\mathbf{X}_i)x_{il}}{\sum_{i=1}^{N}\sum_{l=1}^{L_i}\Gamma_l(j,k,\mathbf{X}_i)} \\ s^*_{jk} &= \frac{1}{d}\frac{\sum_{i=1}^{N}\sum_{l=1}^{L_i}\Gamma_l(j,k,\mathbf{X}_i)\|x_{il}-\mu^*_{jk}\|^2}{\sum_{i=1}^{N}\sum_{l=1}^{L_i}\Gamma_l(j,k,\mathbf{X}_i)}\ , \end{aligned}$$

где

$$\Gamma_l(j,k,X) = \gamma_l(j,X)\frac{P_{jk}p_{jk}(x_l)}{\sum_{k=1}^{m}P_{jk}p_{jk}(x_l)}\ .$$

Доказательство получается естественным объединением доказательства, приведенного для обучения RBF-сетей, (стр. 86) или гораздо более общего доказательства из раздела A.3.3 с опущенным доказательством предложения 38 и оставляется в качестве громоздкого технического **упражнения**. Алгоритм и доказательство легко переносятся со сферических гауссианов на общие (139).

Замечание. Получившаяся модель несколько отличается от HMM, использующей в качестве наблюдаемых ответы нейронной сети. Ее можно представить себе как результат объединения многих (по одной на скрытое состояние) двухслойных нейронных сетей и замены их независимых вторых слоев единой марковской моделью.

Непрерывное время. Для распознавания процессов реального времени (например, речи), можно было бы строить модель, содержащую вместо марковской цепи марковский (или не марковский) случайный процесс. На практике, наблюдаемый процесс (например, акустический сигнал) обычно доступен в виде последовательности замеров с дискретным шагом по времени. Поэтому строят модель на базе марковской цепи, но с контролируемым временем пребывания в каждом состоянии. Действительно, в чисто марковской цепи вероятность длительного пребывания в i-м состоянии управляется только одним диагональным элементом a_{ii} матрицы перехода. Вероятность оставаться в этом состоянии ровно k моментов времени равна $a_{ii}^k(1-a_{ii})$, т.е. экспоненциально убывает по k, что совершенно не соответствует акустическим реалиям. Взамен предполагается, что время пребывания цепи в i-м состоянии описывается распределением из какого-либо другого, более адекватного параметрического семейства, и параметры этих распределений тоже оцениваются при обучении.

Непрерывное пространство скрытых состояний. Во многих задачах, например, в прогнозировании извержений гейзера (стр. 182) пространство скрытых состояний, конечно же, непрерывно, а замена его дискретным — грубое приближение. Реконструкция состояния динамической системы по зашумленным (т.е. случайным) зависящим от состояния сигналам — весьма распространенная практическая задача в радиоэлектронике и вычислительной технике. Но вычислительно она разрешима только для самых простых динамических систем (линейных) и распределений эмиссии (гауссовых). Один из самых распространенных методов называется фильтр Калмана. Это отдельная большая

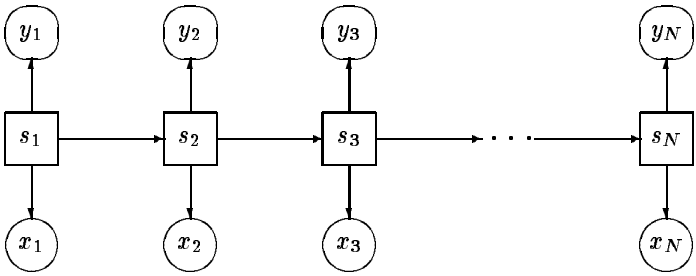

Рис. 14: Схематическое изображение обобщения скрытой марковской модели: и признаки x_i, и ответы y_i зависят от скрытых состояний s_i.

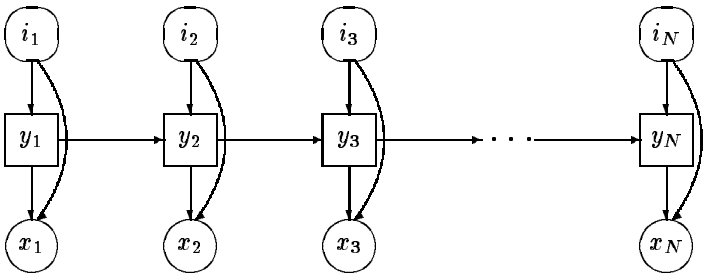

Рис. 15: Другое обобщение скрытой марковской модели — Input Output HMM: и "выходные" наблюдаемые признаки x_j, и скрытые состояния y_j зависят от "входных" наблюдаемых i_j.

наука. А для систем и распределений общего вида единственный практически приемлемый способ что-нибудь посчитать — это аппроксимировать пространство состояний конечным.

Частично наблюдаемые состояния. В некоторых задачах удобны модели, в которых не сами скрытые состояния, но некоторые функции от них наблюдаемы, присутствуют в обучающем наборе, и именно их-то и нужно предсказывать. Например, для распознавания печатного текста может оказаться более адекватной не стандартная HMM, а модель, изображенная на рис. 14. Ее скрытое состояние s_i — это, например, i-й символ вместе со спецификацией шрифта и кегля, которыми он напечатан, а содержательный ответ y_i — сам символ, зависящий от s_i однозначно.

Неоднородность. Упрощающее предположение об однородности скрытой марковской цепи ничем не плохо для моделирования и распознавания воспроизводимых последовательностей, вроде речи или письменного текста, и не вполне хорошо для моделирования уникальных последовательностей, например, численности саранчи или котировок биржевых акций. В этих случаях в модель нужно включить зависимость и скрытых состояний, и наблюдаемых, от тоже случайных, но наблюдаемых внешних параметров — урожая зерновых и политических новостей, соответственно.

Такая модель, изображенная на рис. 15, и методы ее обучения были предло-

жены в работе [9] под названием *IOHMM* (*Input Output HMM*). Эта же модель применима в другой ситуации: когда нужно по последовательности векторов признаков предсказывать соответствующие им ответы и пары (признаки, ответ) почти независимы, но все-таки какая-то зависимость распределения пары от предшествующих ответов возможна.

Модели конечных последовательностей. Стандартная скрытая марковская модель и ее вариации, рассмотренные выше, описывают порождение бесконечных последовательностей наблюдений. Вероятность $P\{X_{[1,L]}\}$ в формуле (269) и далее — это вероятность того, что первые L наблюдений такие, как в последовательности X, а последующие — любые. Но, за исключением задач прогнозирования (стр. 181), в большинстве случаев анализировать нужно только конечные последовательности.

Приспособить HMM к порождению конечных последовательностей можно разными способами. Можно добавить к пространствам скрытых и наблюдаемых состояний дополнительные "завершающие" состояния $y_* \in \mathcal{Y}$ и наблюдения $x_* \in \mathcal{X}$, такие что в состоянии y_* модель с вероятностью 1 порождает наблюдение x_* и остается в состоянии y_*, а в остальных состояниях вероятность наблюдения x_* равна 0. Такая HMM порождает последовательности наблюдений либо не содержащие признак конца x_* вообще, либо кончающиеся бесконечным хвостом x_*; последние можно отождествить с конечными последовательностями отрезанием этого хвоста. Если вероятность попадания модели в конечное состояние y_* равна 1, то с вероятностью 1 модель порождает конечную последовательность наблюдений. Тем самым, модель определяет распределение вероятностей на множестве всех конечных последоваетльностей и вероятность конкретной конечной последовательности $X_{[1,L]}$ равна $P\{(Xx_*)_{[1,L+1]}\}$.

Другой способ модификации HMM для порождения конечных последовательностей состоит в том, то пространства $\mathcal{X}$ и $\mathcal{Y}$ не изменяются, а условие стохастичности матрицы перехода A ослабляется до $\sum_k a_{jk} \leq 1$. На практике чаще применяется третий способ, гибридный, состоящий в том, что добавляется только одно "завершающее" скрытое состояние y_*, из которого нет ни переходов ($a_{*j} = 0$ при всех j), ни эмиссии ($b_{*k} = 0$ при всех k). В графе переходов такой модификации HMM в завершающем состоянии нет петли, присутствующей и раздражающей на рис. 13.

Упражнение. Выпишите вероятность порождения такой моделью последовательности наблюдений $X_{[1,L]}$.

Ненаблюдаемая эмиссия. В ранее рассмотренных моделях последовательности скрытых состояний и наблюдаемых были синхронизированы: каждое скрытое состояние[60] порождало ровно одно наблюдаемое. В некоторых задачах, в том числе при распознавании порождающей последовательности любой длины (стр. 181) естественно применять "асинхронные" модели, в которых скрытые состояния могут порождать целые последовательности наблюдаемых, в том числе и пустые. Зависимости между скрытыми состояниями и наблюдаемыми в таких моделях уже не описываются схемами, вроде изображенной на рис. 10. Порождение непустых последовательностей наблюдаемых можно выразить в терминах обычных "синхронных" моделей, заменив каждое скрытое состояния маленькой (или не очень маленькой) HMM. А вот непорождение наблюдаемых требует специального рассмотрения, поскольку для вычисления условной вероятности $P\{X_{[1,L]}|Y_{[1,M]}\}$ при $M > L$ простая формула (266) неприменима, а значит меняется и вычисление вероятности (269).

[60] Кроме "завершающего" состояния в предыдущем абзаце.

Чтобы формализовать вычисления, будем считать, что к матрице эмиссии B приписан нулевой столбец b_0, элемент b_{j0} которого — это вероятность отсутствия эмиссии в j-м скрытом состоянии. Условию стохастичности теперь удовлетворяет эта расширенная матрица "эмиссии и неэмиссии" $B_0 = b_0 \oplus B$. Для простоты вычислений ограничимся невырожденным случаем, когда все $b_{j0} < 1$, т.е. вероятность эмиссии из каждого состояния положительна. Обозначим через J диагональную матрицу $\mathrm{diag}(b_0)$. Тогда матрица "условной эмиссии" (если эмиссия была, то какая) $(I - J)^{-1}B$ — стохастическая, а матрица "переходов без эмиссии" JA — нет. Вероятность, начав со скрытого состояния j, за l шагов перейти в состояние k и не породить в l пройденных состояниях, кроме последнего, равного k, ни одного наблюдаемого, равна

$$h^l_{jk} = P\{y_{l+1} = k, X = \emptyset | y_1 = j\} = \sum_{y_2,\ldots,y_l} \prod_{i=1}^{l} b_{y_i 0} a_{y_i y_{i+1}} = \left((JA)^l\right)_{jk}$$

(l в правой части равенства — это не верхний индекс, а показатель степени). Положим также $h^0_{jk} = \delta^j_k$. Тогда сумма условных вероятностей, начав со скрытого состояния j, за какое-то число шагов перейти в состояние k и не породить перед этим попаданием в k ни одного наблюдаемого

$$h_{jk} = \sum_{l=0}^{\infty} h^l_{jk} = \sum_{l=0}^{\infty} \left((JA)^l\right)_{jk} = \left((I - JA)^{-1}\right)_{jk}$$

(обратимость матрицы $I - JA$ следует(!) из стохастичности матрицы A и условий $b_{j0} < 1$). Обозначим $H = (I - JA)^{-1}$.

Пусть $Y^* = (y^*_1, \ldots, y^*_L) = (y_{i_1}, \ldots, y_{i_L})$ — последовательность именно тех скрытых состояний какой-то последовательности $Y = Y_{[1,M]}$, которые порождают последовательности наблюдаемых $X = X_{[1,L]}$ (т.е. Y^* входит в Y как подпоследовательность), причем $i_L = M$, (концы подпоследовательности и последовательности совпадают). Вероятность $P\{X, Y^*, Y\}$ такой тройки последовательностей равна

$$\sum_{y'_1,\ldots,y'_L=1}^{n} \left(\pi_{y'_1} h^{i_1-1}_{y'_1 y_{i_1}} b_{y_{i_1} x_1} \prod_{l=1}^{L-1} \left(a_{y_{i_l} y'_{l+1}} h^{i_{l+1}-i_l+1}_{y'_{l+1} y_{i_{l+1}}} b_{y_{i_{l+1}} x_{l+1}} \right) \right) .$$

Тогда сумма $P\{X, Y^*\} = \sum_Y P\{X, Y^*, Y\}$ совместных вероятностей по всем последовательностям Y скрытых состояний, содержащим подпоследовательность с тем же концом, равную Y^* и породившую X, равна

$$\begin{aligned} P\{X, Y^*\} &= \sum_{y'_1,\ldots,y'_L=1}^{n} \left(\pi_{y'_1} h_{y'_1 y^*_1} b_{y^*_1 x_1} \prod_{l=1}^{L-1} \left(a_{y^*_l y'_{l+1}} h_{y'_{l+1} y^*_{l+1}} b_{y^*_{l+1} x_{l+1}} \right) \right) \\ &= (\pi H)_{y^*_1} b_{y^*_1 x_1} \prod_{l=1}^{L-1} \left((AH)_{y^*_l y^*_{l+1}} b_{y^*_{l+1} x_{l+1}} \right) , \end{aligned}$$

а значит, полная вероятность последовательности X равна

$$\begin{aligned} P\{X\} &= \sum_{Y^*} P\{X, Y^*\} = \sum_{y^*_1,\ldots,y^*_L} \left((\pi H)_{y^*_1} \prod_{l=1}^{L-1} (AH)_{y^*_l y^*_{l+1}} \prod_{l=1}^{L} b_{y^*_l x_l} \right) \\ &= \sum_{y^*_1,\ldots,y^*_L} \left((\pi H(I-J))_{y^*_1} \prod_{l=1}^{L-1} (AH(I-J))_{y^*_l y^*_{l+1}} \prod_{l=1}^{L} \left((I-J)^{-1} B \right)_{y^*_l x_l} \right) \end{aligned}$$

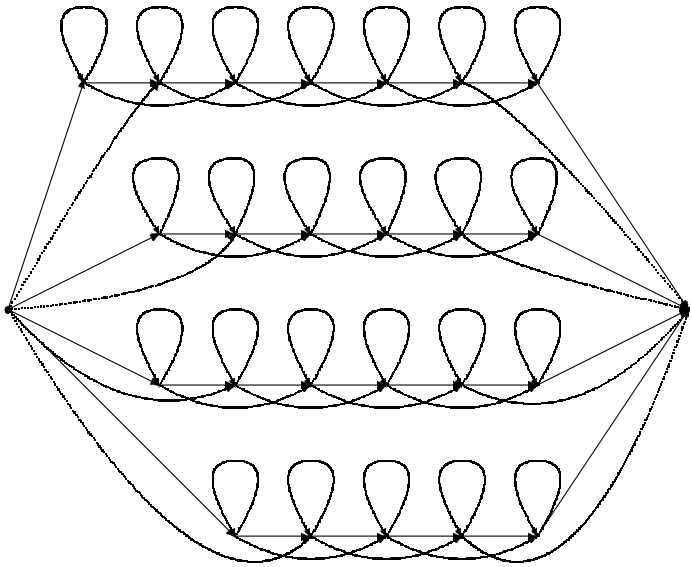

Рис. 16: Пример графа допустимых переходов для марковской модели рукописного символа с несколькими вариантами написания. Эмиссия может быть привязана к переходам, а может быть привязана к состояниям, кроме начального и завершающего.

(ср. с формулой (269)). Вычислять вероятности по этой формуле безнадежно долго, но зато понятно, что вычисление величин $\alpha_l(i, X)$ (формула (270)), $\beta_l(i, X)$ (формула (272)) и последующие вычисления при применении модели изменяются только в трех местах: всюду нужно писать

- $\pi^*_j = (\pi H(I-J))_j = \big(\pi(I-JA)^{-1}(I-J)\big)_j$ вместо π_j,
- $a^*_{jk} = (AH(I-J))_{jk} = \big(A(I-JA)^{-1}(I-J)\big)_{jk}$ вместо a_{jk} и
- $b^*_{jk} = \big((I-J)^{-1}B\big)_{jk}$ вместо b_{jk}.

Обучение марковской модели с ненаблюдаемой эмиссией требует пояснений. Вычислением в пол-строки **можно показать**, что матрица $H(I-J)$ стохастическая, а значит (π^*_j) — вероятностное распределение, а матрица (a^*_{jk}), как и (b^*_{jk}) — тоже стохастическая, и их можно обучить алгоритмом Баума-Велша или его обобщениями. Этого достаточно для прогнозирования и классификации, но не для восстановления последовательности скрытых состояний, поскольку настоящая матрица переходов A остается неизвестной. Полностью обучить модель можно, например, градиентными методами, или максимизацией совместного или оптимального правдоподобия.

Часто применяемыми "техническими" обобщениями HMM являются введение специальных состояний без эмиссии, в частности, начального состояния с начальной вероятностью 1, а также уже упомянутых завершающих состояний из которых нет выхода. Кроме того, применяется привязка эмиссии к переходам между скрытыми состояниями, а не к самим состояниям (при этом из одного состояния в другое может быть несколько разных переходов), что эквивалентно введению в модель дополнительных скрытых состояний "на серединах ребер" и превращению прежних состояний в состояния без эмиссии. Обучение

и применение таких моделей почти не отличается от обучения и применения стандартных HMM. Такие обобщенные модели намного проще объединять друг с другом. Например, на рис. 16 показана модель символа с несколькими вариантами написания (или слова с несколькими вариантами произношения). Если есть какая-либо марковская модель построения слова из символов (или речи и слов), состояния которой соответствуют символам (словам), а переходы — последовательному выписыванию (выговариванию) то, положив, что ее переходы не производят эмиссии, и заменив состояния на модели соответствующих символов (слов), сразу получим объединенную модель. Объединять стандартные HMM было бы заметно менее удобно. Кроме того за счет введения состояний без эмиссии или привязки эмиссии к переходам иногда удается заметно сократить размерность модели.

Литература по построению и применению скрытых марковских моделей весьма обширна и, как правило, перегружена техническими подробностями (увы, необходимыми). Наибольшей популярностью эти модели пользуются в задачах, связанных с распознаванием речи. Для ознакомления можно порекомендовать две вводных статьи [94] и [10], содержащие также многочисленные ссылки.

C.4 Анализ последовательностей в целом

Наряду с построением специфических вероятностных моделей последовательностей, для их статистического анализа можно применять и универсальные методы, если на бесконечномерном пространстве последовательностей определить те же структуры, что определялись на конечномерных пространствах признаков, например, базисные функции (раздел 2.3.1), метрику или ядра (раздел 2.3.2) . Этот подход хорошо применим в задачах, в которых не нужно ничего понять про устройство последовательности, а нужно только научиться характеризовать последовательности в целом, в частности, классифицировать последовательности на “хорошие” и “плохие” (например, оценивать, является ли электронное письмо, т.е. последовательность символов, спамом). Он применим также и для кластеризации последовательностей.

Универсальных теорий того, как распознавать последовательности в целом, нет. В каждой задаче успешно применяются специфические для нее методы, а не только следующие из общей теории статистического обучения. Например, если разработчику программы, распознающей спам, попадется на глаза последовательность символов “V I @ G R a” среди обучающего материала, он немедленно зафиксирует в качестве одного из признаков спама наличие в тексте и этой последовательности, и похожих последовательностей, отличающихся регистрами букв, заменами ‘A’ на ‘@’, ‘I’ на ‘1’ и т.п., а также количеством пробелов между ними, причем ему даже не понадобится смотреть в ответ, действительно ли это письмо является спамом.

Далее в этом разделе приводится несколько несложных примеров построения базисных функций, метрик и ядер на последовательностях.

C.4.1 Базисные функции на последовательностях

Самая простая функция от последовательности $X_{[1,L]}$ элементов пространства признаков $\mathcal{X}$ — это ее длина L. Сама по себе длина для анализа последовательностей довольно бесполезна, но часто используется для нормировки других функций.

Чаще всего базисные функции на последовательностях строят, индуцируя их из функций на пространстве коротких последовательностей фиксированных длин (будем называть их пробными последовательностями или *зондами*.).

Зонды длины k — это то же самое, что векторы в пространстве $\mathcal{X}^k$, и последовательности $X_{[1,L]}$ при $L \geq k$ соответствует $L-k+1$ векторов $X_{[l,l+k]} \in \mathcal{X}^k$, $1 \leq l \leq L-k+1$. Для любой функции f на $\mathcal{X}^k$ можно взять сумму, максимум, среднее, произведение и т.п. от ее значений на этих $L-k+1$ векторах. В частности, для строк, т.е. последовательностей символов некоторого конечного алфавита $\mathcal{X}$, так можно получить количество вхождений в строку каждого символа или его частоту, частоты пар смежных символов, индикатор наличия подстроки "AUG" или число ее вхождений и т.п.

Набор всех базисных функций определяет отображение пространства последовательностей в *спрямляющее пространство* $\mathbb{R}^n$. При не слишком маленькой длине зондов k их количество, а значит и размерность спрямляющего пространства n бывает слишком велика, чтобы обучать распознаватели прямо в нем. Зато при этом последовательности представляются очень разреженными векторами, поскольку бóльшая часть зондов в них не встречается ни разу. В таких случаях обычно на спрямляющем пространстве определяют метрики, ядра и т.п., вычисление которых не зависит от количества не встретившихся зондов.

Такие базисные функции учитывают локальную структуру последовательности, но игнорируют глобальную. Во многих задачах это разумно, например. при статистическом анализе художественных текстов.

Для сравнения опишем способ построения базисных функций на последовательностях, не игнорирующий их глобальную структуру, по крайней мере явно. Предположим, что для последовательностей есть (т.е. придумано) параметрическое семейство порождающих моделей с параметром $\mathbf{W}$, пробегающим некоторую область $\mathcal{W} \subset \mathbb{R}^n$ и в нем выбрана (желательно, обучена) модель $\mathbf{W}^*$. Например, семейством моделей может быть пространство скрытых марковских моделей с фиксированным графом переходов. Тогда для любой последовательности X определен вектор $U_X \in \mathbb{R}^n$, называемый *Fisher score* (*оценка Фишера*):

$$U_X = U_X(\mathbf{W}^*, \mathcal{W}) = \left.\frac{\partial \ln P\{X|\mathbf{W}\}}{\partial \mathbf{W}}\right|_{\mathbf{W}=\mathbf{W}^*}.$$

Отображение $X \mapsto U_X$ — это вектор-функция на пространстве последовательностей, т.е. n функций, предлагаемых в качестве базисных.

Традиционное обозначение U_X вместо подробного $U_X(\mathbf{W}^*, \mathcal{W})$ скрывает то, что получившийся набор базисных функций зависит от модели $\mathbf{W}^*$ и даже от пространства $\mathcal{W}$ и способа его параметризации. Польза от применения этих базисных функций для анализа последовательностей будет в том случае, когда модель $\mathbf{W}^*$ — лучшая или близкая к лучшей в пространстве $\mathcal{W}$, например, получается в результате максимизации правдоподобия $P\{\mathbf{X}|\mathbf{W}\}$ для достаточно представительного обучающего набора $\mathbf{X}$.

В статье [61] предлагается следующее применение отображения $X \mapsto U_X$ для классификации последовательностей. Вместо того, чтобы для каждого класса обучать отдельную HMM (стр. 193), максимизацией правдоподобия обучают одну HMM, не обращая внимания на метки классов y_i. С помощью этой обученной модели вычисляют оценки Фишера $U_{\mathbf{X}_i}$ для всех обучающих последовательностей $\mathbf{X}_i$, а затем на парах $(U_{\mathbf{X}_i}, y_i)$ обучают классификатор в пространстве $\mathbb{R}^n$, например, линейную логистическую регрессию или SVM. В статье [61] приводятся экспериментальные результаты, что получается лучший классификатор, чем при дискриминантном обучении нескольких HMM. Приводится также теоретическое обоснование того, что так и должно быть, впрочем, не вполне строгое и не учитывающее того, что модели для разных классов можно строить с разными пространствами состояний и графами переходов.

C.4.2 Метрики на последовательностях

Любой набор базисных функций осуществляет отображение пространства последовательностей в евклидово пространство $\mathbb{R}^n$ и, тем самым, позволяет любую метрику в нем (не обязательно евклидову) перенести на последовательности. Так получаются разнообразные метрики, основанные на различии частотных характеристик последовательностей. Для сравнения опишем существенно другую метрику, называемую *расстоянием Левенштейна* или *редакторским расстоянием* (*edit distance*).

В исходной работе В.И.Левенштейна [73] расстояние определяется для последовательностей нулей и единиц, но определение немедленно переносится на строки в любом алфавите. Любую строку можно превратить в любую другую с помощью последовательного применения следующих трех операций:

- замена какого-либо символа строки на любой другой символ;
- удаление какого-либо символа строки;
- вставка какого-нибудь символа в любое место строки.

Действительно, достаточно сначала удалить все символы первой строки, а потом последовательно вставить все символы второй строки, а замену символов не использовать вообще. Но это не самый экономный способ. Расстояние Левенштейна между строками определяется как минимальное число операций, достаточных для превращения одной строки в другую.

Упражнение. Проверьте, что так определенное "расстояние Левенштейна" действительно удовлетворяет аксиомам расстояния.

Определение легко обобщается на последовательности элементов любого метрического пространства. Пусть на пространстве элементов с расстоянием $d(\cdot,\cdot)$ определена еще положительная функция g. Введем следующие тарифы:

- замена элемента x на элемент x' стоит $d(x,x')$,
- удаление элемента x стоит $g(x)$,
- вставка элемента x тоже стоит $g(x)$

и определим расстояние**(!)** между последовательностями как минимальную стоимость последовательности операций, превращающих одну последовательность в другую.

Для обобщенного расстояния Левенштейна D очевидным образом выполняются соотношения

$$D(\emptyset,\emptyset) = 0 \tag{291}$$

$$D(Xu,\emptyset) = D(\emptyset,Xu) = D(X,\emptyset) + g(u) \tag{292}$$

$$D(Xu,Yv) = \min(D(X,Y) + d(u,v), D(Xu,Y) + g(v), D(X,Yv) + g(u)) \tag{293}$$

где X и Y — это последовательности, u и v — элементы последовательностей, $\emptyset$ — пустая последовательность, а последовательное выписывание, вроде Xu, — это последовательное выписывание. Значит расстояние $D(X,Y)$ легко вычисляется методом динамического программирования.

Упражнение. Постройте алгоритм, вычисляющий $D(X_{[1,m]}, Y_{[1,n]})$ за время порядка mn на памяти порядка $m+n$.

Замечание. Все-таки вычисление расстояния Левенштейна между очень длинными последовательностями — очень очень медленное. Если вспомнить, что метрические методы распознавания, например, метод ближайшего соседа, сами по себе медленны, получается, что для статистического анализа очень

длинных последовательностей расстояние Левенштейна в чистом виде неприменимо. Но для любой константы $d_0 > 0$ расстояние**(!)** между строками $\min(D(X,Y)$ не уточняющее, насколько именно непохожи совсем непохожие строки, вычисляется за время порядка $d_0(m+n)$ **(!)**.

С.4.3 Ядра на последовательностях

Любой набор базисных функций осуществляет отображение пространства последовательностей в спрямляющее пространство $\mathbb{R}^n$ и, тем самым, позволяет не только любую метрику, но и любое ядро на нем (в частности, скалярное произведение) перенести на последовательности. Например, оценка Фишера $X \mapsto U_X$ индуцирует из скалярного произведения *ядро Фишера*[61] $K(X,X') = (U_X, U_{X'})$. Более того, из любого ядра k на пространстве $\mathbb{R}^n$ индуцируется ядро на последовательностях $K(X,X') = k(U_X, U_{X'})$, тоже иногда называемое ядром Фишера. Ядра Фишера успешно применяются при классификации последовательностей (см. [61]) и не только последовательностей: ведь оценки Фишера и ядра Фишера определены для любых объектов, для которых есть параметрическое семейство вероятностных моделей.

Кроме ядра Фишера есть и совершенно другие способы построения ядер на последовательностях (и не только на последовательностях), получающиеся из вероятностных моделей.

Предложение 41

Для любого распределения вероятностей p на пространстве $\mathcal{X}$ совместное распределение на пространстве $\mathcal{X}\times\mathcal{X}$ пар независимых элементов $\mathcal{X}$ $p(x,x') = p(x)p(x')$ является ядром Мерсера.

Для любого распределения вероятностей p на пространстве $\mathcal{X}\times\mathcal{Y}$ совместное распределение на пространстве $\mathcal{X}\times\mathcal{X}$ пар условно-независимых при любом $y\in\mathcal{Y}$ элементов $\mathcal{X}$

$$p(x,x') = \mathbf{E}_y p(x,x'|y) = \mathbf{E}_y p(x|y)p(x'|y)$$

является ядром Мерсера.

Доказательство. Эти утверждения — всего лишь переформулировка некоторых пунктов предложения 13 и следствия 1, к тому же не зависящая от того, непрерывны или дискретны ли рассматриваемые распределения. □

Удовлетворяющие предложению 41 пары условно-независимых или независимых последовательностей с одинаковым распределением можно порождать с помощью моделей, похожих на HMM. Рассмотрим три последовательности случайных величин x_i, y_i и s_i, зависимости между которыми изображены на рис. 14, последовательность s_i по-прежнему скрытая, а последовательности x_i и y_i — наблюдаемые, но смысл у них теперь другой. Теперь x_i и y_i — это равноправные последовательности в пространстве признаков $\mathcal{X}$, порождаемые скрытыми состояниями одновременно и независимо с одним и тем же распределением, т.е. $p(x_i,y_i|s_i) = p_i(x_i|s_i)p_i(y_i|s_i)$. Для однородных HMM с дискретными пространствами скрытых и наблюдаемых состояний это означает, что

[61] Ядром Фишера чаще называется ядро $I^{-1}(U_X, U_{X'})$ (в матричной записи, если считать градиент вектором-столбцом — ${U_{X'}}^\top I^{-1} U_X$), где I — *информационная матрица Фишера*. Из традиционных обозначений, как и в случае оценки Фишера, заботливо удалены упоминания о модели $\mathbf{W}$ и пространстве моделей $\mathcal{W}$, для которых эта матрица определена. Напоминание: информационная матрица Фишера $I = \mathbf{E}_X(U_X {U_X}^\top)$, где ожидание считается по распределению с параметром $\mathbf{W}$. Так определенное ядро Фишера инвариантно относительно диффеоморфизмов пространства $\mathcal{W}$, зато его очень трудно вычислять. В отличие от оценки Фишера и информационной матрицы Фишера, ядра Фишера Рональд Фишер не придумывал и не использовал, но они названы в его честь.

x_i и y_i порождаются с одной и той же матрицей эмиссии B. Для удобства моделирования конечных последовательностей как и на стр. С.3.2 будем считать, что в модели есть завершающее скрытое состояние s_*, порождающее с вероятностью 1 пару завершающих наблюдаемых состояний. Тогда для любой последовательности скрытых состояний s длины N

$$P\{X,Y|s\} = \prod_{l=1}^{N} (b_{s_l x_l} b_{s_l y_l}) = \left(\prod_{l=1}^{N} b_{s_l x_l}\right)\left(\prod_{l=1}^{N} b_{s_l y_l}\right) = P\{X|s\}P\{Y|s\} \ ,$$

значит по предложению 41 совместная вероятность порождения двух последовательностей наблюдаемых, которую можно неформально интерпретировать как "вероятность замены элементов одной последовательности на соответствующие элементы другой",

$$P_{\text{repl}}\{X_{[1,L]}, Y_{[1,M]}\} = \sum_{S_{[1,N]}} P\{X_{[1,L]}, Y_{[1,M]}|S_{[1,N]}\}P\{S_{[1,N]}\} \tag{294}$$

$$= \begin{cases} 0 & \text{при } L \neq M \\ \displaystyle\sum_{s_1,\ldots,s_L} \left(\pi_{s_1} \prod_{l=1}^{L-1} a_{s_l s_{l+1}} \prod_{l=1}^{L} (b_{s_l x_l} b_{s_l y_l})\, a_{s_L s_*} \right) & \text{при } L = M \end{cases}$$

(ср. с формулой (269)) является ядром Мерсера и на пространстве всех последовательностей.

Если большинство скрытых состояний порождают одно наблюдаемое состояние (каждое — свое) с намного большей вероятностью, чем любое другое, то из ядра (294) после нормировки[62] (раздел 4.1.4) получается разумное расстояниемежду последовательностями: если последовательности одинаковой длины почти везде совпадают, то значение ядра на них довольно велико (но меньше 1), а расстояние между ними довольно мало. К сожалению, отнормированное ядро, как и само ядро (294), никак не замечает того, что одна последовательность длины 1000000 получена из другой последовательности длины 999999 приписыванием всего лишь одного элемента, т.е. почти совпадает с ней: раз длина разная, значит значение ядра на этой паре последовательностей равно нулю.

Совсем другое вероятностное ядро можно получить, если соединить последовательно два экземпляра одной и той же HMM с достижимым с вероятностью 1 завершающим состоянием (стр. 204), и считать, что первый экземпляр в получившемся тандеме порождает последовательность X, а второй — Y. Тогда, если через $P\{\cdot\}$ обозначить вероятность порождения последовательности одинарной HMM, а через $P_{\text{ins}}\{\cdot,\cdot\}$ — вероятность порождения пары последовательностей тандемом HMM, которую можно неформально интерпретировать как "вероятность удаления одной последовательности и вставки вместо нее другой", то $P_{\text{ins}}\{X,Y\} = P\{X\}P\{Y\}$, т.е. последовательности X и Y независимы и по предложению 41 P_{ins} тоже является ядром Мерсера на последовательностях.

Например, если одинарная HMM состоит всего лишь из двух состояний — основного и завершающего, начальная вероятность основного состояния равна α, из основного состояния каждое возможное наблюдаемое значение порождается с вероятностью $\frac{1}{m}$, а переход в завершающее состояние происходит с вероятностью $1-\alpha$, то

$$P^1\{X_{[1,L]}\} = \left(\frac{\alpha}{m}\right)^L (1-\alpha)$$

[62] Без нормировки все длинные последовательности пренебрежимо маловероятны и значения ядра (294) на них исчезающе малы.

и

$$P_{\text{ins}}\{X_{[1,L]}, Y_{[1,M]}\} = \left(\frac{\alpha}{m}\right)^{L+M} (1-\alpha)^2 \ .$$

Это ядро совершенно безразлично к несовпадению длин последовательностей, поскольку зависит только от их суммы.

Рассмотрим теперь порождающую модель пар последовательностей, получаемую вставкой в каждое ребро графа переходов раздвоенной HMM, вычисляющей ядро (294), маленького тандема HMM (можно даже вставлять разные тандемы в разные ребра, но не будем загромождать обозначения). Обозначим множество ее скрытых состояний, унаследованных от раздвоенной HMM и порождающих пары наблюдаемых, через S_2, а множество скрытых состояний тандемов, порождающих только одно наблюдаемое, через S_1. Любая пара последовательностей X и Y может порождаться разными последовательностями $s = (s_1, \ldots, s_N)$ состояний из S_2, как-то перемежаемых состояниями из S_1. Порождаемые этой HMM последовательности наблюдаемых условно независимы: $P\{X, Y|s\} = P\{X|s\}P\{Y|s\}$, а значит по предложению 41 совместное распределение $P\{X, Y\}$ является ядром.

Действительно, обозначим через $\Omega_n(X)$ множество представлений последовательности X в виде конкатенации

$$X_{[1,L]} = x_{\omega_1} X_{[\omega_1+1,\omega_2-1]} \ldots x_{\omega_n} X_{[\omega_n+1,\omega_{n+1}-1]} \tag{295}$$

перемежающихся n элементов и n последовательностей любых длин, возможно, пустых (здесь $\omega_1 = 1$ $\omega_{n+1} = L+1$). Тогда

$$\begin{aligned}
P\{X, Y|s\} &= \sum_{\omega\in\Omega_N(X)} \sum_{v\in\Omega_N(Y)} \prod_{l=1}^{N} \left(b_{s_l x_{\omega_l}} b_{s_l y_{v_l}} P\{X_{[\omega_l+1,\omega_{l+1}-1]}\} P\{Y_{[v_l+1,v_{l+1}-1]}\} \right) \\
&= \left(\sum_{\omega\in\Omega_N(X)} \prod_{l=1}^{N} b_{s_l x_{\omega_l}} P\{X_{[\omega_l+1,\omega_{l+1}-1]}\} \right) \\
&\qquad \left(\sum_{v\in\Omega_N(Y)} \prod_{l=1}^{N} b_{s_l y_{v_l}} P\{Y_{[v_l+1,v_{l+1}-1]}\} \right) \\
&= P\{X|s\}P\{Y|s\} \ .
\end{aligned}$$

То, что таким образом можно получить ядро Мерсера на последовательностях, было показано в статье [119]. Значение этого ядра уже не очень сильно меняется не только при замене, но и при вставке или удалении элемента одного из аргументов, так что, отнормировав его, можно получить расстояние на пространстве последовательностей, напоминающее расстояние Левенштейна.

В заключение опишем способ построения ядер на последовательностях по любым ядрам на элементах, являющийся точным аналогом построения обобщенного расстояния Левенштейна по расстоянию на элементах и не связанный ни с какой вероятностной моделью. В отличие от расстояния Левенштейна, происхождение которого известно, а свойства проверяются очень просто, ни первоисточника этой конструкции ядер, ни ее канонической формулировки автору найти не удалось. Ниже приводятся два разных варианта. Найденное в неопубликованной работе А. Серегина [104] доказательство того, что во втором варианте действительно получаются ядра Мерсера, очень красиво, но довольно сложно. Вместо него приводится доказательство, фактически повторяющее вышеприведенное, использовавшее условную независимость последовательностей, доказательство для вероятностных ядер.

Пусть на пространстве $\mathcal{X}$ определено любое ядро Мерсера k и любая функция g. Определим на пространстве $\mathcal{X}^*$ конечных последовательностей в $\mathcal{X}$ функции "замены"

$$K_{\text{repl}}(X_{[1,L]}, Y_{[1,M]}) = \begin{cases} 0 & \text{при } L \neq M \\ \prod_{l=1}^{L} k(x_l, y_l) & \text{при } L = M \end{cases} \tag{296}$$

и два варианта функции "удаления и вставки"

$$K^1_{\text{ins}}(X_{[1,L]}, Y_{[1,M]}) = \left(\prod_{l=1}^{L} g(x_l)\right)\left(\prod_{l=1}^{M} g(y_l)\right) \tag{297}$$

$$K^2_{\text{ins}}(X_{[1,L]}, Y_{[1,M]}) = \binom{L+M}{M} K^1_{\text{ins}}(X_{[1,L]}, Y_{[1,M]}) \ . \tag{298}$$

Как обычно, из предложений 13 и 14 следует, что K_{repl} и K^1_{ins} являются ядрами Мерсера. Чтобы убедиться, что ядром Мерсера является и K^2_{ins}, достаточно проверить положительную определенность матрицы биномиальных коэффициентов $\binom{m+n}{n} = \frac{(m+n)!}{m!n!}$, симметричность которой очевидна.

Лемма 7 *Матрица* $\left(\binom{m+n}{n}\right)_{m,n=0}^{\infty}$ *положительно определена.*

Доказательство. Положительную определенность матрицы будем доказывать по критерию Сильвестра: достаточно проверить положительность всех левых верхних угловых миноров. Докажем, что каждый такой минор (размера q) равен 1. Обозначим через A^q_l, $1 < l \leq q$ сохраняющую определитель операцию вычитания $(l-1)$-го столбца матрицы из l-го. Поскольку $\binom{m+l}{l} - \binom{m+(l-1)}{l-1} = \binom{(m-1)+l}{l}$, операция A^q_l состоит в сдвиге l-го столбца на 1 вниз и дополнении нулем сверху. Последовательность операций $A^q_l \circ (A^q_{l-1} \circ A^q_l) \circ \ldots \circ (A^q_2 \circ \ldots \circ A^q_l)$ (естественно, выполняемых в порядке справа налево) переводит "прямоугольный треугольник Паскаля" $\binom{m+n}{n}$

$$\begin{array}{ccccc} 1 & 1 & 1 & 1 & \cdots \\ 1 & 2 & 3 & 4 & \cdots \\ 1 & 3 & 6 & 10 & \cdots \\ 1 & 4 & 10 & 20 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{array}$$

в более традиционный "остроугольный" $\binom{m}{n}$

$$\begin{array}{ccccc} 1 & 0 & 0 & 0 & \cdots \\ 1 & 1 & 0 & 0 & \cdots \\ 1 & 2 & 1 & 0 & \cdots \\ 1 & 3 & 3 & 1 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{array}$$

— нижнюю треугольную матрицу с единицами на диагонали и определителями всех левых верхних угловых миноров, равными единице. Доказательство положительной определенности закончено[63]. □

[63] Лемму 7 можно доказать иначе, предъявив спрямляющее отображение, реализующее ядро $K(m,n) = \binom{m+n}{n}$ на множестве $\mathbb{Z}_+$ неотрицательных целых чисел. Годится отображение $\phi : \mathbb{Z}_+ \to L^2(\mathbb{R}_+)$, переводящее число m в функцию $\phi_m(x) = \dfrac{x^m}{m!} e^{-\frac{x}{2}}$. Это следует из общеизвестного и легко проверяемого тождества $\displaystyle\int_0^\infty x^n e^{-x} dx = n!$.

Ядра замены и вставки — это еще не то ядро, которое хочется построить из ядра k и функции g. Сейчас с их помощью будут построены ядра на других пространствах, из которых потом будут индуцированы желаемые ядра на последовательностях.

Ядра замены и вставки на паре пустых последовательностей принимают значение 1, равное произведению пустого множества слагаемых. Любое ядро K на пространстве $\mathcal{X}^*$ конечных последовательностей в $\mathcal{X}$, такое что $K(\emptyset,\emptyset)=1$, можно продолжить на пространство $(\mathcal{X}^*)^*$ конечных последовательностей конечных последовательностей (здесь повторение — не опечатка!) следующим образом. Дополним каждую последовательность последовательностей $X=(X_1,\ldots,$ до бесконечной последовательности $X^\infty=(X_1,\ldots,X_L,\emptyset,\emptyset,\ldots)$ и определим ядро

$$K^*(X,Y)=\prod_{l=1}^{\infty}K(X_l^\infty,Y_l^\infty)\ . \tag{299}$$

По предложению 14 K^* является ядром Мерсера.

Назовем *размеченной последовательностью* в пространстве $\mathcal{X}$ конечную последовательность $X_{[1,L]}$ элементов $\mathcal{X}$ с выделенной в ней подпоследовательностью, возможно, пустой, т.е. пару $(X_{[1,L]},\omega)$, где $\omega\subset\{1,\ldots,L\}$. Определим на пространстве размеченных последовательностей два отображения χ в пространство $\mathcal{X}^*$ обычных последовательностей и ψ в пространство $(\mathcal{X}^*)^*$ последовательностей последовательностей, сопоставляющие размеченной последовательности ее выделенную подпоследовательность и последовательность последовательностей ее невыделенных элементов перед первым выделенным, между первым и вторым выделенными, ..., после последнего выделенного, соответственно. Тогда по предложению 14

$$K^i_{\text{mark}}((X,\omega),(Y,\upsilon))=K_{\text{repl}}(\chi(X,\omega),\chi(Y,\upsilon))\left(K^i_{\text{ins}}\right)^*(\psi(X,\omega),\psi(Y,\upsilon))\quad\text{при } i=1,2 \tag{300}$$

является ядром Мерсера на пространстве размеченных последовательностей.

Каждую последовательность длины L можно разметить 2^L способами — по одному на подмножество множества $\{1,\ldots,L\}$. Обозначим множество этих подмножеств через Ω_L. Таким образом определено отображение пространства последовательностей $\mathcal{X}^*$ в пространство конечных подмножеств пространства размеченных последовательностей. Значит по всеспасительному предложению 14

$$K^i(X_{[1,L]},Y_{[1,M]})=\sum_{\omega\in\Omega_L}\sum_{\upsilon\in\Omega_M}K^i_{\text{mark}}((X,\omega),(Y,\upsilon))\text{ при } i=1,2 \tag{301}$$

является ядром Мерсера. Это и есть желанные ядра на последовательностях.

Напоминаем, что пространство $\mathcal{X}$ и ядро k и функция g нем, с которых начиналось построение — абсолютно произвольные. Неформально, ядро k имеет смысл "вероятности" замены элемента последовательности на другой, а функция g — "вероятности" вставки или удаления элемента, но, в отличие от настоящих вероятностных ядер, эти "вероятности" могут быть и отрицательными, и сколь угодно большими.

Ядра K^1 и K^2 удовлетворяют следующими рекуррентными соотношениями:

$$K^i(\emptyset,\emptyset)=1\text{ при } i=1,2 \tag{302}$$

$$K^i(Xu,\emptyset)=K^i(\emptyset,Xu)=K^i(X,\emptyset)g(u)\text{ при } i=1,2 \tag{303}$$

$$K^1(Xu,Yv)=K^1(X,Y)(k(u,v)-g(u)g(v))+K^1(Xu,Y)g(u)+K^1(X,Yv)g(u) \tag{304}$$

$$K^2(Xu,Yv)=K^2(X,Y)k(u,v)+K^2(Xu,Y)g(v)+K^2(X,Yv)g(u) \tag{305}$$

(ср. с соотношениями (291)–(293)). Это проверяется непосредственной подстановкой формул (296) и (297) или (298) в (299), (300) и (301). Из соотношений (302)–(305) сразу следует способ вычисления ядер $K^1(X,Y)$ и $K^2(X,Y)$, аналогичный вычислению расстояния Левенштейна, за время, пропорциональное произведению длин последовательностей.

Способов построения ядер на последовательностях (а также деревьях, графах и т.п.) весьма много. Некоторые из них сформулированы в весьма общих абстрактных терминах в статье [51], ссылки на более новые имеются в историко-литературном обзоре [105]. Около половины публикаций по применению ядер за 1997–2007 годы, упомянутых в этом обзоре, посвящены построению ядер на строках. Такая популярность ядер на строках вызвана задачами биоинформатики, в которых нужно анализировать молекулы белков, РНК, ДНК и т.п., имеющие структуру строки в алфавите из двадцати (белки) или четырех (РНК, ДНК) букв. Поскольку эти последовательности очень длинные (особенно ДНК — порядка 10^8 пар нуклеотидов), особое внимание уделяется быстрым алгоритмам вычисления ядер или расстояний.

D Анализ изображений; марковские и условные случайные поля

Опять скажу: никто не обнимет необъятного!
К.П.Прутков

Распознавание изображений является одним из самых популярных видов деятельности, использующих статистическое обучение и стимулирующих дальнейшее развитие его методов. В частности, непосредственно из него выросла теория нейронных сетей. Нейронные сети и другие общие методы распознавания, рассмотренные в предыдущих разделах, обычно рассматривали изображение — либо растровое изображение, пересчитанное на растр фиксированного размера, либо какие-то другие вычисленные характеристики изображения, вроде коэффициентов многочленов Фурье — как вектор признаков, по которому нужно научиться вычислять ответ — классификацию или регрессию. А в этом разделе будут рассматриваться методы восстановления изображений по изображениям же, аналогично тому, как в разделе C рассматривалось восстановление последовательностей по последовательностям. То есть, пространство ответов, как и пространство признаков, является очень многомерным вектором, к тому же не фиксированной размерности и обладающим дополнительной двумерной структурой, чаще всего, структурой прямоугольной матрицы. Задачи с еще более сложными и существенно различными пространствами признаков и ответов, например, определение трехмерной структуры опухоли по рентгенограммам в нескольких проекциях, здесь рассматриваться не будут.

D.1 Модельные задачи статистического анализа изображений

Будем использовать обозначения, аналогичные обозначениям раздела C. Через X будем обозначать наблюдаемое изображение, через x_j — его j-й элемент (который в свою очередь может быть вектором, но уже небольшой фиксированной размерности), а через Y и y_j — изображение-ответ и его элемент, соответственно. Двумерная структура изображений в самих обозначениях отсутствует и описывается отдельно. Через $R(Y)$ будем обозначать множество точек изображения Y, а через Y_A — ограничение изображения Y на подмножество $A \subset R(Y)$ (в частности, $Y_{\{j\}} = y_j$).

Далее будут рассматриваться методы статистического анализа изображений, применимые для следующих простейших задач, являющихся составными частями многих реальных задач.

Очистка изображения от шума. Имеется обучающий набор зашумленных растровых изображений $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_N)$ и набор последовательностей соответствующих им исходных изображений $\mathbf{Y} = (\mathbf{Y}_1, \ldots, \mathbf{Y}_N)$. Изображения могут быть черно-белыми, серыми, цветными — любыми, но однотипными. Для каждого i множества точек $R(\mathbf{Y}_i)$ и $R(\mathbf{X}_i)$ совпадают, хотя для разных i они могут быть разными. Нужно научиться по любому изображению X строить по возможности очищенное от шумов изображение Y.

Поиск подозрительных областей. Имеется обучающий набор растровых изображений $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_N)$ (опять же черно-белых, серых или цветных), на которых помечены некоторые области. Например, изображения могут быть рентгенограммами какого-либо органа, на которых помечены опухоли. Нужно научиться по любому изображению X оценивать вероятность пометки каждой точки $R(X)$. С точки зрения реализации пометку на изображении $\mathbf{X}_i$ можно рассматривать как черно-белую картинку $\mathbf{Y}_i$ с $R(\mathbf{Y}_i) = R(\mathbf{X}_i)$, а оценку вероятности — как серую картинку Y.

Разметка изображения (image labeling). Имеется обучающий набор растровых изображений $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_N)$ (опять же черно-белых, серых или цветных), размеченных на области из конечного набора типов. Например, изображения могут быть аэро- или космическими снимками, а разметка — леса, поля, вода, снег, здания, дороги и т.п. Нужно научиться размечать любое изображение X. Опять же, разметку на изображении X можно рассматривать как картинку Y с конечным набором цветов, причем не обязательно с $R(Y) = R(X)$, а, возможно, гораздо меньшего разрешения.

Заметим, что для каждой точки $R(Y)$ в трех вышеописанных задачах требуется обучить регрессию (для черно-белых изображений вырождающуюся в классификацию), логистическую регрессию и (многоклассовую) классификацию, соответственно. Но общие методы статистического обучения для этого неприменимы, поскольку различные точки изображения нельзя считать независимыми, да и все обучающие изображения $\mathbf{X}_i$ могут быть разных размеров и форм.

Однако все три задачи можно решать достаточно стандартным путем, хотя и с различными техническими сложностями, если построить удачное небольшое (см. раздел 1.1.6) семейство вероятностных моделей пары изображений X и Y, не зависящих от устройства множества точек $R(Y)$, и в нем найти ("обучить") хорошую модель . При этом годятся и модели совместного распределения $p(X, Y)$ (порождающие модели), и модели условного распределения $p(Y|X)$ (дискриминантные модели, см. раздел 1.2.3), поскольку искомые ответы зависят только от условного распределения.

Пространство вероятностных распределений на изображениях любого размера необозримо велико, поэтому статистическое обучение во всем этом пространстве не только невероятно трудно технически, но и невозможно теоретически. Радикальное предположение, что разные точки изображения равноправны и независимы, резко уменьшило бы это пространство и свело бы анализ изображений к традиционным задачам классификации или регрессии, но такие простые модели совершенно не соответствуют реальности. Как правило, при моделировании изображений идут промежуточным путем: строят вероятностную модель изображения в небольшой, но все-таки не одноточечной области,

и предполагают, что такие маленькие подобласти большого изображения равноправны, а далекие друг от друга — еще и независимы. Изучением именно таких моделей, являющихся многомерным аналогом однородных марковских цепей, мы и займемся.

D.2 Случайные поля

D.2.1 Марковские случайные поля (MRF, Markov Random Fields)

Рассмотрим неориентированный граф G без петель с S вершинами и семейство случайных величин $Y = (y_1, \ldots, y_S)$, определенных в его вершинах и принимающих значение в одном и том же пространстве $\mathcal{Y}$. В приложениях к анализу изображений граф чаще всего будет квадратной решеткой в некоторой области плоскости (см. рис. 18), но пока годится любой граф, не обязательно даже связный (см. примеры на рис. 17–21). Две вершины называются соседями, если они соединены ребром. Множество соседей вершины v ("проколотую окрестность") будем обозначать через $N'(v)$, а вершину v вместе с ее соседями — $N(v) = \{v\} \cup N'(v)$. Все распределения (или плотности распределений) будем обозначать одной и той же буквой p: $p(y_v)$ — распределение случайной величины в вершине v, $p(Y) = p(y_1, \ldots, y_S)$ — совместное распределение во всех вершинах графа, $p(Y_A)$ — совместное распределение в подмножестве вершин A.

Пара, состоящая из графа и семейства случайных величин на его вершинах, называется *марковским случайным полем* (*MRF*, *Markov random field*), если для любой вершины v выполнено так называемое *локальное марковское свойство*

$$p\left(y_v | Y_{V\setminus\{v\}}\right) = p\left(y_v | Y_{N'(v)}\right) \ , \tag{306}$$

если его левая часть определена, т.е. условное распределение в любой вершине при условии значений в остальных вершинах на самом деле зависит только от значений в ее соседях. Равенство условных вероятностей (306) вместе с оговоркой о его определенности эквивалентно равенству

$$p\left(Y_{N'(v)}\right) p(Y) = p\left(Y_{N(v)}\right) p\left(Y_{V\setminus\{v\}}\right) \ . \tag{307}$$

Предложение 42 *Для марковского случайного поля выполняется следующее* попарное марковское свойство*: для любых двух не соседних вершин v и v'*

$$p\left(y_v, y_{v'} | Y_{V\setminus\{v,v'\}}\right) = p\left(y_v | Y_{V\setminus\{v,v'\}}\right) p\left(y_{v'} | Y_{V\setminus\{v,v'\}}\right) \ . \tag{308}$$

Доказательство. Равенство (307) — это равенство двух функций от S переменных $y_1, \ldots, y_S$. Его можно проинтегрировать по переменной $y_{v'}$. Дальнейшее вычисление оставляется в качестве **упражнения**. □

Будем сокращать словосочетание "марковское случайное поле" до традиционного в англоязычной литературе MRF. Одно и то же семейство случайных величин на одном и том же множестве вершин может быть или не быть MRF в зависимости от того, какие вершины считаются соседними (т.е. соединены ребрами). На полном графе любое семейство является MRF. Наоборот, на графе без ребер, состоящем из изолированных вершин, MRF — это в точности семейства независимых случайных величин. Возникают три естественных желания:

- по условным распределениям $p\left(y_v | Y_{V\setminus\{v\}}\right)$ восстановить совместное распределение $p(Y)$;
- по совместному распределению семейства случайных величин построить граф с минимальным числом ребер, для которого это семейство является MRF;

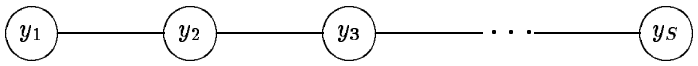

Рис. 17: Случайное поле на одномерной решетке,

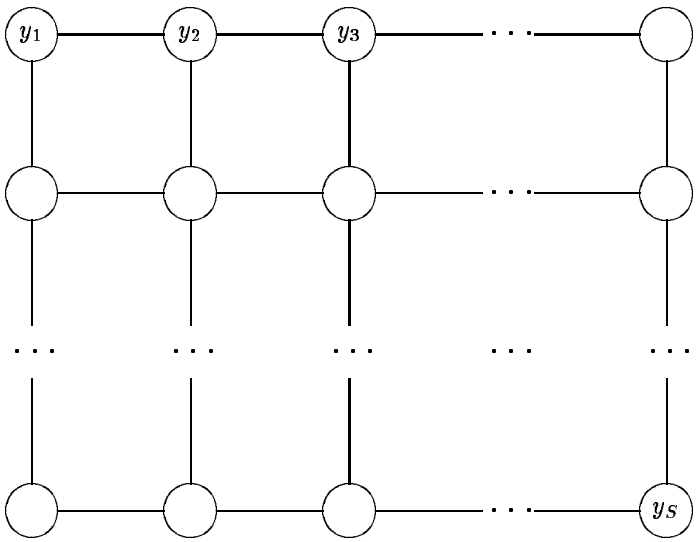

Рис. 18: на двумерной квадратной,

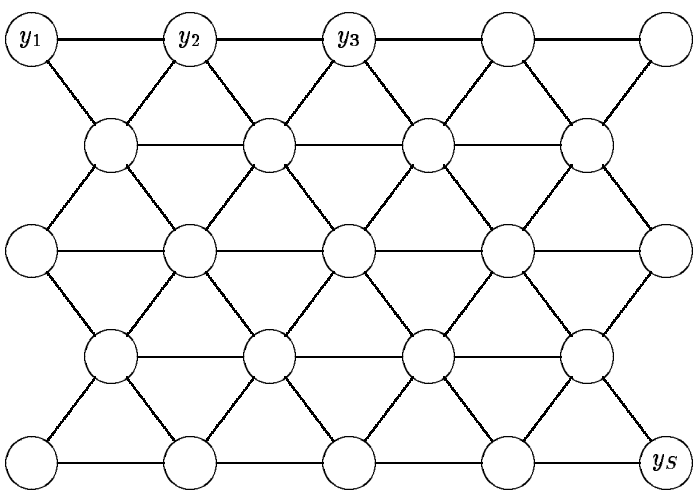

Рис. 19: на двумерной треугольной,

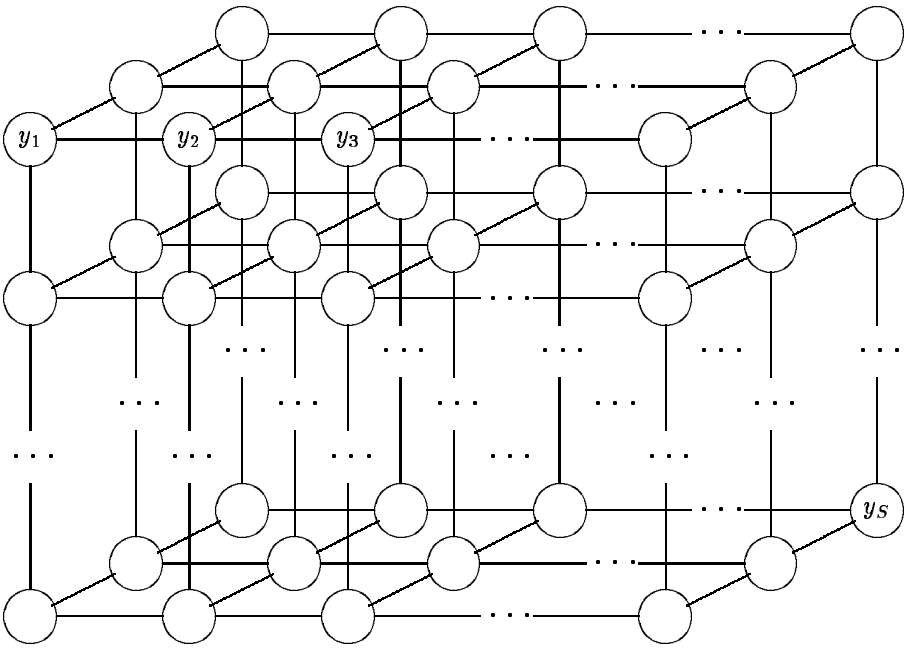

Рис. 20: на трехмерной кубической,

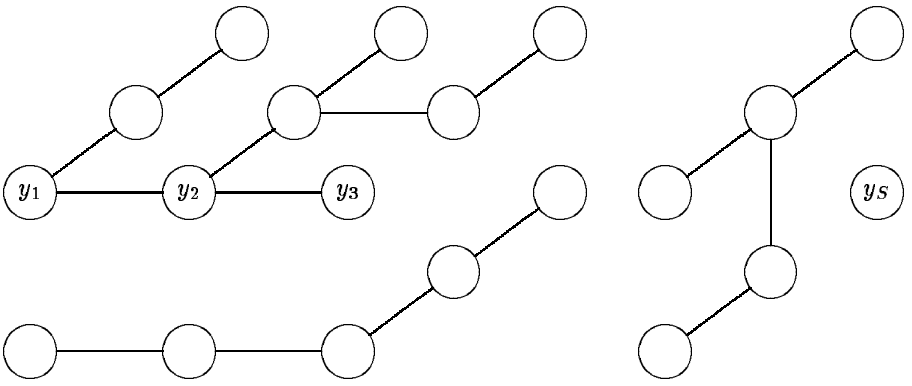

Рис. 21: на нерегулярном лесе.

- по графу построить описание общего вида MRF на нем.

Первое желание вообще никак не связано с марковским условием (306). Оно легко удовлетворяется при строго положительных условных распределениях: для любого фиксированного значения $y^* \in \mathcal{Y}$ выполнено равенство

$$\frac{p\left(y_v, Y_{V\setminus\{v\}}\right)}{p\left(y^*, Y_{V\setminus\{v\}}\right)} = \frac{p\left(y_v|Y_{V\setminus\{v\}}\right)}{p\left(y^*|Y_{V\setminus\{v\}}\right)} \ ,$$

значит

$$p\left(Y\right) = p\left(y_1, \ldots, y_S\right) = p\left(\underbrace{y^*, \ldots, y^*}_{S}\right) \prod_{j=1}^{S} \frac{p\left(y_j|y_1, \ldots, y_{j-1}, \underbrace{y^*, \ldots, y^*}_{S-j}\right)}{p\left(y^*|y_1, \ldots, y_{j-1}, \underbrace{y^*, \ldots, y^*}_{S-j}\right)} \ ,$$

где вероятность $p\left(\underbrace{y^*, \ldots, y^*}_{S}\right)$ однозначно определена вероятностной нормировкой, в частности, положительна.

Условие положительности всех условных распределений можно несколько ослабить, но следующий пример показывает, что, вообще говоря, совместное распределение по условным не восстанавливается.

Пример. Пусть граф G состоит из двух вершин, соединенных ребром, а условные вероятности заданы равенствами $p\left(y_1|y_2\right) = p\left(y_2|y_1\right) = \delta_{y_1}^{y_2}$. Тогда совместное распределение $p\left(y_1, y_2\right)$ — это любое распределение с носителем на диагонали $y_1 = y_2$.

Второе желание, казалось бы, удовлетворяется обращением предложения 42: сначала берется полный граф, а потом ребра между парами вершин, для которых выполнено условие (308), удаляются. В нижеследующих предложении и примере показано, что это действительно получается, если распределение строго положительно, и не получается в общем случае.

Предложение 43 *Если для всех пар вершин случайного поля со строго положительным распределением $p\left(Y\right)$ выполнено попарное марковское свойство, то оно является MRF.*

Доказательство. Удобно начать с полного графа и удалять ребра, удовлетворяющие условию (308) поштучно, проверяя, что при этом сохраняется локальное марковское свойство (306). Для строго положительного распределения $p\left(Y\right)$ условие (308) эквивалентно**(!)** условию

$$p\left(y_v|Y_{V\setminus\{v\}}\right) = p\left(y_v|Y_{V\setminus\{v,v'\}}\right) \ , \tag{309}$$

из условий (306) и (309) при $v' \in N'(v)$ следует**(!)**

$$p\left(y_v|Y_{V\setminus\{v\}}\right) = p\left(y_v|Y_{N'(v)\setminus\{v'\}}\right) \ .$$

Верно и аналогичное равенство с взаимно переставленными вершинами v и v', а значит ребро (v, v') из графа MRF можно удалить. $\square$

Пример. Рассмотрим MRF с полным графом с четырьмя вершинами и случайными величинами в них, принимающими значения 0 или 1 с совместным распределением с носителем на диагонали: $p(0,0,0,0) = p(1,1,1,1) = \frac{1}{2}$. Тогда для любой пары вершин выполнено условие условной независимости (308), так что любое ребро графа, казалось бы, можно удалить без потери марковости (306). Но если удалить все ребра, условие (306) в изолированных вершинах не выполнено. Самый экономный способ сохранить марковость — это любым из трех возможных способов оставить два ребра, связывающие вершины попарно.

Перейдем к выполнению третьего желания — описанию общего вида распределения MRF с данным графом.

Напомним, что полный, т.е. вместе с каждой парой вершин содержащий соединяющее их ребро, подграф графа называется *кликой*. В частности, кликами являются любая пара вершин, соединенных ребром, любая вершина или пустой подграф. Обозначим через $\mathcal{C}(G)$ множество клик в графе G. Распределение семейства Y случайных величин на вершинах графа G со значениями в пространстве $\mathcal{Y}$ называется *распределением Гиббса*, если оно представимо в виде произведения

$$p(Y) = \prod_{C \in \mathbf{F}} \Psi_C(Y_C) \tag{310}$$

по некоторому множеству $\mathbf{F} \subset \mathcal{C}(G)$ клик графа G.

Заметим, что Ψ_C могут быть любыми функциями от своих $\#C$ $\mathcal{Y}$-значных переменных: ни равенства $\Psi_C(Y_C) = p(Y_C)$, ни однозначность разложения (310) не требуются. Поскольку распределение $p(Y)$ неотрицательно, все функции Ψ_C можно считать неотрицательными. Простейшим частным случаем распределения Гиббса является распределение независимых случайных величин.

Предложение 44 *Любое распределение Гиббса на графе является марковским случайным полем.*

Доказательство. Подставляя равенство (310) в равенство

$$p\left(y_v | Y_{V\setminus\{v\}}\right) = \frac{p(Y)}{\int p(Y)\, dy_v}$$

и сокращая совпадающие сомножители, не зависящие от y_v, получаем

$$p\left(y_v | Y_{V\setminus\{v\}}\right) = \frac{\prod_{C\in\mathbf{F}, C\ni v} \Psi_C(Y_C)}{\int \left(\prod_{C\in\mathbf{F}, C\ni v} \Psi_C(Y_C)\right) dy_v} \ .$$

Аналогично, сокращая не зависящие от y_v интегралы (при дискретном пространстве $\mathcal{Y}$ — конечно же, суммы), в выражении

$$p\left(y_v | Y_{N'(v)}\right) = \frac{\int p(Y)\, dY_{V\setminus N(v)}}{\int p(Y)\, dY_{V\setminus N(v)} dy_v} \ ,$$

получаем ту же правую часть:

$$p\left(y_v | Y_{N'(v)}\right) = \frac{\prod_{C\in\mathbf{F}, C\ni v} \Psi_C(Y_C)}{\int \left(\prod_{C\in\mathbf{F}, C\ni v} \Psi_C(Y_C)\right) dy_v} \ . \tag{311}$$

Разбор случая, когда какие-то сомножители в знаменателе равны 0, оставляется в качестве **упражнения**. □

Для всюду положительного распределения верно и обратное утверждение:

Теорема 13 *(J.M.Hammersley, P.Clifford, 1971 [47]). Если распределение $p(Y)$ марковского случайного поля всюду положительно, то оно является распределением Гиббса вида (310).*

Если распределение MRF не всюду положительно, оно может не быть гиббсовским.

Пример. (J.Moussouris, 1974 [87]) Граф G — квадрат с вершинами 1, 2, 3 и 4 и ребрами $[1,2]$, $[2,3]$, $[3,4]$ и $[4,1]$, случайные величины в вершинах принимают два значения 0 и 1, причем из $2^4 = 16$ возможных наборов значений ненулевую вероятность имеют только 8:

$$\begin{aligned} & p(0,0,0,0) = p(0,0,0,1) = p(0,0,1,1) = p(0,1,1,1) \\ = \; & p(1,1,1,1) = p(1,1,1,0) = p(1,1,0,0) = p(1,0,0,0) = \frac{1}{8} \; . \end{aligned}$$

Выполнение условия (306) проверяется непосредственно; для облегчения проверки удобно заметить, что при любых фиксированных значениях на любой диагонали квадрата значение в одной из оставшихся вершин определено однозначно, а в значения в другой равновероятны. С другой стороны, все возможные сочетания значений в вершинах любой клики — т.е. ребра или отдельной вершины — принимаются с ненулевой вероятностью, значит распределение Гиббса (310) было бы положительным всюду. Противоречие.

Для доказательства теоремы Хаммерсли-Клиффорда понадобится чисто комбинаторная лемма, аналогичная формуле включений-исключений. Пусть R — конечное множество, f любая функция на пространстве $\mathcal{Y}^R$, со значениями в коммутативной группе (например, в аддитивной группе вещественных чисел $\mathbb{R}$), $y^* \in \mathcal{Y}$ — произвольный элемент. Для любого подмножества $A \subset R$ и набора $Y = (y_1, \ldots, y_S) \in \mathcal{Y}^R$ обозначим $y_j^A = \begin{cases} y_j & \text{при } j \in A \\ y^* & \text{при } j \notin A \end{cases}$, $Y^A = (y_1^A, \ldots, y_S^A)$ (в отличие от нижнего индекса в обозначении Y_A, означавшего ограничение семейства величин на подмножество индексов A, верхний индекс в Y^A означает фиксацию значений величин с индексами вне A), $f^A(Y) = f(Y^A)$ и

$$g^A(Y) = \sum_{B \subset A} (-1)^{\#(A \setminus B)} f^B(Y) \; . \tag{312}$$

Лемма 8 *Для любого множества $C \subset R$*

$$f^C(Y) = \sum_{A \subset C} g^A(Y) \; , \tag{313}$$

в частности,

$$f(Y) = f^R(Y) = \sum_{A \subset R} g^A(Y) \; .$$

Доказательство. Утверждение леммы получается прямым вычислением:

$$\begin{aligned} \sum_{A \subset C} g^A(Y) &= \sum_{A \subset C} \sum_{B \subset A} (-1)^{\#(A \setminus B)} f^B(Y) = \sum_{B \subset C} \left(\sum_{D \subset C \setminus B} (-1)^{\#D} \right) f^B(Y) \\ &= \sum_{B \subset C} \left(\sum_{k=0}^{\#(C \setminus B)} \binom{\#(C \setminus B)}{k} (-1)^k \right) f^B(Y) = \sum_{B \subset C} 0^{\#(C \setminus B)} f^B(Y) \\ &= f^C(Y) \; . \end{aligned}$$

□

Доказательство теоремы 13. Для MRF с всюду положительным распределением p применим лемму 8 к функции $f(Y) = \ln p(Y)$, заметим, что в получившемся представлении

$$\ln p(Y) = \sum_{A \subset R} g^A(Y)$$

каждая функция $g^A(Y)$ фактически является функцией от ограничения Y_A семейства Y случайных величин на множество A, и покажем, что все слагаемые, соответствующие не кликам, равны нулю.

Пусть $v, v' \in A$ — не соседние вершины. Попарное марковское условие (308), сформулированное в терминах условных вероятностей, можно переписать в терминах абсолютных вероятностей:

$$p\left(y_v, y_{v'}, Y_{V\setminus\{v,v'\}}\right) = \frac{p\left(y_v, Y_{V\setminus\{v,v'\}}\right) p\left(y_{v'}, Y_{V\setminus\{v,v'\}}\right)}{p\left(Y_{V\setminus\{v,v'\}}\right)} \ . \tag{314}$$

Для зафиксированных вершин v и v' слагаемые в определяющей $g^A(Y)$ формуле (312) можно сгруппировать в четверки, по одной на каждое подмножество $B \subset A \setminus \{v, v'\}$:

$$\begin{aligned} g^A(Y) = \sum_{B \subset A\setminus\{v,v'\}} (-1)^{\#B} \Big(& \ln p\left(Y^B\right) - \ln p\left(Y^{B\cup\{v\}}\right) \\ & - \ln p\left(Y^{B\cup\{v'\}}\right) + \ln p\left(Y^{B\cup\{v,v'\}}\right)\Big) \\ = \sum_{B \subset A\setminus\{v,v'\}} (-1)^{\#B} & \ln \frac{p\left(Y^B\right) p\left(Y^{B\cup\{v,v'\}}\right)}{p\left(Y^{B\cup\{v\}}\right) p\left(Y^{B\cup\{v'\}}\right)} \ . \end{aligned}$$

Но, если подставить формулу (314) вместо каждого из сомножителей под логарифмом в каждом слагаемом-четверке, все получившиеся двенадцать сомножителей сокращаются и от слагаемого остается $(-1)^{\#B} \ln 1 = 0$. $\square$

Замечание. Поскольку в доказательстве теоремы 13 использовалось только попарное марковское свойство, то из него и из предложения 44 снова следует утверждение предложения 43.

Таким образом, MRF с положительным совместным распределением всегда является распределением Гиббса вида

$$p(Y) = e^{\sum\limits_{C \in \mathcal{C}(G(Y))} g_C(Y)} \ . \tag{315}$$

В дальнейшем будут рассматриваться только положительные распределения Гиббса, но вместо удобных для доказательства теоремы Хаммерсли-Клиффор обозначений (310) или (315), будут использоваться классические обозначения, принятые в статистической физике:

$$p(Y) = \frac{1}{Z}\ e^{-\sum\limits_{C \in \mathbf{F}} E_C(Y_C)} \ , \tag{316}$$

где $\mathbf{F}$ — некоторое множество непустых клик, $E_C(\cdot) = -\ln \Psi_C(\cdot)$ (ср. с (310)) — функция на ограничении поля на вершины клики C, а множитель $Z = -\ln \Psi_\emptyset$ однозначно определен вероятностной нормировкой распределения.

D.2.2 Модель Изинга и другие примеры

Вероятностные модели вида (316) широко применяются в статистической физике. Действительно, если в распределении Больцмана $p(Y) \propto e^{-\frac{E(Y)}{kT}}$ набор случайных величин Y описывает набор систему частиц, находящихся в вершинах графа G, энергия E системы считается равной сумме энергий взаимодействия групп близких частиц, причем вершины графа соединены ребром в точности тогда, когда они входят в какую-то такую группу, а температура T фиксирована, то распределение имеет в точности вид (316).

Самая простая и самая знаменитая модель такого рода — это модель Изинга, названная в честь Эрнста Изинга, впервые применившего ее в 1920-ые годы для описания ферромагнетизма. Модель устроена следующим образом. Вершины графа G соответствуют атомам кристаллической решетки. У каждого атома v_j и имеется "магнитный спин" s_j, принимающий значения ± 1 (грубо говоря, это проекция магнитного момента на фиксированное направление). Энергия взаимодействия i-го и j-го атома считается равной $-J_{ij}s_is_j$, где для пары соседних атомов коэффициент $J_{ij} = J > 0$ постоянен (т.е. соседним атомам энергетически выгоднее иметь однонаправленные спины), а для не соседних - равен нулю. Взаимодействие групп из более чем двух атомов не учитывается, даже если в кристаллической решетке есть клики, отличные от пар вершин. Если вещество находится в одномерном внешнем магнитном поле напряженностью h_j в j-й вершине, то к энергии попарного взаимодействия атомов добавляются энергии $-h_js_j$ взаимодействия атомов с полем (т.е. спину энергетически выгоднее быть направленным вдоль поля). В результате распределение Больцмана для модели Изинга получается следующим:

$$p\left(Y|h\right) = \frac{1}{Z(T,h)}\, e^{\left(\frac{J}{kT}\sum\limits_{i<j|\{v_i,v_j\}\in\mathbf{F}} s_is_j + \frac{1}{kT}\sum\limits_j h_js_j\right)}, \qquad (317)$$

где $\mathbf{F}$ — множество ребер графа $G(Y)$, а

$$Z(T,h) = \sum_{s_1,\ldots,s_S=\pm 1} e^{\left(\frac{J}{kT}\sum\limits_{i<j|\{v_i,v_j\}\in\mathbf{F}} s_is_j + \frac{1}{kT}\sum\limits_j h_js_j\right)}. \qquad (318)$$

Самым сложным в применении модели (317), является вычисление множителя $Z(T,h)$, хотя бы приближенное, поскольку количество слагаемых в сумме (318), равное 2^S, экспоненциально велико.

В модели Изинга проявляются экспериментально известные свойства ферромагнетиков: при слабом внешнем поле и низких температурах в решетке образуются довольно большие области атомов с одинаковым спином (магнитные домены); при повышении температуры они уменьшаются[64].

Примерно так же, как магнитные спины в ферромагнетике от внешнего поля, зависит чистое черно-белое изображение от изображения, на которое наложен случайный шум: чистое изображение коррелировано с зашумленным, но в нем всегда есть достаточно большие области постоянного цвета, что в зашумленном изображении крайне маловероятно. Эти соображения позволяют применить аналог модели Изинга для очистки изображения от шума. Нужно только научиться подбирать подходящую температуру...

[64] В модели Изинга на неограниченных решетках размерности два и более, как и на практике, домены резко разрушаются при достижении некоторой критической температуры; в одномерной модели, исследованной самим Изингом, такого эффекта нет.

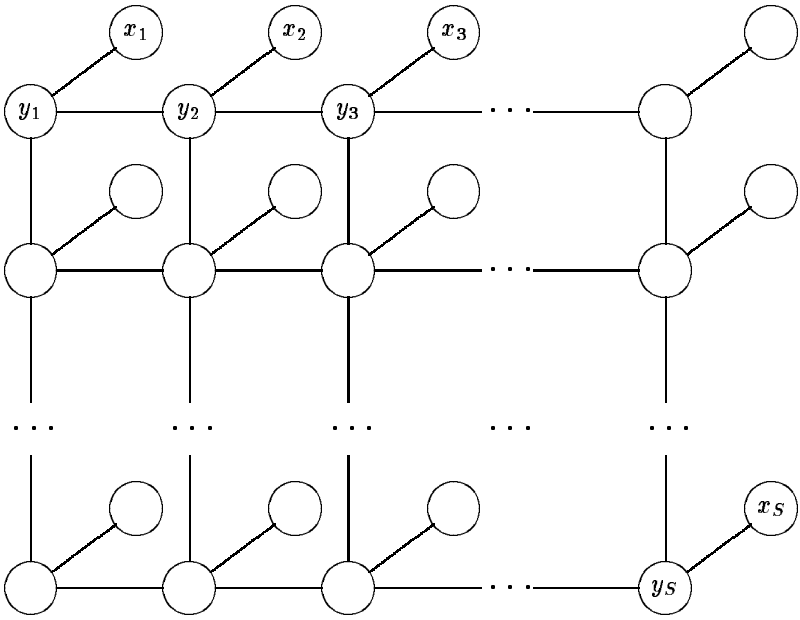

Рис. 22: Граф двумерной модели Изинга.

Пусть G — квадратная решетка в двумерной области, обычно в прямоугольнике, $X = (x_1, \ldots, x_S)$ и $Y = (y_1, \ldots, y_S)$, $X, Y \in \{-1, 1\}^S$ — наблюдаемое зашумленное и ненаблюдаемое чистое изображение, соответственно. Рассмотрим утрированно простую модель условного распределения $p(X|Y)$: шумы в каждой точке независимы и одинаково распределены, вероятность значения x_j (-1 или $+1$, черное или белое) зашумленного изображения в j-м узле зависит только от значения y_j чистого изображения в этом же узле, причем $p(x_j = y_j) > p(x_j \neq y_j)$. Без ограничения общности можно считать, что $p(x_j|y_j) = \frac{e^{\beta x_j y_j}}{e^{\beta} + e^{-\beta}}$, где $\beta > 0$. Априорное распределение $p(Y)$ на пространстве изображений положим равным

$$p(Y) = \frac{1}{Z(\alpha)} \, e^{\left(\alpha \sum\limits_{i<j|\{v_i,v_j\}\in\mathbf{F}} y_i y_j \right)}, \tag{319}$$

как в модели Изинга без внешнего поля. Тогда пара изображений X и Y является MRF, граф которого — двухслойная решетка, изображенная на рис. 22 (ср. с рис. 10 для HMM), а совместное распределение

$$p(X, Y) = p(Y)\, p(X|Y) = \frac{1}{Z(\alpha, \beta)} \, e^{\left(\alpha \sum\limits_{i<j|\{v_i,v_j\}\in\mathbf{F}} y_i y_j + \beta \sum\limits_j x_j y_j \right)}, \tag{320}$$

где $\alpha, \beta > 0$. Пользуясь тождествами $x_j^2 = y_j^2 = 1$ модель (320) можно перепи-

сать в виде

$$p\,(X,Y)=\frac{1}{Z(\alpha,\beta)}\;e^{-\left(\alpha\sum\limits_{i<j|\{v_i,v_j\}\in\mathbf{F}}(y_i-y_j)^2+\beta\sum\limits_j(x_j-y_j)^2\right)} \tag{321}$$

с вдвое меньшими параметрами α и β и другим нормировочным коэффициентом $Z(\alpha,\beta)$. В таком виде она применима не только к черно-белым, но и к серым ($x_j,y_j\in\mathbb{R}$) и цветным ($x_j,y_j\in\mathbb{R}^3$) изображениям. Кроме того, открывается простор для экспериментов с попарными энергиями, отличными от квадратов разностей, например,

$$p\,(X,Y)=\frac{1}{Z(\alpha,\beta)}\;e^{-\left(\alpha\sum\limits_{i<j|\{v_i,v_j\}\in\mathbf{F}}(y_i-y_j)^2+\beta\sum\limits_j|x_j-y_j|\right)}\;.$$

Изображенное на рис. 22 MRF с совместным распределением (320) или (321) является порождающей моделью. При другой вероятностной модели шума (например, когда "грязное" изображение в точке зависит от "чистого" в нескольких соседних точках) получится другое MRF с другим распределением. На самом деле для распознавания изображений вместо совместного распределения $p\,(X,Y)$ достаточно моделировать условное распределение $p\,(Y|X)$, имеющее, ту же самую функцию энергии, но зависящий от X нормировочный множитель. В частности, совместному распределению (321) соответствует условное распределение

$$p\,(Y|X)=\frac{1}{Z(\alpha,\beta,X)}\;e^{-\left(\alpha\sum\limits_{i<j|\{v_i,v_j\}\in\mathbf{F}}(y_i-y_j)^2+\beta\sum\limits_j(x_j-y_j)^2\right)}\;, \tag{322}$$

а из совместного распределения (320) получается (с точностью до обозначений) модель Изинга (317).

Можно показать, что при непрерывном пространстве ответов $\mathcal{Y}$ для вышеописанных модификаций модели Изинга (320)–(322) наиболее вероятное поле ответов Y при данном зашумленном изображении X является сглаженным полем X[65]. Таким образом, вместе с устранением шума эти модели сглаживают контуры изображения. Поскольку контуры, как правило, являются наиболее информативной частью изображений, для устранения шумов при сохранении контуров приходится применять более сложные модели. Опуская технические детали, приведем идею построения MRF для таких моделей, заимствованную из работы [44]. Граф этого MRF изображен на рис. 23. Изображение X, как и в модели Изинга (рис. 22), состоит из значений x_j (черно-белых серых или цветных) в вершинах прямоугольной решетки, а поле ответов — из таких же значений y_j в вершинах решетки, а также двоичных значений $y_{i,j}\in\{0,1\}$ на

[65]Например, максимизация по вещественнозначному полю Y выражения (322) на прямоугольной решетке сводится к решению системы уравнений

$$0=\frac{\partial}{\partial y_j}\left(\alpha\sum_{i<j|\{v_i,v_j\}\in\mathbf{F}}(y_i-y_j)^2+\beta\sum_j(x_j-y_j)^2\right)=-2\left(\alpha\sum_{i|\{v_i,v_j\}\in\mathbf{F}}(y_i-y_j)+\beta(x_j-y_j)\right)\;,$$

т.е. к разностному аналогу эллиптического уравнения в частных производных второго порядка

$$\alpha\nabla^2y-\beta y=-\beta x\;.$$

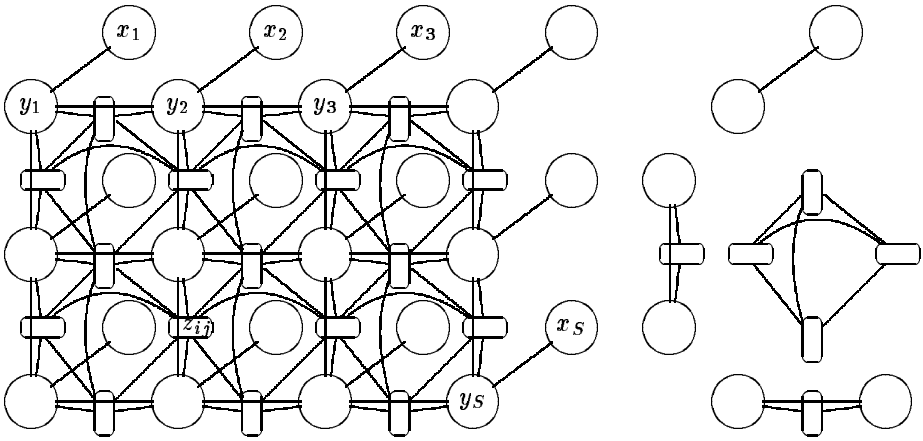

Рис. 23: Граф упрощенного MRF для очистки изображения от шумов (по мотивам работы [44]). Слева изображен фрагмент графа, справа — клики, участвующие в вычислении энергии.

ее ребрах. Удобно считать, что эти значения $y_{i,j}$ находятся в серединах ребер. На рис. 23 слева каждая такая середина ребра соединена с обеими его вершинами и для каждого элементарного квадрата решетки середины всех его ребер соединены друг с другом; справа изображены клики получившегося графа. Равенство $y_{i,j} = 1$ означает, что между соседними вершинами v_i и v_j проходит линия контура, поэтому яркости/цвета y_i и y_j друг от друга не зависят, т.е. энергия их взаимодействия равна нулю. Но и сами контуры в поле ответов также имеют энергию, пропорциональную (с не очень большим коэффициентом γ_2) их суммарной длине, оцениваемой как количество пересекаемых ими квадратов решетки, плюс (с гораздо большими коэффициентами γ_1, γ_3 и γ_4) количества особых точек на них: концов контуров, тройных и четверных точек, соответственно. Обозначим для краткости через Q множество элементарных квадратов прямоугольной решетки, через $n(Y,s)$ — сумму значений $y_{i,j}$ в серединах ребер квадрата s и положим $\gamma_0 = 0$. Обобщением распределения (322) будет распределение

$$p\left(Y|X\right) = \frac{1}{Z(\alpha,\beta,X)}\ e^{-E(X,Y)}\ , \tag{323}$$

где

$$E(X,Y) = \alpha \sum_{i<j|\{v_i,v_j\}\in\mathbf{F}} (1-y_{i,j})(y_i-y_j)^2 + \beta\sum_j (x_j-y_j)^2 + \sum_{s\in Q}\gamma_{n(Y,s)}\ . \tag{324}$$

D.2.3 Условные случайные поля (CRF, Conditional Random Fields)

Тройка, состоящая из графа G, семейства $Y = (y_1,\ldots,y_S)$ случайных величин, определенных в его вершинах и случайной величины X (которая может также являться семейством случайных величин в вершинах графа G, но может и не являться), называется *условным случайным полем* (*CRF*, *conditional random field*), если для любой вершины v выполнено равенство

$$p\left(y_v|Y_{V\setminus\{v\}},X\right) = p\left(y_v|Y_{N'(v)},X\right)\ , \tag{325}$$

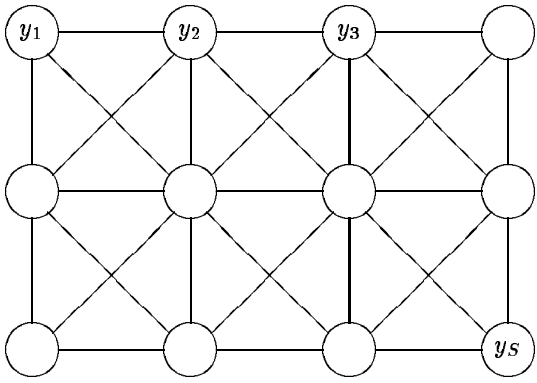

Рис. 24: Кликами этого графа являются единичные квадраты решетки и их подмножества.

где, как и в равенстве (306), через $N'(v)$ обозначено множество соседей вершины v в графе G.

Если распределение $p(Y|X)$ условного случайного поля (CRF) положительно при любых значениях X и Y, то для него справедлива теорема Хаммерсли-Клиффорда 13, причем доказательство переносится с марковских полей на условные без малейших изменений. В дальнейшем будем предполагать распределение CRF положительным и представленным в виде распределения Гиббса (316), зависящего от X как от параметра

$$p(Y|X) = \frac{1}{Z(X)}\, e^{-\sum\limits_{C\in\mathbf{F}} E_C(X,Y_C)}. \tag{326}$$

Выше было показано, что из гиббсовского представления (321) совместного распределения $p(X,Y)$ марковского поля немедленно получается гиббсовское же представление (322) условного распределения $p(Y|X)$, т.е. из MRF получается CRF. Но CRF можно строить и другими способами. Например, разметку аэрофотоснимков можно организовать так.

- Сначала на небольших фрагментах изображений, например, на квадратах фиксированного размера $l \times l$, обучить распознаватель (скажем, нейронную сеть), оценивающий вероятность $p(y|X)$ принадлежности всего фрагмента каждому из возможных классов.

- Затем построить MRF с распределением Гиббса $p(Y)$ следующим образом:

  - каждая вершина графа соответствует квадрату $l \times l$ точек изображения,

  - вершины образуют прямоугольную решетку, но соседними считаются, например соседи и по горизонтали и по диагонали (рис. 24), так что кликами являются четверки вершин квадратов решеток и их подмножества, т.е. непустые клики бывают десяти типов: четверки, тройки четырех возможных ориентаций, пары по вертикали, горизонтали или одной из двух диагоналей и отдельные вершины,

– каждому возможному распределению меток классов Y_C в вершинах клики C типа номер $t(C)$ приписывается некоторая функция $E_{t(C)}(Y_C)$, не зависящая от положения клики C в решетке; при разметке на q классов получается порядка q^4 параметров вне зависимости от размеров изображения.

- И наконец, задать CRF (!) распределением

$$p(Y|X) \propto e^{\left(\gamma \sum\limits_{v\in V} \ln p(y_v|X_v) - \sum\limits_{t=1}^{10} \sum\limits_{C\in\mathcal{C}(G)|t(C)=t} E_t(Y_C)\right)},$$

где X_v — это ограничение изображения X на квадрат, соответствующий вершине v разметки, $f^{y_v}(X_v)$ — даваемая локальным распознавателем оценка вероятности $p(y_v|X_v)$, а параметр γ определяет, насколько можно доверять независимой классификации квадратов, и подбирается вместе с функциями $E_t(Y_C)$ при обучении.

При изучении CRF часто заимствуется из статистической физики идея включать CRF (326) в однопараметрические семейства

$$p(Y|T,X) = \frac{1}{Z(T,X)}\, e^{-\sum\limits_{C\in\mathbf{F}} \frac{E_C(X,Y_C)}{T}}. \tag{327}$$

и терминология

- $E_C(X,Y_C)$ называется *энергией взаимодействия* в клике C;
- $-T\ln Z(T,X)$ называется *свободной энергией*;
- параметр $T>0$ называется *температурой*;
- любой набор $Y=(y_1,\ldots,y_S)$ величин из пространства ответов $\mathcal{Y}$, а не только случайный, называется *полем*.

В последующих разделах мы будем рассматривать семейства CRF на семействах графов немного более общего вида, чем в приведенных примерах. Зафиксируем семейство подмножеств $\mathcal{C}_1,\ldots,\mathcal{C}_M$ множества всех непустых клик всех графов семейства, такое что клики одного подмножества $\mathcal{C}_m$ изоморфны (например, графы считаются вложенными в плоскость — все-таки речь идет о анализе изображений — а изоморфизмы клик порождены параллельными переносами плоскости). Множества $\mathcal{C}_m$ могут пересекаться и даже совпадать. Для каждого индекса m, называемого *типом энергии*, для каждой клики $C\in\mathcal{C}_m$ определена функция энергии $E_m(X_{m,C},Y_C)$, где Y_C — это, как обычно, ограничение случайного поля на вершины клики C, а $X_{m,C}$ — некоторая, зависящая от типа энергии m и клики C часть изображения X, которое не обязательно является случайным полем на том же графе. Тогда обещанное семейство CRF — это семейство случайных полей Y на графах $G(Y)$, вообще говоря, разных, с распределениями Гиббса

$$p(Y|w,X) = \frac{1}{Z(w,X)}\, e^{-\left(\sum\limits_{m=1}^{M} w_m \sum\limits_{C\in\mathcal{C}(G(Y))\cap\mathcal{C}_m} E_m(X_{m,C},Y_C)\right)}, \tag{328}$$

зависящими от вектора коэффициентов $w=(w_1,\ldots,w_M)$.

Обозначим через $E_{m,X,Y} = \sum_{C \in \mathcal{C}(G(Y)) \cap \mathcal{C}_m} E_m(X_{m,C}, Y_C)$ сумму энергий типа m по всем кликам графа $G(Y)$, на котором определено поле ответов Y для изображения X, а через $E_{\mathbf{X},\mathbf{Y}}$ — вектор всех $E_{m,\mathbf{X},\mathbf{Y}}$. Тогда сумма $\sum_{m=1}^{M} w_m E_{m,\mathbf{X},\mathbf{Y}}$ — это скалярное произведение $(w, E_{\mathbf{X},\mathbf{Y}})$, и формула (328) переписывается в более компактном виде

$$p\,(Y|w,X) = \frac{1}{Z(w,X)}\; e^{-(w,E_{X,Y})}\;. \tag{329}$$

Однопараметрическое семейство распределений Гиббса (327), зависящее от температуры, является частным случаем M-параметрического семейства (328). В дальнейшем при обсуждении обучения CRF будет использоваться представление (328) или (329) для условного распределения, а при обсуждении применения фиксированного CRF — любое из представлений (326), (327), (328) или (329).

D.3 Применение случайных полей для анализа изображений

После того, как для поля ответов Y в какой-либо задаче анализа изображений построено семейство CRF (329), возникает потребность в следующих вычислениях:

1. по изображению X и полю ответов Y вычислить вероятность $p\,(Y|X)$;
2. по изображению X вычислить наилучшее поле ответов Y (задачи очистки изображения от шумов или разметки), т.е. максимизировать $p\,(Y|X)$ по Y;
3. по изображению X вычислить распределение вероятностей в каждой точке $p\,(y_v|X)$ (задача поиска подозрительных областей);
4. по обучающему набору $(\mathbf{X},\mathbf{Y})$ выбрать оптимальное CRF в семействе (обучить CRF), например, максимизируя правдоподобие $p\,(\mathbf{Y}|\mathbf{X})$ по параметрам модели.

Вычисление 1 нужно не само по себе, а как первый этап последующих вычислений. Основная трудность в вычислении 1 — это вычисление нормировочного множителя

$$Z(w,X) = \int_{y_1,\ldots,y_S} e^{-(w,E_{X,Y})} dy_1, \ldots dy_S\;,$$

где, как обычно, для дискретного пространства ответов $\mathcal{Y}$ интеграл превращается в сумму

$$Z(w,X) = \sum_{y_1,\ldots,y_S \in \mathcal{Y}} e^{-(w,E_{X,Y})}\;. \tag{330}$$

Вычислять этот интеграл или сумму непосредственно — неприемлемо (экспоненциально по размеру изображения) долго. Сделать это аналитически удается крайне редко. Для CRF на одномерных решетках или деревьях известны быстрые рекурсивные алгоритмы вычисления таких сумм, аналогичные алгоритму "вперед-назад" для HMM (стр. 188). Но для двумерных решеток приемлемо быстрых точных алгоритмов не известно, зато известны разнообразные приближенные, более или менее быстрые, более или менее сложные, гарантирующие или не гарантирующие сходимость к разумному ответу.

К счастью, вычисление 2 (минимизация $p\,(Y|X)$ по Y) не требует вычисления $Z(w,X)$.

Остаток этого раздела состоит из описания некоторых не слишком сложных стохастических алгоритмов проведения вычислений 2–4 в обход честного вычисления 1 (вероятности $p(Y|X)$) и рассмотрения методов приближенной оценки $Z(w,X)$ и распределений $p(Y_R|X)$ ограничений полей ответов на небольшие множества узлов, например, клики. Эти оценки дают альтернативные способы проведения вычислений 3–4.

D.3.1 Поиск наиболее вероятного поля ответов

Для распределения Гиббса (326) или (327) наиболее вероятный ответ для данного изображения X — это поле Y, минимизирующее стоящую в показателе суммарную энергию

$$E(X,Y) = \sum_{C\in\mathbf{F}} E_C(X,Y_C)\ , \tag{331}$$

являющуюся функцией от многих переменных $y_1,\ldots,y_S$. В задачах регрессии с непрерывным пространством ответов, например, $\mathcal{Y}=\mathbb{R}$, для минимизации энергии можно применять любые методы, например, градиентные(некоторые технические подробности приведены в разделе 3.2.4). В исключительно счастливом случае, если энергия выпукла по Y, как, например, в CRF (322), любой разумный градиентный метод сходится к единственному минимуму — глобальному. Если энергия не выпукла, градиентный спуск может привести в локальный минимум, не являющийся глобальным, и ничем не хороший. Для дискретного же пространства $\mathcal{Y}$ градиентные методы вообще не применимы.

Поскольку от каждого значения y_v зависит очень небольшая доля слагаемых в сумме (331), энергию удобно минимизировать покоординатно, циклически обходя все узлы v графа $G(Y)$. Этот способ применим и для дискретного пространства $\mathcal{Y}$, но сходимость к глобальному минимуму энергии он тоже не гарантирует.

Зато для распределений Гиббса минимизировать энергию можно с помощью специального варианта метода имитации отжига (стр. 78), который мы сейчас рассмотрим подробно.

Выборка Гиббса (Gibbs sampler)

Будем предполагать пространство ответов $\mathcal{Y}$ конечным q-элементным. Тогда для распределения Гиббса (327) условное распределение $p\left(y_v|Y_{V\setminus\{v\}},X\right)$ в одной вершине v выписывается явно:

$$p\left(y_v|Y_{V\setminus\{v\}},X\right) = \frac{1}{Z_v(T,X,Y)}\ e^{-\sum\limits_{C\in\mathbf{F},C\ni v}\frac{E_C(X,Y_C)}{T}}\ . \tag{332}$$

В отличие от нормировочного множителя $Z(T,X)$ распределения Гиббса в целом, вычисление нормировочных множителей

$$Z_v(T,X,Y) = \sum_{y^*\in\mathcal{Y}} e^{-\sum\limits_{C\in\mathbf{F},C\ni v}\frac{E_C(X,Y_C^{V\setminus\{v\}})}{T}} \tag{333}$$

никаких трудностей не представляет, так как сводится к суммированию небольшого числа слагаемых в каждой вершине графа. Это позволяет в пространстве случайных полей на графе $G(Y)$ начиная с любого — случайного или нет — поля ответов $Y^{(0)}$ строить случайные последовательности полей $Y^{(i)}$ следующим образом:

1. зафиксируем некоторый порядок обхода $v^{(1)}, v^{(2)}, \ldots$ вершин $v_1, \ldots, v_S$, так чтобы при каком-то $K \geq S$ на любых K последовательных шагах обхода встречалась каждая вершина (например, просто обход в порядке номеров вершин с циклическим повторением);

2. на i-м шаге в вершинах v, отличных от $v^{(i)}$, полагаем $y_v^{(i)} = y_v^{(i-1)}$, а в вершине $v^{(i)}$ генерируем случайное значение $y_{v^{(i)}}^{(i)}$ по распределению $p\left(y_{v^{(i)}} | Y^{(i-1)}_{V\setminus\{v^{(i)}\}}, X\right)$, т.е. условному распределению (332) при условии $Y^{(i-1)}$.

Процесс построения последовательности $Y^{(i)}$ называется *выборкой Гиббса* (*Gibbs sampler* или *Gibbs sampling*). Так же называется и сама последовательность, и ее любой ее элемент (*Gibbs sample*). Трудности перевода...

Последовательность $Y^{(i)}$ очевидным образом является марковской цепью с пространством состояний $\mathcal{Y}^S$. Из пункта 1 ее построения и положительности распределения Гиббса следует, что она эргодична (из любого состояния в любое можно за сколько-то шагов попасть с положительной вероятностью), а из пункта 2 — что распределение Гиббса (327) является ее стационарным распределением.

Теорема 14 (S.Geman, D.Geman ([44])) *Для любого начального поля $Y^{(0)}$ распределение $p\left(Y^{(i)} | Y^{(0)}, X\right)$ при $i \to \infty$ сходится к распределению Гиббса (327).*

Доказательство теоремы несложное, но громоздкое, поэтому ограничимся лишь иллюстрацией его идеи. Во-первых, в формулировке теоремы не уточнено, о какой сходимости идет речь (по вероятности, поточечной, равномерной и т.д.), поскольку рассматриваются распределения на конечных множествах и все виды сходимости равносильны. Во-вторых, из эргодичности марковской цепи и существования стационарного распределения следует, что сходиться она может только к нему. В-третьих, легко проверить, что если обозначить $\Delta = \max_Y E(X,Y) - \min_Y E(X,Y)$ и $\delta_T = \min_{Y,v} p\left(y_v | Y^{V\setminus\{v\}}, X\right)$, то

$$0 < \frac{e^{-\frac{\Delta}{T}}}{q} \leq \frac{1}{1 + (q-1)e^{\frac{\Delta}{T}}} \leq \delta_T \leq \frac{1}{q}$$

и если от i-го до j-го шага пройдены все вершины графа, то для любой пары полей Y и Y' $p\left(Y^{(j)} = Y | Y^{(i-1)} = Y', X\right) \geq (\delta_T)^S$. После чего сходимость, к тому же с явно оцененной скоростью, вытекает из следующей леммы, доказываемой почти непосредственным вычислением:

Лемма 9 *Для любой пары начальных полей Y' и Y'' и любого поля Y*

$$\left| p\left(Y^{(i)} = Y | Y^{(0)} = Y', X\right) - p\left(Y^{(i)} = Y | Y^{(0)} = Y'', X\right) \right| \leq \left(1 - (q\delta_T)^S\right)^{\left\lfloor \frac{i}{K} \right\rfloor} \leq \left(1 - e^{-\frac{S\Delta}{T}}\right)^{\left\lfloor \frac{i}{K} \right\rfloor}. \quad (334)$$

Выборку Гиббса можно строить не только для постоянного распределения Гиббса (327), но и для последовательности распределений: на i-м шаге значение $y_{v^{(i)}}^{(i)}$ генерируется по условному распределению (332) i-го распределения (327). В частности, для последовательности распределений

$$p\left(Y | T_i, X\right) = \frac{1}{Z(T_i, X)} \; e^{-\sum\limits_{C \in \mathbf{F}} \frac{E_C(X, Y_C)}{T_i}} \quad (335)$$

с температурой $T_i \to 0$ при $i \to \infty$ получается имитация отжига. Наличие количественной оценки скорости сходимости в доказательстве теоремы 14 позволяет построить "расписание отжига" (т.е. последовательность T_i), гарантирующее сходимость к минимуму энергии (331) при любом начальном поле $Y^{(0)}$.

Поскольку минимум энергии может достигаться в нескольких точках, понятие сходимости к минимуму необходимо уточнить. Обозначим через $\pi_0(X)$ распределение вероятностей на пространстве полей Y, сосредоточенное на множестве точек глобального минимума энергии (331) и равномерное на нем (напоминаем, что пространство полей конечно). Будем также использовать обозначения, введенные в обсуждении теоремы 14.

Теорема 15 (S.Geman, D.Geman ([44])) *Для любой последовательности температур $T_i \to 0$, для которой найдется такое число i_0, что $T_i \geq \frac{S\Delta}{\ln\left\lfloor \frac{i}{K} \right\rfloor}$ при $i > i_0$, и для любого начального поля $Y^{(0)}$ распределение $p\left(Y^{(i)}|Y^{(0)}, X\right)$ при $i \to \infty$ сходится к распределению $\pi_0(X)$.*

Доказательство этой теоремы также опустим из-за его громоздкости. Легко заметить, что последовательность температур подобрана так, чтобы произведение в правой части аналога неравенства (334) для переменной температуры стремилось к нулю. Внимательно проследив доказательство, приведенное в [44], можно получить еще и явную оценку скорости сходимости.

Теорема 15, казалось бы, позволяет гарантированно находить минимум энергии (331), т.е. строить по изображению X наилучшее поле ответов Y. На самом деле при гарантирующем сходимость расписании отжига температура убывает безумно медленно, и для сходимости требуется неприемлемо большое число шагов. На практике распознавание изображений посредством имитации отжига с помощью выборки Гиббса применяется успешно, но расписание отжига приходится подбирать экспериментально.

Некоторым утешением может служить то, что порождение выборки Гиббса, как и покоординатная минимизация энергии, очень хорошо распараллеливается. Действительно, порядок обхода вершин графа может быть произвольным, а на каждом шаге построения выборки нужно знать только значения поля в вершинах, соседних с некоторой. Поэтому если организовать обход так, чтобы подряд шли большие массивы попарно не соседних вершин, то шаг во всех вершинах каждого такого массива можно делать одновременно.

D.3.2 Оценка распределения поля в точке

В задаче поиска подозрительных областей (стр. 214), а также при некоторых методах обучения CRF, нужно научиться оценивать вероятности $p(y_v|X)$. В отличие от условных вероятностей $p\left(y_v|Y_{V\setminus\{v\}}, X\right)$ (332), вычисляемых легко, вычисление вероятностей[66] $p(y_v|X)$ требует такого же неприемлемо трудоемкого суммирования, как и вычисление нормировочного множителя $Z(X)$ (330):

$$p(y_v|X) = \frac{1}{Z(w,X)} \sum_{Y_{V\setminus\{v\}}} e^{-(w,E_{X,Y})} .$$

[66] В англоязычной литературе совместное распределение подмножества случайных величин (в данном случае — значения y_v случайного поля в одной вершине) называется *marginal distribution* относительно совместного распределения всего множества случайных величин (в данном случае — случайного поля Y). Название произошло от обработки статистических таблиц вручную, когда распределение одной из двух зависимых случайных величин вычисляли суммированием таблицы их совместного распределения по строке (или по столбцу) и писали на полях (margin!) таблицы. В русскоязычной литературе устоявшегося термина нет; иногда используется транслитерация "маргинальное распределение", иногда — перевод "граничное распределение".

Но из теоремы 14 следует, что для выборки Гиббса с любым начальным полем $Y^{(0)}$ распределения $p\left(y_v^{(i)}|X\right)$ сходятся к искомым вероятностям $p\left(y_v|X\right)$. Значит вероятность $p\left(y_v|X\right)$ можно оценить как частоту появления значения y_v в выборке Гиббса, если только не попалась особенно неудачная выборка:

Теорема 16 *С вероятностью 1 для выборки Гиббса* $Y^{(i)}$

$$p\left(y_v|X\right) = \lim_{n\to\infty} \frac{\#\{i \le n : y_v^{(i)} = y_v\}}{n} \ .$$

Аккуратное выведение теоремы 16 из теоремы 14, а также выведение из леммы 9 количественной оценки скорости сходимости (типа "для любых $\epsilon > 0$ и $\eta > 0$ и $n > N(\epsilon,\eta)$ с вероятностью не менее $1-\eta$ выполняется неравенство $\left|\frac{\#\{i\le n: y_v^{(i)}=y_v\}}{n} - p\left(y_v|X\right)\right| < \epsilon$", где $N(\epsilon,\eta)$ предъявляется явно) являются умеренно сложным техническим **упражнением**.

Метод Монте-Карло для марковских цепей (MCMC, Markov Chain Monte Carlo)

Теорема 16 для выборки Гиббса является частным случаем *метода Монте-Карло для марковских цепей* (*MCMC*, *Markov chain Monte Carlo*), который состоит в следующем. Пусть имеется эргодическая марковская цепь с пространством состояний $\mathcal{Y}'$ и стационарным распределением π. Тогда для любой интегрируемой по распределению π функции f и почти любой реализации $y^{(1)},\dots,y^{(t)},\dots$ марковской цепи ожидание функции по пространству равно ее среднему по цепи, т.е.

$$\mathbf{E}_{\pi;y} f(y) = \int f(y) d\pi(y) = \lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} f(y^{(i)}) \ . \tag{336}$$

Замечание. Как и для оценки вероятностей в теореме 16, при наличии количественных оценок эргодичности марковской цепи можно получить оценку длины выборки Гиббса, требующейся для оценки интеграла с заданной точностью и надежностью. Метод MCMC бывает применим и тогда, когда доказательства теорем 14–16 не проходят, например, для выборок Гиббса для CRF с непрерывным пространством состояний $\mathcal{Y}$. Точные формулировки и обоснования этих голословных утверждений чересчур громоздки и поэтому опущены.

Замечание. Метод MCMC для выборки Гиббса применим и для оценки распределений вероятностей $p\left(Y_R|X\right)$ ограничений поля Y на небольшие подмножества $R \subset V$ вершин графа, например, на пары вершин, соединенные ребрами, или на клики. Но поле на r-элементном множестве может принимать q^r значений, поэтому и количество обходов всех вершин в выборке Гиббса, и объем требуемой памяти должны расти с ростом r как q^r, так что для практической применимости метода MCMC подмножества R должны быть действительно весьма небольшими.

D.3.3 Обучение CRF

Обучение CRF вида (329), как правило, производится максимизацией апостериорной вероятности вектора параметров w при наличии какого-либо априорного распределения с плотностью $p_0(w)$ и обучающего набора, состоящего из изображений $\mathbf{X} = (\mathbf{X}_1,\dots,\mathbf{X}_N)$ и соответствующих им полей ответов $\mathbf{Y} = (\mathbf{Y}_1,\dots,\mathbf{Y}_N)$, причем пары (изображение, поле ответов) считаются независимыми, т.е. максимизируется выражение

$$p(w|\mathbf{X},\mathbf{Y}) \propto p(w,\mathbf{Y}|\mathbf{X}) = p_0(w) p\left(\mathbf{Y}|w,\mathbf{X}\right) = p_0(w)\prod_{i=1}^{N} p\left(\mathbf{Y}_i|w,\mathbf{X}_i\right) \to \max_w \ .$$

Как обычно, если никакого разумного априорного распределения нет, полагается $p_0(w) = 1$, и максимизация апостериорной вероятности превращается в максимизацию правдоподобия

$$p\left(\mathbf{Y}|w, \mathbf{X}\right) \to \max_w \ .$$

Семейство распределений $p\left(\mathbf{Y}|w, \mathbf{X}\right) = \prod_{i=1}^{N} p\left(\mathbf{Y}_i|w, \mathbf{X}_i\right)$ также является семейством распределений Гиббса (329) на несвязном объединении $G(\mathbf{Y}_1) \sqcup \ldots \sqcup G(\mathbf{Y}_N)$ с нормировочным множителем $Z(w, \mathbf{X}) = \prod_{i=1}^{N} Z(w, \mathbf{X}_i)$.

Чтобы соответствовать введенной в разделе 1.2.6 терминологии для обучения дискриминантных вероятностных моделей вообще, введем обозначения $\psi(w) = -\ln p_0(w)$ и $L(w|\mathbf{X}, \mathbf{Y}) = \ln p\left(\mathbf{Y}|w, \mathbf{X}\right)$ и будем формулировать обучение CRF как минимизацию функции потерь, равной, с точностью до прибавления константы, минус логарифму апостериорной вероятности:

$$\begin{aligned} E(w, \mathbf{X}, \mathbf{Y}) &= -\ln p(w, \mathbf{Y}|\mathbf{X}) = \psi(w) - L(w|\mathbf{X}, \mathbf{Y}) \\ &= \psi(w) + \ln Z(w, \mathbf{X}) + (w, E_{\mathbf{X},\mathbf{Y}}) \to \min_w \end{aligned} \tag{337}$$

(ср. со стр. 31).

Если функция $\psi(w)$ выпукла (в частности, если априорное распределение $p_0(w)$ — гауссово, лапласово или равномерное), то минимизационная задача (337) выпукла. Это утверждение является обобщением утверждения о выпуклости обучения линейной логической регрессии (86), в которое обучение CRF превращается, если в графе нет ребер, т.е. все клики одноэлементны, причем, хотя CRF заметно сложнее логистической регрессии, доказательства совпадают дословно.

Предложение 45 *Функция $-L(w|\mathbf{X}, \mathbf{Y}) = \ln Z(w, \mathbf{X}) + (w, E_{\mathbf{X},\mathbf{Y}})$ (минус логарифм правдоподобия для семейства CRF (329)) выпукла по w.*

Доказательство. Достаточно доказать выпуклость функции $\ln Z(w, \mathbf{X})$, от которой $-L(w|\mathbf{X}, \mathbf{Y})$ отличается на линейное по w слагаемое. Для этого достаточно доказать выпуклость ограничения функции $\ln Z(w, \mathbf{X})$ на любую прямую $w(\lambda) = w^0 + \lambda w^1$, а для этого достаточно проверить, что вторая производная $\frac{d^2}{d\lambda^2} \ln Z(w(\lambda), \mathbf{X})$ положительна. А это проверить легко. Обозначим через $[Y]$ множество всех полей ответов, определенных в вершинах графа $G(Y)$. Тогда

$$\frac{d}{d\lambda} \ln Z(w(\lambda), \mathbf{X}) = \frac{-1}{Z(w(\lambda), \mathbf{X})} \sum_{Y \in [\mathbf{Y}]} (w^1, E_{\mathbf{X},Y}) e^{-(w(\lambda), E_{\mathbf{X},Y})}$$

и

$$\begin{aligned} \frac{d^2}{d\lambda^2} \ln Z(w(\lambda), \mathbf{X}) = \frac{1}{\left(Z(w(\lambda), \mathbf{X})\right)^2} \left(\sum_{Y \in [\mathbf{Y}]} (w^1, E_{\mathbf{X},Y}) e^{-(w(\lambda), E_{\mathbf{X},Y})} \right)^2 \\ + \frac{1}{Z(w(\lambda), \mathbf{X})} \sum_{Y \in [\mathbf{Y}]} (w^1, E_{\mathbf{X},Y})^2 e^{-(w(\lambda), E_{\mathbf{X},Y})} \geq 0 \ . \end{aligned}$$

□

Выпуклые минимизационные задачи хороши тем, что у них нет локальных минимумов, отличных от глобального, и любой метод градиентного спуска (см. раздел 3.2.4) сходится к глобальному минимуму. Градиент функции потерь (337) легко выписывается (в предположении, что градиент функции $\psi(w) = -\ln p_0(w)$ легко выписывается)

$$\begin{aligned}
\frac{\partial E(w, \mathbf{X}, \mathbf{Y})}{\partial w} &= \frac{\partial \psi(w)}{\partial w} + \frac{\partial \ln Z(w, \mathbf{X})}{\partial w} + E_{\mathbf{X},\mathbf{Y}} \\
&= \frac{\partial \psi(w)}{\partial w} - \frac{1}{Z(w, \mathbf{X})} \sum_{Y \in [\mathbf{Y}]} E_{\mathbf{X},Y} e^{-(w, E_{\mathbf{X},Y})} + E_{\mathbf{X},\mathbf{Y}} \\
&= \frac{\partial \psi(w)}{\partial w} - \sum_{Y \in [\mathbf{Y}]} E_{\mathbf{X},Y} p\left(Y | w, \mathbf{X}\right) + E_{\mathbf{X},\mathbf{Y}} \\
&= \frac{\partial \psi(w)}{\partial w} + \sum_{i=1}^{N} \left(E_{\mathbf{X}_i,\mathbf{Y}_i} - \sum_{Y \in [\mathbf{Y}_i]} E_{\mathbf{X}_i,Y} p\left(Y | w, \mathbf{X}_i\right) \right) \\
&= \frac{\partial \psi(w)}{\partial w} + \sum_{i=1}^{N} \sum_{m=1}^{M} w_m \sum_{C \in \mathcal{C}(G(\mathbf{Y}_i)) \cap \mathcal{C}_m} \\
&\qquad \left(E_m((\mathbf{Y}_i)_C, (\mathbf{X}_i)_{m,C}) - \sum_{Y \in \mathcal{Y}^C} E_m(Y, (\mathbf{X}_i)_{m,C}) p\left(Y | w, \mathbf{X}_i\right) \right) .
\end{aligned} \tag{338}$$

Сумма по всевозможным полям на клике C в формуле (338) является ожиданием энергии $E_m(Y_C, (\mathbf{X}_i)_{m,C})$ по распределению Гиббса $p\left(Y | w, \mathbf{X}_i\right)$. Все эти ожидания можно вычислять приближенно методом MCMC (336), строя для каждого элемента обучающего набора выборку Гиббса и усредняя энергии по ней.

Таким образом имеется все, что нужно для успешного градиентного обучения CRF: способ оценки градиента функции потерь по параметрам и выпуклость функции потерь, гарантирующая сходимость к глобальному минимуму. К сожалению, такой способ обучения на практике довольно медленен, поскольку для каждого шага градиентного спуска для каждого элемента обучающего набора нужно строить довольно длинную выборку Гиббса. Несколько ускорить вычисления можно, распараллеливая вычисление градиентов по элементам обучающего набора или наоборот, применяя стохастический градиентный спуск (стр. 83).

Существенно более быстрыми, чем стохастический метод MCMC, на практике оказались вариационные методы обучения CRF основанные на приближенной оценке нормировочного множителя $Z(w, \mathbf{X})$ или, что равносильно, свободной энергии $-\ln Z(w, \mathbf{X})$, и *локальных распределений* — вероятностей $p\left(Y_R | w, \mathbf{X}\right)$ ограничений полей ответов на отдельные вершины, клики или другие небольшие множества вершин R путем представления их в качестве решений некоторых экстремальных задач.

D.3.4 Оценки свободной энергии

В этом разделе мы для краткости почти всегда будем опускать указание зависимости распределений CRF от изображения X, которое можно считать зафиксированным.

Продолжим и обобщим начатое на стр. 227 напоминание терминологии статистической физики. Пусть имеется система из S частиц, каждая из которых может находиться в каком-то состоянии из конечного множества $\mathcal{Y}$. Тогда состояние всех системы описывается полем $Y = (y_1, \ldots, y_S)$. Каждому состоянию

(т.е. полю) Y приписана некоторая энергия $E(Y)$. Системе в целом приписано также некоторое распределение вероятностей состояний $p(Y)$. Тогда

- *средней энергией* $U(p)$ системы называется ожидание энергии состояния

$$U(p) = \mathbf{E}_{p;Y} E(Y) = \sum_Y p(Y)E(Y) \ ; \tag{339}$$

- *энтропией* $H(p)$ называется обычная энтропия распределения p

$$H(p) = \mathbf{E}_{p;Y} \left(-\ln p(Y)\right) = -\sum_Y p(Y) \ln p(Y) \ ; \tag{340}$$

- *свободной энергией* $F(p)$ системы называется разность

$$F(p) = U(p) - H(p) = \sum_Y p(Y)(E(Y) + \ln p(Y)) \ . \tag{341}$$

Прямое отношение к свободной энергии имеет распределение Гиббса:

Предложение 46 *Распределение Гиббса* $p(Y) = \frac{1}{Z}\ e^{-E(Y)}$ *является точкой глобального минимума свободной энергии* $F(p)$ *по всем распределениям, а минимальное значение свободной энергии равно* $-\ln Z$.

Доказательство. Поиск критической точки (единственной!) лагранжиана

$$L(p, \lambda) = F(p) + \lambda \left(\sum_Y p(Y) - 1 \right) = \sum_Y p(Y) \left(E(Y) + \ln p(Y) + \lambda \right) - \lambda \ ,$$

выписывание свободной энергии в ней и проверка положительной определенности ее гессиана являются простым **упражнением**. □

Для CRF аналогами частиц являются вершины графа G, аналогами состояний частиц — значения y_i поля в вершине, а для распределений Гиббса из семейства (328) свободная энергия равна $-\ln Z(w, X)$. Точное вычисление этой свободной энергии, как неоднократно отмечалось выше, экспоненциально медленно по количеству частиц и, тем самым, практически невозможно. Но в статистической физике разрабатывались методы приближенной оценки свободной энергии распределений Гиббса и некоторые из них оказались вполне применимы для CRF.

Приближение распределения Гиббса с помощью минимизации свободной энергии

Приближенное вычисление распределения Гиббса путем минимизации свободной энергии по всем распределениям невозможно по такой же причине, что и прямое вычисление $Z(w, X)$ — потому, что размерность пространства распределений, равная $q^S - 1$, неприемлемо велика. Но можно пытаться минимизировать свободную энергию по подпространствам распределений приемлемой размерности и надеяться, что получится достаточно хорошее приближение. Или даже минимизировать "аппроксимацию свободной энергии" по подпространствам "не совсем распределений". Несколько вариантов построения таких подпространств и аппроксимаций, успешно применявшихся в статистической физике, мы сейчас кратко опишем, а затем один из них рассмотрим подробно.

Аппроксимация усредненного поля (*mean field approximation*) в статистической физике состоит в том, что вместо системы нескольких взаимодействующих частиц, расположенных в вершинах графа, рассматривается несколько независимых систем, каждая из которых состоит из одной из частиц в поле, определяемом усреднением по состояниям остальных частиц. Пространство независимых распределений в вершинах, т.е. совместных распределений вида $p(Y) = \prod_{j=1}^{S} p(y_j)$, имеет вполне приемлемую размерность $(q-1)S$. Для таких распределений и энергии (331) свободная энергия имеет вид

$$\begin{aligned} F(p_1, \ldots, p_S) &= \sum_Y \prod_{j=1}^{S} p_j(y_j) \left(\sum_{C \in \mathbf{F}} E_C(Y_C) + \sum_{j=1}^{S} \ln p_j(y_j) \right) \\ &= \sum_{C \in \mathbf{F}} \sum_{Y: C \to \mathcal{Y}} E_C(Y) \prod_{j: v_j \in C} p_j(y_j) + \sum_{j=1}^{S} \sum_{y \in \mathcal{Y}} p_j(y) \ln p_j(y) \ . \end{aligned}$$

Если зафиксировать все распределения, кроме j-го, то минимизация свободной энергии по p_j сводится к задаче

$$\sum_{C \in \mathbf{F}: C \ni v_j} \sum_{Y: C \to \mathcal{Y}} E_C(Y) \prod_{k: v_k \in C} p_k(y_k) + \sum_{y \in \mathcal{Y}} p_j(y_j) \ln p_j(y_j) \to \min_{p_j} \ ,$$

которая решается явно (**упражнение!**):

$$p_j(y_j) = \frac{1}{Z_j} \ e^{-\left(\sum_{C \in \mathbf{F}: C \ni v_j} \sum_{Y: C \setminus \{v_j\} \to \mathcal{Y}} E_C(y_j, Y) \prod_{k: v_k \in C \setminus \{v_j\}} p_k(y_k) \right)} \ ,$$

где нормировочный множитель Z_j, естественно, равен сумме экспонент в правой части по всем q возможным значениям $y_j \in \mathcal{Y}$. Выражение в больших скобках в показателе экспоненты имеет смысл энергии взаимодействия j-й частицы с усредненным полем (mean field) в j-й вершине. Начав с каких-то, например, равномерных распределений p_j и, как и при построении выборки Гиббса, многократно обходя все вершины графа, на каждом шаге можно уменьшать свободную энергию.

К сожалению, при этом сходимость свободной энергии к глобальному, а не к локальному минимуму никак не гарантирована. Более того, даже и глобальный минимум и реализующие его распределения $p_j(y_j)$, могут быть совсем не похожи на распределения $p(y_j)$ в отдельных вершинах, получающиеся из аппроксимируемого Гиббса $p(Y)$, и его свободную энергию просто оттого, что минимум ищется в слишком узком классе распределений, в частности, намного меньшем, чем класс условных распределений (326), определяющих распределение Гиббса.

Аппроксимация Бете: анонс. Заметим, что для любого распределения p и энергии (331) в вычислении средней энергии

$$U(p) = \sum_Y p(Y) E(Y) = \sum_{C \in \mathbf{F}} \sum_{Y: C \to \mathcal{Y}} E_C(Y) p(Y) \tag{342}$$

участвуют только вероятности ограничений поля на клики. Возникает естественное желание минимизировать свободную энергию не по очень большому пространству всех распределений на всем графе, а по пространству наборов распределений $\mathcal{P}_{\mathbf{F}} = \{p_C : C \in \mathbf{F}\}$ на кликах. В типичном случае, когда, как

в семействе (328) имеются клики M типов, содержащие не более K вершин каждая и клики каждого типа переводятся друг в друга какими-то преобразованиями графа (например, для двумерной прямоугольной решетки $M = 3$ — вершины, вертикальные ребра и горизонтальные ребра — и $K = 2$), количество клик каждого типа не превосходит количества вершин S и размерность пространства таких наборов не превосходит $(q^K - 1)MS$, что вполне приемлемо.

Однако не всякое семейство $\mathcal{P}_{\mathbf{F}}$ таких локальных распределений p_C на полях на кликах получается из какого-то глобального распределения p на полях на всем графе. Некоторые необходимые условия очевидны: для любой пары клик $C', C'' \in \mathbf{F}$ с непустым пересечением $C' \cap C''$ и непустого подмножества C этого пересечения (подмножество пересечения клик, конечно же, является кликой, но не обязательно принадлежит семейству $\mathbf{F}$) распределение на C должно быть определено корректно:

$$p_C(Y_C) = \sum_{Y_{C'\setminus C}} p_{C'}(Y_C, Y_{C'\setminus C}) = \sum_{Y_{C''\setminus C}} p_{C''}(Y_C, Y_{C''\setminus C}) \ . \qquad (343)$$

Но для графов с циклами есть еще и более сложные условия согласованности.

Пример. ([81]) Граф G состоит из трех вершин 1, 2 и 3, соединенных в треугольник, множество клик $\mathbf{F}$ — из трех пар вершин, пространство ответов $\mathcal{Y}$ двухэлементно, так что каждое совместное распределение $p_{i,j}$ на паре вершин $\{i, j\}$ естественно записывается в виде матрицы 2×2 с положительными элементами и единичной суммой. Возьмем следующие (не зависящие от X) распределения на парах вершин:

$$p_{1,2} = \begin{pmatrix} 0.4 & 0.1 \\ 0.1 & 0.4 \end{pmatrix} \qquad p_{2,3} = \begin{pmatrix} 0.4 & 0.1 \\ 0.1 & 0.4 \end{pmatrix} \qquad p_{1,3} = \begin{pmatrix} 0.1 & 0.4 \\ 0.4 & 0.1 \end{pmatrix}$$

и равномерные распределения p_1, p_2 и p_3 в отдельных вершинах. Тогда в каждой вершине условие (343) выполнено. А теперь предположим, что все локальные распределения $p_{i,j}$ получаются из некоторого глобального распределения p. Все $(2^3 = 8)$-элементное множество $\mathcal{Y}^{\{1,2,3\}}$ полей на графе покрывается объединением трех 4-элементных множеств: $\mathcal{Y}^{1\neq 2} = \{Y : y_1 \neq y_2\}$, $\mathcal{Y}^{2\neq 3} = \{Y : y_2 \neq y_3\}$ и $\mathcal{Y}^{1=3} = \{Y : y_1 = y_3\}$, но

$$\begin{aligned} p(\mathcal{Y}^{1\neq 2}) + p(\mathcal{Y}^{2\neq 3}) + p(\mathcal{Y}^{1=3}) &= p_{1,2}(\mathcal{Y}^{1\neq 2}) + p_{2,3}(\mathcal{Y}^{2\neq 3}) + p_{1,3}(\mathcal{Y}^{1=3}) \\ &= (0.1 + 0.1) + (0.1 + 0.1) + (0.1 + 0.1) = 0.6 < 1 \ . \end{aligned}$$

Противоречие.

Таким образом, пространство наборов локальных распределений, хотя и приемлемо по размерности, но довольно сложно устроено. В работе [13] Ханс Бете впервые применил подход, в современных терминах выглядящий так: ограничиться условиями локальной согласованности (343) (к тому же только самыми простыми из них — для одноточечных множеств C), определить для семейств локальных распределений $\mathcal{P}_{\mathbf{F}}$ энтропию и свободную энергию, в хороших случаях совпадающих с энтропией и, соответственно, свободной энергией глобального распределения p и минимизировать эту аппроксимацию свободной энергию по относительно простому пространству локально согласованных семейств локальных распределений. В отличии от аппроксимации усредненного поля, дающей оценку свободной энергии сверху, аппроксимация Бете гарантированной односторонней оценки не дает.

Такие семейства локальных вероятностных распределений $\mathcal{P}_{\mathbf{R}}$ на ограничениях полей на некотором семействе подмножеств $\mathbf{R}$ вершин графа, удовлетворяющих некоторым условиям локальной согласованности типа (343), но не обязательно получающихся из какого-то распределения на всех полях, называются *довериями* (*belief*).

Замечание. Смысл этого термина не совпадает со смыслом одноименного термина belief, введенного на стр. 30.

Подход Бете и его аппроксимация свободной энергии оказались весьма успешными и в статистической физике, и, впоследствии, в анализе изображений. Во-первых, для минимизации аппроксимации Бете были найдены довольно быстрые алгоритмы. Во-вторых, распределения на кликах — это в точности то, что нужно для вычисления градиента по формуле (338) при градиентном обучении CRF и достаточно для оценки распределения в вершинах (пункт 3, стр. 228). А то, что эти доверия могут не соответствовать никакому глобальному распределению вероятностей, на практике оказалось не очень существенным.

Аппроксимация Бете уже разрекламирована, но еще не предъявлена. Ниже предъявляется более общая конструкция аппроксимации свободной энергии, содержащая аппроксимацию Бете как частный случай, а затем на стр. 239 аппроксимация Бете рассматривается подробно.

Аппроксимация Кикучи: анонс. В работе [65] Риоичи Кикучи (Ryoichi Kikuchi) предложил аппроксимацию свободной энергии, более точную, чем аппроксимация Бете. А именно, он предложил рассматривать удовлетворяющие условиям локальной согласованности, аналогичным (343), семейства доверий $\mathcal{P}_{\mathbf{R}}$ для семейств "областей" — множеств $\mathbf{R}$, уже не обязательно являющихся кликами, а, возможно, несколько бóльших. Каждому множеству $R \in \mathbf{R}$ приписывается средняя энергия доверия (т.е. вероятностного распределения) p_R, равная сумме средних энергий содержащихся в R клик

$$U(p_R) = \sum_{Y:R\to\mathcal{Y}} p_R(Y) \sum_{C\subset R} E_C(Y_C) \ , \tag{344}$$

и энтропия

$$H(p_R) = -\sum_{Y:R\to\mathcal{Y}} p_R(Y) \ln p_R(Y) \ ,$$

а значит и свободная энергия $F(p_R) = U(p_R) - H(p_R)$. Аппроксимация Кикучи $F_{\mathbf{R}}(\mathcal{P}_{\mathbf{R}})$ свободной энергии семейства доверий $\mathcal{P}_{\mathbf{R}}$ имеет вид

$$F_{\mathbf{R}}(\mathcal{P}_{\mathbf{R}}) = \sum_{R\in\mathbf{R}} c_R F(p_R) \ , \tag{345}$$

где целочисленные коэффициенты c_R для максимальных (т.е. не входящих ни в какие другие) множеств семейства $\mathbf{R}$ равны 1, а для остальных подбираются по формуле включений-исключений, чтобы каждое слагаемое из правой части формулы (342) (средняя энергия клики) входило с коэффициентом 1, т.е. чтобы средняя энергия вычислялась точно.

Пример. Пусть граф G — квадратная решетка (рис. 18), а энергия равна сумме энергий по всем возможным кликам, т.е. по вершинам и парам вершин на ребрах, например, определяется распределением Гиббса (322). В качестве семейства множеств $\mathbf{R}$ можно взять все клики (вершины и ребра) и все квадраты решетки (т.е. строго говоря, четверки вершин, являющиеся вершинами элементарного квадрата решетки). Семейство доверий $\mathcal{P}_{\mathbf{R}}$ однозначно определяется доверияими на квадратах, удовлетворяющими условиям согласованности (343) на ребрах и в вершинах, по которым они пересекаются. Для любого квадрата средняя энергия (344) получается суммированием по четырем его ребрам и четырем вершинам, для ребра — по нему самому и по двум его вершинам, а для вершины, естественно, только по ней самой. Легко видеть, что при коэффициентах c_R в формуле (345), равных 1 для квадратов и вершин и -1 для ребер, средняя энергия вычисляется правильно.

Аппроксимация Бете получается как частный случай аппроксимации Кикучи[67], когда семейство множеств $\mathbf{R}$ состоит из клик множества $\mathbf{F}$, участвующих в вычислении энергии (342), и вершин графа, т.е. при $\mathbf{R} = \mathbf{F} \cup V$. Условия согласованности (343) при этом накладываются только в вершинах[68]. При рассмотрении аппроксимации Бете удобно отождествлять одноэлементные клики $\{v\}$ с вершинами v, распределения $p_{\{v\}}$ с распределениями p_v и считать, что в каждой вершине определена энергия $E_v(\cdot) = E_{\{v\}}(\cdot)$ (а если не была определена, положить ее равной 0), а через $\mathbf{F}' = \mathbf{F} \setminus V$ обозначать множество неодноэлементных клик в формуле (342). Тогда семейство доверий $\mathcal{P}_{\mathbf{F}}$ однозначно определяется своим подсемейством $\mathcal{P}_{\mathbf{F}'}$ Обозначим $n_v = \#\{C \in \mathbf{F}' : C \ni v\}$. Тогда коэффициент $c_{\{v\}}$ в формуле (345) равен $1 - n_v$ и *свободная энергия Бете* равна

$$\begin{aligned} F_{\mathbf{F}}(\mathcal{P}_{\mathbf{F}}) &= \sum_{C\in\mathbf{F}'} \left(U(p_C) + \sum_{v\in C} U(p_v) - H(p_C) \right) + \sum_{v\in V} (1-n_v)\,(U(p_v) - H(p_v)) \\ &= \sum_{C\in\mathbf{F}'} \sum_{Y_C : C\to\mathcal{Y}} p_C(Y)\,(E_C(Y_C) + \ln p_C(Y_C)) \\ &\quad + \sum_{v\in V} \sum_{y\in\mathcal{Y}} p_v(y)\,(E_v(y) + (1-n_v)\ln p_v(y)) \qquad (346) \\ &= \left(\sum_{C\in\mathbf{F}} U(p_C) \right) - \left(\sum_{C\in\mathbf{F}'} H(p_C) + \sum_{v\in V} (1-n_v) H(p_v) \right) \\ &= U(\mathcal{P}_{\mathbf{F}}) - H_{\mathbf{F}}(\mathcal{P}_{\mathbf{F}})\ , \qquad (347) \end{aligned}$$

где $U(\mathcal{P}_{\mathbf{F}})$ совпадает со средней энергией (339), а $H_{\mathbf{F}}(\mathcal{P}_{\mathbf{F}})$ называется *энтропией Бете*.

Применение аппроксимации Кикучи для анализа изображений подробно описано в статье [125], где показано, что технически это не намного сложнее, чем применение аппроксимации Бете, а результаты получаются лучше. Мы, однако, ограничимся подробным описанием более наглядной и менее громоздкой аппроксимации Бете.

Аппроксимация Бете бывает точной.

Предложение 47 *Для распределения Гиббса на ациклическом графе*

$$p(Y) = \prod_{C\in\mathbf{F}'} \frac{p_C(Y_C)}{\prod_{v\in C} p_v(y_v)} \prod_{v\in V} p_v(y_v) = \prod_{C\in\mathbf{F}'} p_C(Y_C) \prod_{v\in V} (p_v(y_v))^{1-n_v}\ . \qquad (348)$$

Доказательство. Сначала предположим, что граф G связен, т.е. является деревом. Будем строить расширяющуюся последовательность поддеревьев $G_1 \subset \ldots \subset G_{\#\mathbf{F}} = G$ следующим образом: в качестве G_1 возьмем произвольную вершину $C^{(1)} \in \mathbf{F}'$, а к каждому уже построенному поддереву G_{i-1} будем добавлять очередное ребро $C^{(i)}$, имеющее общую вершину $v^{(i)}$ с G_{i-1} (из ацикличности G следует что ровно одну). Справедливость равенства (348) для ограничений поля Y на поддеревья G_i с кратностями вершин n_v, определяемыми в пределах поддерева, легко проверяется по индукции: база индукции тривиальна, а шаг следует из равенства

$$\frac{p(Y_{G_i})}{p(Y_{G_{i-1}})} = p(Y_{C^{(i)}} | Y_{G_{i-1}}) = p(Y_{C^{(i)}} | y_{v^{(i)}}) = \frac{p_{C^{(i)}}(Y_{C^{(i)}})}{p_{v^{(i)}}(y_{v^{(i)}})}\ .$$

[67] Исторически, конечно же, наоборот: аппроксимация Кикучи появилась как обобщение аппроксимации Бете.

[68] Как правило, аппроксимацию Бете применяют, когда энергия складывается из взаимодействия с внешним полем и попарных взаимодействий, т.е. каждая клика из $\mathbf{F}$ содержит не более двух вершин.

Рассмотрение случая несвязного ациклического графа (т.е. леса) оставляется в качестве тривиального **упражнения**. □

Следствие 8 *Если граф ацикличен, то свободная энергия Бете (346) распределения Гиббса совпадает со свободной энергией (341).*

Доказательство. Непосредственно проверяется, что энтропия (340) для распределения (348) совпадает с энтропией Бете (см. (347)). □

Упражнение. На любом графе максимум энтропии Бете достигается на семействе $\mathcal{P}_{\mathbf{F}}$ равномерно распределенных доверий и совпадает с максимумом энтропии (340). Значит, если средняя энергия (339) мала по сравнению с энтропией (340), то минимум свободной энергии Бете (346) близок к минимуму свободной энергии (341).

Вышеописанные частные случаи, конечно, никак не гарантируют, что минимизация свободной энергии Бете всегда позволяет оценить минимум настоящей свободной энергии, а значит, научиться вычислять распределение Гиббса (пункт 1 на стр. 228). Но попробовать стоит....

Минимизация свободной энергии Бете — это решение минимизационной задачи

$$F_{\mathbf{F}}(\mathcal{P}_{\mathbf{F}}) = \sum_{C\in\mathbf{F}'}\sum_{Y\in\mathcal{Y}^C} p_C(Y)\,(E_C(Y)+\ln p_C(Y)) + \sum_{v\in V}\sum_{y\in\mathcal{Y}} p_v(y)\,(E_v(y)+(1-n_v)\ln p_v(y)) \to \min_{\{p_C(Y),p_v(y)\}} \tag{349}$$

по переменным $p_C(Y)$ (q^k чисел для каждой k-элементной клики $C\in\mathbf{F}'$) и $p_v(y)$ (q чисел для каждой вершины $v\in V$ графа G), удовлетворяющим линейным равенствам и неравенствам

$$p_C(Y)\ge 0 \quad \text{при} \quad C\in\mathbf{F}', Y\in\mathcal{Y}^C \tag{350}$$

$$p_v(y)\ge 0 \quad \text{при} \quad v\in V, y\in\mathcal{Y} \tag{351}$$

$$\sum_{Y\in\mathcal{Y}^C} p_C(Y)=1 \quad \text{при} \quad C\in\mathbf{F}' \tag{352}$$

$$\sum_{y\in\mathcal{Y}} p_v(y)=1 \quad \text{при} \quad v\in V \tag{353}$$

$$\sum_{Y\in\mathcal{Y}^C:y_v=y} p_C(Y)=p_v(y) \quad \text{при} \quad v\in C\in\mathbf{F}', y\in\mathcal{Y}\ , \tag{354}$$

т.е. лежащим в пересечении произведения набора симплексов с плоскостью.

Минимизируемая функция ограничена**(!)** и дифференцируема, область ее определения компактна и выпукла, так что ее можно было бы пытаться минимизировать свободную энергию Бете градиентным спуском. Но мы продемонстрируем другой метод, предложенный в статье [126] и, с теми или иными модификациями, успешно применяющийся. Основная идея состоит в том, чтобы вместо невыпуклой минимизационной задачи решать последовательность выпуклых, для которых почти все методы минимизации хорошо сходятся.

Предложение 48 *Если $f=f^+-f^-$, где f^+ и f^- — выпуклые функции, l_x^- — опорная линейная функция*[69] *к f^- в точке x и $(f^+-l_x^-)(x')<(f^+-l_x^-)(x)$, то $f(x')<f(x)$.*

[69] Т.е. линейная функция, график которой является опорной плоскостью к надграфику выпуклой функции. Не следует путать ее с опорной функцией какого-либо множества.

Доказательство. $f(x') \le (f^+ - l_x^-)(x') < (f^+ - l_x^-)(x) = f(x)$. $\square$

Из этого тривиального предложения и того, что для дифференцируемой выпуклой функции f^- опорная функция l_x^- единственна и равна

$$l_x^-(x') = f^-(x) + (\nabla f^-(x), x' - x) \ ,$$

следует следующий способ минимизации разности гладких выпуклых функций: начиная с начальной точки $x^{(1)}$ последовательно решать выпуклые минимизационные задачи

$$x^{(i+1)} = \arg\min_x f^+(x) - (f^-(x^{(i)}) + (\nabla f^-(x^{(i)}), x - x^{(i)})) \ .$$

Если функция f ограничена и определена на выпуклом компактном множестве, то такая последовательность сходится**(!)** к точке минимума функции f, но, увы, не обязательно к точке глобального минимума. В зависимости от того, как именно представить функцию f в виде разности двух выпуклых, последовательность $x^{(i)}$ может сходиться к разным точкам минимума.

Применим этот способ к минимизации свободной энергии Бете (349), представив ее в виде разности выпуклых**(!)** функций:

$$F^+(\mathcal{P}_\mathbf{F}) = \sum_{C\in\mathbf{F}'} \sum_{Y\in\mathcal{Y}^C} p_C(Y)\,(E_C(Y) + \ln p_C(Y)) + \sum_{v\in V}\sum_{y\in\mathcal{Y}} p_v(y)\,(E_v(y) + \ln p_v(y))$$
$$F^-(\mathcal{P}_\mathbf{F}) = \sum_{v\in V} n_v \sum_{y\in\mathcal{Y}} p_v(y)\ln p_v(y) \ .$$

Обе функции F^+ и F^- дифференцируемы и строго выпуклы, а при превращении какого-либо из неравенств (350) или (351) в равенство производная F^+ по соответствующей переменной равна $-\infty$, что гарантирует отсутствие минимумов F^+ с любой линейной добавкой на границах симплексов и позволяет не следить за условиями (350) и (351) на каждом шаге минимизации F^+. Шаг минимизации F^+ сводится к решению строго выпуклой задачи

$$F^+(\mathcal{P}_\mathbf{F}^{(i+1)}) - \left(F^-(\mathcal{P}_\mathbf{F}^{(i)}) + \left(\nabla F^-(\mathcal{P}_\mathbf{F}^{(i)}), \mathcal{P}_\mathbf{F}^{(i+1)}\right) - \mathcal{P}_\mathbf{F}^{(i)}\right)\right) \to \min_{\mathcal{P}_\mathbf{F}^{(i+1)}}$$

или, что эквивалентно,

$$F^+(\mathcal{P}_\mathbf{F}) - \left(\nabla F^-(\mathcal{P}_\mathbf{F}^{(i)}), \mathcal{P}_\mathbf{F}\right) = F^+(\mathcal{P}_\mathbf{F}) - \sum_{v\in V} n_v \sum_{y\in\mathcal{Y}} p_v(y)\left(1 + \ln p_v^{(i)}(y)\right) \to \min_{\mathcal{P}_\mathbf{F}} \tag{355}$$

при линейных ограничениях (352), (353) и (354). Ее лагранжиан равен

$$\begin{aligned} L(\mathcal{P}_\mathbf{F}, \lambda) = &\sum_{C\in\mathbf{F}'} \sum_{Y\in\mathcal{Y}^C} p_C(Y)\,(E_C(Y) + \ln p_C(Y)) \\ &+ \sum_{v\in V}\sum_{y\in\mathcal{Y}} p_v(y)\left(E_v(y) + \ln p_v(y) - n_v\left(1 + \ln p_v^{(i)}(y)\right)\right) \\ &+ \sum_{C\in\mathbf{F}'} \lambda_C\left(\sum_{Y\in\mathcal{Y}^C} p_C(Y) - 1\right) + \sum_{v\in V}\lambda_v\left(\sum_{y\in\mathcal{Y}} p_v(y) - 1\right) \\ &+ \sum_{C\in\mathbf{F}'}\sum_{v\in C}\sum_{y\in\mathcal{Y}} \lambda_{C,v,y}\left(\sum_{Y\in\mathcal{Y}^C : Y_v = y} p_C(Y) - p_v(y)\right) \ , \end{aligned}$$

где λ — это набор множителей Лагранжа λ_C, λ_v и $\lambda_{C,v,y}$ по одному на каждое условие (352), (353) и (354), соответственно. Приравнивая нулю производные

лагранжиана по доверияm $\mathcal{P}_{\mathbf{F}}$

$$0 = \frac{\partial L(\mathcal{P}_{\mathbf{F}}, \lambda)}{\partial p_C(Y)} = E_C(Y) + \ln p_C(Y) + 1 + \lambda_C + \sum_{v \in C} \lambda_{C,v,y}$$

$$0 = \frac{\partial L(\mathcal{P}_{\mathbf{F}}, \lambda)}{\partial p_v(y)} = E_v(y) + \ln p_v(y) + 1 - n_v \left(1 + \ln p_v^{(i)}(y)\right) + \lambda_v - \sum_{C \ni v} \lambda_{C,v,y} \ ,$$

разрешая полученные равенства относительно доверий $p_C(Y)$ и $p_v(y)$

$$p_C(Y) = e^{-\left(E_C(Y) + 1 + \lambda_C + \sum\limits_{v \in C} \lambda_{C,v,y}\right)} \tag{356}$$

$$p_v(y) = e^{-\left(E_v(y) + 1 - n_v \left(1 + \ln p_v^{(i)}(y)\right) + \lambda_v - \sum\limits_{C \ni v} \lambda_{C,v,y}\right)} \tag{357}$$

и подставляя полученные доверия $p_C(Y)$ и $p_v(y)$ обратно в лагранжиан, получаем двойственный лагранжиан

$$L^*(\lambda) = -\sum_{C \in \mathbf{F}'} \left(\lambda_C + \sum_{Y \in \mathcal{Y}^C} e^{-\left(E_C(Y) + 1 + \lambda_C + \sum\limits_{v \in C} \lambda_{C,v,y}\right)} \right) - \sum_{v \in V} \left(\lambda_v + \sum_{y \in \mathcal{Y}} e^{-\left(E_v(y) + 1 - n_v \left(1 + \ln p_v^{(i)}(y)\right) + \lambda_v - \sum\limits_{C \ni v} \lambda_{C,v,y}\right)} \right) .$$

Как и положено двойственному лагранжиану строго выпуклой задачи, функция $L^*(\lambda)$ строго вогнута и имеет единственную точку глобального максимума, которую и нужно найти. Система уравнений $\nabla L^*(\lambda) = 0$ легко выписывается:

$$0 = \frac{\partial L^*(\lambda)}{\partial \lambda_C} = -1 + \sum_{Y \in \mathcal{Y}^C} e^{-\left(E_C(Y) + 1 + \lambda_C + \sum\limits_{v \in C} \lambda_{C,v,y}\right)}$$

$$0 = \frac{\partial L^*(\lambda)}{\partial \lambda_v} = -1 + \sum_{y \in \mathcal{Y}} e^{-\left(E_v(y) + 1 - n_v \left(1 + \ln p_v^{(i)}(y)\right) + \lambda_v - \sum\limits_{C \ni v} \lambda_{C,v,y}\right)}$$

$$0 = \frac{\partial L^*(\lambda)}{\partial \lambda_{C,v,y}} = \sum_{Y \in \mathcal{Y}^C : y_v = y} e^{-\left(E_C(Y) + 1 + \lambda_C + \sum\limits_{v' \in C} \lambda_{C,v',y}\right)} - e^{-\left(E_v(y) + 1 - n_v \left(1 + \ln p_v^{(i)}(y)\right) + \lambda_v - \sum\limits_{C' \ni v} \lambda_{C',v,y}\right)} ,$$

но не решается аналитически. Однако она разрешима относительно любой переменной при фиксированных остальных:

$$\lambda_C = \ln \sum_{Y \in \mathcal{Y}^C} e^{-\left(E_C(Y) + 1 + \sum\limits_{v \in C} \lambda_{C,v,y}\right)}$$

$$\lambda_v = \ln \sum_{y \in \mathcal{Y}} e^{-\left(E_v(y) + 1 - n_v \left(1 + \ln p_v^{(i)}(y) \right) - \sum_{C \ni v} \lambda_{C,v,y} \right)}$$

$$\lambda_{C,v,y} = \frac{1}{2} \ln \frac{\sum_{Y \in \mathcal{Y}^C : y_v = y} e^{-\left(E_C(Y) + \lambda_C + \sum_{v' \in C, v' \neq v} \lambda_{C,v',y} \right)}}{e^{-\left(E_v(y) - n_v \left(1 + \ln p_v^{(i)}(y) \right) + \lambda_v - \sum_{C' \ni v, C' \neq C} \lambda_{C',v,y} \right)}}$$

т.е. двойственный лагранжиан можно максимизировать по любой одной переменной. Из его строгой вогнутости следует, что последовательная минимизация при многократном обходе всех переменных сходится к искомой точке глобального максимума двойственного лагранжиана. Формулы (356) и (357), примененные в этой точке максимума, дают решение $\mathcal{P}_{\mathbf{F}}^{(i+1)}$ очередного шага (355) минимизации свободной энергии Бете.

Существует много вариантов этого алгоритма, различающихся способами обхода переменных, критериями остановки во внутренних циклах, способами введения множителей Лагранжа и их интерпретации, способом — явным или неявным — представления свободной энергии Бете в виде разности выпуклых функций, см., например, очень изящную формулировку алгоритма в статье [53].

Минимизация свободной энергии Бете: применение. Полученные с помощью этого алгоритма оценки доверий можно использовать при вычислении (минус) градиента логарифма апостериорной вероятности обучающего набора (338) для шага градиентного обучения CRF. (Напоминаем, что не зависящие от X поля в этом разделе рассматривались исключительно для того, чтобы формулы помещались в строки.) Казалось бы, этот способ обучения ничем не лучше метода MCMC, а только хуже: в нем, как и при построении выборки Гиббса нужно многократно обходить все вершины всех полей ответов и в каждой вершине производить какие-то вычисления энергий всех возможных полей на содержащих эту вершину кликах; зато, в отличие от MCMC, нет никаких, даже статистических, гарантий сходимости к правильному ответу. Но на практике оказывается, что обучение минимизацией свободной энергии Бете сходится даже если внутренние циклы (максимизацию двойственного лагранжиана) проходить всего лишь по несколько раз, причем сходится намного быстрее MCMC.

D.3.5 CRF со скрытыми переменными и их обучение

При анализе изображений с помощью вероятностных моделей, в частности, CRF, довольно часто модель описывает не только само растровое изображение, но и какие-то его дополнительные характеристики. Например, MRF (323)–(324) описывает также (предполагаемые) контуры изображения. На применение моделей наличие в них дополнительных переменных никак не влияет. Но обучение моделей с помощью обучающих наборов, не содержащих или не полностью содержащих эти дополнительные характеристики правильных ответов несколько затрудняется. Впрочем, оно полностью укладывается в схему обучения с пропущенными данными из раздела A и может производиться методом максимизации ожидания (EM). Некоторые подробности применения метода максимизации ожидания для обучения CRF приведены в обзоре [107]. Мы же их полностью опустим.

D.4 Историко-литературные ссылки

Марковские случайные поля как обобщение марковских цепей для нужд статистической физики появились в 1960-ые годы ([34]), хотя модель Изинга — намного раньше, в 1920-ые. Теорема Хаммерсли-Клиффорда [47] была доказана в 1971 году, но не опубликована сразу из-за кажущегося излишним требования строгой положительности распределения. Приведенный в статье [87] 1974 года контрпример показал, что оно не излишне. К этому времени уже были опубликованы несколько разных доказательств теоремы.

С работы [44] началось активное применение моделей статистической физики к распознаванию изображений. В частности, в ней впервые были применены выборки Гиббса и доказана сходимость получающегося распределения к распределению Гиббса. Последовавшая за ней лавина работ необозрима. Применению марковских случайных полей для анализа изображений посвящена, например, книга [78], доступная в электронном виде.

Условные случайные поля [71] изначально предназначались для одномерных задач как дискриминантный аналог порождающих скрытых марковских моделей. Для их применения и обучения, как и для порождающих моделей, применимы аналоги алгоритмов "вперед-назад" (а именно, алгоритм распространения доверия [90]), Витерби и Баума-Велша.

Эти же алгоритмы применялись, иногда успешно, и ранее для анализа двумерных изображений с помощью марковских полей, но ни обоснования применимости, ни гарантий сходимости алгоритмов не было. Более того, были реальные примеры их расходимости. Обоснование появилось в статье [124] про связь алгоритма распространения доверия с аппроксимациями свободной энергии Бете и Кикучи, после чего быстро появились и алгоритмы с гарантированной сходимостью ([126], [53] и др.) и более точной аппроксимацией (серия работ [124]–[125] и др.).

Довольно подробное введение в случайные поля с большим количеством ссылок приводится в обзоре [107], а наглядные примеры применения их в задачах анализа изображений — помимо классической работы [44], например, в статьях [70] и [55], а также в необъятном количестве других.

Наряду с исторически сложившейся традицией использования при изучении случайных полей языка, заимствованного из статистической физики, встречается и описание их на стандартном языке теории вероятностей. Например, в книге [15] для моделей более общего вида, чем случайные поля, применимых не только для анализа изображений ([15, глава 8]), на чисто вероятностном языке описаны аналоги выборки Гиббса ([15, глава 11]) и приближенной минимизации свободной энергии ([15, глава 10]).

Список литературы

[1] М.А.Айзерман, Э.М.Браверман, Л.И.Розоноэр. Теоретические основы метода потенциальных функций в задаче об обучении автоматов распределению входных ситуаций на классы. *Автоматика и телемеханика*, 25(6):917—936, 1964.

[2] М.А.Айзерман, Э.М.Браверман, Л.И.Розоноэр. *Метод потенциальных функций в теории обучения машин.* Наука, Москва, 1970.

[3] M.A. Aizerman, E.M. Braverman, and L.I. Rozonoer. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.

[4] N. Aronszajn. Theory of reproducing kernels (`http://www.recognition.mccme.ru/pub/papers/kernels/ReproducingKernels.pdf`). *Trans. Am. Math. Soc.*, 68(3):337–404, 1950.

[5] David Arthur and Sergei Vassilvitskii. How slow is the k-means method? (`http://www.recognition.mccme.ru/pub/papers/clustering/kMeans-socg.pdf`) In *Symposium on Computational Geometry*, pages 144–153, 2006.

[6] N.E. Ayat, M. Cheriet, L. Remaki, and C.Y. Suen. KMOD - A New Support Vector Machine Kernel with Moderate Decreasing for Pattern Recognition. Application to Digit Image Recognition (`http://www.recognition.mccme.ru/pub/papers/SVM/ayat01kmod.pdf`). In *Sixth International Conference on Document Analysis and Recognition*, pages 1215 – 1219, 2001.

[7] Leonard E. Baum and J. A. Eagon. An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology (`http://www.recognition.mccme.ru/pub/papers/EM/baum67inequality.pdf`). *Bull. Amer. Math. Soc.*, 73(3):360–363, 1967.

[8] Leonard E. Baum, Ted Petrie, George Soules, and Norman Weiss. A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains. *The Annals of Mathematical Statistics*, 41(1):164–171, 1970.

[9] Yoshua Bengio and Paolo Frasconi. An Input Output HMM Architecture (`http://www.recognition.mccme.ru/pub/papers/HMM/bengio95input.ps.gz`). In *Advances in Neural Information Processing Systems*, pages 427–434. MIT Press, 1995.

[10] Yoshua Bengio. Markovian Models for Sequential Data (`http://www.recognition.mccme.ru/pub/papers/HMM/Bengio99Markovian.ps.gz`). *Neural Computing Surveys*, 2:129–162, 1999.

[11] K.P. Bennett, A. Demiriz, and J. Shawe–Taylor. A Column Generation Algorithm for Boosting (`http://www.recognition.mccme.ru/pub/papers/boosting/bennett00column.pdf`). In Pat Langley, editor, *Proceedings of Seventeenth International Conference on Machine Learning*, pages 65–72. Morgan Kaufmann, 2000.

[12] C. Berg, J.P.R. Christensen, and P. Ressel. *Harmonic Analysis on Semigroups.* Springer-Verlag, New York, 1984.

[13] H. A. Bethe. Statistical theory of superlattices. *Proc. Roy. Soc. London ser. A*, 150(871):552–575, 1935.

[14] Christopher M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[15] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[16] C.M. Bishop and M.E. Tipping. Variational Relevance Vector Machine (`http://www.recognition.mccme.ru/pub/papers/Bayes/bishop00variational.ps.gz`). In *Processings of the 16-th Conference on Uncertainty in Artificial Intelligence*, pages 46–53. Morgan Kauffman, 2000.

[17] P.E. Böhmer. Theorie der unabhängigen Warscheinlichkeiten. In *Rapports, Memories et Procès-verbaux de Septième Congrès international d'Actuaries*, volume 2, pages 327–343, Amsterdam, 1912.

[18] B.E. Boser, I.M. Guyon, and V.N.Vapnik. A training algorithm for optimal margin classifiers (`http://www.recognition.mccme.ru/pub/papers/SVM/boser92training.ps.gz.ps`). In *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pages 144–152. ACM Press, 1992.

[19] Léon Bottou. Stochastic Learning (`http://www.recognition.mccme.ru/pub/papers/stochastic/mlss-2003.pdf`). In Olivier Bousquet and Ulrike von Luxburg, editors, *Advanced Lectures on Machine Learning*, Lecture Notes in Artificial Intelligence, LNAI 3176, pages 146–168. Springer Verlag, Berlin, 2004.

[20] Leo Breiman, Jerome Friedman, Charles J. Stone, and R. A. Olshen. *Classification and Regression Trees*. Chapman & Hall/CRC, January 1984.

[21] Leo Breiman. Random Forests (`http://www.recognition.mccme.ru/pub/papers/boosting/breiman01random.pdf`). *Machine Learning*, 45(1):5–32, 2001.

[22] Christopher J.C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition (`http://www.recognition.mccme.ru/pub/papers/SVM/burges98tutorial.pdf`). *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[23] F.P. Cantelli. Sulla determinazione empirica della leggi di probabilitá. *Giorn. Ist. Ital. Attuari*, 4:421–424, 1933.

[24] Michael Collins, Robert E. Schapire, and Yoram Singer. Logistic Regression, AdaBoost and Bregman Distances (`http://www.recognition.mccme.ru/pub/papers/boosting/collins00logistic.pdf`). *Machine Learning*, 48(1-3):253–285, 2002.

[25] Corinna Cortes, Patrick Haffner, and Mehryar Mohri. Positive Definite Rational Kernels (`http://www.recognition.mccme.ru/pub/papers/kernels/colt.pdf`). In *In Proceedings of The 16th Annual Conference on Computational Learning Theory (COLT 2003)*, pages 41–56. Springer, 2003.

[26] C. Cortes and V. Vapnik. Support Vector Networks (`http://www.recognition.mccme.ru/pub/papers/SVM/cortes95support.ps.gz.ps`). *Machine Learning*, 20(3):273–297, 1995.

[27] T.M. Cover and P.M. Hart. Nearest Neighbor pattern classification (http://www.recognition.mccme.ru/pub/papers/NN/cover67nearest.pdf). *IEEE Transactions on Information Theory*, 13:21–27, 1967.

[28] D.R. Cox. Some procedures associated with the logistic qualitative response curve. In F.N. David, editor, *Research Papers in Statistics: Festschrift for J. Neyman*, pages 55–71, New York, 1966. Wiley.

[29] D. R. Cox and D. Oakes. *Analysis of Survival Data*. Chapman and Hall, London, 1984.

[30] D. R. Cox. Regression Models and Life-Tables (http://www.recognition.mccme.ru/pub/papers/survival/cox72regression.pdf). *Journal of the Royal Statistical Society. Series B (Methodological)*, 34(2):187–220, 1972.

[31] N. Cristianini and J. Shawe-Taylor. *An Introduction To Support Vector Machines (and other kernel-based learning methods) (*http://www.support-vector.net/references.html*)*. Cambridge University Press, 2000.

[32] Ayhan Demiriz, Kristin P. Bennett, and J. Shawe-Taylor. Linear programming boosting via column generation (http://www.recognition.mccme.ru/pub/papers/boosting/demiriz02linear.pdf). *Machine Learning*, 46(1):225–254, 2002.

[33] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, series B*, 39(1):1–38, 1977.

[34] Р.Л.Добрушин. Описание случайного поля при помощи условных вероятностей и условия его регулярности (http://www.recognition.mccme.ru/pub/papers/CRF/Dobrushin68description.pdf). *ТВП*, 13(2):201–229, 1968.

[35] B. Efron, I. Johnstone, T. Hastie, and R. Tibshirani. Least angle regression (http://www.recognition.mccme.ru/pub/papers/L1/LeastAngle2002.pdf). *Annals of Statistics*, 32(2):407–451, 2004.

[36] R.A. Fisher. The use of multiple measurements in taxonomic problems (http://www.recognition.mccme.ru/pub/papers/SLT/fisherLDA.pdf). *Eugen.*, 7:179–188, 1936.

[37] Yoav Freund and Robert E. Schapire. Experiments with a New Boosting Algorithm (http://www.recognition.mccme.ru/pub/papers/boosting/freund96experiments.pdf). In *In Proceedings of the Thirteenth International Conference on Machine Learning*, pages 148–156, 1996.

[38] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting (http://www.recognition.mccme.ru/pub/papers/boosting/adaboost.pdf). *J. Comput. Syst. Sci.*, 55(1):119–139, 1997.

[39] Yoav Freund. Boosting a weak learning algorithm by majority (http://www.recognition.mccme.ru/pub/papers/boosting/freund90boosting.pdf). In *COLT '90: Proceedings of the third annual workshop on Computational learning theory*, pages 202–216, 1990.

[40] Jerome H. Friedman. Stochastic gradient boosting (http://www.recognition.mccme.ru/pub/papers/boosting/stobst.ps.gz). *Computational Statistics and Data Analysis*, 38:367–378, 1999.

[41] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine (http://www.recognition.mccme.ru/pub/papers/boosting/trebst.ps.gz). *Annals of Statistics*, 29:1189–1232, 2001.

[42] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting (http://www.recognition.mccme.ru/pub/papers/boosting/friedman98additive.pdf). *Annals of Statistics*, 28:337–407, 2000.

[43] G. M. Furnival and R. W. Wilson. Regression by leaps and bounds. *Technometrics*, 16(4):499–511, 1974.

[44] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images (http://www.recognition.mccme.ru/pub/papers/CRF/GemanPAMI84.pdf). *IEEE Trans. Pattern Anal. Mach. Intell.*, 6:721–741, 1984.

[45] V.I. Glivenko. Sulla determinazione empirica di probabilitá. *Giorn. Ist. Ital. Attuari*, 4:92–99, 1933.

[46] P.S. Gopalakrishnan, D. Kanevsky, D. Nahamoo, and A. Nadas. An inequality for rational functions with applications to some statistical estimation problems. *IEEE Trans. Information Theory*, 37(1):107–113, 1991.

[47] J. M. Hammersley and P. Clifford. Markov fields on finite graphs and lattices (http://www.recognition.mccme.ru/pub/papers/CRF/hammersley71markov.pdf), 1971.

[48] Trevor Hastie, Saharon Rosset, Robert Tibshirani, and Ji Zhu. The Entire Regularization Path for the Support Vector Machine (http://www.recognition.mccme.ru/pub/papers/SVM/hastie04entire.pdf). *Journal of Machine Learning Research*, 5:1391—1415, 2004.

[49] Trevor Hastie and Patrice Y. Simard. Metrics and Models for Handwritten Character Recognition (http://www.recognition.mccme.ru/pub/papers/tangent/hastie97metrics.pdf). *Statistical Science*, 13:54–65, 1998.

[50] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning (*http://www-stat-class.stanford.edu/~tibs/ElemStatLearn/*)*. Springer, 2001.

[51] D. Haussler. Convolution Kernels on Discrete Structures (http://www.recognition.mccme.ru/pub/papers/kernels/haussler99convolution.ps.gz). Technical Report UCSC-CRL-99-10, UC Santa Cruz, 1999.

[52] R. Herbrich. *Learning Kernel Classifiers: Theory and Algorithms (*http://www.learning-kernel-classifiers.org/*)*. The MIT Press, 2002.

[53] Tom Heskes. Stable Fixed Points of Loopy Belief Propagation Are Local Minima of the Bethe Free Energy (http://www.recognition.mccme.ru/pub/papers/CRF/heskes02stable.pdf). In *NIPS 15*, pages 343–350, 2002.

[54] Magnus R. Hestenes and Eduard Stiefel. Methods of Conjugate Gradients for Solving Linear Systems
(http://www.recognition.mccme.ru/pub/papers/gradient/hestenes-stiefel-52.pdf).
J. Res. Natl. Bur. Stand., 49:409–436, 1952.

[55] Xuming He, Richard S. Zemel, and Miguel Á. Carreira-Perpiñán. Multiscale Conditional Random Fields for Image Labeling
(http://www.recognition.mccme.ru/pub/papers/CRF/he04multiscale.pdf).
In *In CVPR*, pages 695–702, 2004.

[56] A.E. Hoerl and R.W. Kennard. Ridge regression: Biased estimation for non-orthogonal problems. *Technometrics*, 12:55–67, 1970.

[57] John H. Holland. *Adaptation in natural and artificial systems.* Univ. of Michigan Press (Reprinted by MIT Press, 1992), 1975.

[58] R. M. Hristev. The ANN Book
(http://www.recognition.mccme.ru/pub/papers/ANN/HristevTheANNbook.djvu).
GNU Public Licence, 1998.

[59] C.-W. Hsu and C.-J. Lin. A comparison of methods for multi-class support vector machines
(http://www.recognition.mccme.ru/pub/papers/SVM/hsu01comparison.ps.gz).
IEEE Transactions on Neural Networks, 13(2):415–425, March 2002.

[60] P. Huber. Robust estimation of location parameter. *Annals of Mathematical Statistics*, 53:73–101, l964.

[61] Tommi Jaakkola and David Haussler. Exploiting Generative Models in Discriminative Classifiers
(http://www.recognition.mccme.ru/pub/papers/kernels/jaakkola98exploiting.pdf).
In *In Advances in Neural Information Processing Systems 11*, pages 487–493. MIT Press, 1998.

[62] B. H. Juang and L. R. Rabiner. The Segmental K-Means Algorithm for Estimating Parameters of Hidden Markov Models
(http://www.recognition.mccme.ru/pub/papers/HMM/juang90segmental.pdf).
IEEE Trans. on Acoustics, Speech, and Signal Processing, 38(9):1639–1641, 1990.

[63] E.L. Kaplan and P. Meier. Nonparametric estimation from incomplete observations
(http://www.recognition.mccme.ru/pub/papers/survival/kaplan58nonparametric.pdf).
Journal of the American Statistical Association, 53:457–481, 1958.

[64] S. S. Keerthi and E. G. Gilbert. Convergence of a Generalized SMO Algorithm for SVM Classifier Design
(http://www.recognition.mccme.ru/pub/papers/SVM/keerthi00convergence.pdf).
Machine Learning, 46(1-3):351–360, 2002.

[65] R. Kikuchi. A theory of cooperative phenomena. *Phys. Rev.*, 81(6):988–1003, 1951.

[66] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[67] J.P. Klein and M.L. Moeschberger. *Survival Analysis. Techniques for Censored and Truncated Data.* Springer, 1997.

[68] Teuvo Kohonen. *Self-Organizing Maps*. Springer, Berlin, Heidelberg, 1995.

[69] А.Н.Колмогоров. О представлении непрерывной функции нескольких переменных суперпозицией непрерывных функций одной переменной и сложения. *ДАН СССР*, 114(5):953–956, 1957.

[70] S. Kumar and M. Hebert. Discriminative Fields for Modeling Spatial Dependencies in Natural Images (`http://www.recognition.mccme.ru/pub/papers/CRF/kumar03discriminative.pdf`). In *NIPS 16*, pages 1531–1538, 2004.

[71] J. Lafferty, A. Mccallum, and F. Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Data (`http://www.recognition.mccme.ru/pub/papers/CRF/lafferty01CRF.pdf`). In *International Confernence on Machine Learning*, 2001.

[72] L. LeCam. On some asymptotic properties of maximum likelihood estimates and related Bayes estimate. *Univ. Calif. Public. Stat*, 11, 1953.

[73] В.И.Левенштейн. Двоичные коды с исправлением выпадений, вставок и замещений символов. *ДАН СССР*, 163(4):845–848, 1965.

[74] Yann Le Cun, Léon Bottou, Genevieve B. Orr, and Klaus-Robert Müller. Efficient Backprop (`http://www.recognition.mccme.ru/pub/papers/ANN/lecun-98b.pdf`). In *Neural Networks, Tricks of the Trade*, Lecture Notes in Computer Science LNCS 1524. Springer Verlag, 1998.

[75] Yury Lifshits. The Homepage of Nearest Neighbors and Similarity Search (`http://simsearch.yury.name/references.html`), 2004-2007.

[76] Chih-Jen Lin, Ruby C. Weng, and S. Sathiya Keerthi. Trust region Newton methods for large-scale logistic regression (`http://www.recognition.mccme.ru/pub/papers/logistic/lin07trust.pdf`). In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 561–568, New York, NY, USA, 2007. ACM.

[77] R. J. A. Little and D. B. Rubin. *Statistical Analysis with Missing Data*. Wiley Series in Probability and Statistics. Wiley, New York, 1st edition, 1987. 2-nd edition (2002).

[78] S. Z. Li. *Markov Random Field Modeling in Computer Vision (*`http://www.cbsr.ia.ac.cn/users/szli/mrf_book/book.html`*)*. Springer-Verlag, London, UK, 1995.

[79] S. Lloyd. Least squares quantization in PCM. *Technical report, Bell Laboratories (1957); IEEE Trans. Inf. Theory*, 28:128–137, 1982.

[80] David J.C. MacKay. Bayesian Interpolation (`http://www.recognition.mccme.ru/pub/papers/Bayes/mackay91bayesian.pdf`). *Neural Computation*, 4:415–447, 1991.

[81] D. J. C. Mackay, J. S. Yedidia, W. T. Freeman, and Y. Weiss. A conversation about the Bethe free energy and sum-product (`http://www.recognition.mccme.ru/pub/papers/CRF/mackay01conversation.pdf`). Technical Report TR2001-018, MERL, 2001.

[82] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *Mathematical Biophysics*, 5:115–133, 1943.

[83] J. Mercer. Functions of positive and negative type and their connection with the theory of integral equations. *Philos. Trans. Roy. Soc. London, A*, 209:415–446, 1909.

[84] А.Б.Мерков. *Распознавание образов: Введение в методы статистического обучения.* УРСС, Москва, 2011.

[85] А.Б.Мерков. *Распознавание образов: Построение и обучение вероятностных моделей.* Ленанд, Москва, 2014.

[86] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.

[87] J. Moussouris. Gibbs and Markov systems with constraints. *Journal of statistical physics*, 10:11–33, 1974.

[88] Radford Neal and Geoffrey E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants (`http://www.recognition.mccme.ru/pub/papers/EM/neal98view.pdf`). In *Learning in Graphical Models*, pages 355–368. Kluwer Academic Publishers, 1998.

[89] A. B. J. Novikoff. On convergence proofs on perceptrons. *Proceedings of the Symposium on the Mathematical Theory of Automata*, XII:615–622, 1962.

[90] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, San Francisco, CA, USA, 1988.

[91] J. Platt. Fast Training of Support Vector Machines using Sequential Minimal Optimization (`http://www.recognition.mccme.ru/pub/papers/SVM/smoTR.pdf`). In B. Schlkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning.* MIT Press, 1998.

[92] J. Ross Quinlan. *C4.5: programs for machine learning.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[93] L. R. Rabiner, J. G. Wilpon, and B. H. Juang. A Segmental K-Means Training Procedure for Connected Word Recognition. *AT&T Tech. J.*, 65(3):21–31, 1986.

[94] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition (`http://www.recognition.mccme.ru/pub/papers/HMM/rabiner.pdf`). *Proceedings of the IEEE*, 77(2):257–286, 1989.

[95] W. Reichl and G. Ruske. Discriminative Training for Continuous Speech Recognition (`http://www.recognition.mccme.ru/pub/papers/HMM/reichl96discriminative.ps.gz`). In *Proc. 1995 Europ. Conf. on Speech Communication and Technology*, pages 537–540, 1995.

[96] H. Robbins and S. Monro. A Stochastic Approximation Method (`http://www.recognition.mccme.ru/pub/papers/stochastic/robbinsMonro.pdf`). *Annals of Mathematical Statistics*, 22:400–407, 1951.

[97] F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization of the brain. *Psychological Review*, 65:386–408, 1958.

[98] Sam Roweis. *Machine Learning* (слайды к курсу лекций CSC2515) (`http://www.cs.toronto.edu/~roweis/csc2515-2006/lectures.html`). Toronto University, 2006. Копия слайдов 2003 года (`http://www.recognition.mccme.ru/pub/papers/general/CSC2515/`).

[99] D. Rumelhart, G. Hinton, and R. Williams. Learning internal representations by error propagation in Parallel Distributed Processing. In D.E. Rumelhart, , and J.L. McClelland, editors, *Explorations in the Microstructure of Cognition*, pages 318–362. The MIT Press, Cambridge, MA., 1986.

[100] Robert E. Schapire, Yoav Freund, Peter Bartlett, and Wee Sun Lee. Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods (`http://www.recognition.mccme.ru/pub/papers/boosting/schapire98boosting.pdf`). *The Annals of Statistics*, 26(5):1651–1686, 1998.

[101] Robert E. Schapire. The Strength of Weak Learnability (`http://www.recognition.mccme.ru/pub/papers/boosting/schapire90strength.pdf`). *Machine Learning*, 5(2):197–227, 1990.

[102] Bernhard Schölkopf, John C. Platt, John C. Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the Support of a High-Dimensional Distribution (`http://www.recognition.mccme.ru/pub/papers/SVM/sch99estimating.pdf`). *Neural Comput.*, 13(7):1443–1471, 2001.

[103] Bernhard Schölkopf. The kernel trick for distances (`http://www.recognition.mccme.ru/pub/papers/kernels/tr-2000-51.pdf`). In *Advances in Neural Information Processing Systems*, volume 13, pages 301–307. MIT Press, 2001.

[104] Arseni Seregin. Об одной положительно определенной форме, 2004. доступен черновик (`http://www.recognition.mccme.ru/pub/papers/kernels/Seregin04StringKernel.pdf`).

[105] Martin Sewell. Kernel Methods (`http://www.svms.org/kernels/kernel-methods.pdf`). *http://www.svms.org*, 2007-2009.

[106] Alex J. Smola and Bernhard Schölkopf. A Tutorial on Support Vector Regression (`http://www.recognition.mccme.ru/pub/papers/SVM/smola98tutorial.ps.gz`). Technical Report NC2-TR-1998-030, NeuroCOLT2, 1998.

[107] Charles Sutton and Andrew McCallum. An Introduction to Conditional Random Fields (`http://www.recognition.mccme.ru/pub/papers/CRF/sutton10introduction.pdf`). arXiv:1011.4088v1, 2010.

[108] А.Н.Тихонов, В.Я.Арсенин. *Методы решения некорректных задач*. Наука, Москва, 1974.

[109] Robert Tibshirani. The lasso method for variable selection in the Cox model (`http://www.recognition.mccme.ru/pub/papers/survival/tibshirani97lasso.pdf`). In *Statistics in Medicine*, pages 385–395, 1997.

[110] R. Tibshirani. Regression shrinkage and selection via the lasso (http://www.recognition.mccme.ru/pub/papers/L1/lasso.pdf). *J. Royal. Statist. Soc B*, 58(1):267–288, 1996.

[111] M.E. Tipping and A.C. Faul. Fast marginal likelihood maximization for sparce Bayesian models (http://www.recognition.mccme.ru/pub/papers/Bayes/tipping03fast.ps.gz). In *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, 2003.

[112] M.E. Tipping. The Relevance Vector Machine (http://www.recognition.mccme.ru/pub/papers/Bayes/tipping00relevance.ps.gz). *Advances in Neural Information Processing Systems*, 12:652–658, 2000.

[113] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory.* Springer, 1995.

[114] Vladimir N. Vapnik. *Statistical Learning Theory.* Wiley, 1998.

[115] Vladimir N. Vapnik. An Overview of Statistical Learning Theory (http://www.recognition.mccme.ru/pub/papers/SLT/Vapnik99overview.pdf). *IEEE Transactions on Neural Networks*, 10(5):988–999, September 1999.

[116] В.Н.Вапник, А.Я.Червоненкис. О равномерной сходимости относительных частот событий к их вероятностям. *Теория вероятностей и приложения*, 16(2):264–280, 1971.

[117] В.Н.Вапник, А.Я.Червоненкис. *Теория распознавания образов.* Наука, Москва, 1974.

[118] A. J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260—-269, 1967.

[119] Chris Watkins. Dynamic Alignment Kernels (http://www.recognition.mccme.ru/pub/papers/kernels/dynk.ps.gz). In *Advances in Large Margin Classifiers*, pages 39–50. MIT Press, 1999.

[120] L.R. Welch. Hidden Markov Models and the Baum-Welch Algorithm (http://www.recognition.mccme.ru/pub/papers/EM/Baum-Welch.pdf). *IEEE Information Theory Society Newsletter*, 53(4), 2003.

[121] J. Weston and C. Watkins. Multi-class Support Vector Machines (http://www.recognition.mccme.ru/pub/papers/SVM/multi-class-support-vector.ps.gz). Technical Report CSD-TR-98-04, Royal Holloway Univ. of London, 1998.

[122] C. F. Jeff Wu. On the Convergence Properties of the EM Algorithm (http://www.recognition.mccme.ru/pub/papers/EM/wu83convergence.pdf). *The Annals of Statistics*, 11(1):95–103, 1983.

[123] Larry S. Yaeger, Brandyn J. Webb, and Richard F. Lyon. Combining Neural Networks and Context-Driven Search for Online, Printed Handwriting Recognition in the NEWTON (http://www.recognition.mccme.ru/pub/papers/ANN-HMM/Yaegeretal.AIMag.pdf). *AI Magazine*, 19(1):73–90, 1998.

[124] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Generalized Belief Propagation (http://www.recognition.mccme.ru/pub/papers/CRF/yedidia00generalized.pdf). In *NIPS 13*, pages 689–695. MIT Press, 2000.

[125] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Constructing Free Energy Approximations and Generalized Belief Propagation Algorithms (http://www.recognition.mccme.ru/pub/papers/CRF/yedidia05constructing.pdf). *IEEE Transactions on Information Theory*, 51(7):2282–2312, 2005.

[126] A. L. Yuille. CCCP algorithms to minimize the Bethe and Kikuchi free energies: Convergent alternatives to belief propagation (http://www.recognition.mccme.ru/pub/papers/CRF/yuille02CCCP.pdf). *Neural Computation*, 14:2002, 2002.

[127] Hui Zou and Trevor Hastie. Regularization and variable selection via the Elastic Net (http://www.recognition.mccme.ru/pub/papers/L1/elasticnet.pdf). *Journal of the Royal Statistical Society B*, 67:301–320, 2005.

Предметный указатель