МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО АТОМНОЙ ЭНЕРГИИ

РОССИЙСКАЯ АКАДЕМИЯ НАУК

РОССИЙСКАЯ АССОЦИАЦИЯ НЕЙРОИНФОРМАТИКИ

МОСКОВСКИЙ ИНЖЕНЕРНО-ФИЗИЧЕСКИЙ ИНСТИТУТ
(ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ)

ИНСТИТУТ ОПТИКО-НЕЙРОННЫХ ТЕХНОЛОГИЙ РАН

НАУЧНАЯ СЕССИЯ МИФИ–2007

НЕЙРОИНФОРМАТИКА–2007

**IX ВСЕРОССИЙСКАЯ
НАУЧНО-ТЕХНИЧЕСКАЯ
КОНФЕРЕНЦИЯ**

ЛЕКЦИИ
ПО НЕЙРОИНФОРМАТИКЕ

Часть 2

По материалам Школы-семинара
«Современные проблемы нейроинформатики»

Москва 2007

УДК 001(06)+004.032.26(06) Нейронные сети
ББК 72я5+32.818я5
М82

НАУЧНАЯ СЕССИЯ МИФИ–2007. IX ВСЕРОССИЙСКАЯ НАУЧНО-ТЕХНИЧЕСКАЯ КОНФЕРЕНЦИЯ «НЕЙРОИНФОРМАТИКА–2007»: ЛЕКЦИИ ПО НЕЙРОИНФОРМАТИКЕ. Часть 2. – М.: МИФИ, 2007. – 148 с.

В книге публикуются тексты лекций, прочитанных на Школе-семинаре «Современные проблемы нейроинформатики», проходившей 24–26 января 2007 года в МИФИ в рамках IX Всероссийской конференции «Нейроинформатика–2007».

Материалы лекций связаны с рядом проблем, актуальных для современного этапа развития нейроинформатики, включая ее взаимодействие с другими научно-техническими областями.

Ответственный редактор
Ю. В. Тюменцев, кандидат технических наук

ISBN 5–7262–0708–4

Содержание

***С. А. Терехов*. Гениальные комитеты умных машин** **11**

Введение: Почему вообще возможно повышение точности? . . . 12

Бустинг (усиление) ансамбля классификаторов 17

Ранние варианты бустинга 18

Семейство алгоритмов AdaBoost 20

Обсуждение AdaBoost 24

Параллельные ансамбли равноправных моделей 28

Комбинирование классификаторов с использованием бутстрепа 28

Почему бэггинг работает? 30

Бэггинг-комитеты, обучаемые в реальном времени 31

Специализация экспертов и другие методы 32

Карты экспертов 33

Смеси экспертов 33

Распределенное обучение экспертов на фрагментах данных 35

Перспективы 37

Благодарности 38

Литература 38

Приложение. Большие наборы данных для экспериментов 41

Задачи 41

Задача 1. Созидательное (?) разрушение 41

Задача 2. Вероятность ошибки алгоритма Boost1 42

Задача 3. (Без решения) 42

УДК 001(06)+004.032.26(06) Нейронные сети

С. А. ТЕРЕХОВ
ООО «Нейрок Техсофт»,
г. Троицк, Московская обл.
E-mail: alife@narod.ru

ГЕНИАЛЬНЫЕ КОМИТЕТЫ УМНЫХ МАШИН

Аннотация

Лекция посвящена актуальной проблеме повышения точности обучения машин путем объединения их в комитеты. Основная цель — обсуждение алгоритмов для обработки очень больших наборов обучающих данных, пригодных к использованию на современных и перспективных многоядерных компьютерах.

S. A. TEREKHOFF
Neurok Techsoft, LLC,
Troitsk, the Moscow Region
E-mail: alife@narod.ru

INGENIOUS COMMITTEES FROM CLEVER MACHINES

Abstract

This Lecture is devoted to committee methods in the machine learning. The main objective is to review most promising algorithms for very large scale classifiers, capable of training on modern and forthcoming multicore computers.

УДК 001(06)+004.032.26(06) Нейронные сети

> . . . Эфиопские драконы в поисках лучших пастбищ регулярно переплывают Красное море, направляясь в Аравию. Для этого четыре-пять драконов, переплетаясь, образуют некое подобие плота, причем головы их торчат над водой.
>
> *Х. Л. Борхес. Книга вымышленных существ*

Введение: Почему вообще возможно повышение точности?

Основным предметом этой лекции является статистическая задача классификации в постановке, принятой в области обучения машин (machine learning). Имеется множество векторов $\{\vec{x}\}$ из многомерного числового пространства, координаты которого будем называть признаками. Векторы независимо получены выборкой из некоторого (неизвестного) вероятностного распределения, обладающего плотностью $\rho(\vec{x})$. Для части векторов известно значение (метка) класса из оговоренного конечного набора возможных классов. Требуется установить, к каким классам относятся оставшиеся векторы, а также любые другие векторы, порожденные той же плотностью[1].

Из векторов-примеров с известными метками можно образовать обучающую выборку, для которой с использованием алгоритмов обучения машин (например, искусственных нейронных сетей или деревьев правил), строится функция-классификатор $h(\vec{x})$. Выходом классификатора является метка класса для каждого входного вектора $\vec{x}$.

С практической точки зрения, обучаемые классификаторы различаются по таким показателям, как точность, производительность, устойчивость к погрешностям в исходных векторах, а также по затратам на их обучение и тестирование.

[1] Более строгие постановки задачи классификации, включающие формализацию понятия ошибки и ее измерения, риска, требования к признаковому пространству и др., приведены в специальной литературе (см., например, [2–4]).

В лекции речь пойдет о рациональном распределении усилий на обучение машин при решении сложных задач классификации и регрессии для больших объемов данных (100 тысяч и более примеров). Масштабирование алгоритмов распознавания классов становится одной из основных проблем [1] в таких областях, как анализ финансовых данных, обработка транзакций в сетях электронной торговли, мониторинг операций с кредитными картами, фильтрация почтовых сообщений, анализ белка и генных структур и др.

Дилемма состоит в поиске ответа на вопрос: что проще — построить и обучить один глобальный классификатор, либо сегментировать проблему на серию задач меньшего масштаба и обучить, а затем объединить, большое количество моделей, желательно, меньшей сложности. Последнее может оказаться предпочтительным, если при объединении моделей достигается требуемая точности прогнозирования.

Откуда может появиться резерв точности множества моделей, каждая из которых настроена на свою подзадачу, и, вообще говоря, не обязана быть высокоточной на всем исходном множестве данных?

Эмпирически феномен повышения точности и надежности результата при объединении нескольких мнений в единое решение был замечен давно[2]. Первые серьезные математические основания для этого факта, по-видимому, связываются с центральной предельной теоремой теории вероятности. Ранние формулировки[3] теоремы появились только в середине XVIII века.

В нашем контексте суть теоремы состоит в том, что последовательности частичных средних, вычисленных по наборам из n независимых случайных величин, даже имеющих большую дисперсию σ, стремятся к нормальному распределению с дисперсией, в корень из n раз меньшей. Тем самым, при определенных условиях, если использовать в качестве результата среднее от значений прогнозов отдельных моделей, то неопределенность такого результата окажется ниже неопределенности отдельной модели.

В реальности условия ЦПТ могут оказаться трудно выполнимыми[4],

[2]Занятно, что около тысячи лет назад, согласно исторической легенде, царь хазар при решении вопроса о выборе веры для своего народа руководствовался мнением трех мудрецов различного вероисповедания. Вопрос, который он задавал каждому — какая, из оставшихся двух, вера ближе к твоей? (см. *Артур Кестлер*. Тринадцатое колено. Крушение империи хазар и ее наследие. – С.-Петербург: Евразия, 2006).

[3]Работа А. Муавра, 1733 год.

[4]Существует еще и проблема скорости сходимости.

поэтому сама теорема может рассматриваться, как «не запрещающая» понижать ошибку путем комбинирования прогнозов.

Прежде, чем приступить к обсуждению практических путей и алгоритмов целенаправленного объединения моделей, рассмотрим простой пример.

Пусть в нашем распоряжении имеются три алгоритма, каждый из которых решает определенную задачу бинарной классификации с вероятностью успеха p, независимо от остальных. Тогда при классификации очередного примера возможны 8 исходов, когда все классификаторы выдали верный ответ, либо два из трех не ошиблись (три варианта), либо не ошибся лишь один (еще три варианта), и, наконец, когда ошиблись все три алгоритма одновременно. Вероятности этих комбинаций исходов равны, очевидно, p^3, $3p^2(1-p)$, $3p(1-p)^2$ и $(1-p)^3$.

Если из данных алгоритмов образовать *комитет большинства*[5], принимающий коллегиальное решение путем простого голосования, то вероятность благоприятного исхода определяется двумя из четырех комбинаций и составляет величину

$$q = p^3 + 3p^2(1-p) = 3p^2 - 2p^3. \tag{1}$$

График зависимости $q(p)$ приведен на рис. 1.

Эта зависимость аккумулирует в себе многие ключевые вопросы, связанные с использованием ансамблей классификаторов.

НАБЛЮДЕНИЕ 1. Кривая имеет два существенно различных участка $p < 1/2$ и $p > 1/2$. Если точность отдельного классификатора хуже точности угадывания класса путем честного бросания честной монетки, то комитет таких моделей только *ухудшит* точность[6].

[5]В лекции термины «комитет», «ансамбль» будут использоваться в широком смысле «объединенная модель», выходом которой является функциональная комбинация выходов отдельных моделей. Конкретные формализации способов композиции выходов будут вводиться по ходу изложения. Этот подход представляется автору оправданным вследствие большого разнообразия возможностей таких объединений, что приведет к значительной перегрузке текста множеством определений. Строгие аксиоматические построения в области композиций алгоритмов развиты в работах академика *Ю. И. Журавлева* [6]. См. также электронные лекции [7].

[6]Это очень важно в контексте обучения нейронных сетей методами повышенных порядков. Такие методы могут очень быстро приводить к раннему останову в ближайшем локальном минимуме, причем состояние обученности такого классификатора может оказаться произвольным. Если при обучении большого числа нейросетей этот процесс не контролировать, то итоговый комитет окажется весьма далеким от желаемой цели.

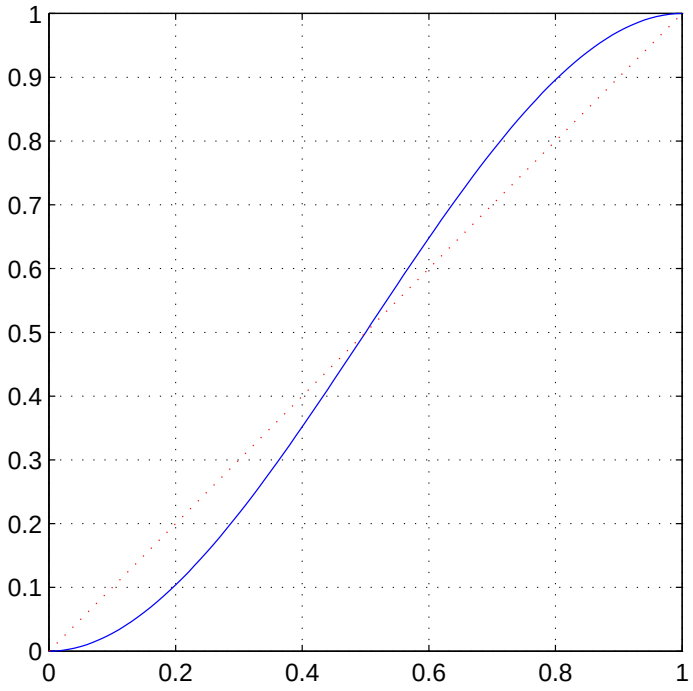


Рис. 1. Вероятность верной классификации комитетом большинства (*по оси ординат*), как функция точности отдельного классификатора (*ось абсцисс*).

Наблюдение 2. Выигрыш в точности в области обученных до $p > 1/2$ классификаторов, определяемый степенью превышения сплошной кривой над пунктирной на рис. 1, значительно варьируется в зависимости от достигнутого p.

Наблюдение 3. Важно, что *во всей области* $1 > p > 1/2$ объединение алгоритмов в комитет систематически превышает точность отдельной модели. Поэтому, потенциально, можно было бы неограниченно повышать точность путем построения комитетов их комитетов, затем комитетов из комитетов комитетов и т. д. В реальности, этого можно достигнуть при отсутствии противоречий в данных, но лишь для *ошибки обучения*. Растущая система комитетов начинает наизусть выучивать информационный шум в

данных, и точность на новых примерах (*ошибка обобщения*) деградирует.

Итак, точность ансамбля моделей можно улучшить, если: 1) повысить точность каждой отдельной модели и, одновременно, 2) обеспечить статистическую независимость ошибок разных членов ансамбля.

Как нередко бывает, эти требования, отчасти, противоречат друг другу. Действительно, для каждого классификатора, достижение точности *при условии независимости* его ошибок от других классификаторов является более сложной задачей, чем аналогичное обучение *при отсутствии ограничивающих условий*.

Проверка статистической независимости распределений ошибок в общем случае может оказаться самостоятельной проблемой, поэтому на практике обычно останавливаются на более простой ситуации анализа линейных корреляций ошибок всех пар классификаторов комитета. Если коэффициенты корреляции для различных пар отличаются, то можно предложить более эффективную схему выбора относительных весов отдельных членов комитета, а именно [5]:

$$w_i = \frac{\sum_{j=1}^{M} (C^{-1})_{ij}}{\sum_{k=1}^{M} \sum_{j=1}^{M} (C^{-1})_{kj}}. \tag{2}$$

Здесь M — число моделей в ансамбле, C^{-1} — матрица, обратная к матрице парных корреляций. Строго говоря, этот результат справедлив для задачи регрессии линейной комбинацией моделей

$$h(\vec{x}) = \sum_i w_i h_i(\vec{x}), \tag{3}$$

однако общий смысл зависимости (2) сохраняется: чем больше похож данный член комитета на остальных, тем меньшую пользу он приносит, и тем меньше его вес.

Коэффициенты корреляций в (2) невозможно точно вычислить для будущих данных, однако их можно оценить по некоторой, не зависящей от обучающих данных, выборке[7].

В дальнейшем изложении в лекции будут обсуждаться различные методические подходы к оптимальному комбинированию моделей, включая как

[7] Для этих целей некорректно будет использовать ту *валидационную* выборку, по которой проводится прогноз окончательной ошибки обобщения модели, поскольку оптимальный выбор весов членов комитета является частью процесса обучения.

общие идеи, так и готовые к применению алгоритмы. Также будут затронуты вопросы оптимизации самих моделей с целью более эффективного использования их в комитетах.

В завершающем разделе рассматриваются вопросы масштабирования ансамблей классификаторов, особенно актуальные в связи с грядущим распространением общедоступных параллельных (многоядерных) процессоров и открывающимися широкими возможностями по массовому использованию высокопроизводительных обучающихся машин.

В приложении приведены некоторые задачи, цель которых стимулировать критическое отношение к изложенному в лекции материалу.

Бустинг (усиление) ансамбля классификаторов

При формировании ансамбля моделей необходимо содержательно оптимизировать два критерия — качественное обучение отдельного классификатора и оптимальное их объединение. Голосующие алгоритмы принято разделять на два класса [12]: алгоритмы, которые для новых классификаторов изменяют распределение обучающих примеров исходя из показателей точности предыдущих моделей, и те, в которых новые члены ансамбля обучаются независимо от остальных. Первый класс еще 5–10 лет назад выглядел более привлекательным, поскольку позволял целенаправленно улучшать обучение. Однако в контексте обучения систем на больших объемах данных возможность распараллеливания обучения ансамблей независимых моделей становится все более актуальной, и, по мнению автора, основные усилия разработчиков в ближайшее время будут сконцентрированы вокруг второго класса ансамблей, обладающих естественной параллельностью[8].

Поэтому начнем рассмотрение с систем последовательно обучающихся классификаторов.

Идея *бустинга* (boosting) предложена *Робертом Шепайре* (Robert E. Schapire) в конце 80-х годов (см. обзор [8]) в контексте фундаментального вопроса об эквивалентности *слабого и сильного* обучения. Смысл постановки проблемы состоит в следующем.

Пусть имеется некоторый алгоритм классификации, точность которого для заданной плотности распределения данных лишь незначительно пре-

[8]Логика параллельной обработки информации вносит свои коррективы в приоритеты не только в области обучения машин. Аналогичные примеры все чаще появляются и в других, более традиционных, областях, таких как вычислительная линейная алгебра, методы оптимизации, численное решение уравнений математической физики.

вышает точность угадывания: $p = 1/2 + \varepsilon$. Возможна ли, с использованием только такого (слабого) классификатора и набора данных, практическая[9] реализация программы по достижению сильного (сходящегося по вероятности) обучения, такого, что для всяких $0 \leqslant \gamma$, $\delta \leqslant 1/2$ с вероятностью более $1 - \gamma$ ошибка окажется меньше δ.

Ранние варианты бустинга

Приведенный во введении пример показывает, что на пути к повышению точности желательно обеспечивать независимость ошибок классификаторов. Напрашивающийся первый шаг в этом направлении — построение цепочки последовательно обучающихся классификаторов, каждый из которых осведомлен об ошибках предыдущих. Так как мы имеем дело с одним алгоритмом, способным достигнуть слабого обучения, то единственный[10] способ внести разнообразие в ошибки получающихся членов ансамбля — это использовать при обучении каждый раз другие данные. В первых вариантах бустинга [10] (см. также [9,11]) рассматривались тройки последовательно обучающихся слабых алгоритмов, достигающих уровня ошибки $\alpha = (1 - p) < 1/2$. Ниже приводится прототипический алгоритм их комбинирования.

Алгоритм Boost1

1. Первый классификатор h_1 обучается на множестве из m примеров.
2. Второй классификатор также обучается на m примерах, выбираемых так, что первый классификатор ровно на половине из них дает точный ответ.
3. Наконец, третий классификатор обучается на таких m примерах, на которых мнения первого и второго классификаторов расходятся.

Окончательная гипотеза формируется голосованием трех классификаторов

[9] Теоретически, как было видно из вводной части лекции, сильного обучения достигнуть не сложно, если иметь набор, вообще говоря, разных моделей с гарантированной независимостью ошибок.

[10] При обучении нейросетей с ранним остановом в локальном минимуме определенное разнообразие достигается при различном случайном выборе начальных весов нейронов. Однако такое разнообразие отражает собой лишь статистику локальных минимумов ошибки и никак не связано с независимостью ошибок сетей, находящихся в разных минимумах. Улучшение алгоритма обучения (и, например, переход к онлайн-обучению или постраничному обучению) нивелирует искусственный эффект влияния начальных весов.

по большинству:

$$h(x) = \mathrm{sign}\Big(\sum_{k=1}^{3} h_k(x)\Big). \qquad (4)$$

В формуле (4) принято, что ответы классификаторов принадлежат множеству $\{-1, +1\}$. Несложные рассуждения и арифметика вероятностей приводят к вероятности ошибки для тройки $3\alpha^2 - 2\alpha^3$ (совпадение с формулой (1) для независимых ошибок случайно!), что меньше ошибки каждого h_k.

Смысл алгоритма `Boost1` состоит в последовательной фильтрации примеров предыдущими обученными классификаторами, причем так, что задача для каждого последующего классификатора становится труднее. Проиллюстрируем логику работы алгоритма на примере игрушечной задачи о классификации двух множеств точек (50/50) на плоскости на два класса линейными пороговыми классификаторами (т. е. единичными формальными нейронами). Распределение точек и результат применения `Boost1` показаны на рис. 2. Сформированное в результате обучения решающее правило имеет вид ломаной линии, сепарирующей классы значительно лучше, чем каждый из классификаторов в отдельности.

Интересно увидеть, как оказались распределены обучающие примеры для самой сложной задачи последнего классификатора. Они включают только множество тех точек, по которым результаты первых двух нейронов не совпадают (выборка для обучения необходимой длины 100, равная по объему исходному множеству примеров, была получена случайным ресамплингом[11] с возвратом).

Характерная особенность этой выборки состоит в том, что исходная плотность распределения оказывается начисто забытой (о ней «помнит» только первый из классификаторов) в угоду точности. Это сохраняется и для других методов бустинга — целью обучения является лишь аппроксимация геометрической границы классов, поэтому результаты бустинга алгоритмов при наличии шума могут значительно уклоняться от оптимального байесова решающего правила, существенно учитывающего характер плотности данных во всем пространстве.

Базовая идея `Boost1` получила дальнейшее развитие в семействе адаптивных алгоритмов, позволяющих комбинировать произвольное число ал-

[11] Применение ресамплинга, строго говоря, не предусматривалось в оригинальной версии алгоритма, формулировка которого проведена для некоторого «оракула» (неограниченного источника, порождающего новые данные).

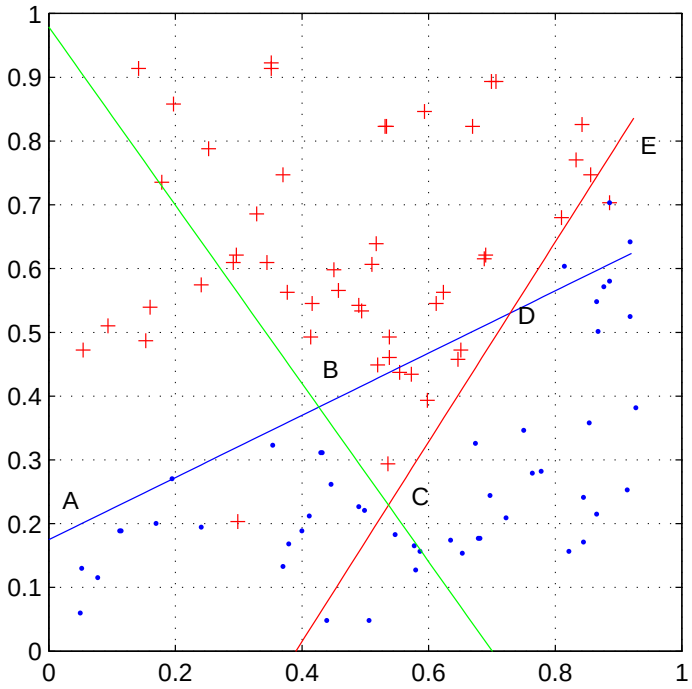


Рис. 2. Обучение ансамбля линейных пороговых классификаторов методов `Boost1`. Решающее правило соответствует границе ABCDE. Первый классификатор — линия ABD, второй — CDE, третий — BC.

горитмов с близким к оптимальному сочетанием их относительных весов. Такие алгоритмы могут, при отсутствии шума и противоречий в данных, достигать произвольной точности обучения, т. е. являются сильными алгоритмами в статистической терминологии PAC (probably asymptotically correct — асимптотически точные по вероятности).

Семейство алгоритмов AdaBoost

Примерно через 5 лет после опубликования оригинальной идеи бустинга, ее автору с коллегами удалось существенно обобщить этот подход. В

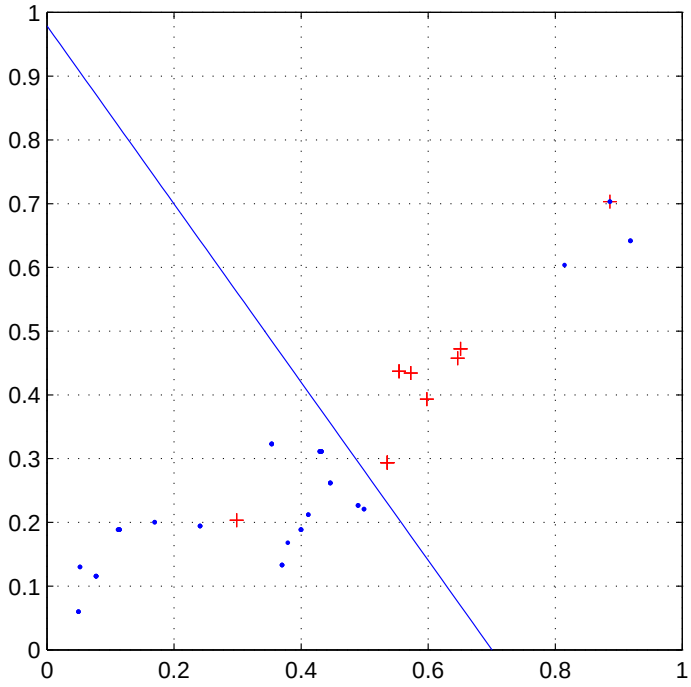


Рис. 3. Обучающие примеры для третьего классификатора в методе `Boost1`

новой работе [13][12] предложен алгоритм `AdaBoost` (Adaptive Boosting — адаптивное усиление), в котором в ансамбль могут объединяться произвольное число моделей, обучение проходит на одной обучающей выборке, точности различных моделей могут отличаться (но сохраняется требование слабой обучаемости), что учитывается в относительных весах моделей в коллегиальном решении.

Ниже приводится вариант алгоритма в чуть более поздней формулировке [8].

Пусть классификатор со слабым обучением формирует гипотезу (функцию) из пространства входных признаков во множество двоичных ответов $h_t : X \to \{-1, 1\}$. Качество этой гипотезы измеряется в количестве оши-

[12]Только по статистике сайта `CiteSeer` эта работа процитирована около 500 раз.

бок относительно указанного распределения D_t примеров с известными метками классов $x_i \in X \to y_i \in \{-1, 1\}$:

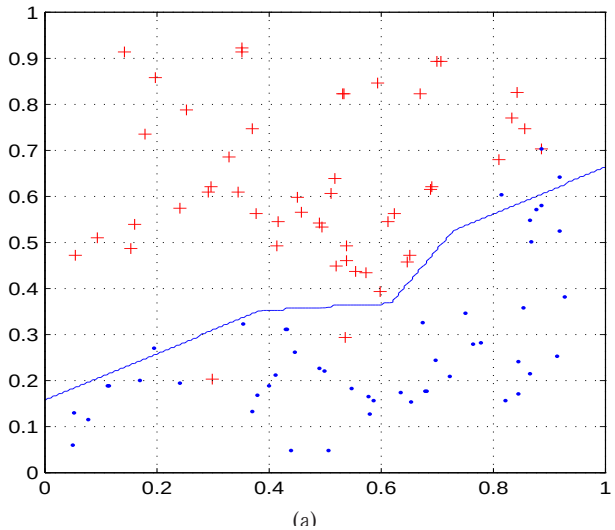
$$\varepsilon_t = \Pr_{(x,y)\sim D_t}[h_t(x_i) \neq y_i] = \sum_{i:h_t(x_i)\neq y_i} D_t(i). \tag{5}$$

Оценка ошибки (5) существенно зависит от характера распределения D_t, которое в вычислениях описывается относительными весами примеров в выборке. Выражение (5) также справедливо для случая нескольких классов, что отражено в приведенном ниже алгоритме.

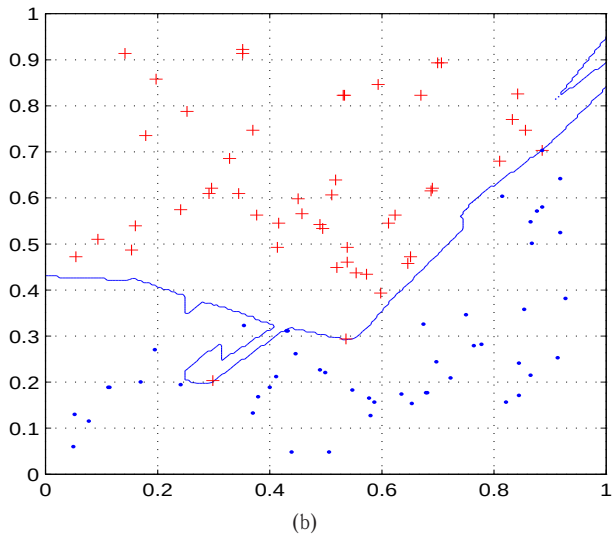
Алгоритм AdaBoost

1. Дан набор обучающих примеров $(x_1, y_1), \ldots, (x_m, y_m)$ с метками $y \in \{1, \ldots, k\}$. Начальные веса примеров одинаковы: $D_1 = 1/m$. Шаги 2–4 повторяются для итераций $t = 1, \ldots, T$.
2. Обучить классификатор на распределении D_t, получить классифицирующую гипотезу h_t с оценкой ошибки
$$\varepsilon_t = \sum_{i:h_t(x_i)\neq y_i} D_t(i).$$
3. Вычислить
$$\alpha_t = \frac{1}{2} \cdot \ln\Big(\frac{1-\varepsilon_t}{\varepsilon_t}\Big).$$
4. Обновить распределение примеров
$$D_{t+1}(i) = \frac{D_t(i) \cdot \exp[-\alpha_t y_i h_t(x_i)]}{Z_t},$$
где Z_t — нормировка суммы D_{t+1} на единицу (так что получается нормированное весовое распределение).
5. Сформировать окончательную гипотезу-классификатор
$$H(x) = \operatorname{sign}\Big(\sum_{t=1}^{T} \alpha_t h_t(x)\Big)$$
для случая двух классов, или
$$H(x) = \operatorname*{argmax}_{y\in\{1,\ldots,k\}}\Big[\sum_{t:h_t(x)=y} \alpha_t\Big]$$
для k классов.

Суть алгоритма заключается в последовательном, адаптивном к результатам предыдущих классификаторов, повышении относительного веса при-



(a)



(b)

Рис. 4. Решающее правило после четырех итераций метода `AdaBoost`: а) `AdaBoost` для линейного порогового классификатора; б) обучение нейронной сети с двумя нейронами в скрытом слое методом `AdaBoost`.

меров, на которых допускались ошибки. Индивидуальный вес α_t очередного классификатора остается положительным, если его ошибка ниже 50% на том распределении, на котором он обучался. Чем больше ошибка классификатора, тем меньше его вес.

Характер работы алгоритма иллюстрируется на рис. 4 на решении модельной задачи классификации, описанной в предыдущем разделе. Для сравнения приведены результаты для линейного порогового классификатора (1 нейрон) и нейронной сети с двумя нейронами в скрытом слое.

При прямом применении `AdaBoost` к достаточно гибким классификаторам наблюдается переобучение.

Обсуждение AdaBoost

Одно из достоинств алгоритма `AdaBoost` состоит в возможности проведения всего цикла обучения для одного исходного набора обучающих примеров, путем повторного использования их на разных шагах.

Основной теоретический результат [14] относительно ошибки обучения итогового ансамбля сформулирован в теореме: Если алгоритм `AdaBoost` порождает гипотезы с ошибками

$$\varepsilon_{D_t}(h_t) = 1/2 - \gamma_t,$$

то ошибка *обучения* комитетной гипотезы

$$E(H) \leqslant \exp\bigl(-2\sum_t \gamma_t^2\bigr).$$

Таким образом, ошибки обучения в благоприятной ситуации убывают экспоненциально. Доказательство теоремы приведено в [14] и многократно воспроизведено в других работах. Интуитивно, суть теоремы можно понять, если заметить, что если пример неверно классифицирован комитетом, то он также ошибочно классифицирован большинством индивидуальных моделей. Следовательно, такой пример имеет (экспоненциально) большой относительный вес. А так как распределение весов нормировано, то доля таких примеров (экспоненциально) мала.

Вопрос о качестве обобщения моделей, обученных по `AdaBoost`, широко дискутируется в литературе, в основном, с экспериментальной точки зрения. Теоретические оценки степени обобщения, оперирующие поня-

тием VC-размерности[13] классификатора d, приводят к слишком широким интервалам, чтобы представлять практический интерес. Ошибка имеет тенденцию уменьшаться как $\sim \sqrt{Td/m}$.

Экспериментально обнаружено, что алгоритм весьма эффективен, но может приводить к переобучению при большом числе шагов, хотя это наблюдается относительно нечасто. Это — хорошая новость, поскольку, по построению, `AdaBoost` все жестче концентрируется на малом числе трудных примеров в процессе добавления новых членов комитета, что должно приводить к переобучению. Отчасти факт отсутствия переобучения во многих практических примерах объяснен автором алгоритма [14], исходя из рассмотрения динамики *зазора* (или *отступа*, margin) между классами.

Зазор (удаление от границы классов) для индивидуального примера равен $y_i h(x_i)$. Эта величина положительна только тогда, когда пример классифицирован безошибочно. Суть алгоритма `AdaBoost` заключается в последовательной минимизации на каждом шаге величины

$$Z_t = \sum_t D_t(i) \cdot \exp(-\alpha_t y_i h_t(x_i)) \tag{6}$$

как функции α_t, при этом предыдущие веса $\alpha_1, \ldots, \alpha_{t-1}$ остаются неизменными. Таким образом, алгоритм является «жадным» покоординатным методом максимизации зазора, поскольку экспоненциальная функция в (6) является мажорантой (верхней границей) риска ошибки, который равен 1, если зазор отрицателен, и нулю (нет ошибки) при положительном зазоре. Это иллюстрирует рис. 5.

В статистической теории обучения [3] максимизация зазора интерпретируется, как малый риск ошибки обобщения — чем больше ширина зазора, тем менее вероятно его преодоление и попадание нового примера за границу класса.

Соотношение межу понятиями «большой зазор» и «качество обобщения» вызывает определенную полемику в технической литературе. «Мягкий» текущий вывод ([20], с. 168) из этой полемики таков: бустинг эффективен для классификаторов, которые слегка недообучаются (в противоположность рассмотренному далее в лекции бэггингу, для которого рекомендованы алгоритмы, которые слегка переобучаются на данных).

[13]Имеются замкнутые оценки для размерности Вапника-Червоненкиса [3] некоторых типов классификаторов, в частности, для линейных решающих правил размерность имеет масштаб числа свободных параметров. Рассмотрение этой теории выходит за рамки данной лекции.

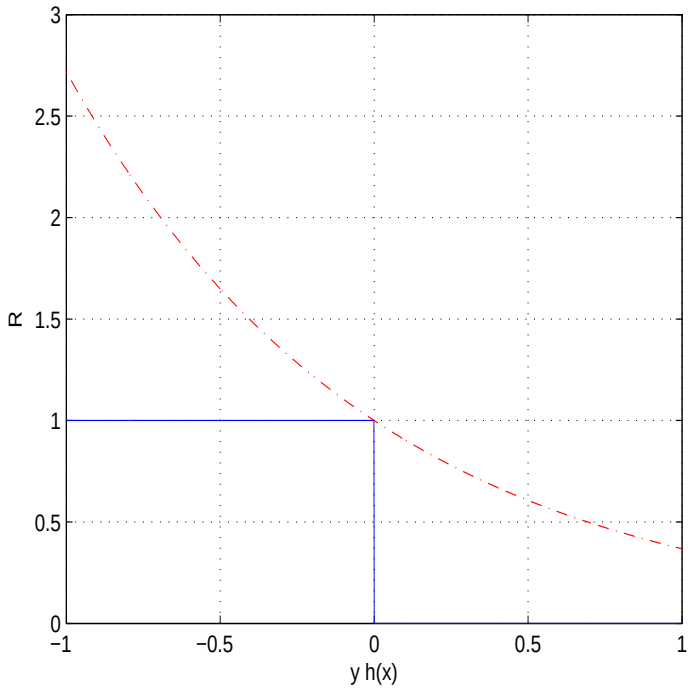


Рис. 5. Зависимость теоретического риска ошибки (сплошная пороговая линия) и его оценки в `AdaBoost` (штрих-пунктир) от величины зазора (margin)

Методы максимизации зазора, и в частности, основанные на них машины опорных векторов (`SVM` — `Support Vector Machine`) обсуждались в предыдущей лекции автора [15]. Отметим здесь, что возможна формальная постановка `AdaBoost`, как задачи оптимизации, аналогичной задаче оценивания плотности[14] алгоритмом `SVM` [16,17]. При этом отличие в постановках состоит в минимизации разных норм весовых векторов — нормы L_2 для `SVM` и L_1 (суммы модулей) для `AdaBoost`. Это отличие, однако оказывается существенным — в признаковом пространстве X решение для `AdaBoost` оказывается разреженным (число членов комитета конечно и

[14]Имеется в виду алгоритмы «`SVM` для классификации в один класс» (`1-class SVM`), которые оценивают носитель области в признаковом пространстве, занятой данными.

относительно невелико), решение же для SVM приводит к плотному вектору (оно является разреженным в дуальном пространстве весов примеров). В итоге, несмотря на определенное сходство математического аппарата, получаемые обученные классификаторы дают разные приближения к границе классов.

Формально, алгоритм AdaBoost применим и к общему случаю k классов, но его прямое использование при $k > 2$ затруднено, поскольку требование к ошибке $\varepsilon_t < 1/2$ трудно удовлетворить для множества классов. В литературе обсуждаются многочисленные модификации алгоритма, позволяющие обойти данную трудность [8,18,19].

Относительно недавний прогресс [21] в разработке методов семейства AdaBoost связывается с учетом того факта, что многие алгоритмы классификации (включая нейронные сети) способны выдавать на выходе не только метку класса со значениями $\{-1, 1\}$, но и вещественнозначную оценку уверенности в прогнозе этой метки. В частности, для этих целей может использоваться неконтрастированный выход нейронной сети в интервале $[-1, 1]$.

Первый шаг состоит в формальном сохранении алгоритма в исходном виде, но теперь в качестве гипотезы отдельного классификатора следует понимать значение в интервале $h(x) \in [-1, 1]$. Это приводит к более консервативному выбору весов примеров $D(i)$ и лучшему приближению покоординатного поиска α_t (правда, за счет увеличения числа членов комитета). Следующий шаг сводится к замене приближенного значения α_t (шаг 3 алгоритма AdaBoost) прямой численной минимизацией функции (6). Авторы [21] также предлагают использование целевой функции (6) при обучении самих классификаторов. Так, например, алгоритм обучения нейронной сети на основе обратного распространения ошибок легко может быть обобщен на оценку (6), поскольку производные оценки по выходу нейросети вычисляются явно.

Другие варианты целевого улучшения базового алгоритма могут быть получены путем использования в формуле (6) иных оценок риска (вместо величины зазора $y_i h(x_i)$). На этом пути предложены методы классификации на несколько классов и для ситуации, когда пример может принадлежать сразу нескольким классам [21, с. 13]. Имеются также алгоритмы на основе AdaBoost для задачи регрессии [9] (классификации в континуум классов).

Методы AdaBoost весьма популярны и эффективны для задач малой и средней сложности (до нескольких десятков тысяч примеров). В контексте

этой лекции, основными сдерживающими моментами к широкому применению методов бустинга к задачам *увеличивающегося масштаба* являются:

- Принципиальная последовательная структура алгоритмов `AdaBoost`. Обучение каждой новой модели ансамбля может начаться только после получения результатов предыдущей модели.
- Все члены комитета обучаются на масштабных выборках, требующих просмотра и пробного применения классификатора ко всем обучающим примерам на каждой итерации. Это требование сразу ограничивает снизу скейлинг алгоритма — сложность заведомо не лучше, чем линейная по числу примеров. Это уже может оказаться серьезным препятствием для ряда приложений.

Нашей следующей целью будут алгоритмы, способные преодолеть эти ограничения.

Параллельные ансамбли равноправных моделей

Желаемого разнообразия моделей и определенной независимости их ошибок, что благоприятно сказывается на точности при объединении в комитет, можно достигнуть и без использования одних моделей для изменения других. Модели могут обучаться независимо, что является принципиальным ресурсом при использовании параллельных вычислений.

Комбинирование классификаторов с использованием бутстрепа

Равноправные модели становятся разнообразными при обучении на различных выборках данных. Если в нашем распоряжении имеется лишь один набор из m примеров, то различающиеся подвыборки с близкой статистикой можно получить путем применения *бутстрепа* [22] — случайной выборки с возвратом.

Эмпирическая плотность распределения данных, оцененная по набору примеров, дается выражением

$$\rho_{emp}(x) = \frac{1}{m}\sum_{j=1}^{m}\delta(x - x_j). \tag{7}$$

Эту плотность можно использовать для генерации новых выборок. А именно, для получения очередного примера следует случайно выбрать одно из

слагаемых в (7) и далее формально выполнить выборку из плотности, даваемой этим слагаемым. В случае дельта-функции результат, очевидно, детерминирован и равен соответствующему примеру x_j. Эта процедура и есть бутстреп-метод.

Вероятность появления в выборке каждого примера, по построению, равна

$$p_{bs} = 1 - (1 - 1/m)^m \xrightarrow[m\to\infty]{} 0.632.. \qquad (8)$$

т. е. некоторые примеры встречаются несколько раз, а какие-то ни разу. Многие статистические особенности исходного набора данных воспроизводятся и в бутстреп-выборках, что позволяет использовать этот метод для построения перекрестных оценок[15].

Подход, основанный на независимом обучении отдельных моделей на бутстреп-выборках из обучающего набора данных и объединении полученных моделей в комитет большинства, был предложен в 1994 году *Лео Брейманом* (Leo Breiman) [23]. Метод получил название «бэггинг» (bagging — *b*ootstrap *agg*regat*ing*).

Формальный алгоритм весьма прост:

АЛГОРИТМ BAGGING

1. Дан набор обучающих примеров $(x_1, y_1), \ldots, (x_m, y_m)$ с метками $y \in \{1, \ldots, k\}$.
2. Получить T бутстреп-выборок D_t.
3. Независимо (параллельно) обучить t классификаторов h_t, каждый на своей выборке D_t.
4. Сформировать окончательную гипотезу-классификатор

$$H(x) = \operatorname{sign}\Big(\sum_{t=1}^{T} h_t(x)\Big)$$

для случая двух классов, или

$$H(x) = \operatorname*{argmax}_{y\in\{1,\ldots,k\}} \Big[\sum_{t:h_t(x)=y} 1\Big]$$

для k классов.

[15] Обращение с бутстреп-выборками требует известной аккуратности. К ним нельзя относиться как к неиссякаемому источнику новых данных, отдельные выборки являются специфически *зависимыми*. В частности, если исходная выборка примеров была неудачной по отношению к реальной плотности распределения данных (например, давала смещенную оценку матожидания), то все бутстреп-реплики также окажутся неудовлетворительными с большой вероятностью.

Метод бэггинга демонстрирует устойчивые результаты на практике[16], и очень удобен в реализации. Разброс значений выходов различных классификаторов может быть использован для дополнительной оценки ошибки (ее верхней границы). При последовательной реализации комитет может расширяться поэтапно, при этом первые результаты классификации система начинает выдавать уже после обучения первых моделей.

Полностью аналогично алгоритм бэггинга формулируется и для задачи регрессии, в которой модели образуют линейную комбинацию с одинаковыми весами.

Почему бэггинг работает?

Теоретические оценки достижимой точности [23] основываются на общепринятой декомпозиции квадратичной ошибки на два члена — «смещение» (bias) и «дисперсию» (variance). Дальнейшие выкладки показывают, что бэггинг уменьшает дисперсию, не изменяя смещение оценки ансамбля.

Однако, ситуация до конца не ясна, в частности, обнаружено, что бэггинг может улучшать оценки, даже не приводя к уменьшению дисперсии. Суть вопроса в том, что при построении комитета борются две тенденции — благоприятное усреднение моделей и факт того, что каждая модель обучалась на неполном (и искаженном) наборе данных.

Экспериментально показано [5,12,23], что бэггинг обеспечивает *равномерное* улучшение точности на всех задачах (хотя бустинг демонстрирует лучшие результаты на многих из них, но в ряде случаев ухудшает точность). Наилучшие результаты достигаются при комбинировании так называемых неустойчивых моделей (к ним относятся нейронные сети и деревья правил)[17], но не улучшает и не ухудшает устойчивые к изменениям в наборе данных алгоритмы (такие, как метод k–ближайших соседей).

[16] В 2001–2002 годах коллективом под руководством автора лекции была разработана система `CNet` для задач классификации и регрессии, использующая бэггинг-ансамбль нейросетевых моделей. Вычислительный сервер `CNet` успешно применялся для промышленных приложений в области бытовой химии, неразрушающего контроля трубопроводов, а также в нефтедобывающей отрасли. В настоящее время он продолжает использоваться в проектах, выполняемых компанией `Нейрок Техсофт`.

[17] Увы, общепринятый в математической статистике термин здесь не отражает существа дела. Речь идет о *гибкости* моделей в их способности отражать изменения в данных. Неустойчивость же обычно ассоциируется с отсутствием сходимости — однако отмеченные «неустойчивые» алгоритмы прекрасно сходятся, если сходятся соответствующие последовательности наборов данных.

С практической точки зрения интересен также вопрос — можно ли добиваться улучшения точности при бэггинге моделей, если объем обучающих бутстреп-выборок *меньше* объема исходного набора данных. Как далеко можно двигаться в сторону уменьшения выборок без существенных последствий для точности? Автору известны лишь экспериментальные оценки для частных случаев.

По-видимому, численные эксперименты являются основным инструментом при анализе бэггинга — сам *Лео Брейман*, отвечая на вопрос, сколько моделей следует использовать в комитете, апеллирует к экспериментальным оценкам — не более 50 для задач классификации и вдвое меньше для регрессии.

Бэггинг-комитеты, обучаемые в реальном времени

В исходной постановке метод бэггинга требует наличия всех обучающих примеров до начала процесса обучения. Это нельзя признать полностью соответствующим многим практическим потребностям. В приложениях данные могут поступать в систему последовательно и обучение целесообразно было бы совместить с процессом сбора данных.

Техническая трудность состоит в онлайн-генерации бутстреп-выборок. Более точно, в принятии решения по каждому примеру в отдельности, сколько раз (с каким весом) он должен участвовать в обучении каждой из моделей комитета.

Возможное решение состоит в аппроксимации бутстреп-распределения путем генерации случайного веса примера из распределения Пуассона со средним, равным единице. Вероятность встретить пример ровно k раз для $k = 0, 1, \ldots$ в законе Пуассона равна

$$P(K = k) = \exp(-1)/k! \,. \tag{9}$$

При поступлении нового примера он копируется искомое число раз (свое для каждого классификатора) и далее участвует в обучении онлайн-алгоритмом. Такой способ построения выборки не вполне точно воспроизводит бутстрэп. В частности, по получении m примеров нет гарантии того, что ровно такое же число примеров будет отобрано для каждой модели. Это, потенциально, может вносить дополнительные искажения в статистику (впрочем, существенные только с теоретической точки зрения).

В другом подходе [24] предлагается рассматривать бутстреп-выборку как исходную выборку, но с весами, принимающими точные значения

$$\left\{0, \frac{1}{m}, \frac{2}{m}, \ldots\right\},$$

причем сумма весов равна единице. Тогда возможна байесова интерпретация весов примеров, как вектора вероятностей, следующего m-мерному распределению Дирихле. Если алгоритм обучения может оперировать с весами примеров (в частности, аддитивная по примерам ошибка нейронной сети может суммироваться с весами), то искомое онлайн-решение — это учет каждого примера при обучении каждой модели с последовательной генерацией весов. В этом случае все классификаторы автоматически являются равно нагруженными примерами.

Оценка комитета может формироваться после каждого шага обучения на новом примере и использоваться одновременно с обучением (примеры с известными метками поступают в каналы обучения, а для примеров с неизвестными метками вычисляется класс из комитета оценок текущих классификаторов).

Специализация экспертов и другие методы

Рассмотренные в предыдущих разделах методы формирования ансамблей классификаторов неявно предполагают, что ошибки отдельной модели относительно *равномерно* распределены в пространстве признаков, и, при ее оптимальном добавлении в комитет, достаточно воспользоваться единой оценкой ошибки.

Это предположение может нарушаться в больших неоднородных наборах данных. Различные области пространства могут качественно описываться разными членами комитета, поэтому их объединение должно учитывать такую специализацию. Прямое применение этого наблюдения приводит к подходу динамической интеграции классификаторов [25].

Суть идеи состоит в том, что после обучения все примеры сохраняются вместе с оценками ошибок, которые делались на них всеми членами ансамбля. Далее, при поступлении нового примера, в базе данных производится поиск ближайших прототипов из обучающей выборки и в голосовании участвуют только те члены комитета, которые допускали малые ошибки на найденных прототипах. Таким образом, происходит динамическое объединение *некоторых* членов комитета для решения каждой новой задачи.

Карты экспертов

Идею динамического объединения нетрудно развить дальше, если вместо хранения прототипов использовать кластерную структуру (например, карту Кохонена), построенную на обучающей выборке без меток классов, и приписать каждому кластеру список моделей, демонстрировавших лучшие результаты на данных этого кластера. Для классификации теперь достаточно найти ближайший кластер и применить ассоциированные с ним модели [18].

Каждый член комитета приписывается ко всем кластерам, на данных из которых он мало ошибается, таким образом, кластеры ведут свои рейтинг-листы моделей.

Нейросетевая карта специализированных экспертов может использоваться не только в процессе распознавания, но и для аналитических целей. В частности, на основе карты по мере поступления запросов можно сделать вывод о «проблемных областях» комитета. Им соответствуют кластеры, заполненные данными, но бедные связанными с ними классификаторами. Это дает новую стратегию обучения и пополнения комитета новыми моделями.

Смеси экспертов

Карта экспертов является наглядным простым примером более общей методологии смесей экспертов [26–28]. Смеси экспертов очень широко обсуждаются в специальной литературе [19]. В Российской школе нейроинформатики [29] также предложен свой оригинальный подход к этой проблематике. В целом, данная тема достойна отдельной лекции, поэтому остановимся только на общей структуре системы.

Объединение экспертов в ансамбль при решении задачи классификации нового примера производится с весами, которые определяются выходами управляющей нейронной сети (рис. 6).

Задача управляющей нейронной сети — выбрать веса смеси, наиболее соответствующие классифицируемому в данный момент входному вектору. В простейшем варианте управляющая сеть состоит из одного слоя нейронов, число которых равно числу членов комитета. Выход сети определяется

[18] Предлагается назвать такую гибридную архитектуру картой экспертов (`ExpertMap`).

[19] Работа [26] также является одним из лидеров по цитированию, наравне со статьей *Schapire* об `AdaBoost`.

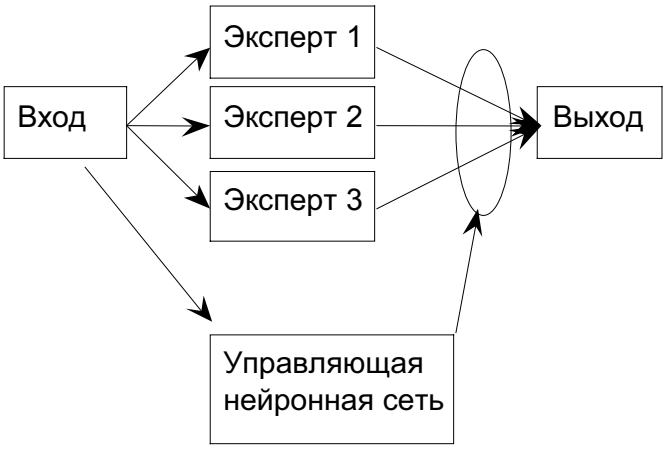


Рис. 6. Архитектура модели смеси экспертов

методом «мягкого максимума» (softmax):

$$c_i = \sum_j x_j w_{ij}, \quad g_i = \frac{\exp(c_i)}{\sum_j \exp(c_j)}\,. \tag{10}$$

Выход всей системы дается выражением

$$H_{mix} = \sum_i g_i h_i(x)\,. \tag{11}$$

При использовании квадратичной функции ошибки, выражения для обучения управляющей сети получаются ее дифференцированием и применением обратного распространения:

$$\frac{\partial E(x,y)}{\partial g_i} = h_i(x) \cdot \Big(\sum_j g_j h_j(x) - y\Big)\,. \tag{12}$$

В различных вариантах модели в качестве управляющей сети используются гауссовы смеси (радиальные базисные функции) с обучением EM–алгоритмом [30], системы нечетких правил, иерархические структуры [26] и другие модели.

Смеси экспертов являются примером *нелинейного* объединения алгоритмов в комитет — веса в выражении (11) зависят от координат входного вектора. В общем случае, получение теоретических построений, описывающих точность таких моделей, затруднено. На практике используют обычные методы перекрестного оценивания и валидационные выборки.

Распределенное обучение экспертов на фрагментах данных

Дальнейшее движение в сторону усложнения проблемы и роста ее масштаба состоит в рассмотрении задач, в которых данные распределены по узлам сети, а сбор всех данных в одном месте является одновременно и дорогим, и нерациональным, поскольку данные заведомо не помещаются в память одного компьютера.

Распределение данных по некоторой сети может соответствовать деловой или производственной логике прикладной области (например, географически распределенное производство компонент оборудования, сетевая структура поставщиков и потребителей торговых сетей, медицинские и социологические данные о населении, архивы специальных государственных структур, метеорологические данные, банки генома и др.). В таких приложения допустим лишь сетевой *обмен моделями* или группами моделей, обучаемыми локально.

Реалистическим приближением к этой ситуации является распределенное обучение ансамбля классификаторов, выполняемое на многопроцессорной ЭВМ, причем весь объем данных превышает возможности отдельных узлов. Такая задача рассмотрена в работе [1] в контексте использования суперкомпьютерных вычислений[20] для обучения машин задаче классификации.

Подход состоит в построении независимых комитетов на каждом процессорном узле и далее в объединении всех моделей в один ансамбль путем простого голосования. Алгоритм разделения на малые подмножества (pasting small votes — “bites”) был предложен *Лео Брейманом* еще в 1999 году [31]. В работе [1] он обобщен на случай распределенной обработки данных без коммуникаций.

[20] Работа [1] частично спонсировалась Сандийскими Национальными лабораториями США.

АЛГОРИТМ DIVOTE

1. Дан набор обучающих примеров $(x_1, y_1), \ldots, (x_m, y_m)$ с метками $y \in \{1, \ldots, k\}$.
2. Разделить данные на T непересекающихся подмножеств D_t, разместить каждое из подмножеств на отдельном процессорном узле.
3. В каждом узле получить бутстреп-выборку размера N и обучить первый набор классификаторов. Далее повторять шаги 4–6 до выполнения критерия останова (стабилизации ошибки или заданное число итераций).
4. На шаге k вычислить ошибку $e(j)$ на данных, не включенных в подвыборку, использованную при обучении (речь идет только о данных локального множества D_t на каждом процессорном узле), и вычислить вероятность $c(j)$ переиспользования примеров без ошибок:

$$e(j) = p \cdot e(j-1) + (1-p) \cdot r(j), \quad p = 3/4,$$

$r(j)$ — вероятность ошибки текущего комитета большинства из j классификаторов на всем локальном множестве D_t,

$$c(j) = e(j)/(1 - e(j)).$$

5. Сформировать новую обучающую выборку из N примеров. Примеры просматриваются случайно, с возвратом. Если пример ошибочно классифицируется множеством классификаторов данного узла, для которых он не встречался в их обучающих выборках[21], то он включается безусловно. Пример, верно классифицируемый этими классификаторами, включается в подвыборку с вероятностью $c(j)$.
6. Обучить $(j+1)$–й классификатор $h_{t,j+1}$ на выборке, полученной на шаге 5.
7. Собрать все полученные модели классификаторов на одном процессорном узле и сформировать окончательную гипотезу-классификатор

$$H(x) = \operatorname{sign}\Big(\sum_{t=1}^{T} \sum_{j} h_{t,j}(x)\Big)$$

для случая двух классов, или

$$H(x) = \operatorname*{argmax}_{y \in \{1,\ldots,k\}} \Big[\sum_{t,j:h_{ij}(x)=y} 1 \Big]$$

для k классов.

Алгоритм `DIvote` был интенсивно экспериментально исследован [1] в сравнении с онлайн-методами бустинга и другими алгоритмами. Полученные высокие показатели по точности комитета авторы объясняют высокой

[21] Такая оценка ошибки обычно называется *out-of-bag error* (ошибка на оставшихся данных).

степенью попарного разнообразия классификаторов, оцениваемой по так называемой $\varkappa$–метрике:

$$\theta_1 = \frac{\sum_{i=1}^{k} C_{ii}}{m}, \quad \theta_2 = \sum_{i=1}^{k}\Big(\sum_{j=1}^{k}\frac{C_{ij}}{m}\cdot\sum_{j=1}^{k}\frac{C_{ji}}{m}\Big), \quad \varkappa = \frac{\theta_1-\theta_2}{1-\theta_1}. \qquad (13)$$

Здесь матричные элементы C_{ij} равны числу примеров, которые первый классификатор относит к i-му, а второй к j-му классу. Показатель $\varkappa$ равен нулю, если два классификатора дают совпадающие ответы случайно, и близок к единице, если классификаторы точно соответствуют друг другу на каждом примере. Экспериментальные зависимости ошибок от статистики пар показывают, что малые значения ошибок наблюдаются при высокой степени диверсификации моделей ансамбля.

Перспективы

Рассмотренные в лекции алгоритмы разделения больших задач обучения на множество отдельных подзадач с последующим объединением моделей в ансамбли позволяют значительно повысить точность прогнозирования в условиях ограничения на вычислительные ресурсы и возможность одновременной обработки всего массива данных. Однако, в своем большинстве, алгоритмы обучения многократно переиспользуют данные, хотя и в параллельном режиме.

Дальнейшее масштабирование систем классификации, по мнению автора, возможно при разработке алгоритмов, использующих ровно *один проход* по подмножеству данных (one-pass learning algorithms). Алгоритмы такого класса позволят значительно расширить области применения масштабных систем классификации с наступлением эры доступных многоядерных процессоров. В настоящий момент типичным является применение двухядерных систем, но не за горами 8 и 16-ядерные процессоры для персональных компьютеров. Такой уровень параллельности еще нельзя назвать массивным, поэтому основной ресурс — это обращение к дисковой памяти (к базе данных) при потоковой обработке.

Автор прогнозирует значительное повышение интереса и рост числа публикаций по однопроходным параллельным алгоритмам.

Второе важное обстоятельство — отсутствие строгих теоретических обоснований для многих, интересных в прикладном плане, разработок.

В частности, затруднительно признать окончательно, что поняты все причины повышения точности алгоритмов с бутстреп-выборками (класс бэггинга). Эти алгоритмы наиболее эффективно распараллеливаются, но нет ясных, *оцениваемых одновременно с обучением*, критериев выбора их параметров — числа моделей в комитете, достаточного объема каждой из подвыборок, а также рациональной сложности каждой из моделей.

Возможности по независимому обучению комитетов классификаторов по географически или организационно распределенным фрагментам данных могут в будущем значительно повлиять на бизнес-логику обращения с данными в крупных компаниях и государственных учреждениях. *Обмен моделями* вместо дорогого или затрудненного по соображениям информационной безопасности *обмена данными* может стать основным механизмом синхронизации бизнес-процессов. Это обстоятельство уже широко осознается специалистами в области мультиагентных информационных систем. Очередь за распределенными системами поддержки принятия решений.

Благодарности

Автор благодарит коллег по компании Нейрок Техсофт, стимулирующих поисковые разработки своими достижениями в практическом использовании методов обучения машин в договорных работах, проводимых компанией.

Литература

1. *Nitesh V. Chawla, Lawrence O. Hall, Kevin W. Bowyer, W. Philip Kegelmeyer.* Learning ensembles from bites: A scalable and accurate approach // *Journal of Machine Learning Research.* – **5**(Apr):421–451, 2004.
   URL: http://www.jmlr.org/papers/v5/chawla04a.html
2. *Bishop C. M.* Neural Networks for Pattern Recognition. – Oxford University Press, 1995.
3. *Vapnik V. N.* The nature of statistical learning theory. – Springer-Verlag, 1995.
4. *T. Hastie, R. Tibshirani, J. Friedman.* The Elements of statistical learning. Springer, 2001
5. *Gavin Brown.* Diversity in neural network ensembles. – PhD Thesis, Univ. of Birmingham, 2003.
   URL: ftp.cs.bham.ac.uk/pub/authors/J.L.Wyatt/gxb_thesis.pdf

6. *Журавлев Ю. И.* Об алгебраическом подходе к решению задач распознавания или классификации // В сб. *«Проблемы кибернетики»*, 1978, т. 33, с.5.

7. *Воронцов К. В.* Лекции по алгоритмическим композициям. – 2006.
URL: http://www.ccas.ru/voron/download/Composition.pdf

8. *Yoav Freund, Robert E. Schapire.* A short introduction to boosting // *Journal of Japanese Society for Artificial Intelligence.* – **14** (5): 771–780, September, 1999.
URL: http://www.cs.princeton.edu/simschapire/uncompress-papers.cgi/FreundSc99.ps

9. *R. Avnimelech, N. Intrator.* Boosting regression estimators // *Neural Computation*. – **11**: 491–513, 1999.
URL: http://citeseer.ist.psu.edu/avnimelech99boosting.html

10. *Robert E. Schapire.* The strength of weak learnability // *Machine Learning*. – **5** (2): 197–227, 1990.
URL: http://citeseer.ist.psu.edu/schapire90strength.html

11. *Drucker H.* Boosting using neural networks // In: *Combining Artificial Neural Nets: Ensemble and Modular Multi-Net Systems* (Edited by *A. J. C. Sharkey*). – 1998.
URL: http://www.boosting.org/papers/Dru99.pdf

12. *Eric Bauer, Ron Kohavi.* An empirical comparison of voting classification algorithms: Bagging, boosting, and variants // *Machine Learning*. – **36** (1999) pp. 105–142.
URL: http://citeseer.ist.psu.edu/bauer99empirical.html

13. *Yoav Freund, Robert E. Schapire.* A decision-theoretic generalization of on-line learning and an application to boosting // *Second European Conference on Computational Learning Theory.* – 1995.
URL: http://citeseer.ist.psu.edu/89601.html

14. *Robert E. Schapire.* Theoretical views of boosting and applications // *Algorithmic Learning Theory*, 10th International Conference, ALT '99, Tokyo, Japan, December 1999, Proceedings.
URL: http://citeseer.ist.psu.edu/231645.html

15. *Терехов С. А.* Технологические аспекты обучения нейросетевых машин // Лекция для VIII школы-семинара *«Современные проблемы нейроинформатики»*. – М.: МИФИ, январь 2006. – с. 13–73.

16. *Gunnar Ratsch, Bernhard Schokopf, Sebastian Mika, Klaus-Robert Muller.* SVM and boosting: One class. – Tech. Report GMD RIRST, 2000.
URL: http://citeseer.ist.psu.edu/516656.html

17. *Gunnar Ratsch, Manfred K. Warmuth.* Efficient margin maximizing with boosting // *JMLR*. – **6** (Dec): 2131–2152, 2005.
URL: http://jmlr.csail.mit.edu/papers/volume6/ratsch05a/ratsch05a.pdf

18. Материалы Интернет-узла http://www.boosting.org/

19. *Gunther Eibl, Karl-Peter Pfeiffer.* Multiclass boosting for weak classifiers // *JMLR*. – **6** (Feb): 189–210, 2005.
URL: http://jmlr.csail.mit.edu/papers/volume6/eibl05a/eibl05a.pdf

20. *R. Meir, G. Ratsch.* An introduction to boosting and leveraging // In: *S. Mendelson* and *A. Smola*, editors, *Advanced Lectures on Machine Learning*, LNCS, pp. 119–184. – Springer, 2003.
URL: http://www.boosting.org/papers/MeiRae03.pdf

21. *Robert E. Schapire, Yoram Singer.* Improved boosting algorithms using confidence-rated predictions // *Machine Learning*. – **37** (3): 297-336, 1999.
URL: http://citeseer.ist.psu.edu/schapire99improved.html

22. *T. Hastie, R. Tibshirani, J. Friedman.* The elements of statistical learning: Data mining, inference, and prediction. – Springer, 2001.

23. *Leo Breiman.* Bagging predictors // *Machine Learning*. – **24** (1996), 123–140. Also, Univ. California Technical Report No. 421, September 1994.
URL: http://citeseer.ist.psu.edu/breiman96bagging.html

24. *Herbert K. H. Lee, Merlise A. Clyde.* Lossless online Bayesian bagging // *JMLR*. **5** (2004). – pp. 143-151.
URL: http://jmlr.csail.mit.edu/papers/volume5/lee04a/lee04a.pdf

25. *Alexey Tsymbal, Seppo Puuronen.* Bagging and boosting with dynamic integration of classifiers // In *Proceedings of PKDD 2000*, Lyon, France, *Lecture Notes in Artificial Intelligence*, Springer Verlag, volume 1910, pp. 116–125, 2000.
URL: http://www.boosting.org/papers/upload_30017_tsymbal1.pdf

26. *Jordan M. I., Jacobs R. A.* Hierarchical mixtures of experts and the EM algorithm // *Neural Computation*. – **6** (1994), pp. 181–214.
URL: http://citeseer.ist.psu.edu/jordan93hierarchical.html

27. *Steve Waterhouse, David Mackay, Tony Robinson.* Bayesian methods for mixtures of experts // *Advances in Neural Information Processing Systems* (NIPS), 1996.
URL: http://citeseer.ist.psu.edu/waterhouse96bayesian.html

28. *R. A. Jacobs, M. I. Jordan, S. J. Nowlan, G. E. Hinton.* Adaptive mixtures of local experts // *Neural Computation*. – **3** (1): 79–87, 1991.

29. *Гилев С. Е., Горбань А. Н., Миркес Е. М.* Малые эксперты и внутренние конфликты в обучаемых нейронных сетях // *Доклады Академии Наук СССР*. – 1991. – т. 320, № 1. – с. 220–223.

30. Справочник по прикладной статистике. В 2-х томах, под ред. *Э.Ллойда, У. Ледермана*. – М. Финансы и Статистика, 1989.

31. *L. Breiman.* Pasting bites together for prediction in large data sets // *Machine Learning*, **36** (2): 85–103, 1999.
URL: http://citeseer.comp.nus.edu.sg/137761.html

Приложение. Большие наборы данных для экспериментов

Разрабатываемые масштабируемые алгоритмы для решения задач классификации требуют, как правило, экспериментальной отработки и верификации. Встает вопрос об источниках данных соответствующего замыслу масштаба.

Пример такой базы данных — это данные по прогнозированию лесного покрытия на основе картирования (без применения удаленного зондирования территорий).

URL–адрес базы данных `Forest CoverType`:
http://kdd.ics.uci.edu//databases/covertype/covertype.html

Данные собраны в университете штата Колорадо (Remote Sensing and GIS Program, Department of Forest Sciences, College of Natural Resources, Colorado State University). Массив включает более полумиллиона примеров с 54 признаками и 7 выходными классами. Задача осложняется неравномерным распределением классов, но благоприятным моментом является отсутствие пропусков в данных.

Достигаемая точность классификации составляет примерно 70%.

Задачи

Задача 1. Созидательное (?) разрушение

Некто имеет в своем распоряжении N строго одинаковых копий одного классификатора, обладающего точностью p. Желая повысить точность путем комбинирования моделей, он вносит в выход каждой из копий свой источник случайных ошибок, причем ошибки от действия различных источников статистически независимы. Далее, полученные классификаторы объединяются в комитет. Можно ли таким способом достигнуть точности выше, чем исходная p?

Изменится ли ответ задачи, если независимые ошибки прибавляются ко *входам* тождественных копий обученного классификатора, объединенных в комитет?

Задача 2. Вероятность ошибки алгоритма Boost1

Используя арифметику вероятностей, показать, что ошибка тройки классификаторов в алгоритме `Boost1` составляет $3\alpha_2 - 2\alpha_3$, где α — вероятность ошибки одного классификатора.

Задача 3. (Без решения)

Получите оценки скорости деградации точности бэггинг-комитета при уменьшении числа примеров в каждой из бутстреп-выборок при сохранении общего числа членов комитета. Имеет ли смысл в каждой бутстеп-выборке иметь *большее* число примеров, чем в исходном наборе данных?

Сергей Александрович ТЕРЕХОВ, кандидат физико-математических наук, заместитель Генерального директора ООО «Нейрок Техсофт» (г. Троицк, Московская обл.). Область научных интересов — анализ данных при помощи искусственных нейронных сетей, генетические алгоритмы, марковские модели, байесовы сети, методы оптимизации, моделирование сложных систем. Автор 1 монографии и более 50 научных публикаций.