г Троицк, Московской обл.
Ул. Нагорная, д.9, офис 6.

Отдел Интеллектуальных Технологий, НейрОК Техсофт

Технологические аспекты обучения нейросетевых машин

Сергей А. Терехов

Лекция для VIII Всероссийской научно-технической конференции "Нейроинформатика -2006"

Технологические аспекты обучения нейросетевых машин

Сергей А. Терехов
ООО Нейрок Техсофт

В лекции представлен очерк эффективных методов практических вычислений в задачах обучения нейросетевых и других статистических моделей. Рассматриваются как особенности часто встречающихся информационных задач, так и алгоритмический “инструментарий”. Значительное внимание уделено намечающейся стандартизации в области обучающихся вычислительных машин.

В приложении приведен краткий обзор стандартизованного представления моделей на основе XML-структур языка PMML

This Lecture introduces a family of methods for effective practical neural and statistical computations. Both, the peculiarities of information processing tasks and algorithmic toolboxes are considered. Significant attention is paid to standardization of learning machine models.

A short review of Predictive Models Markup Language, an XML application, is also presented.

> *В основе твоих размышлений и рассуждений должны лежать данные, которые проверил ты сам*
>
> Дао Toyota

Введение: Технологии обучающихся машин для анализа данных

Технология в энциклопедическом определении это “...совокупность методов обработки, изготовления, измерения состояния, свойств, формы сырья, материала или полуфабриката, осуществляемых в процессе производства продукции”. В компании Toyota, производящей машины, использование технологий базируется на принципе “новая технология должна поддерживать человека, процесс и ценности”.

Искусственная нейронная сеть – это тоже специфический вид *обучающихся* машин, предназначенных для статистического прогнозирования и систематизации больших объемов информации. Создание таких машин – это производственный процесс, сырьем для которого служат данные[1], ресурс данных является определяющим[2] в

[1] Данные понимаются в широком смысле – числовые, знаковые и текстовые, мультимедийные, и др.
[2] Один из гигантов, компания Yahoo! первой ввела специальную должность – директор по данным (Chief Data Officer). Им стал Usama Fayyad.

производственном процессе. Продукция или товар – это компьютерные программы или их компоненты, электронные и оптические схемы и чипы, а также результаты их использования при практической обработке данных. Товаром также являются и *сами технологии*, позволяющие производить и тиражировать обучающиеся машины.

Под обучаемой машиной в таком, утилитарном, смысле понимается такая машина или ее программная модель, производство которой допускает максимально гибкое использование сырья (т.е. данных), при этом значительная часть[3] производственного процесса *выполняется самой создаваемой машиной*.

На западе, в США, ключом к успеху принято считать формализацию и документирование как самой технологии, так и процесса ее совершенствования. Это позволяет снизить стоимость владения технологией, а также относительно безболезненно обновлять кадровый состав. При высокой степени формализации многие этапы технологического процесса могут выполняться специалистами более низкой квалификации (более дешевыми, и менее требовательными). В японских фирмах (особенно в Toyota) считают, что технология, прежде всего, должна быть хорошо проверенной и содержать “встроенное” в нее качество итоговой продукции. Успех технологии не в документах, а в людях.

При современном уровне развития отрасли обучающихся машин многие технологические этапы просто не мыслимы без людей – носителей этих технологий. Однако, если сбросить со щита аспекты стандартизации, повторяемости и доказательности свойств обучения, то использование нейронных сетей рискует оказаться непомерно дорогим для большинства потребителей. В области обучения машин приемлемым, по-видимому, можно считать такой уровень формализации технологии, при котором *этой технологии* можно обучать специалистов (не путать с обучением *нейросетевым алгоритмам* и архитектурам!).

Предыдущие лекции автора [9-11] ограничивались вопросами теории и алгоритмов информационного моделирования. Данная лекция адресована, прежде всего, начинающим специалистам, которые разрабатывают[4] технологии моделирования и прикладные обучающиеся системы. Автор надеется, что лекция будет также полезна и специалистам, использующим или внедряющим в своих компаниях нейросетевые и информационные решения.

В этой лекции немного формул, поскольку она, в основном, содержит рекомендательные положения и примеры, относящиеся к технологическим аспектам моделирования и обучения моделей. Значительная часть материала относится не только к нейросетевым прикладным алгоритмам, но и другим статистическим методам с обучением, причем особое внимание уделено симбиозу подходов. При этом автор не отклонился от своего принципа подробного

---

[3] Идеалом служит ситуация, когда в процессе производства человек лишь формулирует *цели* и предоставляет источник *данных*, а все остальное делает сама машина. Известные автору примеры самовоспроизводящихся программ, к сожалению, не преследуют иную цель, кроме самовоспроизводства.

[4] Для читателей, заинтересованных в основном, конкретными нейросетевыми архитектурами и методами их обучения можно порекомендовать вводный курс лекций автора [15], URL http://alife.narod.ru/, а также серию издаваемых в МИФИ брошюр по материалам лекций для ежегодных конференций “Нейроинформатика”. В конце 2005 года в издательстве “Вильямс” выходит перевод классической монографии Саймона Хайкина “Нейронные сети”. Не пропустите!

изложения в каждой лекции одного из современных алгоритмов; здесь это метод SVM (support vector machine).

Первый раздел лекции вводит в проблематику практических задач, для которых предназначены обучающиеся машины, обсуждается специфическая “трудность” этих задач. Далее рассматриваются особенности самих машин и алгоритмов, отличающие поисковые разработки от производственных вариантов. Даются практические рекомендации по “портфелю” алгоритмов, которые в совокупности могут обеспечить достаточно полный анализ прикладной задачи. Некоторые элементы технологий обучения и тестирования иллюстрируются на примерах конкурсных задач классификации (WCCI 2006). В приложении к лекции дан краткий обзор стандартизованного XML-языка описания моделей обучающихся машин PMML (Predictive Model Markup Language) и обсуждаются преимущества стандартизации моделей. Как обычно, лекция завершается списком литературы, преимущественно доступной в сети Интернет, и учебно-исследовательскими задачами, которые автор готов обсудить по электронной почте.

Задачи, которые решают нейронные сети

Практические потребности в обучающихся машинных методах обработки информации можно условно разбить на две большие группы. К первой относятся задачи поиска закономерностей в имеющихся данных, компактное аналитическое описание *уже известной* пользователю информации. Это направление ассоциируется с термином “промысел данных” (data mining). Вторая группа задач связана с прогнозированием значений *будущих* наблюдений. Иногда только это, второе, направление относят к области обучения машин (machine learning).

Между этими направлениями нет принципиального антагонизма, ведь алгоритмы, моделирующие известные данные, могут использоваться и для предсказаний, с другой стороны, обоснование прогнозов обучающимися машинами возможно, во многом, только путем статистических экспериментов на имеющихся к этому моменту данных. Поэтому в лекции под обучением машин будут подразумеваться общие для обоих направлений принципы.

Далее будут рассмотрены основные типы информационных задач. Нужно сразу отметить, что формальное распределение всех информационных задач “по полочкам” также выглядит искусственным, и этот стиль изложения, скорее, дань традиции.

Принципиально то, что заказчики и потребители решений, различают разные задачи по их *целевому назначению*, а не по используемым типам алгоритмов. Очень трудно внедрить в производство “систему классификации клиентов банка”, но можно и нужно внедрять “систему снижения расходов банка за счет дифференцированного обслуживания разных групп клиентов”. Разница здесь отнюдь не в названии! Целевая формулировка принципиально влияет на критерии оптимальности требуемой обучающейся машины и предполагает такую математическую постановку проблемы, при которой результат может быть встроен в имеющийся технологический или бизнес-процесс.

В практике автора встретился следующий отрицательный пример. В разработке требовалось сегментировать предприятия и отделы сети клиентских услуг по наборам услуг, наиболее востребованным клиентами в каждой группе предприятий. Определение цели такой сегментации было отложено заказчиком “на потом”. В результате, стремясь к максимальной устойчивости системы, надежному выявлению всех типов имеющихся предприятий, объяснению и прогнозированию структуры их прибыли, было выявлено около 50 групп. Система позволяла удобно

представлять результаты работы визуально и показывала высокую точность при классификации вновь вводимых предприятий. Однако перед этапом внедрения выяснилось, что структура взаимоотношений с поставщиками, а также логистическая организация сети допускает дифференцированный подход к управлению не более, чем пятью категориями предприятий. Более глубокая дифференциация бизнеса на данном этапе противоречит отраслевому способу производства и из-за скрытых потерь и необходимости уменьшения объемов партий поставок экономически не целесообразна. Замечательный робастный классификатор с инновационными алгоритмическими достижениями был положен на полку[5].

Различные примеры прикладных задач в разных контекстах обсуждаются по всему ходу лекции.

Задачи классификации

Обучаемая машина при решении задачи классификации относит каждый вектор данных к одному из заранее выбранных классов. В естественно-научном смысле *класс* – это совокупность объектов, выделенных по некоторому набору признаков. Если признаки классов четко определены и легко устанавливаются у всех объектов, то отнесение объекта к классу составляет простую логическую процедуру. Такая процедура может быть выполнена программируемой машиной-автоматом.

Необходимость в *обучаемых* машинах возникает тогда, когда объекты описываются большим множеством опосредованных и ненадежно наблюдаемых (зашумленных и противоречивых) признаков, и путем логического анализа установить принадлежность каждого объекта к одному из классов не представляется возможным. Вместе с тем, имеется набор объектов, для которых принадлежность к классам уже каким-то образом установлена, например, по факту свершения некоторого события с этим объектом в прошлом. Этот набор объектов с известными классами играет роль *обучающей выборки*. При обучении машина модифицирует свою структуру или параметрическое описание, с целью снижения ошибки классификации известных данных до такого уровня, при котором *ожидаемая* ошибка классификации будущих новых примеров будет минимальной.

Поскольку основное свойство классов – формальное описание точными признаками – в реальности утрачено, то в прикладных задачах термин *класс* может трактоваться самым широчайшим набором способов. Под выбором класса может пониматься ответ на вопрос – убедит ли покупателей шампуня новый рекламный ролик, вырастет ли завтра курс ценной бумаги, проводится ли покупка в интернет-магазине по контрафактной кредитной карточке, и множество других вопросов, обращенных к *будущим* свойствам или возможным событиям с классифицируемыми объектами. Именно направленность в будущее составляет главную ценность обучаемых машин для делового заказчика. Целевая установка для машины – *точность*.

Впрочем, остаются также и задачи классификации уже имеющихся объектов или примеров. Здесь, в основном, целевая потребность состоит в автоматизации рутинного ручного труда, а основной целью, помимо точности, выступает *экономия времени*.

[5] Разумеется, в последствии разработанная технология использовалась в нескольких успешных проектах, но первый ее заказчик остался неудовлетворенным.

Классификация является самой фундаментальной и наиболее ясно воспринимаемой задачей в области обучения машин. Минимальный шаг в переходе от *программируемых* автоматов и компьютерных программ с предписанной функциональностью к *обучаемым* машинам и программам состоит во встраивании обучаемого классификатора, автоматически выбирающего одну из заранее приготовленных программ. Классификатор в таких системах принимает решение на основе вектора *внешних* наблюдаемых признаков той ситуации, в которой сейчас находится машина.

Нужно отметить, что весомое большинство используемых на практике обучающихся систем, которые сами потребители называют *интеллектуальными*, сегодня не пошло далее этого минимального шага. Более того, осторожные производители, зачастую доверяют классификаторам только тогда, когда, фактически, любая из исполняемых программ не приведет к аварии или другому недопустимому поведению машины. Классификатор лишь позволяет выбрать либо более экономную, либо более "комфортную" программу в данной ситуации, повышая качество потребительских свойств результата, но не определяя их суть. В таких приложениях классификатор, по существу, является лишь опционной компонентой, и этот "интеллект" можно без больших потерь отключить.

Такова реальность, и разработчики технологий обучаемых машин должны ее учитывать.

> **Совершенствование автоматики автомобиля**. В процессе движения автомобиль может снабжаться дополнительными датчиками, измеряющими такие параметры, как химический состав выхлопов, вибрации механизмов, поля температур узлов и др. Вектор наблюдений обрабатывается бортовым классификатором состояний, результат классификации используется для выбора режимов подачи топлива и управления передачей.

Что составляет суть "интеллектуальности" машины или программы, содержащей обучаемый классификатор? Такая программа получает новые степени свободы за счет возможности использования дополнительной плохо формализованной информации из внешнего мира, которая без блока обучения вынужденно игнорируется.

При согласовании постановки задачи классификации с потребителями необходимо убедиться в следующих позициях:

- Для обучения системы имеется достаточный объем достоверных обучающих данных, и, при необходимости, есть возможность собрать дополнительные данные. На практике обучение рекомендуется проводить на последовательно увеличивающихся выборках данных, что позволяет ответить на вопрос о достаточности объема экспериментально. При росте числа примеров оптимальная сложность классификатора также растет.

- Данные должны содержать примеры для *всех* затребованных классов. На практике могут встречаться ситуации, когда у заказчика имеются примеры только одного класса, и требуется отличить этот класс от всего остального. Такая задача должна ставиться, как задача оценивания области, занятой данными класса в пространстве признаков (нахождения *носителя* класса).

- Определены условия, критерии, и данные для тестирования системы. Заказчик может проводить также независимое тестирование разработанной системы самостоятельно, с использованием наборов данных, составляющих его интеллектуальную собственность. В некоторых случаях результаты такого тестирования могут составлять важную промышленную тайну, охраняемую от конкурентов. Вы можете быть и не оповещены даже о факте внутреннего тестирования, но результаты согласованных *базовых* тестов должны быть доступны. В противном случае, не возможно совершенствование системы.

- Определено место задачи классификации в технологическом или производственном процессе у потребителя. Система классификации должна иметь возможность получить входную информацию той же структуры, что и при обучении и тестировании, и определен способ, как и *когда* будет извлечен и использован результат. “Когда” здесь также является важным. Нетрудно представить ситуацию, в которой примеры для классификации поступают в систему потоком, скажем, один пример в 0.001 секунды. Классификатор по итогам тестирования имеет высокую точность в 99%. Если при ошибочной классификации в использующем классификатор приборе подается звуковой сигнал или вспыхивает лампочка, то она будет непрерывно гореть!

- Применение обучаемого классификатора должно быть обосновано. Факт наличия программы или алгоритма у разработчика и абстрактное стремление к инновациям не является достаточным основанием! Необходимость применения классификатора должна быть ясно осознана потребителем. Исключение составляют случаи, когда обучаемая машина просто экономнее или более производительна, а потери от неконтролируемых статистических ошибок невелики.

Основная постановка задачи классификации обычно формулируется для случая двух классов. При большем числе классов задача может быть сегментирована на бинарные подзадачи различения каждого класса от всех остальных, либо различения каждой пары классов друг от друга, с последующим голосованием. Многие типы информационных задач могут рассматриваться, как варианты классификации с несколькими классами.

Аппроксимация данных

Задача с несколькими дискретными исходами представляет собой задачу классификации. Если число возможных вариантов велико или непрерывно, то обучаемая машина имеет дело с аппроксимацией или регрессией данных. Результатом такой машины является функциональная зависимость выхода от вектора входов.

При формулировке задачи регрессии нужно учитывать, что получаемая зависимость следует из данных только в вероятностном смысле. Искомая функциональная связь, в причинно-следственном смысле, может в действительности и не существовать, либо реальности соответствует функция, зависящая от еще каких-то переменных, которые отсутствуют в имеющихся данных.

Вследствие конечности набора данных, на практике трудно различить две ситуации, когда дисперсия регрессионной зависимости вызвана шумом в данных, и когда она обусловлена не учетом важных факторов. В последнем случае, модель данных может в новых условиях вводить потребителя в серьезное заблуждение, хотя формально статистические критерии достоверности выявленной зависимости могут быть удовлетворены.

Еще один подводный камень в задаче аппроксимации заключается в возможной неравномерности распределения ошибки в пространстве признаков. Потребности заказчика, использующего модель, могут в течение длительного времени быть сосредоточены в какой-то относительно небольшой области пространства. Это совершенно естественно – например, происходит отработка конкретного режима технического устройства в ограниченной области параметров. Если ошибка модели *в этой области* велика, то возникают неприятности, хотя в целом, на полном корпусе данных, модель удовлетворяет всем выставленным требованиям по точности.

Существуют регулярные способы избежать этой проблемы. Для этого при обучении машины не следует ограничиваться оцениванием только математического ожидания значения функции в точке – что, например, делает аппроксимирующая нейронная сеть, уменьшающая квадрат ошибки. Одновременно с моделью среднего может обучаться и модель, аппроксимирующая локальную дисперсию и более высокие моменты распределения невязок. При этом совместное обучение максимизирует функцию правдоподобия (likelihood) в выбранной параметрической форме, с учетом статистики распределения шума. Математические вопросы обучения такой модели рассматривались в лекции [25]. В итоге, потребитель информируется о фактическом уровне ожидаемой точности в области интересов.

Тестирование регрессионных моделей является более сложным процессом, в сравнении с классификаторами. Особенно это касается способов использования аппроксиматора, связанных с его дифференцированием. Примером такого использования являются задачи управления, где нейронная сеть обучается модели *отклика* управляемой системы, а применяется для оценки *якобиана* выходных переменных по входам. Приближение функции не означает сходимость в приближении ее производных, если не приняты соответствующие меры[6]. Потребитель должен быть об этом четко предупрежден.

Вообще, при высокой размерности данных разумно избегать постановки задачи в форме непрерывной аппроксимации, если можно ограничится задачей классификации с несколькими классами. Классы соответствуют попаданию значений прогнозируемой функции в определенные диапазоны значений.

Заказчик может предлагать постановку в форме регрессии просто исходя из того, что выходные значения данных принимают произвольные значения (обычно из некоторого отрезка). При обсуждении же *целей* создания обучаемой машины устанавливается, что *использование* разработки состоит в принятии конечного числа решений при достижении прогнозируемой переменной некоторых пороговых уровней (цены, доходности, прочности, массы, размера области коррозии и т.п., в зависимости от приложения). В таких условиях применение классификатора обеспечит более устойчивое и робастное к шуму решение.

[6] Одновременная аппроксимация нейронной сетью и функции, и ее производных по дискретному набору зашумленных значений *только* самой функции многих переменных представляет собой серьезную математическую проблему. Приложения таких алгоритмов связываются с обратными задачами и задачами управления. Аспиранты, обратите внимание.

Кластеризация и визуализация данных.

В практике возникают задачи с данными, основной корпус которых не содержит меток классов. Здесь потребителя интересуют вопросы о самом количестве возможных классов и общей структуре данных.

> **Анализ потребительских корзин**. Многочисленными исследованиями установлено, что мы с вами, как покупатели товаров в супермаркетах, магазинах косметики, аптеках и т.п. действуем согласно некоторому набору поведенческих сценариев и предпочитаем покупать товары или пользоваться услугами в достаточно устойчивых сочетаниях. Однако, для категорий покупателей разных торговых сетей типичные "корзины" покупок могут значительно различаться. Это обуславливает потребность в проведении специализированных исследований и, часто, разработку индивидуальных алгоритмических решений для конкретных заказчиков. Проблема состоит в выявлении групп схожих наборов покупок в базах данных кассовых транзакций или данных маркетинговых опросов. При этом "вектора" покупателей являются крайне разреженными (при ассортименте в несколько тысяч наименований, одна покупка может содержать 3 - 30 товаров). Они также могут частично сопровождаться метками классов, такими как, участвует ли покупатель в программе скидок, или имеет ли он карточку постоянного покупателя, выбирает ли товары по рассылаемым каталогам, или пользуется еще какими-то преимуществами современных маркетинговых технологий. Конечная цель внедрения обучаемых систем анализа данных состоит в упрощении процесса управления поставками, снижении складских запасов, уменьшении потерь при продаже скоропортящейся и дорогостоящей продукции, и, разумеется, повышении комфорта покупателей, что положительно сказывается на устойчивости и расширении бизнеса.

Обучение машины производится *без учителя*, поскольку его сигналы отсутствуют либо не полны или не достоверны. В задачах подобного типа основное внимание нужно уделять статистической устойчивости выявленных кластерных образований в массиве данных. С потребительской точки зрения, решение должно сопровождаться наглядной визуализацией результатов – ведь при помощи машины заказчик, возможно, впервые наблюдает целостную картину данных.

При визуализации решается проблема "упаковки" многомерного набора признаков в двумерное пространство экрана компьютера или листа отчета. Однозначного наилучшего во всех смыслах решения этой задачи не существует, и в практике используются различные приближения – нейросетевые карты самоорганизации Кохонена [5], методы главных или независимых компонент [6], и другие методы, устанавливающие одностороннюю связь между метрическими отношениями в многомерном пространстве с двумерными координатами. Одно из простейших решений состоит в применении отображении Саммона [5], предложенного еще в 60-х годах. Карта Саммона строится путем оптимизации функционала согласования относительных расстояний между центрами кластеров в исходном многомерном и двумерном пространстве:

$$Q(\vec{x}) \sim \sum \frac{\left(D\left(\vec{X}_i - \vec{X}_j\right) - d\left(\vec{x}_i - \vec{x}_j\right)\right)^2}{D\left(\vec{X}_i - \vec{X}_j\right)^2}$$

Здесь большие буквы относятся к многомерным, а малые – к двумерным координатам кластеров. Оптимизация проводится по значениям двумерных

координат. Хотя задача построения такой карты относительно трудоемка и не масштабируется на большое число кластеров, она служит удобным подспорьем в ежедневной работе.

Особо нужно остановиться на проблеме визуализации многомерных данных, изменяющихся во времени (многомерных временных рядов). Здесь цель состоит не в группировке схожих состояний, а в кластеризации *схожих сценариев изменения* состояний.

> **Анализ динамики складских запасов**. Снижение непроизводительных затрат на хранение излишков запасов и убытков, связанных с недостатком сырья или товара на складе, в магазине, или на участке конвейерной линии, требует понимания, как устроена динамика потребления запасов. Выявление схожих сценариев потребления позволяет производить пополнение складов “пакетами” однотипно используемых ресурсов, что стабилизирует отношения с оптовыми поставщиками и упрощает логистику. Обучаемая система призвана выявить паттерны схожей динамики на разных масштабах времени, определяемых технологическими процессами предприятия.

Решение такой задачи может основываться на построении цепочки кластерных структур. Первичная карта кластеров объединяет схожие состояния, безотносительно моментов времени их возникновения. Динамика процесса на первичной карте выглядит, как последовательные “перескоки” вектора состояния из кластера в кластер. Для заданного масштаба времени собирается статистика таких перескоков (в виде частот посещения кластеров, или частот переходов между парами кластеров). Гистограммы, соответствующие выбранной статистике, служат *входными* векторами для следующей, *вторичной*, кластерной карты. Вторичная карта объединяет схожие гистограммы, которые являются отражением схожих типов динамики. Процесс построения цепочки карт, при необходимости, может быть продолжен.

Построенные машиной визуализируемые кластерные структуры могут для наглядности снабжаться метками тех примеров, для которых такие метки известны. Приписывание наиболее часто встречающейся метки всем примерам, попадающим в данный кластер, есть, суть, первое приближение к решению уже обсуждавшейся задачи классификации.

Прогнозирование временных рядов и оценивание рисков

Общим свойством этих задач является то, что запрашиваемый результат заведомо носит вероятностный характер, и это четко осознается потребителем. Цель обучаемой машины состоит в предсказании распределений вероятности следующего или нескольких будущих состояний динамики системы или процесса. При формулировке задачи важными являются две посылки. 1) Предыдущая история может быть суммирована в виде вектора состояния, процесс эволюции которого является Марковским. Это означает, что распределение вероятности сценариев будущего определяется только текущим вектором состояний. 2) Справедливо предположение о схожести динамики будущего, если она стартует из близких состояний настоящего. Например, предположение о том, что рынки ценных бумаг схожим образом реагируют на похожую информацию, а клиенты, берущие кредит, имеют одинаковые причины и мотивы для его потенциального невозврата, если их финансовые, социальные и другие параметры одинаковы.

Проверка выполнимости этих требований и связанный с ними выбор переменных вектора состояния составляют основную сложность задачи прогнозирования. Если вектор признаков, отражающих базовые свойства прошлой истории финансового

ряда, или составляющих содержательное описание клиента банка, уже построен, то задача прогнозирования сводится к классификации векторов или, реже, аппроксимации вектор-функций.

Часто набор признаков ограничивается естественными возможностями заказчика по сбору данных о клиенте или рынке. Тогда задача обращается в плоскость поиска наиболее достоверной степени подробности прогнозируемого распределения вероятности, при ограничениях на информативность входных векторов. Методы синтеза обучаемой машины могут включать алгоритмы поиска оптимума с ограничениями.

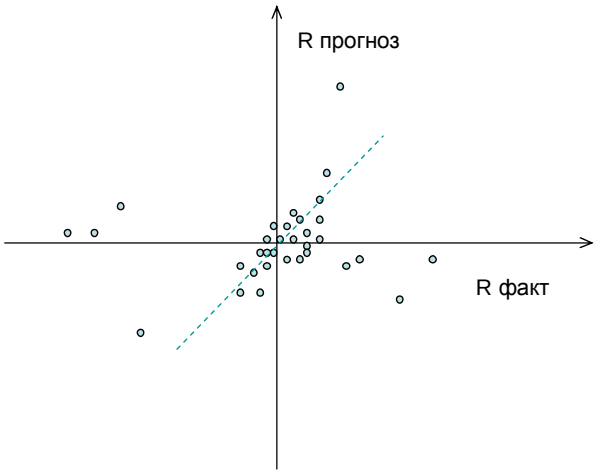

Рис 1. “Плохой” прогноз финансового временного ряда при высоком уровне корреляции предсказываемой и фактической доходности.

Подробность представления распределения вероятности должна учитывать также цели и способ использования потребителем. Так, риски нежелательных исходов оцениваются интегралами “хвостов” распределений, точность же представления основной массы типичных примеров не существенна и не будет востребована.

На Рис.1 приведен схематический пример результатов работы системы прогнозирования рынка, когда формальные критерии точности противоречат практическим потребностям. Прогнозы доходности при ее высоких абсолютных значениях, представляющие основной интерес для трейдера или управляющего портфелем, ошибочны даже с точностью до знака. Меж тем, прекрасные результаты и высокий коэффициент корреляции обусловлены высокой точностью прогнозов случаем с малой доходностью, т.е. там, где это никому не нужно.

К задаче оценивания рисков мы еще вернемся по ходу лекции.

Другие задачи

Приложения обучаемых машин полны также и другими задачами, которые могут быть напрямую и не связаны с прогностическим анализом данных. Среди них – оптимизация суррогатных моделей сложных технических систем, использование нейросетей, как пробных функций в вариационных математических постановках задач математической физики. Имеются задачи ремонта данных, выявление выбросов и расстроек динамики временных рядов. Важное место занимает оптимизация при обучении с подкреплением, с попутным решением сторонних практических задач. Уже отмечалась задача управления, которая в ряде

практических случаев может сводиться к выбору одного из режимов или одной из программ управления.

Для получения разносторонней картины имеющегося многообразия применений можно попробовать[7] задать вопрос типа “Machine learning applications” поисковой системе Google.

Почему информационные задачи сложны?

Программирование машин относится к одному и самых сложных видов созидательной деятельности. Программирование *обучаемых* машин, по теоретическому замыслу, должно было бы стать более простым для человека – ведь значительную часть работы берет на себя сама машина. На деле, аппетит приходит во время еды, и повсеместное внедрение компьютеров и других автоматов, смещает тяжесть проблем в иную, системную плоскость. Круг потребителей машин существенно расширился, повысились требования к надежности, простоте управления и, главное, интуитивной “предсказуемости” того, как должна была бы вести себя машина, если бы на ее месте был я[8].

“Золотое время”, когда модели обучаемых машин не покидали стены научных лабораторий и компьютеры разработчиков-ученых, уходит. В исследовательских условиях в центре внимания были собираемые вручную таблицы чисел, результаты использовались при проведении экспериментов на соседнем стенде, и все изготавливалось в штучных количествах.

Если миллионы машин каждое мгновение решают миллионы задач классификации в причудливой смеси физической и виртуальной реальности, то все статистически достоверные и недостоверные события обязательно происходят, все данные, которые могли быть неполными или испорченными, обязательно такими окажутся, а чудеса с данными случаются каждые 10^{-3} секунды. Реально они будут происходить гораздо чаще, но мы с вами, с нашими скоростями обработки нервных импульсов, просто не сможем этого заметить.

Сложность создания обучаемых машин носит системный характер. Коснемся некоторых составляющих этой сложности, которые нужно учитывать уже сейчас.

- **Неполнота обучающих данных** и информации об их природе. Неполные данные – это отсутствие некоторых значимых входов или сигналов на имеющихся входах, неточные входные сигналы, недостоверные выходы. Статистика распределения данных всегда принципиально не полна.

- **Противоречивость данных** и другие источники информационного шума. Данные поступают от внешних сенсоров и систем накопления, передаются по неидеальным каналам, хранятся на неидеальных носителях. При росте объемов данных дополнительно вмешивается фактор времени. Часть признаков в векторах данных успевает устаревать к моменту поступления остальных фрагментов. Остановки и синхронизации при сборе данных не всегда допустимы, поэтому на практике мы имеем дело со смесью признаков и параметров, относящихся к изменяющимся

[7] Нужен ли вам для ориентации в полученном наборе из 20,000,000 ссылок помощник? Им может стать обучаемая машина!

[8] А иначе зачем мне нужна такая непослушная машина?

объектам в непредсказуемые моменты времени. Такие данные почти наверняка содержат противоречия.

- **“Проклятие” размерности**. С ростом числа анализируемых признаков геометрия многомерных пространств работает против статистики. Можно было бы ожидать, что при разумном числе обучающих примеров достижимы близкие к теоретическим оценки вероятности правильной классификации вектора данных, так как в его окрестности имеется достаточное число близких примеров с известными классами [24]. Но в многомерных пространствах это не так. Например, даже для покрытия 10% объема 10-мерного куба требуется покрыть 80% длины каждого ребра. Таким образом, локальная статистика каждой области данных автоматически оказывается грубой.

- **Проблема масштабирования** алгоритмов. Задачу обучения классификатора для таблицы из 10,000 примеров и 10 входных признаков, по-видимому, нужно признать закрытой. Имеется большое число научных публикаций, в которых обоснованы устойчивые качественные алгоритмы обучения машин для таких масштабов. В свете практических потребностей первостепенное значение приобретают показали роста вычислительной сложности и ресурсов требуемой памяти алгоритмов при увеличении числа содержательных обучающих примеров. Реальность такова, что даже линейный рост сложности по числу примеров и размерности задачи начинает быть неприемлемым. Например, при обучении машины SVM (см. далее в лекции) на 1,000,000 примеров число получаемых базовых векторов памяти машины может достигать 10,000 для достижения сходимости валидационной ошибки. Такой размер классификатора оказывается нетехнологичным и по памяти и по скорости вычисления прогнозов.

- **Выбор модели**. Проблема выбора модели, обладающей преимуществами в точности решения задачи, является мета-проблемой, стоящей над задачей обучения. В идеале, обе задачи – и обучения, и оптимизации структуры должны решаться машиной самостоятельно и одновременно. Это лишь частично и приближенно выполнимо на практике, поскольку требования сходящейся точности обучения и оптимальности структуры не являются строго коллинеарными, и мы имеем дело с задачей многокритериальной оптимизации. Статистическое же сравнение множества моделей с целью выбора затрудняется проблемой масштабирования.

- **Проблемы безопасности и приватности**. Машина, выполняющая задачу от имени ее владельца, становится дополнительной угрозой возможности вторжения в его частную и деловую жизнь. Система классификации знает пароль доступа к источнику данных. Система управления заказами и покупками распоряжается финансовыми ресурсами и банковскими счетами, параметры которых она должна для этого в себе содержать. Это отдельная большая тема, выходящая далеко за рамки лекции, но не исключено, что именно проблемы безопасности (в разных

смыслах) окажутся центральными при массовом внедрении обучаемых машин.

- **Технические и организационные сложности.** При массовом использовании обучаемых машин встает проблема технологического сопряжения с другими программными компонентами, машинами, сенсорными устройствами. Корень зла здесь в том, что все должно производиться, внедряться и сопрягаться очень быстро. Но до сих пор зарядные устройства мобильных телефонов не совместимы, а кодировок текстов в компьютере больше, чем языков, используемых людьми. И все тут.

На этом, позитивно-ободряющем, фоне мы далее переходим к обсуждению машинных моделей и технологий для решения информационных задач. По ходу изложения вопросы системной сложности неизбежно будут возникать вновь.

Нейронные сети, которые решают задачи

Название раздела несет, как минимум, двойную смысловую нагрузку – в нем сочетаются и математические методы, составляющие основу обучающихся машин, и наши с вами нейронные сети, играющие не последнюю роль при решении информационных задач. И хотя основное содержание касается, прежде всего, алгоритмов и машин, начнем мы именно с мнений людей.

Консалтинговой компанией KDNuggets проводятся регулярные опросы специалистов по многим базовым аспектам в области обучения машин. Один из периодических опросов связан с использованием в приложениях различных алгоритмов и технологий анализа данных. Основные результаты [1] последнего (2005 г) опроса, в котором приняло участие 784 посетителя сайта, приведены ниже в таблице. Вопрос формулировался так: “Какие технологии анализа данных вы используете наиболее часто?”

Таблица 1. Наиболее часто используемые технологии анализа данных.

Технология или алгоритм	Количество “голосов”	%
Деревья решений/правил	107	14%
Кластеризация	101	13%
Методы регрессии	90	11%
Статистика	80	10%
Визуализация	63	8%
Нейронные сети	61	8%
Ассоциативные правила	54	7%
Методы оценок по ближайшим соседям	34	4%
SVM (методы минимизации структурного риска)	31	4%

Байесовы методы	30	4%
Анализ временных рядов и последовательностей	26	3%
Boosting (взвешивание и отбор данных для повышения точности комитетов классификаторов)	25	3%
Гибридные методы	23	3%
Bagging (усреднение прогнозов комитета моделей, обученных на выборках с возвратом)	20	3%
Генетические алгоритмы	19	2%
Другие	20	3%

Методики нейронных сетей занимают место в середине таблицы, лидируют методы решающих деревьев. Однако, причина, по которой в лекции приведена эта таблица, конечно, не в том, чтобы при выборе метода для решения прикладной задачи руководствоваться рейтингами популярности. Таблица иллюстрирует тот факт, что в современной практике нет единого универсального средства для всех задач анализа данных. Поэтому в дальнейшем изложении мы остановимся на описании группы методов, сравнительное и совместное использование которых позволит увидеть особенности проблемы с разных сторон. Выбор, по-прежнему, остается субъективным и отвечает предпочтениям автора.

Вероятностные деревья для задач классификации

Классификатор входных векторов в форме иерархии правил, по-видимому, является самым наглядным и простым в использовании инструментом. Джером Фридман (Jerome Friedman), один из классиков в области статистического анализа данных, называет деревья правил “методом, который всегда должен находится рядом, на полке” [24].

Существует несколько алгоритмов синтеза деревьев и вариантов “понимания” того, что является результатом их применения. Дерево может пониматься, как последовательный логический алгоритм с точками ветвления и гарантированным остановом в одном из терминальных состояний (листьев). В другом варианте, дерево определяет геометрическую сегментацию области многомерного пространства, занимаемой данными, на параллелепипеды, в пределах которых классифицируемые вектора относятся к одному классу.

Перечисленные трактовки весьма привлекательны и непосредственно воспринимаемы пользователями классификаторов. Но их особенность состоит в обращенности модели в *прошлое*, ориентации на объяснение данных, которые уже имеются к данному моменту.

Однако, если подходить к задаче классификации как к протяженному во времени технологическому процессу, при котором пользователи имеют дело с потоком *новых* примеров для классификации, более последовательной представляется вероятностная интерпретация алгоритма. Здесь дерево понимается, как модель условной плотности вероятности отнесения входного примера к одному из классов. Процесс классификации состоит в понижении условной энтропии распределения

классов для данного примера при прохождении его по узлам дерева. Идеальная классификация состояла бы в достижении нулевой энтропии (полной определенности) в листьях.

Основанный на вероятностном подходе алгоритм построения дерева рассматривался в одной из предыдущих лекций автора [9]. Этот алгоритм может быть рекомендован, и как самостоятельная методика обучения машины задаче классификации. Здесь же нас, прежде всего, интересует вопрос, как метод деревьев технологически сопрягается с другими методами.

Во многих случаях использование классифицирующего дерева является экспресс-методом, *первым шагом* на пути получения окончательного решения. В практике решение почти всегда разумно начинать с применения дерева. Это вызвано как высокой вычислительной эффективностью, так и пониженными (в сравнении с другими методами) требованиями к предобработке данных. В частности, данные могут непосредственно обрабатываться без предварительного масштабирования, нормализации, с использованием тех размерных шкал, в которых они исходно представлены.

Например, при классификации свойств химических смесей, концентрации разных компонент могут описываться как в массовых, так и объемных единицах, при этом основные компоненты могут измеряться в граммах, а микродобавки – в других удобных единицах. Это упрощает первичный процесс прохождения данных и снижает вероятность ошибок на первых этапах, когда задача для исследователя является новой, и терминология между потребителем и разработчиком еще не полностью согласована.

Второй важный момент – дерево выявляет наличие противоречий в данных и позволяет получить оценку степени информационного шума в данных. Эти показатели вычисляются на основе статистики “чистоты” листьев дерева. Наличие противоречивых примеров – когда одни и те же или близкие входные вектора относятся к разным классам – при обучении дерева не может быть устранено или “растворено” в результирующей модели (как, например, это происходит в нейронной сети). Соответствующие листья дерева в этом случае будут иметь остаточную энтропию, несмотря на малое число относящихся к ним примеров. Противоречия в последних, далее, легко выявляются (прямым просмотром).

Выявление ошибок и противоречий в данных является необходимым элементом любой технологии их анализа. Раннее информирование заказчика о реальном качестве данных позволит сэкономить уйму лишней работы на следующих этапах проекта – ведь часть противоречий в данных может возникать вследствие ошибок при их сборе, некачественной работы измерительного оборудования, недостоверной отчетности, или из-за ошибок в программном обеспечении сбора данных.

Третье достоинство вероятностных деревьев – возможность оценки сравнительной значимости признаков. По алгоритму построения дерева, решающее правило в каждом узле понижает суммарную энтропию условной вероятности при его применении. В каждом правиле задействован какой-то один признак. Уменьшение энтропии (прирост информации или *нег*-энтропии) измеряется количественно. В итоге, с каждым признаком ассоциировано суммарное уменьшение энтропии от всех правил, основанных на этом признаке. Входные признаки, указание значений которых приводит к значительному уменьшению энтропии, наиболее информативны.

Технологическая цепочка анализа данных обязательно должна включать оценки информативности признаков. Это не только позволяет построить более эффективные прогностические модели (см. Задачу 1), но и само по себе является важным результатом анализа. Иногда основная потребность заказчика как раз и состоит в том, чтобы ответить на вопрос, какие столбцы таблицы данных важны для прогноза, а сбор каких данных можно в дальнейшем не проводить. Результаты такого типа могут приводить к прямому экономическому эффекту.

> **Оценка кредитных рисков.** Поиск значимых факторов находится в центре потребностей в такой, в последнее время ставшей особенно актуальной, проблеме, как оценка кредитных рисков. При решении задачи классификации клиентов по вероятности несвоевременного возврата кредита имеется "идеологическое" противоречие полноты и "прямоты" собираемой информации с достоверностью этой информации, особенно, когда вопросы касаются фактических уровней и источников доходов, из которых будет погашаться кредит. Часто косвенные вопросы, типа "насколько заранее Вы планируете отпуск и услугами каких турфирм пользуетесь?" дают для классификатора более точную информацию, нежели прямые вопросы о степени стабильности доходов. Важную роль здесь также играют вопросы приватности и другие юридические аспекты, поэтому при создании систем классификации для оценки рисков целесообразно проработку объема и состава собираемых данных проводить во взаимодействии экспертов из различных областей (в т.ч., и в области анализа данных).

Детальный анализ обученных вероятностных деревьев может дать много другой полезной информации (например, выделить области наиболее нестабильного поведения прогнозируемой переменной, указать интервалы изменений переменных, которые существенны[9] для классификации).

Вместе с тем, алгоритмы на основе деревьев имеют и ряд специфических недостатков. Среди них [24] неустойчивость к изменению набора обучающих данных. Удаление или добавление нескольких примеров может существенно повлиять на структуру результирующего дерева, ошибки накапливаются при продвижении к листьям. Классические модели деревьев не позволяют построить гладкие приближения в задачах регрессии – вероятное значение прогнозируемой переменной меняется от ветви к ветви скачками. С точки зрения представимости функций моделями деревьев, затруднено выявление возможной аддитивной структуры функциональной зависимости. Если существенное поведение прогнозируемой величины определяется взвешенной суммой нескольких входных признаков, то этот факт не будет явно обнаружен. Это вызвано тем, что дерево на каждом уровне использует правила лишь с одной переменной.

На практике задачи с аддитивной (или почти аддитивной) структурой могут возникать, когда компоненты входных векторов соответствуют высокой степени линейной независимости признаков. В таких задачах классы близки к линейно

[9] В быту, при измерении температуры тела, чаще всего нас интересует интервал в окрестности значения 36.6, а, скажем, точность измерения в диапазоне 37-38 не столь важна. В других задачах классификации такие интервалы и важные точки в значения признаков не всегда заранее указаны и очевидны. Их нахождение может помочь потребителям модели целенаправленно изменить схему сбора данных, например, использовать для измерений более дешевое оборудование, или, наоборот, ввести дополнительные позиции в бланки запрашиваемой отчетности у дочерних отделений компании.

разделимым, и (часто к удивлению заказчиков) качественная классификация может быть получена простейшей нейронной сетью с 2-3 нейронами в скрытом слое. Решение же на основе дерева приводит к сложной иерархии трудно интерпретируемых правил. Поэтому проверку степени линейной разделимости классов также целесообразно включать на одной из ранних стадий разработки.

Таким образом, методы деревьев составляют важный элемент успешной аналитической технологии (особенно ее первых этапов), но в общем случае их разумно использовать в сочетании с другими подходами.

Методы кластеризации данных

Семейство алгоритмов кластеризации весьма обширно. Разнообразие обусловлено как особенностями постановок задач и типов обрабатываемых данных, так и базовыми статистическими принципами в основе алгоритма.

В наиболее общей форме задача кластеризации (таксономии) состоит в отыскании в массиве обучающих данных групп примеров (таксонов[10]), обладающих схожими групповыми свойствами. Последовательный подход в построении алгоритма кластеризации состоит в явном формулировании определенного целевого функционала качества с последующей его минимизацией.

Простейший функционал качества – суммарное по всем примерам расстояние от каждого примера до центра ближайшего к нему кластера в выбранной метрике. Этот функционал определен для условий, когда число кластеров *K* известно заранее. Переменными поиска являются координаты центров кластеров.

Минимизация функционала качества должна производиться по всем возможным распределениям *N* примеров по *K* кластерам. Это и определяет фундаментальную сложность задачи кластеризации. Даже если на возможные положения центров кластеров наложено дополнительное условие точного совпадения центра с одним из примеров, число возможных решений равно числу сочетаний C_N^K.

Комбинаторные задачи такого типа не могут быть решены точно за полиномиальное время, поэтому все известные алгоритмы кластеризации являются *принципиально* приближенными, и от достижения точного минимума придется отказаться сразу (исключение могут составлять некоторые вырожденные задачи классификации на небольшое число классов).

Для некоторых алгоритмов удается доказать их сходимость (к одному из локальных минимумов), однако, учитывая приближенный характер решения, на практике часто используются и итерационные алгоритмы, сходимость которых теоретически не установлена. Алгоритм останавливается принудительно по числу итераций.

К такому классу алгоритмов относится, пожалуй, самый популярный алгоритм *K-средних* (k-means). Алгоритм стартует с K наудачу выбранных примеров из обучающего набора, на каждой итерации все примеры перераспределяются по степени близости к K выбранным примерам. После перераспределения центр каждого кластера помещается в “центр тяжести” его примеров, либо в точку примера, ближайшего к положению центра тяжести. Процесс повторяется либо до достижения наперед указанного числа итераций (*эпох* обучения), либо до состояния, когда распределение примеров по кластерам стабилизируется.

---

[10] *Таксон* - группа дискретных объектов, связанных той или иной степенью общности свойств и признаков и благодаря этому дающих основание для присвоения им определенной таксономической категории.

Для использования на практике предложен широкий набор вариантов улучшения этого базового кластерного алгоритма:

- Методы “отжига”, стабилизирующие итерации и позволяющие достигнуть путем стохастического поиска более глубоких минимумов функционала качества (например, метод Тишби [2])

- Методы с вероятностной принадлежностью примеров к кластерам. Оптимизация производится на основе EM-алгоритма [7]. В процессе итераций плотности распределения примеров по кластерам обужаются, что соответствует росту целевой функции правдоподобия или, в Байесовой постановке, максимизации апостериорной вероятности объяснения обучающих примеров кластерами.

- Методы с принадлежностью примеров кластерам, определяемые нечеткими множествами, нечеткой логикой, а также интервальными и “мягкими” метриками.

Эти и другие модификации метода K-средних положены в основу промышленно используемых программных разработок, но полезно помнить, что основные недостатки этого алгоритма – комбинаторная сложность, зависимость качества результатов от удачной начальной инициализации кластерной структуры и отсутствие глобальной сходимости – не устранены.

Проблема построения кластерной структуры при известном числе кластеров часто является подчиненной проблеме выбора наилучшего числа кластеров. Эта задача (которую иногда называют задачей *категоризации* данных) также может предшествовать построению классификаторов, поскольку она отвечает на вопрос о числе классов, на которые *объективно* разбиваются данные.

Поиск числа кластеров или классов также опирается на оптимизацию функционала качества. Поиск состоит в нахождении такой степени подробности кластерной структуры, при которой и уменьшение, и увеличение числа кластеров приводит к ухудшению качества кластеризации. С вероятностной точки зрения наиболее последовательным выглядит совмещение процессов построения кластеров с вариацией их числа. Для робастного поиска используются функционалы качества с регуляризацией (например, штрафами за излишнюю сложность структуры). С такими функционалами имеют дело популярные методы минимизации длины описания модели [3].

Различают два подхода к построению кластерных структур с изменяющимся числом кластеров – рост числа кластеров с расщеплением уже имеющихся, и наоборот, последовательное объединение (агломерация) отдельных примеров в мелкие группы, которые затем объединяются в более крупные кластеры. Второй подход обычно более устойчив, т.к. критерий объединения проще сформулировать в виде задачи оптимизации, нежели предложить эффективную эвристику для расщепления кластеров. При большом числе примеров вычислительно более эффективным является процесс роста путем расщепления.

Обучающаяся машина должна уметь “фотографировать” интересные кластерные структуры в процессе их роста. Потребителей обычно не устраивает какой-то один вариант кластерной структуры, для анализа нужно понимать внутреннюю структуру крупных кластеров.

Набор инструментов для анализа данных и построения обучающихся машин обязательно должен включать хотя бы один алгоритм кластеризации с возможностью изменения типа используемой метрики и поиском оптимальной кластерной структуры. Здесь важно учитывать, что в реальности данные могут образовывать несколько компактных структур на различных масштабах подробности. Это сопровождается наличием нескольких минимумов у оптимизируемого функционала. Обратное также верно – данные могут и не образовывать четких кластерных структур вообще. В последнем случае можно остановиться на удобном для понимания результатов числе кластеров, обеспечивающих как достаточную подробность представления информации, так и отражающих общий характер пространственного расположения многомерных данных.

Эффективную кластеризацию обеспечивают гибридные методы с кластерной структурой в форме дерева решений [4]. В отличие от классификаторов, элементарное решающее правило в каждом узле дерева состоит в отнесении примера к одному из двух кластеров – потомков данного узла. Кластеризация далее продолжается до достижения одного из листьев, примеры которого, по построению, формируют компактный кластер. Итоговая кластерная структура имеет характер вложенных многомерных ячеек Вороного.

Гибридные методы классификации и кластеризации

При теоретическом рассмотрении принято различать задачи обучения машин с *учителем*, в частности классификацию в условиях, когда метки классов обучающих примеров известны, и обучение *без учителя* – кластеризацию примеров безотносительно их меток или с отсутствующими метками классов. В первой задаче искомой является условная вероятность класса при заданном примере, во второй задаче цель состоит в построении модели совместной (полной) плотности распределения примеров.

В приложениях эти задачи в чистом виде встречаются относительно редко, особенно когда компания-заказчик только приступает к систематическому анализу накопленных данных, либо, наоборот, когда уже проведено внедрение типовых решений на основе рыночных продуктов, и после использования базовых процедур классификации и/или кластеризации возникает потребность в дополнительном углубленном анализе данных и результатов.

Сформулированные при постановке задачи классы не обязательно являются в математическом смысле строгими для потребителя, а кластерный анализ может приводить к выявлению обособленных групп данных, для практического обслуживания которых целесообразно ввести новые дополнительные классы.

Даже если проблема ставится, как задача кластеризации, обучающие примеры (или их часть) могут иметь метки отношения к некоторой классификации. Естественным, ожидаемым для потребителя результатом работы алгоритма служит преимущественное попадание примеров одного класса в один или близкие кластеры. Полное игнорирование меток является отбрасыванием части существенной информации, которая в действительности имеется у заказчика.

Другой пример – собранные данные имеют метки классов, и рассматривается задача классификации. Однако, фактическая достоверность меток крайне не высока. Например, метки получены при “ручной” предварительной классификации с участием нескольких экспертов, применявших отличающиеся друг от друга нечетко сформулированные критерии определения классов. Классы могли быть назначены машиной, автоматически считывающей документы и использующей алгоритм распознавания образов, с ощутимой вероятностью ошибки. Встречается

множество и других ситуаций, когда указанным классам примеров нельзя доверять безусловно.

В таких случаях целесообразно использовать гибридные подходы, сочетающие классификацию и кластеризацию. При этом "ведущим" выступает алгоритм кластеризации, так как метрические отношения близости входных векторов первичны по отношению к малодостоверным меткам классов. Можно выделить два варианта искомой гибридной технологии:

- *Априорное* использование меток классов и формирование кластерной структуры с преимущественным отнесением каждого примера к кластеру с примерами "дружественного" класса. К семейству таких методов относятся алгоритмы обучающегося векторного квантования (LVQ – learning vector quantization [5, с.175]).

- Использование меток классов *апостериори*, путем "раскраски" готовой кластерной структуры. Для каждого кластера собирается статистика (гистограмма) частот встречаемости каждого класса. С учетом априорной вероятности классов, эта статистика служит оценкой плотности распределения условной апостериорной вероятности классов, определенной на всем пространстве данных.

В последнем случае рекомендуется использовать кластерную иерархию с отбором кластерных структур, максимизирующих апостериорную вероятность наблюдения классов. Полученный гибридный классификатор обладает ценным свойством – автоматически при классификации оценивается вероятность ее ошибки.

> **Оптимизация рисков при финансовом прогнозировании**. В задачах управления портфелем финансовых активов разумная цель прогнозирования (предсказания будущего направления изменений цен, доходности на заданный период, оценивания будущей волатильности и др.) состоит не в создании оракула, знающего наперед, как поведет себя рынок, а в информационном обеспечении алгоритмов принятия инвестиционных решений. Гибридный классификатор, результатом работы которого является *распределение* вероятности будущих событий, позволяет вычислить суммарную вероятность (риск) неблагоприятных событий. Обоснованные управляющие решения далее принимаются с учетом анализа уровня рисков, а не благодаря "счастливому" точному прогнозу, выдаваемому таинственной нейронной сетью.

Рассмотренные здесь методы классификации деревьями правил и кластеризации с метрическими отношениями соседства принципиально основаны на моделировании плотности вероятности (условной или безусловной) в пространстве данных. Представление плотности является одним из фундаментальных принципов синтеза обучающихся машин [8].

В последнее время появилось много сообщений об успешном применении машин, основанных на иных принципах, в частности на моделировании *геометрической структуры* пространства обучающих примеров в окрестности границ классов. Основной представитель этого семейства – машины базовых векторов (дословно[11],

[11] Хотя истоки метода SVM находятся в России, устойчивого эквивалента понятию support vector в отечественной литературе пока нет.

"поддерживающих" векторов – SVM, Support Vector Machines). Методы SVM ранее не затрагивались в лекциях автора, поэтому рассмотрим их более подробно.

Машины базовых векторов (SVM)

Рассмотрение метода SVM мы начнем с задачи классификации множества примеров $\vec{x}_k \in R^m, y_k \in \{-1,1\}, k = 1..n$ на два линейно разделимых[12] класса, задаваемых выходной переменной y. Решение этой задачи с использованием нейронной сети, состоящей из единственного нейрона, может быть получено градиентным алгоритмом минимизации суммарной квадратичной ошибки.

(1) $$\hat{y}_k = sign(\vec{w} \cdot \vec{x}_k + b)$$

(2) $$\Delta \vec{w}_k \propto \delta \cdot \vec{x}_k$$

Соотношение (1) определяет базовую модель – пороговое решающее правило с весовыми параметрами (w,b), поправки к которым при обучении (2) пропорциональны обучающим векторам и невязкам модели на каждом примере.

Суммирование поправок при стохастической оптимизации дает решение в виде взвешенной суммы *всех* обучающих векторов:

(3) $$w = \sum_k \alpha_k \vec{x}_k$$

Характер поведения решения изображен на Рис. 2а. Вектора примеров в затененной области дают больший вклад в решение, но масса остальных векторов также влияет на значения параметров.

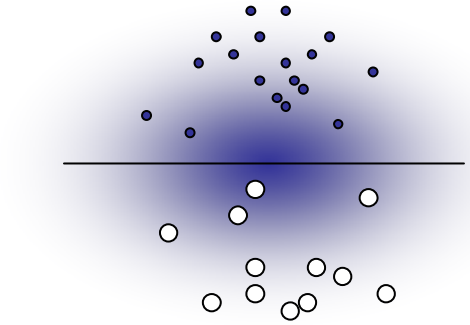

Рис 2а. Разделение классов персептроном

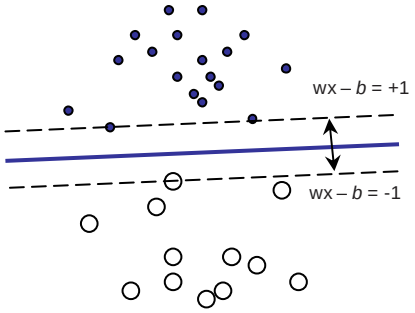

Рис 2.б Классификатор максимального зазора (maximum margin classifier)

Подход, предложенный еще в 60-70-х годах[13] Владимиром Вапником [13], выделяет для описания решающего правила только вектора, находящиеся вблизи разделяющей границы. При этом решающее правило (Рис 2.б) среди всех линейных

[12] Первоначальное представление об основных терминах, понятиях и алгоритмах обучения нейронных сетей можно получить из электронного учебника автора [15].

[13] Как это, к сожалению, часто бывает, признание и широкое распространение метод нашел гораздо позже, только после зарубежных публикаций...

классификаторов выделят тот, который максимизирует “зазор” между крайними точками множеств. Вектора, лежащие точно на границах разделительной полосы, названы здесь базовыми (support vectors), а решение имеет вид линейной комбинации только базовых векторов:

$$w = \sum_{k \in \{SV\}} \alpha_k \vec{x}_k \tag{4}$$

Окончательно классификатор произвольного примера *x* имеет вид взвешенной суммы скалярных произведений с базовыми векторами:

$$\hat{y}(\vec{x}) = sign\left(\sum_{k \in \{SV\}} \alpha_k \cdot (\vec{x} \cdot \vec{x}_k) + b \right) \tag{5}$$

Важно понимать принципиальную разницу между формулами (3) и (4), первая из которых имеет статистическую природу, а вторая – геометрическую. Геометрическое решение тесно связано с классификацией *малых* множеств данных, когда статистические гипотезы о распределения могут оказаться несостоятельными.

В математическом и вычислительном плане, *трудная* задача оптимизации многомерной нелинейной функции при обучении нейросетевой модели заменяется *трудной* (но другой) задачей *выбора* из множества векторов тех, которые определяют геометрию границы. Как будет показано ниже, этот подход также соответствует определенной задаче оптимизации.

Как и алгоритм обучения нейрона, метод базовых векторов может формулироваться и для случая, когда точного линейного разделения классов нет. Из аналитико-геометрических выкладок, квадрат величины зазора между множествами обратно пропорционален квадрату нормы решения $\left\|\vec{w}\right\|^2$. Если допустить наличие ошибок, связанных с попаданием векторов в полуплоскость примеров другого класса, то оптимальный геометрический классификатор дается решением следующей задачи:

$$\min_{w,b,\xi} \left\|\vec{w}\right\|^2 + C\sum_{i=1}^{n} \xi_i \tag{6}$$

$$\xi_i \geq 0,\ y_i \hat{y}(\vec{x}_i) \geq 1 - \xi_i,\ \ i = 1..n \tag{7}$$

Коэффициент *C*, являющийся параметром алгоритма[14], характеризует относительную роль штрафа за нарушения условий точной линейной разделимости.

Большинство (?) практических задач классификации приводит к нелинейным границам классов, более того, области классов могут оказываться не выпуклыми и не односвязными. Следующее фундаментальное обобщение машины базовых векторов состоит в переходе к нелинейным решающим правилам.

Базовая идея такова: вместо рассмотрения задачи классификации в исходном векторном пространстве нужно перейти к пространству новых переменных *большей* размерности, в котором уже возможно построение линейного классификатора. Новые переменные связаны с исходными координатами нелинейными функциональными соотношениями

(8) $$\varphi_j(\vec{x}), \quad j = 1...$$

В новых координатах классификатор будет иметь вид:

(9) $$\hat{y}(\vec{x}) = sign\left(\sum_{j \in \{SV\}} \alpha_j \cdot \left\langle \varphi(\vec{x}) \cdot \varphi(\vec{x}_j) \right\rangle + b \right) = sign\left(\sum_{j \in \{SV\}} \alpha_j \cdot K(\vec{x}, \vec{x}_j) + b \right)$$

Если внутреннее произведение в новых координатах вычислимо с помощью некоторой *ядерной* функции *K* , то система функций (8) формально может быть и бесконечной! Соответствующий математический аппарат Гильбертовых пространств со скалярным произведением, порождаемым ядром ко времени выхода работ В. Вапника уже существовал[15]. В данном контексте существенно, что для получения линейного классификатора не требуется вычисление (и даже существование явного вычислимого вида) базисных функций и формальных сумм для их скалярных произведений, содержащих потенциально бесконечное число слагаемых. Все построения могут быть выполнены использованием одной лишь ядерной функции *K*.

В новых переменных прямая задача оптимизации (6)-(7) в явном виде содержит коэффициенты *w* разложения по базисным функциям. Для вычислений пригодна дуальная формулировка, которая выводится использованием стандартного лагранжевого формализма [16]. Функция Лагранжа строится с применением свободных множителей (неотрицательных двойственных переменных) для всех ограничений

(10) $$L(w, \xi; \eta, \alpha) = \tfrac{1}{2}\|w\|^2 + C\sum \xi_i - \sum_i \eta_i \xi_i - \sum_i \alpha_i \left[y_i \hat{y}(x_i) - 1 + \xi_i \right]$$

[14] У читателя может остаться законный вопрос, почему (в каком смысле?) максимизирующее зазор между классами решение является наилучшим с точки зрения классификации новых векторов. В теории В. Вапника построены оценки верхней границы ошибки обобщения для классификационных алгоритмов, сходящиеся по вероятности. Наилучшим назван алгоритм, который минимизирует наихудшую верхнюю оценку ошибки (функционал структурного риска в терминологии Вапника). Оценки опираются на введенное в теории понятие VC-размерности (размерности Вапника-Червоненкиса) классификатора, описывающее его геометрическую сложность. Для ядерных классификаторов семейства SVM оптимизация структурного риска формально приводит к задаче (6)-(7).

[15] Теория пространств, порождаемых ядрами (reproducing kernel Hilbert space) была независимо разработана N. Aronszajn и S. Bergman в 1950 г.

Соотношения для седловой точки в двойственных переменных приводят к искомой постановке задачи:

$$\max_{\alpha} \sum_i \alpha_i y_i - \tfrac{1}{2} \sum_{ij} \alpha_i \alpha_j K(\vec{x}_i, \vec{x}_j) \quad (11)$$

$$\sum \alpha_i = 0, \ \min(0, Cy_i) \le \alpha_i \le \max(0, Cy_i) \quad (12)$$

Данная задача является задачей квадратичной оптимизации с простыми ограничениями. Основная трудность в решении вызвана тем, что число неизвестных переменных α равно числу обучающих примеров. В частности, матрица *K* для задач с числом примеров масштаба 10,000 уже не может быть целиком размещена в памяти компьютера, поэтому прямое применение процедур оптимизации затруднено.

Как уже отмечалось, в ожидаемом решении большинство примеров не участвует в описании классифицирующей границы, и коэффициенты α таких примеров в точке оптимума равны нулю. Таким образом, решением задачи (11)-(12) является разреженый вектор с относительно небольшим числом ненулевых компонент (на практике, в зависимости от сложности задачи и уровня шума в данных, можно ожидать до 99% нулевых компонент).

Это обстоятельство позволяет разработать специализированные методы оптимизации, в которых разреженность решения встроена в алгоритм поиска. К таким методам относятся методы покоординатного расщепления, в частности алгоритм последовательной минимальной оптимизации (SMO – sequential minimal optimization) [17]. Этот алгоритм является важным практическим частным случаем общего метода Немировского-Нестерова, разработанного еще в 80-х годах.

Суть метода состоит в выделении пар переменных α для пары базовых векторов, наиболее близко примыкающих к границе классов и лежащих по разные стороны от границы. Решение задачи квадратичного программирования для пары переменных с ограничениями строится аналитически в явной форме. После шага оптимизации одна или обе переменные могут обратиться в ноль, и соответствующие вектора исключаются из списка кандидатов. Процесс завершается, когда не удается найти пару векторов, оптимизация параметров которых может улучшить решение. Подробно варианты алгоритма изложены в специальной литературе по оптимизации [18,19].

Регулярность решения в методе SVM определяется параметром *C* в условиях (12) а также выбором ядерной функции. Стандартным выбором является использование радиальных базисных функций

$$K(\vec{x}_i, \vec{x}) = \exp\left(-\gamma \left\| \vec{x} - \vec{x}_i \right\|^2\right) \quad (13)$$

В итоге, метод параметризуется двумя величинами *C* и γ, оптимальные значения которых для каждой практической задачи могут быть выбраны на основе статистических кросс-валидационных экспериментов или бутстреп-выборок. Для оценок γ полезно провести оценки гистограммы распределения квадратов расстояний между векторами в подвыборке обучающих данных.

Хотя основной областью применения SVM являются задачи классификации многомерных векторов, имеются и обобщения метода на задачи регрессии и аппроксимации плотности полной вероятности. Однако в задачах регрессии ощутимого преимущества от использования базовых векторов нет, т.к. требуется эффективное покрытие всего множества данных. В этом случае имеется серьезная альтернатива – нейронные сети с локальными нормализованными сплайнами, о которых шла речь в предыдущей лекции автора [25, 11-12]. Нейросетевые сплайны свободны от ограничений ядерных методов, связанных с фиксацией положения базисных функций строго в точках обучающих данных.

Сторонники геометрических машин SVM часто отмечают “неоспоримые преимущества” ядерных методов над нейросетевыми. Значительная доля таких заявлений преследует, конечно, маркетинговые цели – ведь потребители машинных технологий, как и все другие покупатели, нуждаются в обновлении парка алгоритмов. В реальности SVM, как и другие алгоритмы, имеет свои недостатки и свои границы применимости:

- Обучение SVM-машины сводится к решению задачи квадратичного программирования, которая, при прочих равных условиях, проще общей задачи нелинейной оптимизации, возникающей при обучении нейросетевых машин. Однако число свободных весовых параметров в практически используемых нейронных сетях редко превышает 2-3 тысячи, число же переменных в задаче SVM равно числу примеров, и во многих приложениях (например, в задачах классификации текстовых документов) может достигать миллионов. Поэтому проблема масштабирования SVM стоит весьма остро.

- Для масштабных задач число результирующих базовых векторов в SVM модели может достигать нескольких тысяч. Это затрудняет использование обучающихся по алгоритму SVM машин в портативных и встраиваемых устройствах, обладающих ограниченной памятью и быстродействием. Для нахождения результата классификации для каждого нового примера необходим расчет расстояний от него до каждого базового вектора и вычисление такого же числа нелинейных ядерных функций. Быстродействие таких моделей на современных PC не превышает $10^{-2} – 10^{-3}$ секунды на один пример. Это может оказаться слишком медленным, например, для систем управления и реального времени.

- Принцип минимизации структурного риска, положенный в основу SVM классификаторов, дает оценку ошибки обобщения в вероятностном (не абсолютном) смысле. Поэтому получаемая граница между классами оптимальна лишь в смысле статистики примеров *в ее окрестности*. Например, перемещение вдоль границы значительных объемов примеров, находящихся на удалении от нее, никак не отражается на классификаторе. Напротив, решение, даваемое оптимальными Байесовыми классификаторами и нейросетевыми алгоритмами, будет адекватно отражать изменения в пространственном расположении примеров.

Методы SVM целесообразно использовать в сочетании с пониманием особенности задачи, достигнутым при помощи других алгоритмов и совместно с ними. Например, может быть предложена интересная гибридная методика, в которой для

обучения нейронной сети задачи классификации используются не все обучающие данные, а лишь базовые вектора, выделенные алгоритмом SVM. При достижении сравнимой точности классификации нейросетевая модель будет значительно выигрывать в вычислительной эффективности.

Далее мы переходим к обзорному рассмотрению нейронных методов анализа данных для задач классификации, кластеризации и регрессии.

Нейросетевые обучающиеся машины

К *вычислительным* нейросетевым методам относятся алгоритмы, использующие в своей основе векторные операции с массивным параллелизмом и нелинейные базисные функции *одной* переменной. Структурно, вычисления производятся в узлах сети с направленными связями, при этом каждый узловой элемент (*нейрон*) обрабатывает только информацию, поступившую к нему от других, непосредственно связанных с ним, нейронов.

Базовая операция нейрона, как правило, состоит в вычислении скалярного произведения вектора входных сигналов с хранящимся в локальной памяти весовым вектором, и в последующем применении нелинейной переходной функции к этому скалярному произведению:

$$n(\vec{x};\vec{w}) = f\left(\sum x_j w_j + w_0\right) \tag{14}$$

Вместо скалярного произведения формально может рассматриваться и расстояние между входным вектором и вектором памяти, но это принципиально не меняет сути вычислений:

$$\begin{aligned} & n\left(\left\|\vec{w}-\vec{x}\right\|^2\right) = n\left(\left\|w\right\|^2 + \left\|x\right\|^2 - 2\vec{w}\cdot\vec{x}\right) = \\ & = n\left([\left\|x\right\|^2, \vec{x}];\ [w_0 + \left\|w\right\|^2, 1, -2\vec{w}]\right) \equiv n(\vec{y};\vec{v}) \end{aligned} \tag{15}$$

В набор входных сигналов добавлена новая компонента, равная норме входного вектора (общая “интенсивность” сигнала), и переобозначены компоненты весового вектора.

При помощи выбора весовых коэффициентов, наборов входных сигналов и вида переходной функции *f* могут быть вычислены и другие операции, встречающиеся в нейросетевой литературе, такие как нахождение победителя при соревновательном функционировании, нормализация и контрастирование компонент векторов. Более сложные операции, например масштабирование одной группы входов другими входами, реализуются введением дополнительных элементов с базовой функцией (14).

Функция, вычисляемая всей нейронной сетью в целом, дается рекурсивным применением элементарных нейронных функций (14):

$$NN(\vec{x};W) = f\left(w_0 + \sum w_j f_j\left(w_{j0} + \sum \ldots\right)\ldots\right) \tag{16}$$

Разнообразие имеющихся нейросетевых архитектур не противоречит основной модели (16). Так, для описания моделей с рекуррентными связями, к (16) добавляются условия попарного равенства некоторых векторов памяти и тождественности векторов входных сигналов. Нейронные сети с конечными временными задержками представляются дополнительными слоями рекурсивного

ряда (16). Специальные архитектуры с симметричными (не направленными) связями также описываются дублированием слоев и условиями на равенство весовых векторов. В ряде моделей выходом (результатом применения) нейросети служит не только итоговое значение формулы (16), но дополнительно и значения некоторых из переменных, полученных на предыдущих стадиях рекурсии, например значения отдельных нейронных функций (14) для выбранной группы нейронов. В этом случае нейронная сеть осуществляет, в общем случае, нелинейное отображение входного вектора $\vec{x}$ в выходной вектор $\vec{z} = NN(\vec{x})$, другой или той же размерности.

Таким образом, далее в лекции без ограничения общности, рассматриваются только рекурсивные вычислительные алгоритмы класса (16). Нужно заметить, что общность вида (16) в специальной литературе часто завуалирована конструктивными описаниями архитектуры, способа функционирования и обучения. Такое, общепринятое, описание наиболее часто используемых моделей в интуитивной форме можно найти, например, в электронном учебнике [15].

Минимально достаточной моделью вида (16) является нейронная сеть с одним скрытым слоем равноправных нейронов, называемая также сетью прямого распространения (feed-forward):

$$(17) \qquad FF(\vec{x};V,W) = g\left(v_0 + \sum v_j f_j\left(w_{j0} + \sum .w_{jk} x_k\right).\right)$$

Для задач регрессии данных выходная функция g может быть выбрана линейной. Алгоритмы функционирования и дифференцирования модели (17) также подробно рассмотрены в электронном учебнике, в приложении к которому имеется интерактивные иллюстрации[16].

Под обучением нейросетевой машины понимается модификация свободных весовых коэффициентов W и, реже, самой структуры связей, с целью оптимизации выбранного критерия качества обучения. Нейронные сети имеют принципиальную общность с методами статистического обучения, при этом в обучении и тестировании моделей используются одни и те же функционалы качества и методы регуляризации. В основе обучения лежит алгоритм многомерной оптимизации[17].

Важным практическим достоинством нейронного алгоритма (16) является возможность эффективного вычисления градиента функции качества по компонентам весов W. Рекурсивная структура позволяет применить правила дифференцирования сложной функции, при этом необходимые промежуточные вычисления проводятся заранее, при прямом применении формулы (16). Такой способ дифференцирования получил название алгоритма обратного распространения ошибки.

В семейство алгоритмов (16) формально попадают и многие алгоритмы, которые традиционно не относят к нейронным сетям (например, ранее рассмотренная машина SVM). В обширной литературе по вычислительным нейронным сетям часто выделяются дополнительные свойства, сужающие класс (16), например, *ассоциативность* памяти весовых векторов. Под ассоциативностью можно

[16] Исходный код программы с использованием javascript доступен в HTML-тексте страницы.
[17] В некоторых случаях явная формулировка оптимизируемого функционала может быть затруднена. Например, в широко используемых картах самоорганизации Кохонена [5] алгоритм обучения формулируется конструктивно.

понимать[18] тот факт, что обучающие примеры, или их подмножества, не хранятся в узлах сети по определенным адресам (как в ядерных алгоритмах), а в сжатой форме “распределены” в значениях всех весовых коэффициентов. Приближение к выходному вектору $\vec{z} = NN(\vec{x})$ может быть получено по неточной копии входного вектора $\vec{x}$.

Что принципиально при синтезе нейросетевых моделей?

Нейросетевые модели представляют собой существенно нелинейные функции с большим числом степеней свободы. Такая гибкость и широта класса представимых функций накладывает особые требования к технологии их использования. Нейросеть является тонким прецизионным инструментом, предполагающим аккуратность в обращении.

- **Данные.** Нейронная сеть представляет собой статистическую модель обучающих данных. Первичным здесь являются именно данные. Значительная доля всех затрат (и временных, и человеческих) при синтезе моделей связана со сбором, согласованием, и обеспечением достоверности данных.

  Математические вопросы возникают при предобработке и кодировании данных. Предобработка преследует, в целом, одну цель – повышение информативности данных и связанную с ней регулярность алгоритма обучения. Важность снижения шума в данных иллюстрируется задачей 1, приведенной в конце лекции.

  Масштабирование данных и преобразование координат позволяют упростить задачу оптимизации, в частности статистически выравниваются компоненты градиента. Перед проведением обучения необходимо изучить гистограммы распределений координат обучающих примеров. Если визуальная инспекция невозможна, то требуется разработка алгоритмов, собирающих статистические параметры гистограмм. При высокой неравномерности распределений, и особенно их многомодовости, выравнивание[19] достигается применением кумулятивных табличных функций распределения.

  Программные системы, предназначенные для промышленных применений нейросетевых моделей обычно содержат развитую библиотеку средств предобработки. Полная автоматизация процесса предобработки[20] представляет собой сложную интеллектуальную и алгоритмическую проблему.

- **Оптимизация.** Обоснованные алгоритмы обучения нейросетевых моделей базируются на решении той или иной задачи

---

[18] Это, скорее, вопрос условностей и истории. Так, например, при аппроксимации значений некоторой функции системой полиномов, коэффициенты этих полиномов также являются сжатой формой представления таблицы значений, при этом сами значения нигде не запоминаются. С другой стороны, в задаче *интерполяции* данных, полином Лагранжа содержит табличные значения в явном виде.

[19] Или “выбеливание” – whitening.

[20] Разумеется, предобработка может, наоборот, приводить и к усложнению задачи и ухудшению результатов ☺

оптимизации. По существу, обучение машины – это оптимизация ее структуры и параметров. Конечно, для практической пригодности машины требование ее оптимальности не является обязательным, скорее, наоборот, оно избыточно. Достаточно достигнуть некоторого *допустимого* состояния в пространстве параметров, в котором функциональность машины может быть *обоснована* (путем проведения формальных тестов). Однако достижение допустимого состояния параметров из некоторого исходного состояния, суть, выполнение процедуры оптимизации с ее остановом по достижении критерия допустимости. Поэтому разработка или применение мощного алгоритма оптимизации является одним из ключей к успешному созданию модели.

Нужно понимать, что многие опубликованные алгоритмы обучения нейросетевых моделей носят иллюстративный характер, являются "концептуальными автомобилями", показывающими, как *в принципе* может возникать обучение[21] данной нейросети. Часто авторы алгоритмов преследуют далекие от технологических потребностей цели, например, биологические аналогии или реализуемость обучения в специальных условиях. Перед машиной может ставиться задача адаптации в нестационарных условиях. Так, автомат, собирающий информацию в недоступных человеку условиях, вообще может иметь своей целью как можно более длительное выживание при произвольной блуждающей активности. Конечно, алгоритмы, разработанные для таких специфических задач, не всегда пригодны для вычислительных применений при анализе данных.

- **Статистическая интерпретация.** Обучение нейросети останавливается в одном из множества локальных минимумов функционала качества. Поэтому *точное* воспроизведение одной и той же модели при повторном обучении из произвольной начальной точки не вероятно. Близость моделей должна пониматься в вероятностном смысле – сходности статистических оценок точности и др. Это может привести в замешательство как пользователей, так и, например, специалистов ИТ-подразделений предприятия, внедряющего нейросетевые модели. С их точки зрения, программный комплекс, содержащий нейросеть, не удовлетворяет одному из важных принципов повторяемости результатов при неизменности условий использования. Так, например, системы баз данных возвращают по одному и тому же запросу одинаковое число записей, если содержимое таблиц не изменилось. Эта проблема в целом не так проста, как может показаться на первый взгляд – ведь работа предприятий может базироваться на установленных технологических требованиях к программному обеспечению и процедурах обеспечения его устойчивого функционирования.

  Это замечание также справедливо и по отношению к другим методикам, использующим рандомизированную инициализацию

[21] Классический пример – алгоритм обратного распространения ошибки с градиентным обновлением весов. В его исходной формулировке, этот алгоритм для обучения нейросетей на практике не пригоден.

или стохастику[22] в алгоритме обучения. К ним относятся, например, машины SVM, обучающиеся итерационными методами расщепления.

Никакая практическая модель не может быть получена обучением *ровно одной* нейросети, хотя бы потому, что неудачная начальная инициализация не позволяет достигнуть допустимого обученного состояния. В приложениях широко используются комитеты из множества нейросетей, обученных на различных подвыборках данных. Сравнительный анализ нескольких моделей различной сложности позволяет содержательно оценить ошибки. Разные нейросети могут использоваться для моделирования отдельных областей пространства данных, сочетаясь с методами кластеризации или деревьев решающих правил.

- **Встроенное оценивание точности.** Уже отмечалось, что для улучшения потребительских свойств, целесообразно проводить одновременное обучение среднему значению выхода и, как минимум, его дисперсии. Нейросеть с несколькими выходами (или набор отдельных нейросетей) аппроксимирует несколько первых моментов условной вероятности выходов при известных входах.

- **Тестирование.** Все затраты по сбору данных и созданию моделей нужно отнести к убыткам, если доказательство правильности работы обученной машины не проведено или невозможно. Хотя имеются уже вошедшие в учебники теоретические построения для оценок ошибок обобщения нейронных сетей, зачастую проверка выполнения условий их применимости к конкретной задаче затруднена. Некоторые оценки могут требовать достижения окрестности точного минимума функционала качества. Другие предъявляют жесткие требования к статистике и объему данных. Маргинальные оценки обеспечивают границы ошибки с доверительной вероятностью. Другими словами, на практике статистического экспериментирования с нейросетевыми моделями не избежать.

  Одной из рекомендованных технологий тестирования является кросс-валидация. Процедура кросс-валидации заключается в обучении множества нейронных сетей тестируемой архитектуры на частичных выборках данных, полученных последовательным удалением части обучающих примеров. Оценивание производится по статистике ошибок на удаленных примерах. Здесь важны два момента. 1) Значение ошибки обобщения является *случайной величиной*, поэтому способ усреднения экспериментальных результатов должен обеспечивать состоятельность и несмещенность ее оценки. Для корректного проведения статистического анализа рекомендуется обратиться к литературе по математической статистике и методам анализа результатов экспериментов (например, [7]). 2) Обеспечивающее достоверную оценку расщепление обучающей выборки на кросс-валидационные

[22] Популярное онлайн-обучение является примером стохастического метода. Результаты зависят от порядка, в котором предъявляются обучающие вектора.

подмножества зависит от объема данных и сложности модели (здесь – числа степеней свободы весовых параметров). Часто рекомендуемое расщепление выборки на 10 (или 2, или ...) частей может использоваться только для первичных оценок. Этот вопрос также относится к области матстатистики [20,21].

Среди важных прикладных вопросов находятся также проблемы стандартизации[23] моделей и технологий их внедрения в производственные системы потребителей. Эта тема затронута в Приложении.

Иллюстрации применения технологий информационного моделирования

Для апробации разрабатываемых технологий на этапах, предшествующих практическому внедрению, целесообразно экспериментально выяснить границы их применимости и сравнить с другими решениями. Один из доступных полигонов – конкурсы решения информационных задач, проводимые в рамках научных конференций или, независимо, компаниями, заинтересованными в сравнительном оценивании их собственных технологий.

Конечно, конкурсные задачи это только набор частных случаев, кроме того, конкурсы могут дополнительно акцентировать некоторые специальные свойства алгоритмов (например, конкретные формальные показатели качества обучения). Технологии широкого применения редко занимают в конкурсах лидирующие позиции, побеждают обычно специализированные разработки из университетов. Так и должно быть – поток инноваций из научных коллективов должен быть впереди тщательно отбираемых практических методов.

На Рис. 3 приведены результаты одного из таких конкурсов (EUNITE Symposium 2003). Цель состояла в прогнозе по 29 входным переменным динамики 5 выходных переменных, описывающих качественные показатели продукции в производстве стекла. Точность оценивалась как по каждой из переменных в отдельности, так и совокупности[24].

[23] Полезно вспомнить слова Генри Форда "… если вы понимаете стандарт, как ограничение, прогресс остановится" (Дж. Лайкер. Дао Toyota, М. Альпина, 2005, стр. 190)
[24] http://www.eunite.org/eunite/events/eunite2003/competition2003-winners.htm

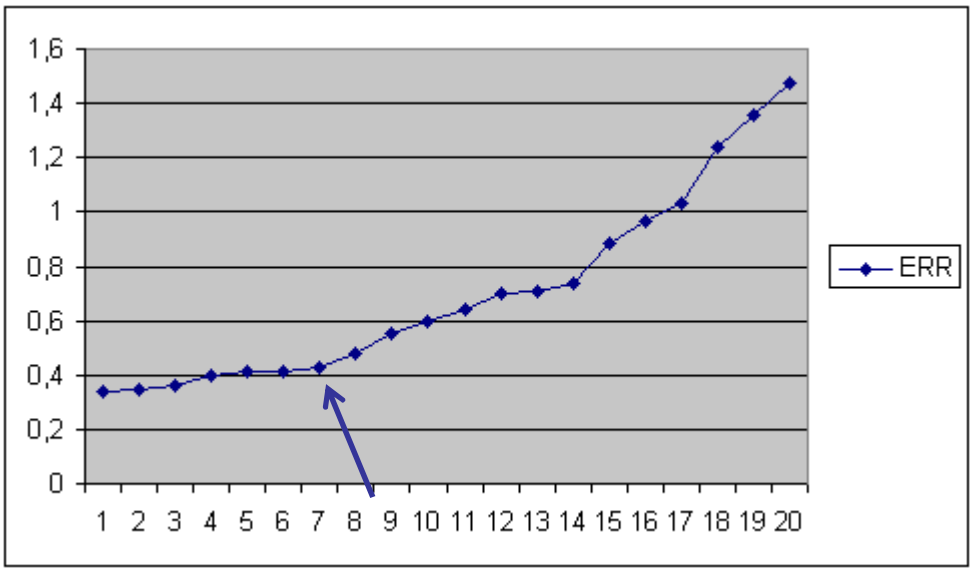

Рис 3. Итоговый показатель точности прогноза в конкурсе EUNITE-2003. Результат коллектива автора отмечен стрелкой.

Технология АФИНА (см. заключительную часть лекции) показала седьмой результат (на 15% уступив лучшему результату, и почти вдвое опережая средний показатель). Это – типичный ожидаемый уровень для промышленных технологий, не адаптировавшихся специально к условиям конкурса.

Ниже мы рассмотрим задачи одного из новых конкурсов, в котором читатель лекции может еще успеть принять участие.

Задачи конкурса WCCI-2006

Конкурс объявлен в рамках[25] Всемирного конгресса по компьютерному интеллекту (IEEE WCCI-2006). На конкурс предложены 5 задач классификации на два класса. Цель – построить машину, обладающую не только наименьшей ошибкой обобщения, но и получить *наилучшую оценку* этой ошибки. Ошибка обобщения будет вычисляться по итогам конкурса на тестовых данных, с учетом баланса относительных частот встречаемости классов:

$$BER = \frac{1}{2}\left(\frac{b}{a+b} + \frac{c}{c+d}\right)$$

где a и b – число правильных и ошибочных классификаций примеров класса -1, d и c – правильные и ошибочные срабатывания классификатора для примеров, истинный класс которых +1.

Пять наборов данных (с собственными именами) отличаются числом примеров, размерностью пространства входов, одна из задач задача классификации ставится для разреженной матрицы данным в пространстве размерности более 10,000. Для каждой задачи имеется набор обучающих данных с известными метками классов,

[25] Сайт конгресса: http://www.wcci2006.org/home.html. Адрес страницы с условиями конкурса и наборами данных http://www.modelselect.inf.ethz.ch/index.php.

набор валидационных данных, метки которых будут опубликованы на поздних этапах конкурса, и набор тестовых данных, метки которых не будут известны участникам.

В лекции эти задачи будут использоваться для иллюстрации некоторых элементов технологии разработки классификаторов на примере одной из задач. Следуя общей идее лекции, мы не ограничиваемся применением только нейронных сетей.

Задача ADA

Таблица обучающих данных содержит 4147 примеров размерности 48. Валидационных данных в 10 раз меньше, тестовых – в 10 раз больше.

Шаг 1. Визулизация данных. Для размерности масштаба 10-100 анализ можно начать с визуальной инспекции гистограмм распределений значений входов. Цель – общее понимание характера данных, наличие выбросов, резкой неравномерности, постоянных входов и других аномалий в данных. Поскольку, по условиям, входная часть информации из тестового набора может быть свободно использована, то гистограммы строятся по все имеющимся данным (включая тесты). На этом этапе выявлено, что около половины входов являются булевыми (0 или 1), при этом имеются переменные с высокой асимметрией распределения.

При большей размерности данных визуальный анализ гистограмм затрудняется[26]. Для автоматизации анализа аномалий и выбросов необходима разработка специализированного программного обеспечения. Некоторые коммерчески распространяемые пакеты статистического анализа включают такой анализ.

Шаг 2. Предобработка. Характер гистограмм данных в задаче ADA показывает, что целесообразно применить выравнивание распределений. Для этого строятся кумулянты гистограмм, и полученные табличные зависимости используется для перевода данных в отрезок -1..1. Наиболее густонаселенные области данных отображаются в широкие поддиапазоны, а участки, на которых данные отсутствуют, сжимаются. Предобработка выходов не требуется – они явно представлены в виде кодов классов.

Шаг 3. Линейный анализ. Анализ линейных корреляций, линейной разделимости данных, выделение главных компонент и другие линейные методы всегда предшествуют нелинейным подходам. Последние используются для уточнения линейных приближений, при этом линейный метод автоматически дает нижнюю границу оценок ошибки и вычислительной сложности. При анализе линейных корреляций большим значениям соответствуют индивидуально значимые входы, но отсутствие корреляций не означает неинформативность входа при его использовании в нелинейной модели.

В данной задаче корреляции входов с выходами не образуют компактной группы входов, объясняющих значительную часть дисперсии. Модули коэффициентов корреляции практически равномерно заполняют отрезок 0.05 .. 0.45. Таким образом, для дальнейшего анализа сохраняются все входные переменные.

Шаг 4. Планирование статистических экспериментов. В цели конкурса (и часто, в цели практических разработок) входит не только получение наилучшей ошибки обобщения, но и ее оценка на этапе обучения. Универсальных гарантированных теоретических оценок для всех условий не существует, поэтому

[26] Хотя в ответственных приложениях его все равно рекомендуется провести. Это – рабочие будни нашей профессии.

экспериментальный анализ фактически неизбежен. Однако, перед началом серий расчетов необходимо как можно более четко сформулировать цели экспериментов, выбрать его план, и, по возможности, представить себе ожидаемые результаты.

В нашей ситуации цели таковы

- Получение результатов и оценок точности более чем одной моделью

- Для каждого типа моделей определение уровня ее структурной сложности, который для имеющихся данных дает наилучшую ошибку обобщения

- Поскольку прогноз ошибки на тестовых данных индуктивно выводится из качества работы на данных с известными классами, то необходимо оценить надежность такой индукции, путем постепенного увеличения числа обучающих примеров.

Итак, план эксперимента включает минимум две модели. Это первая размерность матрицы плана эксперимента. Для задачи классификации таким набором могут быть метод вероятностных деревьев правил, и нейронная сеть прямого распространения. В лекции для основных иллюстраций выбраны нейронные сети прямого распространения с одним скрытым слоем. Алгоритм обучения основан на решении задачи оптимизации методом BFGS. Функционал качества обучения включает штраф за рост весов. Собственно, нейросетевые вычисления, в целом, следуют книге [22].

Во вторых, для каждой модели необходимо варьировать ее сложность. В выбранных моделях простейшим из вариантов является однопараметрическое изменение числа нейронов скрытого слоя нейросети и ограничение роста дерева по числу ветвей.

В более общем случае методы контроля сложности могут включать параметры регуляризирующих функционалов ошибки при обучении (например, ранний останов итераций), а также различные алгоритмы конструктивного упрощения структуры (прореживание связей и ветвей, и др.) При использовании комитетных методов объединения нескольких моделей (Adaboost, bagging и др.), контроль сложности должен вестись с учетом параметров всего комитета. Так или иначе, для экспериментального оценивания сложность формализуется небольшим числом параметров. Если используется какая-то теоретическая модель выбора или контроля сложности, не стоит полностью отказываться от вычислительных экспериментов. Они помогут установить применимость предпосылок модели для конкретного набора данных.

Третья размерность матрицы плана – доля примеров из обучающей выборки, которая будет использоваться в конкретном эксперименте. Для выяснения характера зависимости оценки ошибки расечты будут проводиться на изменяющемся от 10% до 100% объеме данных.

Нас интересует ошибка обобщения, которую можно получить при помощи кросс-валидации. Для этого отобранные для обучения данные разбиваются на несколько групп, и в каждом расчете одна из групп не участвует в обучении, а используется для оценивания ошибки. Методика кросс-валидации чувствительна к пропорции обучающих и валидационных данных, этот вопрос уже обсуждался в лекции. Для

уменьшения фактора смещенности оценки при больших объемах данных можно рекомендовать рекомендовать результат [20], согласно которому каждая группа должна содержать $\sim 1/\sqrt{2m}$ примеров, где m – число свободных параметров модели (число весов для нейронной сети или удвоенное число решающих правил для дерева).

Таким образом, определены наборы из трех параметров, составляющих план эксперимента. В нашем случае можно выполнить *полный* план с перебором всех троек. На практике число экспериментов можно сократить, используя частичный план (например, латинский гиперкуб). При частичных планах интерпретация результатов должна проводиться путем построения регрессионной модели, согласованной с типом плана.

Перед проведением расчетов определимся с объемом собираемой информации. В соотвествием с целями, будем фиксировать тип модели, параметр ее сложности (число нейронов или правил), объем используемых данных, ошибку обучения (для каждого класса), ошибку валидации, ошибку тестирования на неиспользуемых данных из обучающей выборки. Дополнительно могут также собираться такие данные, как время вычислений, тип и значение критерия останова обучения, и другие данные, хакартеризующие расчет.

И, наконец, обсудим, что в целом ожидается в результате эксперимента. Ясно, что при малом объеме данных ошибка обучения будет быстро уменьшаться с ростом сложности модели. Оцениваемая ошибка обобщения в идеале должна (сверху) стремиться ошибке валидации и обучения с ростом числа данных. При большом объеме данных повышается вероятность отсутсвия сходимости итерационного обучения нейросетевой модели, поэтому из статистики следует исключить случаи раннего останова обучения в локальных минимумах. Это будет проявляется в выбросах во времени обучения. Для оцениваемой ошибки тестирования хотелось бы получить тенденцию к насыщению при увеличении объема выборки. Теоретически, кривая ошибки тестирования и валидации может иметь минимум при росте сложности модели, но при высокой скорости сходимости обучения этот минимум может быть слабо выраженным.

Шаг 5. Проведение экспериментов с базовыми нелинейными моделями. Экспериментальные вычисления удобно оформить пакетными заданиями, если это позволяет программное обеспечение. При проектировании программ нужно учитывать такую практическую потребность, иначе поисковые расчеты приведут к сильному удорожанию технологии.

Основные результаты экспериментов приведены ниже на рисунках 4-6.

Методика вероятностных деревьев вычислительно более эффективна, поэтому ее всегда целесообразно использовать для экспресс-оценок и с нее начинать выбор моделей. Для удобства практического использования кривых ошибкок удобно рисовать (и экстраполировать) данные в зависимости от обратного числа примеров.

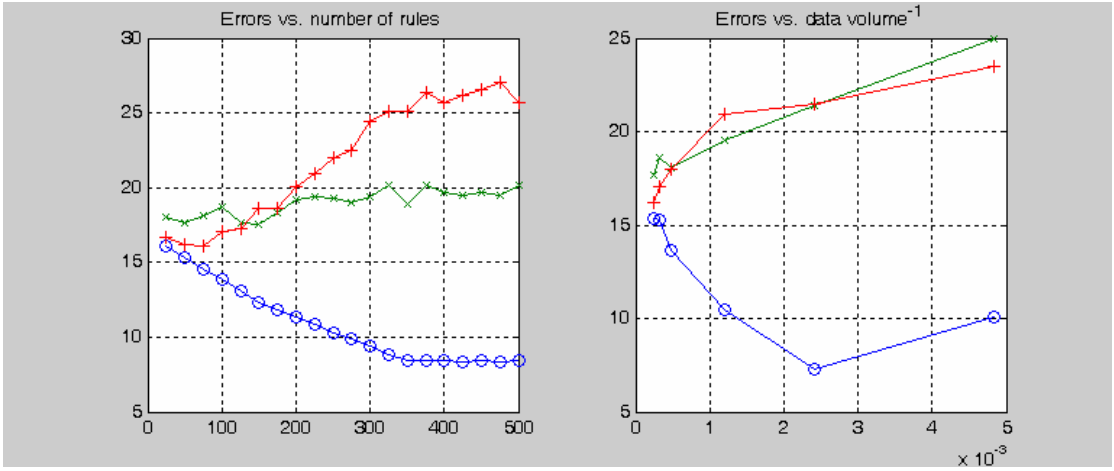

Рис. 4. Зависимость ошибок обучения (ooo), валидации (xxx) и тестирования (+++) от числа решающих правил вероятностного дерева (левый рисунок). Сходимость ошибок для модели из 50 правил.

Валидационная ошибка дерева демонстрирует практически линейный тренд и сходится к значению 17.5%. Эту оценку, видимо, следует считать наилучшим вариантом для данной модели. Эксперименты с нейронными сетями должны показать, можно ли ее улучшить.

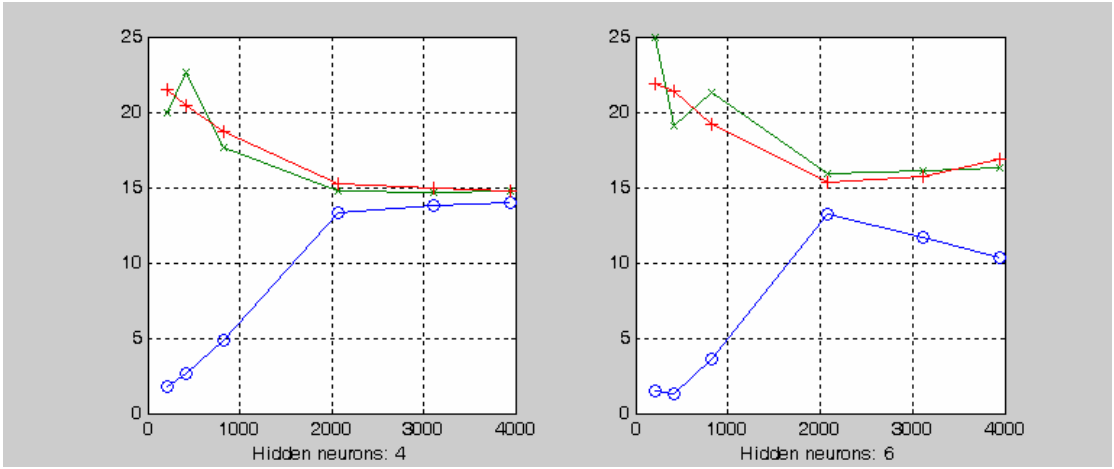

Рис 5. Зависимость ошибок обучения (ooo), валидации (xxx) и тестирования (+++) от объема данных для нейронной сети с переобучением (6 нейронов, правый рисунок), и оптимальной модели (4 нейрона, график слева).

По итогам расчетов, оптимальная нейросетевая модель для этой задачи имеет 4 нейрона в скрытом слое. На рис. 6 наблюдается тенденция к сходимости ошибок, но практическая сходимость еще не достигнута. Экстраполированная оценка при возрастании числа примеров 14.7%, что несколько уточняет экспресс-результат, полученный вероятностным деревом.

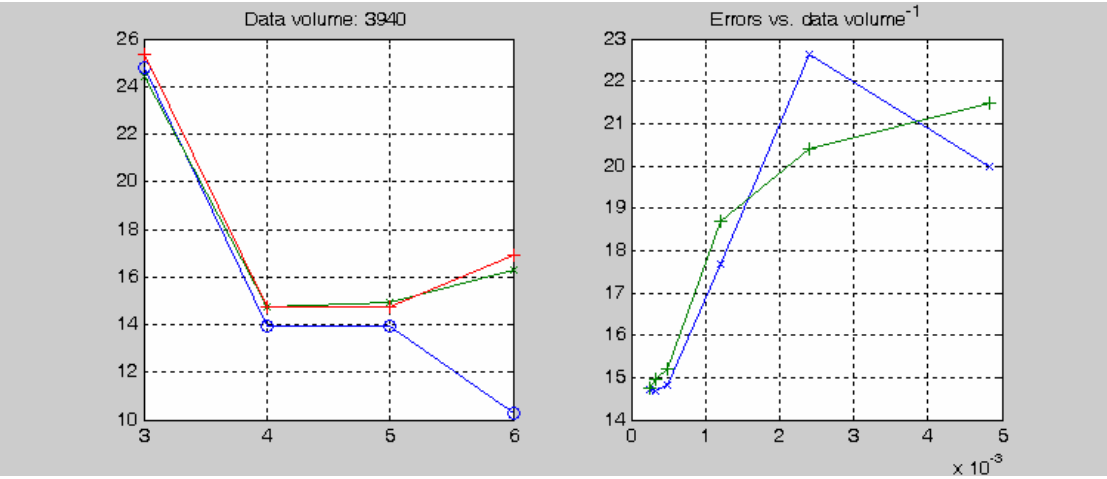

Рис 6. Зависимость ошибок от числа нейронов в скрытом слое (слева) и сходимость ошибок тестирования и валидации в зависимости от обратного числа обучающих примеров.

Шаг 6. Применения моделей. После выбора оптимального уровня сложности моделей проводится обучение итоговых систем на полном наборе обучающих данных. При создании окончательных вариантов нужно учитывать, что сходимость обучения гарантируется только в вероятностном смысле. Поэтому должно быть опробовано несколько вариантов, отличающихся инициализаций свободных параметров. Отбираются модели по соответствию их показателей данным предыдущих экспериментов. Обычной практикой может служить выбор варианта, обладающего ошибкой обучения, равной медиане значений всех пробных вариантов.

Обсуждение

Наблюдаемое в задаче ADA соотношение точности нейросетевых классификаторов и моделей деревьев является типичным для практики, но бывают и исключения (см. Задачу 1 в приложении).

Другие задачи конкурса имеют свои особенности. Две наиболее важные из них – рост размерности пространства входов, и разреженность данных (задача NOVA имеет около 16,000 входных переменных). Такого типа задачи могут встречаться при анализе текстов по статистике встречаемых слов, а также в упоминавшихся в лекции задачах анализа покупательских корзин.

Для устойчивой классификации при таких больших размерностях необходимо сжатие признакового пространства. Как отмечено в [23, с.83], при объемах данных, незначительно превышающих размерность пространства, более эффективной является использование методов кластеризации данных. Кластеризация обычно обеспечивает сжатие в 20-100 раз без существенной потери свойств разделимости классов[27]. В качестве новых, более компактных, координат примеров могут служить взвешенные расстояния (или функции от них) от примера до каждого кластера. Кластеризация сильно разреженых векторов требует специальных методик – при простой метрической кластеризации данные объединяются по

[27] Конечно, значения 20-100 являются некоторыми условными оценками того, что можно ожидать на практике. Известны задачи, когда сжатие можно довести до 1-2 переменных, сохранив точность классификатора.

признаку *отсутсвия* тех или иных компонент, что не всегда соотвествует сути решаемой задачи.

Описанные этапы решения информационной задачи не должны рассматриваться, как прямое руководство к действию. Иногда некоторые этапы можно опустить, случается, что суть задачи можно проявить, сосредоточившись на предобработке данных и методе оценивания модели. Есть ситуации, где многое зависит от параметров и алгоритмов самого метода обучения.

Послесловие

Вы ознакомились с лекцией о практической стороне и технологиях создания вычислительных нейронных сетей и других обучаемых машин. Многие положения и утверждения в тексте носят философский полемический характер. В них отражена точка зрения автора, основанная на 15-летнем опыте работы в отрасли.

Трудности в построении изложения технологий в виде перечня рекомендуемых действий, которые нужно проделать для гарантированного получения качественного решений, носят системный характер. Мы имеем дело с современной практикой в условиях начала массового внедрения обучаемых элементов во все сферы деятельности.

Критически возрастают скорости и объемы обрабатываемой машинами информации. Кто-то из ведущих математиков сказал, что каждый раз, когда размерность анализируемой матрицы возрастает на порядок, следует кардинально пересматривать всю схему вычислений. В области обучаемых машин и алгоритмов мы имеем дело с прогрессирующим ростом размерностей на несколько порядков.

В итоге картина, нарисованная в лекции, скорее мозаична, в стиле художников-примитивистов, поскольку на ней изображен еще не существующий, только нарождающийся образ.

Но этот образ уже начинает радовать глаз – сообщения об успехах в практическом внедрении *умнеющих* машин поступают все чаще, формируется соответствующий рынок услуг, потребители потихоньку привыкают к особенностям и доверяют машинным системам все большую часть аналитической работы.

Система АФИНА. Ряд и научных, и программно-тенических, и организационных аспектов, затронутых в лекции, воплощен в комплексе технологий и программ АФИНА[28] (Анализ и Формализация Интеллектуальных Нейросетевых Алгоритмов), с момента выхода первой версии которого в 2005 г. исполнилось 10 лет. В разное время в создании комплекса под руководством автора принимали участие А. Квичанский, Н. Федорова, Е. Диянкова, С. Диянкова, Р. и Т. Мухамадиевы, Е. Воленко, Ю. Кузнецов, Ю. Конотоп и многие другие специалисты – математики, программисты, электроники и инженеры. Финансовая поддержка начальных этапов работ оказывалась Международным Научно-Техническим Центром (МНТЦ).

Сегодня семейство технологий АФИНА включает несколько комплексов и программных модулей с общими принципами и подходами к формированию заказных решений для потребителей в областях бытовой химии, нефтедобычи и

[28] АФИНА – в греческой мифологии, богиня разума, изящных и боевых искусств. Название технологического комплекса происходит от транслитерации AFINA (Algorithms For Intelligent Neural Applications).

транспорта нефти, финансового прогнозирования, проектирования вычислительных систем и других. В компании Нейрок Техсофт (г.Троицк) происходит постоянное совершенствование и расширение спектра алгоритмов семейства АФИНА, разрабатываются новые подходы к созданию прикладных разработок и оказанию консалтинговых услуг в области моделирования и обработки информации, составляющих основу деятельности компании.

Благодарности

Автор выражает глубокую благодарность разработчикам системы АФИНА - коллективу соратников и коллег и поздравляет их с 10-летним юбилеем технологии.

Литература

1. WWW-страница компании KDNuggets.
URL: http://www.kdnuggets.com/polls/2005/data_mining_techniques.htm

2. Naftali Tishby, Fernando Pereira, and William Bialek. **The Information Bottleneck Method**. Invited paper to the 37th annual Allerton Conference on Communication, Control, and Computing, 1999.
URL: http://www.cs.huji.ac.il/labs/learning/Papers/allerton.ps.gz

3. Mark Hansen and Bin Yu. **Model selection and Minimum Description Length principle**. J. Amer. Statist. Assoc., 1998
URL: http://cm.bell-labs.com/cm/ms/departments/sia/binyu/ps/mdl.ps

4. Сергазы Нарымов. **Алгоритм растущего нейронного дерева**. Нейроинформатика – 2005, Сборник трудов, МИФИ, 2005

5. T. Kohonen. **Self-Organizing Maps**. Springer-Verlag, 1995.

6. A. Hyvarinen, J.Karhunen, E. Oja. **Independent Component Analysis**. Wiley, 2001.

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

8. Терехов С.А. **Нейросетевые аппроксимации плотности в задачах информационного моделирования**. Лекция для школы-семинара "Современные проблемы нейроинформатики", Москва, МИФИ, 25-27 января 2002 года.

9. Терехов С.А. **Введение в Байесовы сети**. Лекции по нейроинформатике. М. МИФИ, 2003 ч.1., с.149.

10. Терехов С.А. **Нейродинамическое программирование автономных агентов**. Лекция для Школы-семинара “Современные проблемы нейроинформатики”. М. МИФИ, 2004. Часть 2., с.111-139.

11. W. C. Mead, S. K. Brown, R. D. Jones, P. S. Bowling, and C. W. Barnes. **Optimization and Control of a Small-Angle Negative Ion Source Using an On-Line Adaptive Controller Based on the Connectionist Normalized Local Spline Neural Network**. Nuclear Instruments and Methods Vol. A352, 309 (1994).

12. R. D. Jones, Y. C. Lee, S. Qian, C. W. Barnes, K. R. Bisset, G. M. Bruce, G. W. Flake, K. Lee, L. A. Lee, W. C. Mead, M. K. O'Rourke, I. Poli, and L. E. Thode **Nonlinear Adaptive Networks: A Little Theory, A Few Applications**., in Proceedings of the First Los Alamos Conference on Cognitive Modeling in System Control, June 10-14, 1990, Santa Fe, NM.

13. Вапник В.Н. **Восстановление зависимостей по эмпирическим данным**. М. Наука, 1979

14. Vladimir N. Vapnik, **The Nature of Statistical Learning Theory**, Springer-Verlag, 1995.

15. Терехов С.А. **Вводные лекции по нейроинформатике**. URL: http://alife.narod.ru/lectures/neural/Neu_index.htm

16. Д. Бертсекас. **Условная оптимизация и методы множителей Лагранжа**. М. Радио и связь, 1987.

17. S.S. Keerthi, S.K. Shevade, C. Bhattacharyya, K.R.K. Murthy. **Improvements to Platt's SMO Algorithm for SVM Classifier Design.** Tech Rep., Univ of Singapore, 1999. URL: http://citeseer.ist.psu.edu/244558.html

18. Antoine Bordes, Seyda Ertekin, Jason Weston, Léon Bottou. **Fast Kernel Classifiers with Online and Active Learning**. JMLR, 6(Sep):1579--1619, 2005.

19. E. Osuna, R. Freund, and F. Girosi. **Support vector machines: Training and applications.** Technical Report AIM-1602, MIT A.I. Lab., 1996. URL: ftp://ftp.ai.mit.edu/pub/cbcl/sv-memo.ps.gz

20. S. Amari, N. Murata, K.-R. Müller, M. Finke, H. Yang. **Statistical Theory of Overtraining Is Cross-Validation Asymptotically Effective?** NIPS, 1996. URL: http://citeseer.ist.psu.edu/amari96statistical.html

21. Michael Kearns. **A Bound on the Error of Cross Validation Using the Approximation and Estimation Rates, with Consequences for the Training-Test Split**. Tech. Rep ATT, 1996 URL: http://www.informatik.uni-bonn.de/III/lehre/seminare/Mustererkennung/WS99/cv.ps.gz

22. Bishop, C. M. **Neural Networks for Pattern Recognition**. Oxford University Press. 1995

23. Ежов А.А., Шумский С.А. **Нейрокомпьютинг и его приложения в экономике и бизнесе.** М. МИФИ, 1998.

24. T. Hastie, R. Tibshirani, J. Friedman. **The Elements of Statistical Learning.** Springer, 2001

25. Терехов С.А. **Адаптивные нейросетевые методы в многошаговых играх с неполной информацией.** Лекции по нейроинформатике. М. МИФИ, 2005, с.92.

26. Миркес Е. **Нейрокомпьютер. Проект стандарта**. Н. Наука, 1999

Приложение. О стандартизации описания моделей обучающихся машин (PMML – Predictive Model Markup Language)

При расширении областей применения новых информационных методов вопросы стандартизации приобретают первостепенное значение. Нужно отметить, что вопросы стандартизации уже поднимались в отечественной литературе [26]. При этом в книге [26] была предпринята попытка стандартизации всех, в т.ч. *исследовательских* процессов (структуры данных, алгоритмов, архитектуры, способов оценивания и т.д.), сопровождающих разработку программных систем, составляющих основу моделей нейрокомпьютеров.

Два аспекта здесь представляются особенно важными.

1. Потребности предприятий и организаций, использующих методы анализа накопленных ими данных при помощи обучающихся машин, выходят за рамки только нейронных сетей. На единой платформе должны унифицироваться самые разнообразные системы (математическая статистика, деревья решений, методы исчисления Байесовых вероятностей и др.) Совместная стандартизация всех математических и технологических аспектов всех этих систем крайне затруднена.

2. В современной технологии создания обучающихся машин присутствуют этапы, на которых принципиально задействован естественный интеллект (т.е. человек). Стандартизация этих этапов также крайне затруднена.

С точки зрения автора, к таким этапам и компонентам относятся

- Процессы первичного отбора признаковых переменных, определяющих задачу. Такой отбор опирается на априорные экспертные знания, не обязательно содержащиеся и выводимые из имеющегося набора данных.

- Ограничения на переменные и результаты. Ограничения могут также возникать из соображений, внешних по отношению к решаемой задаче анализа данных.

- (Не-) использование выводов обучающихся машин в условиях невозможности получения полноценного формального обоснования их применимости (включая ограничения на доступный объем тестирования).

- Процессы принятия решений на основе выводов обучающихся машин *и других* (не учтенных) особенностей задачи.

Наличие трудно формализуемых этапов технологии с одной стороны, и потребностей потребителей с другой, ограничивает направления стандартизации. А именно, *стандартизируется только то, что непосредственно внедряется*, т.е. встраивается в бизнес-процессы потребителей. Объектом стандартизации являются окончательные модели обучающихся машин, а также способы получения ими исходных данных и способы выдачи “на-гора” результатов.

За рамками промышленной стандартизации остаются алгоритмы и методы обучения и другие исследовательские и технологические процессы и "открытия", которые позволили получить[29] итоговую модель, готовую к применению.

По этому пути пошли разработчики языка описания прогностических моделей (PMML - The Predictive Model Markup Language). Этот язык, являясь приложением языка разметки XML[30], формализует понятие "модель для анализа данных и прогнозирования" и позволяет использовать такие модели совместно различными приложениями в разнообразных условиях, определяемых потребителем.

Процесс разработки PMML был начат в 1998 году. О поддержке PMML уже в 2001 году заявили такие компании, как Angoss, IBM, Magnify, Microsoft, MINEIT, NCDM, NCR, Oracle, Salford Systems, SPSS, SAS и Xchange. К моменту написания лекции вышла версия PMML - 3.0, и список компаний насчитывает около 20 ведущих поставщиков информационных и аналитических систем.

Разработка PMML координируется независимой группой DMG (The Data Mining Group), включающей представителей компаний-членов. WWW-адрес DMG: http://www.dmg.org/pmml-v3-0.html

Общие принципы PMML

Объектом стандартизации в PMML являются прогностические модели, которые на основе значений определенных входных переменных (или факторов) позволяют получить статистические оценки значений целевых выходных переменных. Способ стандартизации состоит в фиксировании формальной схемы (XML-schema), которой обязан удовлетворять XML-объект, описывающий модель или набор моделей с общим предметным пространством данных.

Язык выделяет следующие компоненты

- Служебная и идентификационная информация (заголовок)
- Блок описания примененного способа получения модели. Содержимое этого блока не специфицировано и оставлено на усмотрение поставщика. В применении модели этот блок никак не используется.
- Словарь данных, описывающий способ извлечения и интерпретации входных и выходных данных моделей. В частности, могут указываться столбцы во входных потоках или файлах, поля баз данных и др.
- Словарь преобразований. Здесь описываются преобразования (арифметические, логические и функциональные), способы

[29] Сказанное ни в коей мере не означает, что разработчики и поставщики информационных технологий, моделей и консалтинговых услуг не должны стремиться к стандартизации и унификации своих внутренних процессов синтеза моделей. Такая стандартизация, упорядочение и документирование, наряду с человеческими ресурсами, составляет основу бизнес-потенциала компании или иного коллектива разработчиков, обеспечивая конкурентные преимущества и среду для плодотворного творчества. Упоминавшаяся в тексте лекции система АФИНА является примером такой внутренней стандартизации процесса синтеза нейросетевых моделей, применяемого в компании Нейрок Техсофт.

[30] Что само по себе уже является устойчивой почвой для создания стандартов в области обработки данных.

кодирования переменных и другие преобразования, которые переводят наборы сырых данных в числовые данные, которыми непосредственно оперируют модели

- Блоки, описывающие модели, включенные в спецификацию. В версии 3.0 формализованы 12 наиболее часто используемых типов моделей – ассоциативные правила, кластерные модели, модели обобщенной регрессии, нейронные сети, системы формальных правил, машины SVM (support vector machines), модели анализа текстов, деревья правил и некоторые другие. Каждый тип моделей описывается своим соответствующим набором элементов, при этом выделены общие сущности, такие как тип решаемой задачи (классификация, регрессия, кластеризация, анализ последовательностей, выявление ассоциаций), описание выходных переменных, сведения о способе верификации модели.

Остановимся несколько подробнее на спецификации нейросетевых моделей, имеющих непосредственное отношение к тематике лекции. Модель нейронной сети определяет элементарные вычислительные нейроны и связи между ними. Нейроны могут объединяться в слои, функционирующие в том порядке, в каком они описаны. Дополнительные свойства (например, ширина радиальной базисной функции) могут быть приписаны индивидуальному нейрону, слою, либо сети в целом. Для выходов нейронов слоя могут применяться общие процедуры (например, выявление победителя или нормировка).

“Веса” и “нейроны” представлены как различные объекты, формирующие сеть с практически не ограниченной топологией соединений.

Спецификация PMML предполагает возможность расширений и дополнений, вносимых различными поставщиками, что позволяет либо расширить базовую функциональность либо использовать некоторые уникальные дополнительные возможности моделей, разработанных и используемых в программном обеспечении данного поставщика. Сам *способ* проектирования расширений специфицирован, при этом наличие расширений не препятствует стандартному использованию базовых возможностей. Так, в случае нейросетевых моделей, которые предполагается в дальнейшем *дообучать*, в качестве дополнительных параметров могут быть специфицированы текущие значения производных по значениям весов, либо значения последних внесенных поправок к весовым коэффициентам.

Отметим, что, например, такой элемент как функционал оценки обучающейся нейросети языком PMML не специфицирован, поскольку он не относится к этапу *применения* нейросетевой модели. Однако он может быть добавлен к модели через механизм расширений.

Примеры нейросетевых моделей в формате PMML можно найти на WWW-узле разработчиков.

Что дает применение PMML?

Применение стандарта PMML дает целый ряд важных технологических и организационных преимуществ в сравнении с (возможно, более вычислительно эффективными и компактными) уникальными спецификациями.

Прежде всего, **процесс синтеза и обучения моделей отделен от условий применения**. Разработка модели может проводиться на производительных UNIX-станциях или кластерах. Используется же модель в самых разных условиях, вплоть до карманных компьютеров, коммуникаторов, смартфонов и встраиваемых

(embedded) систем и в составе самого разнообразных программ – от WWW-страниц и электронных таблиц EXCEL до промышленных CRM-систем

Во-вторых, **процесс тестирования отделен от процесса разработки**. В компаниях могут создаваться специализированные стенды для тестирования и экспериментального обоснования применений для прогностических моделей. Облегчается сравнительное тестирование моделей, основанных на разных принципах. Организационно, тестирование проводится специалистами по тестированию, при этом процесс приемки моделей может быть эффективно формализован.

Упрощается **процесс документирования и учета моделей**. Этот аспект важен в условиях, когда разрабатываются библиотеки моделей (например, аппроксимационные модели химических или механических свойств материалов, используемые при математическом моделировании), или при одновременной эксплуатации множества моделей для отдельных элементов бизнес - процессов (например, прогностические модели работы сети поставщиков или торговых предприятий).

Снимается ряд ограничений на **использование моделей в программных системах от сторонних разработчиков**, особенно в ситуации с “закрытым” исходным кодом.

Упрощается процесс внедрения программных решений. Окончательные версии модели могут быть встроены в систему на последних этапах внедрения, не сдерживая общий процесс.

Упрощается процесс сопровождения и поддержки – путем поставок сменных PMML-модулей с обновленными версиями моделей, без перекомпиляции кода программной системы. Большая часть поддержки может быть обеспечена дистанционно.

В итоге, применение стандарта направлено на минимизацию целевой функции “время-до-потребителя”. В современном мире часто именно этот показатель оказывается решающим при принятии решений об использовании инновационных разработок.

Задачи

Задача 1. Таблица данных со случайными признаками

Эта задача иллюстрирует поведение методик классификации при обучении на данных, содержащих неинформативные входы. Таблица обучающих данных содержит 10 столбцов-входов со случайными значениями в диапазоне [-1..1]. Выходной столбец содержит два возможных значения {-1,+1}, определяющие код класса. Значение +1 соответствует примерам, у которых значения *первой* из 10 координат удовлетворяют условию.

(31.1) $$\sin\left(5 \cdot \pi \cdot x_1\right) \geq 0.1$$

Остальные входы не участвует в определении класса примера.

На Рис. П1 и П2 приведены результаты двух обученных классификаторов. 1) Нейронная сеть сигмоидальных нейронов при разном числе нейронов в скрытом слое, 2) вероятностное дерево с критерием останова по достижении заданного числа ветвей. Каждый из классификаторов обучался на выборках из 1,000 примеров (рис. П1) и 10,000 примеров (рис. П2). Тестирование проводилось на независимой выборке из 10,000 примеров.

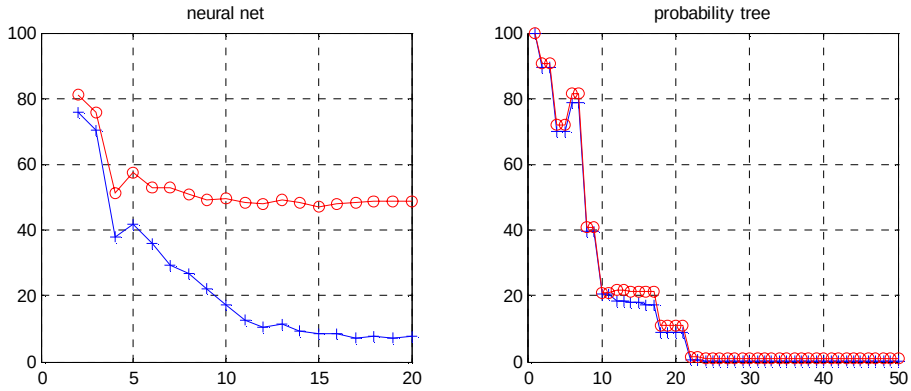

Рис. П1. Ошибка обучения (+++) и тестирования (ooo) как функция сложности модели при обучении на 1,000 примеров. Левый рисунок – нейронная сеть с числом нейронов от 2 до 20. Правый рисунок – вероятностное дерево с числом ветвей от 2 до 50.

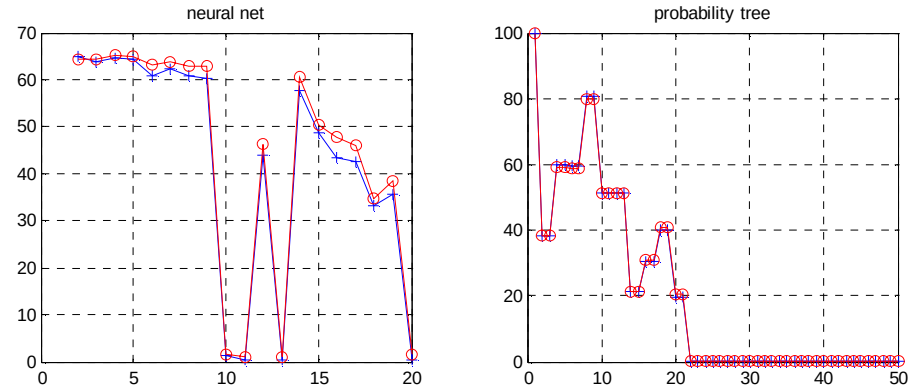

Рис. П2. Те же зависимости при 10,000 обучающих примерах.

Результаты утверждают следующее. Методика вероятностного дерева устойчиво обучается до практически нулевой ошибки и обучения и обобщения при числе ветвей около 25. Сходимость достигается и для тысячи, и для 10 тысяч примеров.

Нейронная сеть при малом числе примеров систематически уменьшает ошибку обучения, при этом ошибка тестирований остается на уровне 50% (это уровень "подбрасывания монеты"). При большом числе примеров (10,000) изменения ошибок обучения и тестирования согласованы, при этом обучение достигается при числе нейронов более 10 и лишь в 4 из 20 вариантов.

Являются ли эти результаты неожиданными? Как их объяснить?

Задача 2. Свойства ядерной аппроксимации

Цель состоит в аппроксимации ступенчатой функции

$$f(x)=\begin{cases}1, & |x|\le 1\\ 0, & |x|>1\end{cases} \tag{32.1}$$

с использованием ряда ядерных функций с фиксированным значением параметра ширины ядра γ

$$\hat{f}(x;\gamma)=\sum_j \alpha_j K\left(|x_j-x|^2;\gamma\right)=\sum_j \alpha_j \exp\left(-\gamma\cdot|x_j-x|^2\right) \tag{32.2}$$

Свободными переменными являются наборы действительных чисел - узловых точек x_j и коэффициентов α_j.

Сходится ли аппроксимация (32.2) к функции (32.1) в метрике $\mathbf{L_2}$ (интеграл от квадрата разности на действительной оси)? Если нет, то можно ли указать оценку точной нижней грани уклонения, как функции параметра γ?

$$E(\gamma)=\inf_{\gamma}\int\left|f(x)-\hat{f}(x;\gamma)\right|^2 dx \tag{32.3}$$

Существует ли функция (32.2), дающая наилучшее приближение? Можно ли указать соответствующие оценки, если сумма (32.2) содержит не более *N* слагаемых?