ЧЕМУ НЕ УЧАТ В АНАЛИЗЕ ДАННЫХ И МАШИННОМ ОБУЧЕНИИ

Александр Дьяконов
(*факультет ВМК, МГУ имени М.В. Ломоносова*)
альфа-версия! Об ошибках и неточностях сообщайте по почте djakonov (a) mail (точка) ru

Цель этой лекции – обратить внимание на некоторые тонкости решения задач анализа данных и машинного обучения, которые часто не освящаются в классических курсах по этим дисциплинам. Материал целиком построен на примерах из жизни, автор постарался отобрать наиболее интересные случаи из недавней практики. Для лучшего понимания материала желательно знание основ анализа данных и машинного обучения [1–2].

Какими нетрадиционными методами решают задачи анализа данных

Мы начнём с очень поучительной истории про то, как иногда надо решать задачи анализа данных. О ней полезно помнить студентам, которые слишком увлекаются «сложными алгоритмами» и верят, что чем больше математической составляющей в алгоритме, тем качественнее его решение.

В конце 1990х годов Интернет-телевидение ещё не было достаточно развито в России, как, впрочем, и сам рунет. Но «обычный» телевизор стал основным средством проведения досуга, появилось множество каналов и передач, а реклама во время трансляций стала приносить телекомпаниям и рекламодателям солидные прибыли[1]. Естественно, процесс получения прибыли захотелось оптимизировать и провести анализ, когда, где и какую рекламу следует показывать. Возникло множество задач анализа данных, одна из которых – определение рейтингов телепередач и репрезентативной выборки пользователей (по которой эти рейтинги можно вычислять).

В телевизоры устанавливались (с разрешения владельцев) специальные приборы, которые регистрировали, какие каналы и передачи телезрители смотрят [3–4]. По многим причинам невозможно поставить приборы во все телевизоры, поэтому небольшой набор устройств надо «раскидать по телезрителям» так, чтобы их показания отражали общую картину популярности в целом (например, в рамках одного города).

Про эту задачу нам даже рассказывали в университетском курсе (тогда автор был ещё студентом ВМК МГУ), в ней множество аспектов:

1) надо грамотно изначально распределить датчики (например, чтобы они были установлены у людей из разных социальных групп),
2) затем надо переустанавливать датчики, одновременно исследуя, все ли группы мы охватили, не изменились ли интересы у какой-то, пусть и малочисленной, группы, не появились ли новые группы,
3) надо кластеризовать[2] пользователей (по интересам), ставить датчики только у небольшой группы представителей кластера, по которой можно восстановить информацию обо всём кластере.

Если задуматься, то задача очень интересная и непростая. Кстати, она актуальна до сих пор, просто поменялась формулировка и сфера приложения. Например, аналогичная задача возникает при размещении банеров на сайте (тут надо выставлять популярные – клики по ним приносят доход, но также проводить разведку, а что стало популярным именно сейчас). Математические техники решения этих задач схожи. Также подобные задачи возникают в системах предложения услуг, например билетов на авиарейсы. Надо понимать, кому и по

[1] До 1990 года было всего несколько телевизионных каналов, а рекламы практически не было. Сейчас в это уже сложно поверить…
[2] Напоминание: **кластеризация –** объединение схожих объектов в группы.

какой цене предлагать (здесь похожие проблемы: кластеризация пользователей, определение стратегий для кластеров, изменение стратегий).

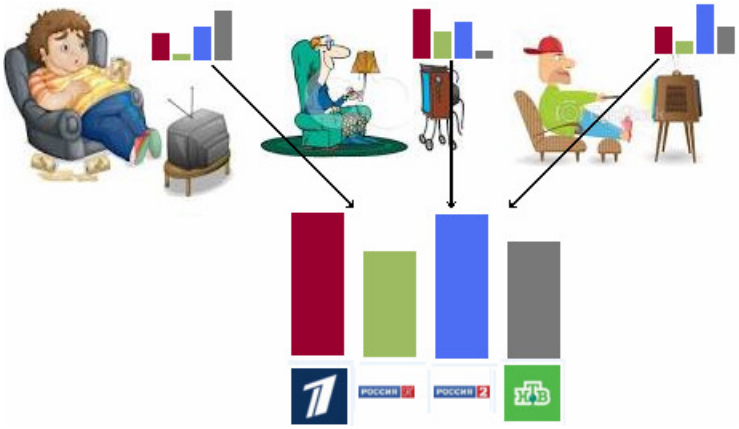

Рис. 1. Получение рейтингов каналов.

В 1990е годы целые группы математиков-прикладников начали решать подобные задачи. Однажды в компанию, которая занималась размещением датчиков, приняли на работу социолога – человека с нематематическим образованием. Он узнал про проблему и очень удивился: «А нельзя ли обойтись одним прибором, причём более примитивным, он будет измерять шум и вибрации? Его надо установить в городской канализации…»

Идея была гениальна! Когда домохозяйки смотрят телесериал, они не моют посуду и не готовят обед, т.е. вода не утекает из квартиры (в 1990е годы телевизор в семье был, как правило, один, и он не стоял на кухне). Когда мужики смотрят бокс, хоккей или футбол, никто из них не побежит в туалет, скажем, во время серии послематчевых пенальти, а вот после – очень может быть. Таким образом, «затишье» в канализационных трубах связано с трансляцией интересных телепередач (ведь телевизор был одним из немногих развлечений), а «шумы и вибрации» возникают во время рекламных пауз и после окончания популярных телепередач. Эта гипотеза была проверена, и оказалось, что с одним более примитивным датчиком[3] задача определения рейтинга передач решается ничуть не хуже, чем с N датчиками (которые распределяются по телезрителям).

Мораль: **нельзя пренебрегать простыми способами решения задач анализа данных**. Неверно, что «чем больше интегралов в решении, тем оно правильнее». Честно говоря, большинство математиков (особенно выпускников мехмата МГУ) очень трудно переубедить, что **задачи можно решать простыми, и даже нематематическими методами**! В реальной жизни у действительно важных задач постановка вообще нематематическая, а способы её формализации могут уводить нас от исходной цели. Как мы увидим дальше, это возникает уже на этапе придумывания функционала качества решения.

Прежде чем решать задачу

Когда автор только начинал разрабатывать алгоритмы классификации, один профессор передал ему несколько задач для тестирования алгоритмов. Первая же задача классификации на два непересекающихся класса сильно удивила. Она была описана в двух csv-файлах[4] с признаковыми таблицами, первый файл – для обучения, второй – для контроля. Открыв файл для обучения, чтобы просто посмотреть на реальные данные, автор заметил простую

[3] Справедливости ради, отметим, что одним датчиком обойтись нельзя, но один действительно заменяет сотни более продвинутых.

[4] Обычный текстовый файл, значения признаков перечисляются через запятые.

закономерность[5]: если значение одного из признаков было равно 4 или 7, то объект относился к первому классу, а в противном случае – ко второму. Файл с тестовой признаковой матрицей подтвердил, что найденный метод решения задачи абсолютно точен – он и контрольную выборку классифицировал со 100%-м качеством.

12	0	4	1	98	1
4	0	7	1	50	1
11	1	5	0	99	2
4	1	10	0	99	2
10	0	7	0	98	1
5	0	9	1	99	2
4	1	4	0	50	1
11	0	3	0	50	2

Рис. 2. Фрагмент таблицы объект-признак с найденной закономерностью.

Это была задача из известного репозитория задач, более того, на ней было принято тестировать свои методы специалистами по машинному обучению. Автор долго отслеживал в различных сборниках конференций таблички с подобными результатами тестирования. С помощью хитроумной бустинговой схемы[6] удалось довести качество решения до 97% и обогнать лидировавшую до этого модификацию SVM[7], но 100% пока никто не получил…

Конечно, не надо путать решение одной конкретной задачи и разработку достаточно универсального метода. Если метод не даёт 100% там, где их можно получить, это не говорит о том, что это плохой метод. Здесь настораживает другое:

1) огромное число специалистов на протяжении нескольких десятков лет решает одну задачу… и не может решить (в самом простом – математическом – смысле этого слова), и даже не видит, что **задача-то скорее искусственная, чем реальная**.
2) очень многие задачи из этого репозитория (особенно на ранней стадии его развития) не были реальными (или же были модельными, или соответствовали совершенно неприкладным задачам, или объёмы данных были совсем ничтожными), когда вокруг было много «настоящих» задач, которые можно и нужно было решать методами машинного обучения. Тем не менее, тестировались именно на этих неинтересных (и никому не нужных) задачах. Некоторые до сих пор решают задачи 1970х годов! Хотя с того времени **изменились не только задачи, но и объёмы данных и требования к решениям…**

Согласитесь, что найденная закономерность делает задачу немного бессмысленной. По крайней мере, на ней не стоит сравнивать различные модели алгоритмов, а если и стоит (для проверки «чувствительности к подобным закономерностям»), то разумно породить ещё несколько задач с подобными «тривиальными решениями». Итак**, прежде чем решать задачу, необходимо её понять, посмотреть на данные, попытаться придумать простейшее решение.**

Какие бывают закономерности

Вот ещё одна поучительная история, которая произошла на одном из соревнований по анализу данных. Организаторы допустили ошибку при создании признаковых таблиц, и

[5] Закономерность умышленно изменена, чтобы читатель не смог точно идентифицировать задачу…

[6] Итерационный способ построения ансамбля алгоритмов. На каждый итерации в ансамбль включается алгоритм, учитывающий ошибки предыдущих, см. [2].

[7] В простейшем варианте алгоритм решает задачу классификации с двумя классами с помощью разделения объектов двух классов «оптимальной» гиперплоскостью, см. [1-2].

целевой вектор[8] на 80% восстанавливался по значениям других признаков (а при наличии достаточной статистики мог быть восстановлен и полностью).

В задаче классификации требовалось определить, воспользуется ли клиент услугами, предлагаемыми в рассылке. Часть признаков описывала клиента:

пол,
идентификационный номер,
регион;

часть специфику услуги/товара:

стоимость,
скидка,
категория и т.п.;

оставшиеся – поведение клиента:

сколько рассылок ему делалось,
сколькими услугами он воспользовался и т.д.

Нетрудно видеть, что если мы можем идентифицировать клиента (это делалось по признакам «идентификационный номер» и «регион», поскольку номер был уникален только в пределах одного региона), тогда, упорядочив его статистику по признаку «сколько рассылок ему делалось», из столбца «сколькими услугами он воспользовался» мы восстанавливаем целевой признак. Если после прошлой рассылки число услуг увеличилось, то он воспользовался предыдущим предложением, иначе – нет. Не восстанавливаются лишь последние значения целевого вектора для каждого клиента, также проблемно восстанавливать в случае пропусков в статистике (см. рис. 3).

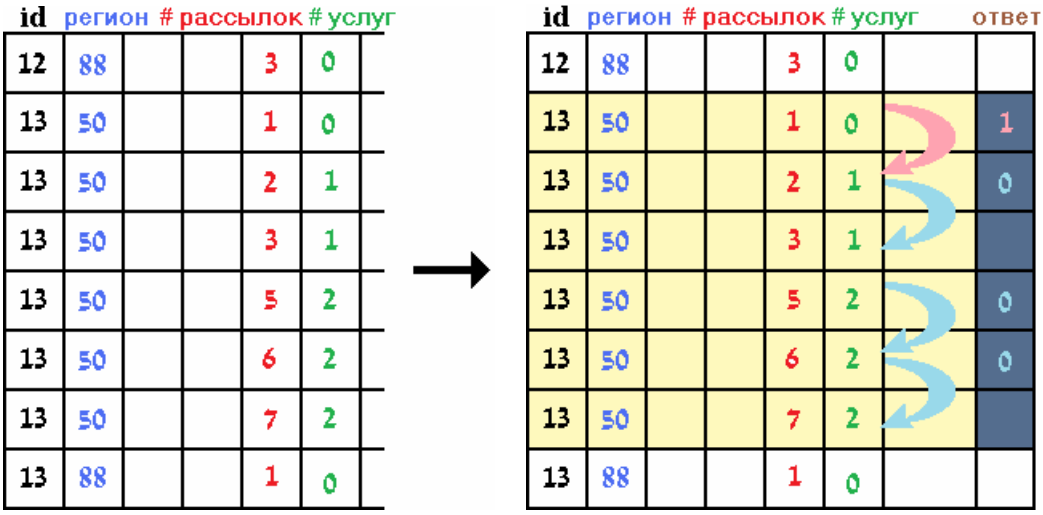

Рис. 3. Восстановление значений целевого вектора без обучения.

Самое интересно, что из 55 активных участников соревнования описанный «дефект данных», заметили лишь пятеро (а на первой стадии соревнования – только трое)! Отметим, что участники были студентами, аспирантами и молодыми учёными из разных стран мира, б***о***льшая часть участников специализируется именно на анализе данных. А ведь явные закономерности в данных следовали просто из названий признаков!

Исследователи, которые решают задачи, часто даже не читают названия признаков! Ну, или читают, но не стараются понять. Общепринятая схема решения: взять исходную

[8] Напомним: **целевой вектор –** вектор, который надо научиться восстанавливать по значениям остальных признаковых векторов (на рис. 3 он помечен словом «ответ»).

таблицу и запихнуть в чёрный ящик типа RandomForest[9], а дальше решение сводится к борьбе с переобучением[10] и настройке параметров алгоритма. В принципе, ничего страшного в этом нет. Просто именно эти специалисты и создают рекомендательные сервисы, системы обработки информации, прогнозные модули и т.п., которые нас окружают. Интересно, насколько бы это всё лучше работало, если бы они задумывались над смыслом данных?

Отметим, что описанная идея «узнавания» целевого вектора реализуется следующим простым MATLAB-кодом.

```
myid = data(:,1:2)*[100 1]'; % уникальный id

U = unique(myid)'; % множество различных id

myans = nan([size(data2,1) 1]); % целевой вектор (для заполнения)

for u = U
    X = data(myid ==u,:); % подматрица ~ один клиент
    X = sortrows(X, nfeatureNumLetters); % сортируем по числу рассылок/услуг
    myansX = nan([size(X,1) 1]); % локальный шаблон (для заполнения)
    I  = diff(X(:,nfeatureNumLetters))==1; % есть статистика
    I0 = diff(X(:,nfeatureNumAction))==0; % потенциальный 0
    I1 = diff(X(:,nfeatureNumAction))==1; % потенциальный 1
    % заполнение локального шаблона
    myansX(find(I&I0)) = 0;
    myansX(find(I&I1)) = 1;
    % в ответ
    myans(myid ==u) = myansX;
end;
```

Код для восстановления значений целевого вектора.

Здесь нет никакого обучения, и в несколько строк находятся метки для примерно 80% объектов. Кстати, для решения задач на остальных объектах хорошо подходил обычный метод ближайшего соседа[11]. Для классификации объекта, т.е. пары «пользователь–услуга» надо было просто найти пользователей из этого региона и такого же пола, которым предлагалась эта же услуга. Оказалось, что в конкретном регионе пользователи одного пола примерно одинаково реагируют на услугу.

Реализация описанной идеи тоже довольно проста: по любой разумной метрике на группе признаков надо было найти всех соседей, находящихся на минимальном расстоянии от классифицируемого объекта (всегда было несколько соседей на нулевом расстоянии). Усреднённые значения целевого признака этих соседей были хорошим ответом в задаче. **Описанный метод ближайшего соседа, который анализирует соседей на фиксированном расстоянии, вы не встретите ни в одной книге по машинному обучению**, хотя метод очень естественный, имеет понятный смысл и успешно работал в реальной прикладной задаче! Впрочем, всех успешных методов не перечислишь – **надо понимать основные идеи алгоритмов и уметь реализовать их на практике.**

Автору часто приходят письма, в которых его просят отыскать закономерность в массиве данных. Особенно удивляют письма с содержанием «Я слышал Вы в этом специалист, вот, я

[9] **Случайный лес** – популярный метод решения задач классификации и регрессии. Дальше в лекции он ещё будет встречаться.

[10] Напомним: **переобучение –** эффект, когда на контроле качество алгоритма хуже, чем на обучении (см. дальше в лекции).

[11] Здесь, **ближайший сосед** – простейший метод в анализе данных, основанный на поиске ближайших объектов к рассматриваемому по некоторой метрике.

придумал модельную задачу, сможете ли Вы её решить?» Человеку, вовлечённому в решение прикладных задач, не хватает времени даже на поиск всех «естественных закономерностей». Более того, отыскать все закономерности попросту невозможно! Представьте, что в описанной выше задаче мы бы не знали названия признаков (часто в соревнованиях так и делают – не сообщают смысл признаков), тогда как можно догадаться создать уникальные идентификаторы клиентов, затем для каждого отсортировать подматрицу по значениям одного из столбцов и смотреть на изменения другого столбца?! А ведь наверняка есть реальная задача, даже с известными признаками, но не до конца понятым их физическим смыслом, которую успешно решает похожий алгоритм! Именно поэтому автор не верит в существование универсальных алгоритмов поиска закономерностей.

Функционалы качества

Заголовок этого раздела ко многому обязывает, ведь в задачах машинного обучения с учителем функционалы качества (ошибки) играют важную роль[12]! Справедливости ради, стоит отметить, что если вам надо быстро получить «приемлемое решение», то можно даже не обращать внимания на эти функционалы. Все они устроены одинаково – чем ближе полученное решение к истинному, тем лучше. Но когда задача решается в рамках соревнования или тендера, или заказчик очень заинтересован в улучшении качества даже на доли процентов, не обращать внимания на функционал качества нельзя! Это один из первых уроков, которые автор получил при решении реальных задач: **исследователь должен понять, как решение будет оцениваться.**

В функционале качества главное не то, какой смысл (физический или геометрический) мы вкладываем в понятие «ошибка», а то, **какое решение минимизирует этот функционал**. Вводя функционал, мы фактически определяем, какое решение нас устроит. Приведём простой пример.

Один заказчик, настаивал, чтобы задача с двумя непересекающимися классами, где правильный ответ для каждого объекта – метка из множества $\{0,1\}$, решалась в регрессионной форме, т.е. алгоритм получал степень принадлежности к классу 1, при этом ошибка такого решения вычислялась по формуле

$$|y_i - a_i| \cdot \begin{cases} 0.8, & y_i = 1, \\ 0.2, & y_i = 0, \end{cases} \qquad (*)$$

где $y_i \in \{0,1\}$ – верная классификация i-го объекта, $a_i \in [0,1]$ – ответ нашего алгоритма. Заказчик оговаривал важность того, что алгоритм должен выдавать промежуточные значения из отрезка [0,1], которые можно будет интерпретировать, например, как вероятности принадлежности к классу 1.

Допустим, мы знаем, что i-й объект принадлежит к классу 1 с вероятностью p. Посчитаем матожидание[13] нашей ошибки:

$$0.8|1-a_i|p + 0.2|a_i|(1-p) =$$
$$= 0.8p - 0.8pa_i + 0.2a_i - 0.2pa_i =$$
$$= 0.8p - (p - 0.2)a_i .$$

[12] Функционал качества формализует понятие качества решения, т.е. чем он больше, тем лучше. Функционал ошибки – «плохость» решения, чем больше, тем хуже. По сути, один превращается в другой умножением на (-1).

[13] Надеемся, что читатель знаком с этим термином (это «**средние потери**», с точность до множителя можно считать, что это суммарная ошибка на классифицированной одинаково большой выборке, p-я доля объектов которой принадлежит классу 1).

Учитывая, что $a_i \in [0,1]$, получаем, что оптимальное решение (которое минимизирует матожидание ошибки)

$$a_i = \begin{cases} 0, & p < 0.2, \\ 1, & p \geq 0.2. \end{cases}$$

Нетрудно видеть, что требования заказчика противоречивы: функционал (*) вынуждает нас выдавать значения из множества $\{0,1\}$, т.е. границы отрезка[14]! **Именно функция ошибки определила класс решений.**

Совет: для незнакомого функционала полезно провести процедуру, описанную выше, чтобы понять, какие решения он считает «подходящими».

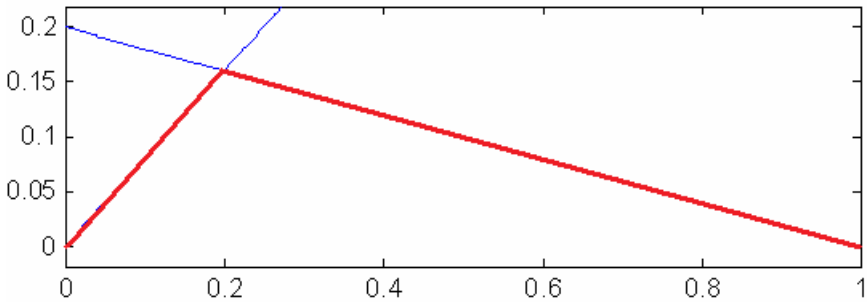

Рис. 4. Матожидание ошибки в зависимости от *p*.

В задачах прогнозирования временных рядов часто используют функционал вида

$$2\sum_i \frac{|a_i - y_i|}{a_i + y_i},$$

поскольку его значения можно интерпретировать в терминах процентов ошибки[15]. Поэтому заказчики часто требуют минимизировать его. Но на рядах с большим числом нулей наблюдается интересный эффект: если прогноз равен нулю, а истинное значение отлично от нуля, или наоборот, то выражение

$$2\frac{|a_i - y_i|}{a_i + y_i}$$

сразу даёт ошибку 200%! Таким образом, при прогнозе 11 вместо 10 получаем ошибку в

$$2\frac{|11-10|}{11+10} = \frac{2}{21} \approx 9.52\%,$$

а при прогнозе 0.001 вместо 0 получаем

$$2\frac{|0.001-0|}{0.001+0} = 2\frac{1}{1} = 200\% !$$

Поэтому прогнозирование сводится к угадыванию нулей/не нулей, а это часто не то, что хочет заказчик. Кстати, ряды с нулями встречаются достаточно часто, например, при прогнозировании продаж товаров: некоторые товары покупаются достаточно редко (максимум 1–2 шт. в месяц) и на них сложно минимизировать подобный функционал – надо угадать, когда именно товар будет куплен.

Совет: Всегда обращайте внимание, какой штраф будет в вырожденных и крайних случаях.

---

[14] Это общее свойство функций ошибки, в которые входят модули разностей истинной величины и прогнозируемой, см. [1].

[15] Считается, что 0/0=0, поскольку это соответствует прогнозу 0 при правильном ответе 0, т.е. нулевой ошибке.

Приведём ещё одну интересную функцию качества решения $a_i \in [0,1]$ в задаче с двумя непересекающимися классами $\{0,1\}$:

$$\begin{cases} \log(a_i), & y_i = 1, \\ \log(1-a_i), & y_i = 0. \end{cases}$$

На первый взгляд, она выглядит очень «экзотической». За неправильный ответ $a_i = 1 - y_i$ вместо y_i она даёт штраф «$-\infty$», и решение сразу становится «ненужным», поэтому алгоритм должен быть очень осторожным (избегать крайностей – ответов из $\{0,1\}$). Проведём описанную выше процедуру – посчитаем матожидание ошибки – получим

$$p\log(a_i) + (1-p)\log(1-a_i). \qquad (**)$$

Теперь минимизируем это выражение, для этого возьмём производную и приравняем к нулю

$$\frac{p}{a_i} - \frac{1-p}{1-a_i} = 0.$$

Отсюда, $a_i = p$. Не правда ли, очень естественно!? Если мы знаем вероятность того, что объект принадлежит к классу 1, то её и надо выдавать в качестве ответа! Заказчик, который хотел интерпретировать ответ как вероятность, должен был требовать минимизацию именно этого функционала[16]! Кстати, подставив $a_i = p$ в (**) получим выражение для энтропии с точностью до знака, которое также известно в анализе данных, например при построении решающих деревьев. Теперь проницательный читатель понимает, **почему при построении решающих деревьев используется энтропия**.

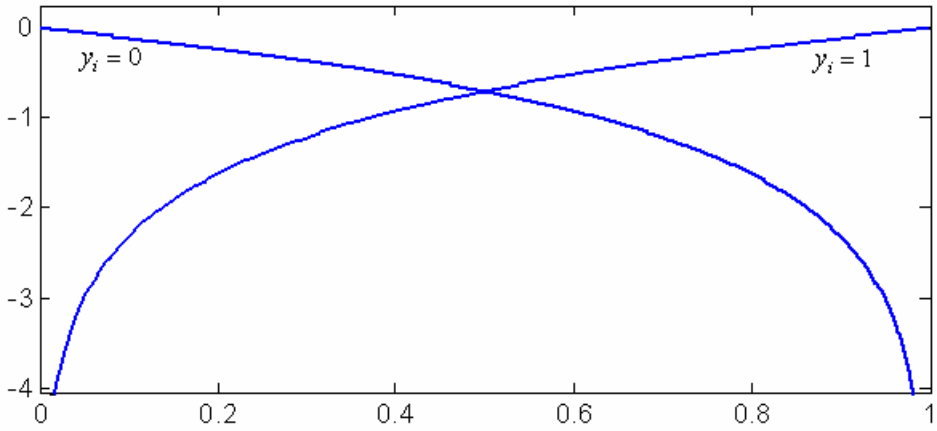

Рис. 5. График функций $\log(a_i)$ и $\log(1-a_i)$ от a_i.

Что значит «придумать алгоритм»?

При решении задач анализа данных и машинного обучения слово «алгоритм» может пониматься в узком и широком смыслах. Рассмотрим этапы решения задачи. Сначала мы коллекционируем данные, потом их предобрабатываем, потом анализируем, строим модель, а потом переводим ответ в нужный формат… В узком смысле – алгоритм решения нашей задачи – это настроенная модель. В широком смысле – алгоритм задействован на всех этапах: предобработка данных, работа модели и перевод ответа в нужный формат. Об этом нельзя забывать, поскольку **предобработку данных и окончательное оформление ответа тоже должен придумать исследователь**!

[16] Здесь можно усмотреть логическую ошибку… если мы знаем вероятность, то её надо выдать в качестве ответа. Но если наш алгоритм выдаёт ответ, имеем ли мы право считать его вероятностью (даже при условии небольших значений (**))? Предлагаем над этим подумать читателю.

Приведём такой пример. Выпускник одного из ведущих вузов в своём дипломе решал непростую реальную задачу классификации. Не будем описывать её точную постановку, приведём итоговую табличку с результатами экспериментов из дипломной работы:

	Процент верных ответов:		
Метод:	SVM	RandomForest	kNN
Нормировка по максимуму[17]	**85.0%**	78.2%	75.5%
Стандартизация[18]	82.2%	77.9%	79.8%

Работа была признана очень сильной, и никто не обратил внимания на огромный методический недостаток. Результат RandomForest не должен зависеть от нормировок признаков (поэтому эксперименты с разными нормировками для него можно было не делать). Результат SVM, напротив, сильно зависит от нормировок признаков (это следует просто из определения алгоритма). И мы видим, что при двух нормировках получились совершенно разные результаты. А что если попробовать третью нормировку, четвёртую…? Где гарантия, что мы не получим ещё более качественное решение?

Как раз именно в этой задаче, где уже найдена неплохая модель, зависящая от предобработки**, решение сводится к поиску этой удачной предобработки**! Большое число прикладных задач автор успешно решил именно перебором нормировок для SVM или метода ближайшего соседа!

Нелинейные методы для линейных задач

Теперь поговорим о последнем этапе решения задачи – переводе в нужный формат. **Это вовсе не запись в файл с требуемым расширением, это, в том числе, получение решения с нужными свойствами**. Автор, будучи уже довольно опытным в решении задач классификации, сделал для себя интересное открытие. Допустим, мы решаем задачу с двумя непересекающимися классами {0,1} методом случайного леса (RandomForest). При построении каждого дерева мы используем энтропийный критерий расщепления[19]. Оценка принадлежности к классу вычисляется как среднее ответов деревьев из леса, и это очень хорошее приближение вероятности, а качество решения определяется функцией (**). Казалось бы, всё сходится: и метод построения деревьев, и метод получения ответа, и функция ошибки идеально подходят друг другу! Но оказалось, что часто **такое решение не является наилучшим**!

Случайный лес можно назвать «линейным методом», поскольку среднее арифметическое (деревьев) – линейная функция. Конечно, сами решающие деревья – это нелинейные функции (от значений признаков), но можно считать, что каждое дерево оценивает вероятность принадлежности классу 1, тогда получаем линейную функцию от оценок. Оказывается, на практике решение можно улучшить, причём неожиданным способом – добавив в решение нелинейность и «искривив ответ».

[17] Все значения конкретного признака делятся на максимальное по модулю значение. Это и следующее преобразование проводится для всех признаков.
[18] Из значений конкретного признака вычитается среднее, затем результат делится на дисперсию.
[19] Мы уже говорили об энтропии в этой лекции. Подробнее о критерии см. в [2]. Здесь важно, что использование энтропийного критерия обеспечивает хорошую оценку вероятности.

Рассмотрим работу построенного алгоритма на отложенном контроле[20]. Пробежимся по отрезку $[0,1]$ окном ширины 2ε, для каждого значения $t \in [0,1]$ вычислим, какой процент объектов с ответами нашего алгоритма из отрезка $[t-\varepsilon, t+\varepsilon]$ принадлежит классу 1[21]. Получим график, изображённый на рис. 7. Если бы наш алгоритм выдавал вероятность принадлежности, то график был бы очень близок к прямой. Но он отличен от прямой и похож на сигмоиду

$$\varphi(t) = \frac{1}{1+e^{-t}}.$$

Если же мы построим график нашей функции (процент точек первого класса) в зависимости от $\varphi(t)$, т.е. «с точки зрения алгоритма, к которому применили преобразование φ», то наш график превратится в прямую.

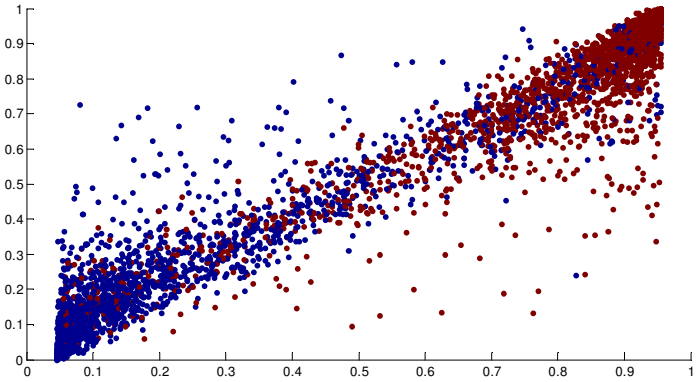

Рис. 6. Объекты двух классов реальной задачи, по осям отложены ответы двух алгоритмов.

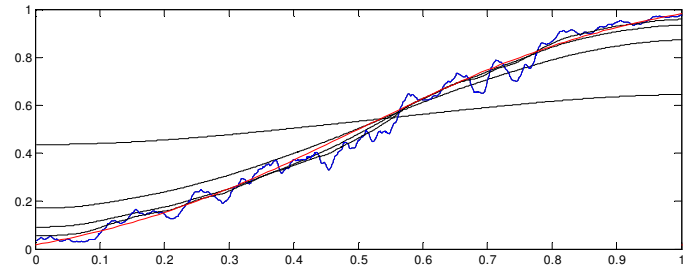

Рис. 7. Функции, полученные при разных усреднениях (параметре ε).

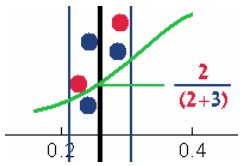

Рис. 8. Значение функции в точке $t \in [0,1]$ с окном $[t-\varepsilon, t+\varepsilon]$[22].

Поэтому, чтобы интерпретировать ответы алгоритма как вероятности, его выход надо «исказить» нелинейным преобразованием φ. Интересно, что **в нашей линейной задаче вдруг возникла нелинейность в виде сигмоиды, которая, например, часто используется как функция активации в теории нейронных сетей** [1–2]. Автор, после того, как в

[20] Исходные данные часто разбивают на две части: на первой – настраивают алгоритм, а на второй (это и есть **отложенный контроль**) – оценивают качество работы алгоритма.
[21] Подумайте, как выбрать ε.
[22] По горизонтали отложены ответы нашего алгоритма. Что отложено по вертикали – не имеет значения.

нескольких реальных задачах увидел необходимость использования сигмоиды, поверил, что концепция нейронных сетей очень правильна. По крайней мере, в ней используются правильные функции активации.

При решении описанной задачи пришлось искать оптимальное преобразование в целом классе сигмоид

$$\varphi(t) = b\left(\frac{1}{1+e^{-a(t-0.5)}} - \frac{1}{2}\right) + \frac{1}{2}. \qquad (***)$$

Вопрос. Попробуйте догадаться, почему был выбран именно такой класс? Форма записи (***) должна вам подсказать.

Что нужно знать об алгоритмах?

Ещё один пример из истории защит дипломных работ. Один выпускник рассказывал о решении задачи классификации с большим числом бинарных признаков. Он разработал алгоритм, который сравнил с классическими на одной конкретной задаче. Особую гордость дипломник испытал существенно обойдя решающие деревья, которые, как он сказал, «очень популярны на западе». Отрыв составил порядка 10% (по качеству решения)!

Правда состоит в том, что само по себе решающее дерево очень слабый метод! Это следует хотя бы из того, что в подобной задаче, где число признаков велико, оно не использует всю информацию, поскольку строится по некоторому подмножеству признаков. Но вот уже бустинговые схемы над деревьями, а также простые ансамбли деревьев – RandomForest, являются очень мощными конструкциями! На среднестатистической реальной задаче они превосходят одно дерево на 15–20%! Поэтому, кстати, отрыв в 10% не так уж и велик… С ансамблями деревьев, конечно, дипломник свой алгоритм не сравнивал… **Необходимо знать, какие алгоритмы и в каких задачах показывают хорошие результаты**.

Ещё один пример из жизни специалистов по анализу данных, которым не хватило времени для экспериментов при решении одной реальной задачи. Поскольку в задаче неплохо работал бустинг над деревьями, при получении итогового решения они в несколько раз увеличили число деревьев в бустинговой схеме. Как следствие – произошло переобучение.

Эта ошибка из разряда «детских», поскольку автор говорит о ней уже на втором семинаре занятий по машинному обучению. В случайных лесах увеличение числа деревьев не ухудшает решение (даже наоборот, причём часто деревьев нужно более 2000), а в бустинговых схемах при увеличении числа деревьев ошибка на отложенном контроле сначала падает, а потом резко возрастает.

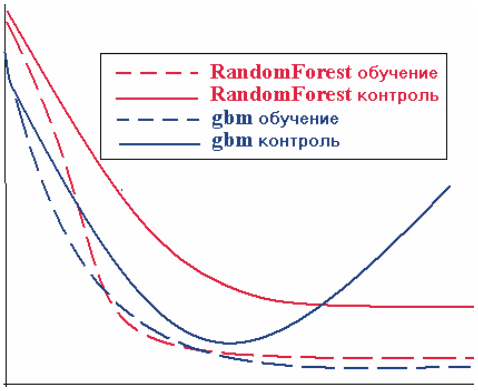

Рис. 9. Графики функции ошибки, по горизонтали – число деревьев в ансамбле.

Более того, для осознания, что параметр «число деревьев» в бустинге очень важен, даже не обязательно знать, что такое бустинг и что такое дерево, достаточно просто внимательно изучить интерфейс соответствующих функций в классических средах решения задач машинного обучения. Например, в среде R [5] в пакете gbm (generalized boosted regression models), который указанные специалисты и использовали, у функции настройки модели есть специальные аргументы, которые позволяют проводить контроль качества настраиваемой модели (cv.folds). В пакете randomForest ничего подобного нет. Следовательно, в случае gbm контроль важен! Таким образом, **со многими алгоритмами можно работать как с чёрными ящиками, надо только внимательно изучить «пульты управления» этих ящиков**, а заодно поэкспериментировать с ними на практике.

Заключение

Когда автор закончил описание всех поучительных историй, большинство из которых произошло в последний год, стала видна интересная закономерность. Истории не подбирались специально, и их порядок определился лишь настроением автора в процессе написания. Но если проследить, о чём была каждая, то чётко вырисовываются этапы решения реальной задачи.

1. Разобраться, а стоит ли задачу решать математическими методами.
2. Посмотреть на данные, понять их смысл.
3. Найти тривиальное решение.
4. Понять и исследовать функционал качества решения.
5. Уделить внимание предобработке данных.
6. Уделить внимание постобработке данных.
7. Грамотно настраивать алгоритмы, учитывая их специфику.

Надеемся, теперь для читателя это список станет не формальным руководством, а поучительным назиданием, и он не пополнит коллекцию подобных историй. **Ведь важны не правила, а умения!**

Список литературы, рекомендованной студентам по теме лекции

[1] *Дьяконов А.Г.* Анализ данных, обучение по прецедентам, логические игры, системы WEKA, RapidMiner и MatLab // Учебное пособие, выложено в электронном виде
http://www.machinelearning.ru/wiki/images/7/7e/Dj2010up.pdf
Опубликовано в виде двух книг:
Дьяконов А.Г. Практикум на ЭВМ кафедры математических методов прогнозирования (логические игры, обучение по прецедентам): Учебное пособие. – М.: Издательский отдел факультета ВМиК МГУ им. М.В. Ломоносова; МАКС Пресс, 2010. – 164с.: ил. (ISBN 978-5-89407-431-3)
Дьяконов А.Г. Практикум на ЭВМ кафедры математических методов прогнозирования (системы WEKA, RapidMiner и MatLab): Учебное пособие. – М.: Издательский отдел факультета ВМиК МГУ им. М.В. Ломоносова; МАКС Пресс, 2010. – 133с.: ил. (ISBN 978-5-89407-432-0)

[2] *Воронцов К.В.* Курс лекций Математические методы обучения по прецедентам, МФТИ, 2004-2008. см. www.MachineLearning.ru

[3] Рейтинг Нильсена // Википедия http://ru.wikipedia.org/wiki/Рейтинг_Нильсена

[4] Интервью «Как измерить телезрителя» // Итоги №36/274(11.09.01)
http://www.itogi.ru/archive/2001/36/107466.html
(в 2001 году автор как раз слушал курс, на котором обсуждались математические проблемы, возникающие при подсчёте ТВ-рейтинга).

[5] «R» // Бесплатная система для статистических вычислений и анализа данных
http://cran.gis-lab.info/

[6] *Дьяконов А.Г.* Научно-популярная лекция «“Шаманство” в анализе данных»
http://alexanderdyakonov.narod.ru/lpotdyakonov.pdf

[7] *Дьяконов А.Г.* Научно-популярная лекция «Введение в анализ данных»
http://alexanderdyakonov.narod.ru/intro2datamining.pdf