Введение в анализ данных. Ответы на вопросы к коллоквиуму.

Егор Соловьёв

NB: это ответы на теоретические вопросы к предстоящему коллоквиуму по курсу майнора. Они не претендуют на абсолютную точность/полноту и (надеюсь) должны помочь систематизировать материал для подготовки. Многое здесь взято из лекций, которые можно и нужно читать:

- https://github.com/esokolov/ml-minor-hse/tree/master/lectures-2017 – собственно слайды лекций по курсу майнора
- https://github.com/esokolov/ml-course-hse/tree/master/2016-fall/lecture-notes – конспекты лекций курса по машинному обучению на ПМИ ФКН.

1 Что такое объекты и признаки в машинном обучении? Для чего нужен функционал качества? Что такое алгоритм/модель?

Объекты – это некоторые ситуации (прецеденты). Например, если мы хотим предсказать прибыль ресторана в зависимости от места его размещения, и нам доступна информация о том, какую прибыль приносили другие рестораны, то в данном случае объектами являются рестораны.

Множество примеров – объектов, для которых мы знаем точное значение предсказываемой величины (или класса, в зависимости от поставленной задачи), будем называть *обучающей выборкой*.

Признаки – это некоторые характеристики объектов, с которыми, в частности, умеет работать компьютер (потому что машины не умеют оперировать с, казалось бы, привычными человеку понятиями). Они бывают не только числовыми (число посетителей ресторана в некоторый день), но и бинарными (принимающими значения 0 - ложь или 1 - истина) и некоторыми другими (об этом ниже).

Допустим, мы хотим предсказывать некоторую величину y для произвольного объекта (например, прибыль нового ресторана за первый год работы). Имея некоторую обучающую выборку – то есть объекты $\{x_1, x_2, ..., x_n\}$ и ответы (например, их прибыли за первый год работы) к ней $\{y_1, y_2, ..., y_n\}$, мы можем построить некоторую *модель (алгоритм)*, который предсказывает ответ для произвольного объекта. По сути, алгоритм – это некоторая функция $a(x)$, которая для произвольного объекта x возвращает предсказание значения некоторой целевой величины (в нашем случае это та самая прибыль ресторана).

Большинство возникающих задач не имеют точного решения (точная прибыль ресторана

зависит от слишком большого количества внешних факторов), но достаточно получить относительно точное приближение. Естественно, хочется получить наиболее точную модель (нам не очень подходит алгоритм, утверждающий, что прибыль любого ресторана всегда составит условные 10 долларов). Для этого вводят *функционал качества*. Чем меньшее (или большее – в зависимости от того, как устроен этот функционал) значение он выдает, тем алгоритм лучше. Например, функционалом качества является среднеквадратичная ошибка (MSE):

$$Q(a, X) = \frac{1}{N}\sum_{i=1}^{N}(a(x_i) - y_i)^2$$

То есть $a(x_i)$ – это значение, предсказанное алгоритмом на объекте x_i, а y_i – истинное значение целевой переменной. Квадрат их разности – это квадратичная ошибка на данном объекте. Если сложить их для всех объектов и разделить на их число (проще говоря, усреднить), получится как раз формула выше.

2 Чем задача классификации отличается от задачи регрессии? Приведите примеры задач классификации и регрессии.

Задача классификации заключается в определении принадлежности объекта одному (или нескольким, в зависимости от задачи) классам. Например, пользователей поисковых систем можно разделить на несколько (допустим, непересекающихся) классов: студенты, школьники и т. д. Определение того, к какому классу относится пользователь, исходя из его поисковых запросов – это решение задачи классификации.

Задача регрессии возникает тогда, когда целевая переменная представляет собой (действительное) число, и нам нужно восстановить некоторую закономерность между объектом и этой целевой переменной. Например, предсказание прибыли ресторана представляет собой задачу регрессии.

3 Что такое вещественные (числовые), бинарные, категориальные признаки? Приведите примеры.

- *Вещественные (числовые) признаки* выражены (как ни странно) числами. Например, это вес или рост человека.
- *Бинарные признаки* принимают два значения – 0 или 1. Их можно представлять как ответ на вопрос, который допускает лишь ответы «да» или «нет» («Зарегистрирован ли данный пользователь на сайте?»)
- *Категориальные признаки* определяют принадлежность объекта некоторой категории (пол, страна и т. д.), при этом эти категории неупорядочены и их количество конечно.

4 В чём заключается обобщающая способность алгоритма машинного обучения? К чему приводит её отсутствие? Что такое переобучение?

Обобщающая способность алгоритма машинного обучения – это способность выдавать правильные (с некоторой степенью точности) ответы для неизвестных объектов, будучи обученным на наборе известных объектов (обучающей выборке). Например, алгоритм, всего лишь запомнивший все объекты в обучающей выборке и выдающий абсолютно точный правильный ответ на каждом из них, будет бесполезен на объектах с неизвестным заранее ответов (так же, как нельзя наизусть заучить учебник с решениями некоторых задач и после этого легко решать такие же задачи, но со слегка изменёнными числами).

Отсутствие такой способности у алгоритма приводит к переобучению.

Переобучение – явление, когда построенная модель хорошо объясняет примеры из обучающей выборки, но плохо работает на новых объектах, не участвовавших в обучении.

При использовании градиентного спуска (например, для линейной регрессии) характерный признак переобучения – большие веса в модели. Для борьбы с ним применяют регуляризацию.

5 Что такое отложенная выборка? Что такое кросс-валидация? (скользящий контроль)? Как ими пользоваться для выбора гиперпараметров?

Отложенная выборка – это часть объектов обучающей выборки, которая не используется в построении модели для того, чтобы на ней можно было измерить качество построенной модели (чтобы понять, что модель не переобучилась).

Кросс-валидация – метод оценки модели и её поведения на независимых данных. При оценке модели имеющиеся в наличии данные разбиваются на k частей. Затем на всех частях, кроме одной производится обучение модели, а оставшаяся часть данных используется для тестирования (оценки качества). Процедура повторяется k раз, чтобы каждая из k частей данных использовалась для тестирования. Такой метод позволяет лучше оценить обобщающую способность алгоритма.

При выборе гиперпараметров можно для каждой комбинации гиперпараметров провести кросс-валидацию модели с данными гиперпараметрами и выбрать тот набор гиперпараметров, которому соответствует меньшая ошибка (большее качество) на кросс-валидации.

6 В чём заключается гипотеза компактности?

Гипотеза компактности заключается в предположении о том, что близкие объекты, скорее всего, лежат в одном классе. Близость объектов можно определять, вводя некоторую функцию расстояния между объектами.

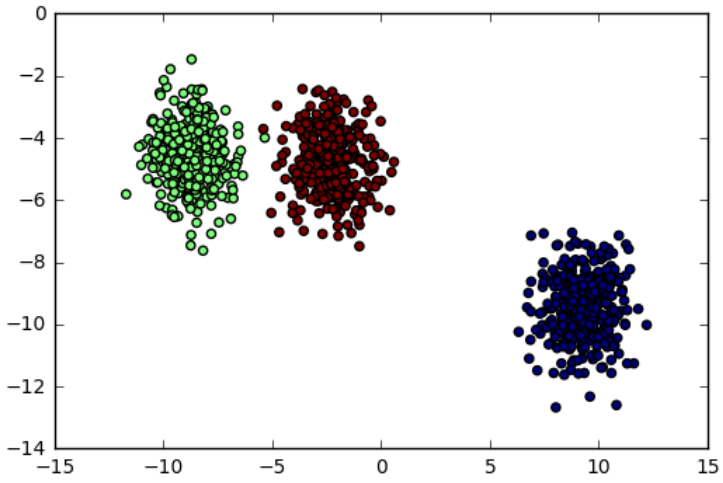

Рис. 1: Пример выборки, для которой выполняется гипотеза компактности.

7 Как метод k ближайших соседей определяет класс для нового объекта?

Допустим, что у нас есть некоторое множество объектов, для которых известно, к каким классам они принадлежат. Пусть теперь поступил новый объект, и его необходимо классифицировать. Необходимо найти k ближайших к нему известных объектов (k – некоторый зафиксированный гиперпараметр). Отнесём новый объект в тот класс, которому принадлежит больше всего из его k соседей.

Формально:

$$a(x) = \arg\max_{y \in \mathbb{Y}} \sum_{i=1}^{k} [y_{(i)} = y]$$

То же, но по-русски. Пусть $y_{(1)}, y_{(2)}, ..., y_{(k)}$ – это номера классов, к которым принадлежат k ближайших соседей нового объекта x. Тогда $[y_{(i)} = y] = 1$, если i-й сосед объекта x принадлежит классу y, и 0 иначе. Тогда, если мы зафиксируем некоторый класс y, то $\sum_{i=1}^{k} [y_{(i)} = y]$ – это как раз число соседей, которые входят в число k ближайших к новому объекту x и принадлежат классу y (потому что каждый такой сосед вносит вклад 1 в сумму, а остальные порождают нули). Если мы рассмотрим такую сумму для всех существующих классов (здесь $\mathbb{Y}$ – множество всех классов) и возьмем тот класс y, для которого она максимальна, получим, что нужный нам класс как раз выражается формулой выше.

8 Опишите метод k ближайших соседей с парзеновским окном. Какие в нём есть параметры?

В формуле, приведённой в предыдущем ответе, каждый сосед вносит вклад 1 в сумму, которую можно интепретировать как число соседей, принадлежащих некоторому классу. Можно предположить, что близкие соседи должны иметь больший вес и влияние на принимаемое решение, чем дальние, поэтому формулу можно модифицировать, добавив веса:

$$a(x) = \arg\max_{y \in \mathbb{Y}} \sum_{i=1}^{k} w_i [y_{(i)} = y]$$

где $w_i = K\left(\frac{\rho(x, x_{(i)})}{h}\right)$ – парзеновское окно ($\rho(x, x_{(i)})$ – расстояние между объектами x и $x_{(i)}$). Функция K называется его ядром, а число h – шириной.

9 Запишите формулу метода kNN для регрессии.

$$a(x) = \frac{\sum_{i=1}^{k} w_i y_{(i)}}{\sum_{i=1}^{k} w_i}$$

То же, но по-русски. В вопросах выше метод ближайших соседей применялся для классификации, то есть он возвращал номер класса, к которому принадлежит новый объект x. Здесь он используется для регрессии (то есть для восстановления некоторой числовой величины). Можно, например, предположить, что значение неизвестной для объекта величины равно среднему значению этой величины среди его k ближайших соседей. Однако, такой метод не будет учитывать сами расстояния до соседей, поэтому можно добавить к значению величины каждого из соседей некоторый вес w_i, который может зависеть от расстояния (например, с помощью парзеновского окна – см. выше).

10 Что такое градиент? Какое его свойство активно используется в машинном обучении?

Градиент функции нескольких переменных в некоторой точке – это вектор, составленый из частных производных данной функции в этой точке. Пусть есть функция $f(x_1, x_2, ...x_n)$, тогда её градиент в некоторой точке x_0 (у которой есть n координат $(x_0^{(1)}, x_0^{(2)}, ...x_0^{(n)})$):

$$\text{grad} f(x_0) = \nabla f(x_0) = \left(\frac{\partial f}{\partial x_1}(x_0), \frac{\partial f}{\partial x_2}(x_0), ..., \frac{\partial f}{\partial x_n}(x_0)\right)^T$$

Градиент функции характеризуется тем, что он указывает направление наискорейшего роста функции в данной точке. Это свойство активно используется в алгоритме градиентного спуска.

11 Опишите алгоритм градиентного спуска.

Алгоритм градиентного спуска позволяет находить минимум функции нескольких переменных, если можно находить ее градиент.

Градиент функции $\text{grad} f = \nabla f$ указывает направление наискорейшего роста функции в данной точке, тогда вектор антиградиента $-\nabla f$ (вектор, у которого все координаты равны координатам вектора градиента с другим знаком) указывает направление наискорейшего спуска функции. Тогда можно начать оптимизацию в некоторой точке, сдвинуться в направлении антиградиента, пересчитать антиградиент, снова сдвинутся в его направлении и повторять до тех пор, пока не выполнен один из критериев останова (грубо говоря, когда сдвиг становится очень мал).

Формально: задаём некоторую начальную точку $x^{[1]}$. Следующую точку будем получать из предыдущей соотношением

$$x^{[k+1]} = x^{[k]} - \lambda^{[k]} \nabla f(x^{[k]})$$

до тех пор, пока не настанет сходимость, определяемая критерием останова (например, модуль разности $|x^{[k+1]} - x^{[k]}| < \epsilon$, где ϵ – заданный параметр). Здесь $\lambda^{[k]}$ - величина шага на данном этапе. Ее можно выбирать, например, постоянной или некоторым образом зависящей от k (например, $\lambda^{[k]} = \frac{1}{k}$)

12 Как обучается линейная регрессия?

Линейная регрессия восстанавливает зависимость между объектом x, представленным вектором признаков $(x_1, x_2, ...x_n)^T$, и некоторой целевой переменной y в виде:

$$a(x) = w_0 + w_1 x_1 + ... + w_n x_n$$

(возможно, без w_0).

Если модель использует сдвиг w_0, имеет смысл ввести признак x_0, всегда равный единице, тогда зависимость можно будет записать следующим образом:

$$a(x) = w_0 \cdot 1 + w_1 x_1 + ... + w_n x_n = w_0 \cdot x_0 + w_1 x_1 + ... + w_n x_n = \langle w, x \rangle$$

где $\langle \cdot, \cdot \rangle$ – скалярное произведение.

Обучение заключается в подборе весов w_i, минимизирующих некоторый функционал качества (например, среднеквадратичную ошибку) на тренировочной выборке. Это можно сделать с помощью линейной регрессии. Например, если в качестве функционала качества выступает $MSE = \frac{1}{N} \sum_{i=1}^{N} (a(x_i) - y_i)^2$, необходимо минизировать функцию:

$$\sum_{i=1}^{N} ((w_0 + w_1 x_1 + ... + w_n x_n) - y_i)^2$$

(постоянный множитель $\frac{1}{N}$ был опущен, потому что минимум функции, умноженной на ненулевое число, будет достигаться там же, где и исходной).

13 Почему наличие линейно зависимых признаков представляет проблему при обучении линейной регрессии?

Наличие линейно зависимых признаков приводит к переобучению линейной модели (в частности, к появлению больших весов). Веса становится сложно интерпретировать и судить о важности различных признаков.

14 Что такое регуляризация? Как она помогает бороться с переобучением?

Явный признак переобучения в линейной модели – большие веса w_i. Для борьбы с переобучением можно ввести штраф за большие веса и минимизировать не функционал

$$\frac{1}{N}\sum_{i=1}^{N}((w_0 + w_1x_1 + ... + w_nx_n) - y_i)^2$$

а регуляризованный:

$$\frac{1}{N}\sum_{i=1}^{N}\left(((w_0 + w_1x_1 + ... + w_nx_n) - y_i)^2\right) + \lambda||w||^2$$

где $||w||^2 = \sum_{i=1}^{n} w_i^2$ – регуляризатор, а λ – правильно подобранный гиперпараметр.

15 Чем L1-регуляризация отличается от L2-регуляризации?

L1-регуляризация заключается в использовании L_1-нормы

$$||w||_1 = \sum_{i=1}^{n}|w_i|$$

и минимизации следующего регуляризованного функционала ошибки:

$$\frac{1}{N}\sum_{i=1}^{N}\left(((w_0 + w_1x_1 + ... + w_nx_n) - y_i)^2\right) + \lambda||w||_1$$

L2-регуляризация использует норму $||w||^2 = \sum_{i=1}^{n} w_i^2$, а функционал ошибки выглядит так:

$$\frac{1}{N}\sum_{i=1}^{N}\left(((w_0 + w_1x_1 + ... + w_nx_n) - y_i)^2\right) + \lambda||w||^2$$

16 Что такое масштабирование (шкалирование) признаков? Как его проводить? Зачем это нужно?

Масштабирование признаков – это некоторые действия над ними, которые делают их лежащими в одном масштабе. Это улучшает сходимость градиентных методов.

Например, можно вычесть из каждого признака его среднее и разделить на стандартное отклонение. Если x_j – некоторый признак, ему будет соответствовать модифицированный признак

$$x'_j = \frac{x_j - \mu_j}{\sigma_j}$$

где

$$\mu_j = \frac{1}{l}\sum_{i=1}^{l} x_i^j$$

$$\sigma_j = \sqrt{\frac{1}{l}\sum_{i=1}^{l}\left(x_i^j - \mu_j\right)^2}$$

17 Как выглядят модели линейной классификации в случае двух классов?

Пусть есть два класса $\{-1, +1\}$. Тогда запишем линейный классификатор следующим образом:

$$a(x) = \text{sign}\left(w_0 + \sum_{j=1}^{d} w_j x^j\right)$$

где d – число признаков, $\text{sign}(t)$ - функция, возвращающая знак числа t (-1 или 1).

Если добавить единичный признак x_0, то $\left(w_0 + \sum_{j=1}^{d} w_j x^j\right) = \left(w_0 x_0 + \sum_{j=1}^{d} w_j x^j\right) = \left(\sum_{j=0}^{d} w_j x^j\right)$ $= \langle w, x\rangle$ (скалярное произведение), и классификатор запишется в следующем виде:

$$a(x) = \text{sign}\langle w, x\rangle$$

То же, но по-русски. Уравнение $\langle w, x\rangle = 0$ задает гиперплоскость в $(d+1)$–мерном пространстве, которая разделяет это пространство на две «половины». Объекты, попавшие в одну из них, попадают в класс -1 (для них $\langle w, x\rangle < 0$), попавшие в другую - в класс $+1$. Чем дальше объект отстоит от этой разделяющей плоскости, тем с большей уверенностью можно утверждать, что его классификация была произведена корректно.

18 Что такое отступ? Для чего он нужен?

Определим для объекта x_i из обучающей выборки отступ:

$$M(x_i) = y_i\langle w, x_i\rangle$$

Если отступ положителен, то классификатор даёт верный ответ на данном объекте, если отрицателен – он ошибается. Причём чем больше отступ (и, соответственно, больше скалярное произведение $\langle w, x_i\rangle$), тем классификатор «увереннее» обработал объект.

19 Как обучаются линейные классификаторы (общая схема с верхними оценками)?

Как и для регрессии, здесь необходимо минимизировать некоторый функционал ошибки для того, чтобы найти оптимальные параметры алгоритма. Например, можно минимизировать долю неверных ответов, тогда функционал ошибки запишется следующим образом:

$$Q(w, X) = \frac{1}{l}\sum_{i=1}^{l}[a(x_i) \neq y_i]$$

Используя введённое понятие порога $M_i = y_i\langle w, x_i\rangle$, функционал ошибки можно переписать вот так:

$$Q(w, X) = \frac{1}{l}\sum_{i=1}^{l}[y_i\langle w, x_i\rangle < 0]$$

Однако, оптимизировать такую ошибку достаточно сложно (и вряд ли получится с помощью градиентного спуска). Можно попробовать оценить функцию $[M_i < 0] = [y_i\langle w, x_i\rangle < 0]$ сверху некоторой хорошей (гладкой, поддающейся оптимизации) функцией $L(M_i)$, тогда

$$Q(w, X) = \frac{1}{l}\sum_{i=1}^{l}[y_i\langle w, x_i\rangle < 0] = \frac{1}{l}\sum_{i=1}^{l}[M_i < 0] \leq \frac{1}{l}\sum_{i=1}^{l} L(M_i)$$

Тогда можно минимизировать эту оценку $\frac{1}{l}\sum_{i=1}^{l} L(M_i)$ и надеяться, что при достижении им минимума будет достигаться и минимум исходной ошибки (однако, это не всегда так).

20 Для чего может понадобиться оценивать вероятности классов?

Оценивать вероятности классов может потребоваться тогда, когда решение об классификации объекта будет принято позже. Примером может являться задача кредитного скоринга (выдавать клиенту кредит или нет). Если банк обладает большим количеством свободных средств, он может принять решение выдавать кредиты большему количеству людей, в том числе и менее надёжным. В период экономического кризиса, напротив, кредиты будут выдаваться исходя из стратегии минимизация риска их невозврата. Тогда можно рассчитать для каждого клиента вероятность того, что он вернёт кредит в срок, и решать, что делать с этой информацией, исходя из текущих условий.

21 Как обучается логистическая регрессия? Запишите функционал и объясните, откуда он берётся.

Возьмём в качестве верхней оценки $L(M_i) = \log(1 + \exp(-M_i))$. Тогда задача построения линейного классификатора сводится к задаче минимизации верхней оценки функционала ошибки

$$\sum_{i=1}^{l} \log(1 + \exp(-M_i))$$

Эта верхняя оценка получается из следующих соображений. Пусть наш алгоритм возвращает для объекта x некоторое число $b(x)$ в пределах от 0 до 1. Такое число можно интерпретировать как вероятность принадлежности объекта x классу $+1$. Для того, чтобы значения $b(x)$ лежали в данных пределах, можно положить $b(x) = \sigma(\langle w, x\rangle)$, где $\sigma(x) = \dfrac{1}{1+e^{-x}}$ – хорошая (принимающая значение 0 на $-\infty$, $1/2$ в нуле, 1 на $+\infty$ и монотонно возрастающая) и гладкая функция.

Для оценки модели можно использовать функционал правдоподобия (чем он больше, тем лучше).

$$\prod_{i=1}^{l} b(x_i)^{[y_i=+1]}(1 - b(x_i))^{[y_i=-1]}$$

Функционал правдоподобия получается из следующих соображений (подробнее и с доказательством в начале этой лекции). Понятно, что в выборке может быть несколько объектов с одинаковыми значениями признаков (поскольку это, вообще говоря, один объект, обозначим его как x) и разными значениями целевой переменной (номерами классов). Пусть задана некоторая вероятность того, что объект принадлежит классу $+1$, обозначаемая как $p(y = +1|x)$. Тогда если количество объектов x в выборке достаточно велико (стремится к бесконечности), то доля положительных объектов среди них будет стремиться как раз к этой вероятности. Вспомним теперь, что мы конструируем алгоритм, возвращающий числа от 0 до 1. Таким образом, необходимо выбрать такой алгоритм, которому будет оптимально возвращать $p(y = +1|x)$ в качестве ответа для x. Логарифм функции потерь обладает как раз таким свойством.

Максимизация функционала правдоподобия аналогична максимизации его логарифма и минимизации его логарифма, взятого с противоположным знаком, который выглядит следующим образом (и в такой форме носит название log-loss):

$$-\sum_{i=1}^{l}\left([y_i=+1]\log b(x_i)+[y_i=-1]\log(1-b(x_i))\right)$$

Теперь можно подставить $b(x_i)=\dfrac{1}{1+\exp(-\langle w,x_i\rangle)}$ и получить, что функционал выглядит следующим образом:

$$-\sum_{i=1}^{l}\left([y_i=+1]\log\frac{1}{1+\exp(-\langle w,x_i\rangle)}+[y_i=-1]\log\left(1-\frac{1}{1+\exp(-\langle w,x_i\rangle)}\right)\right)=$$
$$-\sum_{i=1}^{l}\left([y_i=+1]\log\frac{1}{1+\exp(-\langle w,x_i\rangle)}+[y_i=-1]\log\left(\frac{\exp(-\langle w,x_i\rangle)}{1+\exp(-\langle w,x_i\rangle)}\right)\right)=$$
$$-\sum_{i=1}^{l}\left([y_i=+1]\log\frac{1}{1+\exp(-\langle w,x_i\rangle)}+[y_i=-1]\log\left(\frac{1}{1+\exp(\langle w,x_i\rangle)}\right)\right)$$

Теперь можно заметить, что под экспонентой перед скалярным произведением стоит минус, если объект принадлежит классу $+1$ и плюс, если он принадлежит классу -1, тогда функционал можно записать так:

$$-\sum_{i=1}^{l}\left([y_i=+1]\log\frac{1}{1+\exp(-y_i\langle w,x_i\rangle)}+[y_i=-1]\log\left(\frac{1}{1+\exp(-y_i\langle w,x_i\rangle)}\right)\right)=$$
$$-\sum_{i=1}^{l}\left([y_i=+1]+[y_i=-1]\right)\log\frac{1}{1+\exp(-y_i\langle w,x_i\rangle)}$$

Сумма $[y_i=+1]+[y_i=-1]=1$, поскольку ровно одно из слагаемых равно 1, а другое 0. Вспоминая свойство логарифма $\log\frac{1}{x}=-\log x$, получаем, что функционал стал выглядеть намного проще:

$$\sum_{i=1}^{l}\log(1+\exp(-y_i\langle w,x_i\rangle)))=\sum_{i=1}^{l}\log(1+\exp(-M_i))$$

22 Как в логистической регрессии строится прогноз для нового объекта?

Рассчитывается значение $\sigma(x)=\dfrac{1}{1+\exp(-\langle w,x\rangle)}$ и сравнивается с некоторым заданным порогом t. Если значение больше порога, объект относится к классу $+1$, если меньше – к классу -1.

23 Как можно использовать категориальные признаки в линейных моделях?

Можно воспользоваться one-hot-кодированием: нужно завести столько признаков, сколько возможных значений у данного категориального признака (каждый из признаков будет

соответствовать одному из возможных значений и равен 1, если это истинное значение объекта, и 0 иначе).

24 В чём заключается использование квадратичных признаков в линейных моделях? Для чего это нужно?

Квадратичные признаки в линейных моделях добавляют для того, чтобы модель могла восстанавливать более сложные зависимости. Для нужного признака необходимо завести ещё один признак, все значения которого суть квадраты соответствующих значений исходного признака.

25 Что такое доля правильных ответов? В чём заключаются её проблемы?

Доля правильных ответов $\frac{1}{l}\sum_{i=1}^{l}[a(x_i) = y_i]$ – это простое отношение числа объектов, для которых алгоритм предсказал верный класс, к числу всех объектов. Её проблема заключается в том, что по ней сложно судить о качестве модели. Например, допустим, что в тренировочной выборке 95% объектов принадлежат классу -1, а оставшиеся 5% – классу $+1$. Рассмотрим «глупый» классификатор, который заявляет, что любой объект принадлежит классу -1. Доля правильных ответов составит 0.95, что близко к 1 и кажется достаточно неплохим показателем (однако такая модель совершенно бесполезна на деле).

26 Что такое точность и полнота? Что такое F-мера?

Рассмотрим следующую матрицу (матрицу ошибок).

	y = 1	y = -1
a(x) = 1	True Positive (TP)	False Positive (FP)
a(x) = -1	False Negative (FN)	True Negative (TN)

Например, FP – это количество объектов, принадлежащих отрицательному (-1) классу, но ошибочно (False) отнесёнными классификатором к положительному (Positive).

Точность (precision) – это доля верных предсказаний среди всех предсказаний принадлежности объекта к положительному классу. Она позволяет ответить на вопрос, можно ли доверять классификатору при $a(x) = 1$:

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Полнота (recall) – это показатель того, сколько положительных объектов находит классификатор, то есть доля верных предсказаний для положительных объектов:

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

F-мера – это среднее гармоническое между *precision* и *recall* для данного алгоритма:

$$\text{F} = \frac{2}{\frac{1}{\text{precision}} + \frac{1}{\text{recall}}} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

27 В чём заключается разница между метриками Accuracy и Precision?

Precision характеризует, сколько полученных от классификатора положительных ответов являются правильными. Accuracy характеризует, сколько полученных от классификаторов ответов являются правильными.

28 Что такое ROC-кривая? Что такое AUC-ROC? Для чего он используется?

ROC-кривая – это кривая, одна из координат которой соответствует доле неверно принятых объектов (False Positive Rate, FPR), а другая доле верно принятых объектов (True Positive Rate, TPR) для различных порогов выбора:

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}, \text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}}.$$

Площадь под данной кривой называется AUC-ROC и принимает значения от 0 (совсем плохо) до 1 (идеально). Если алгоритм выдаёт случайные ответы, то AUC-ROC будет близок к 0.5.

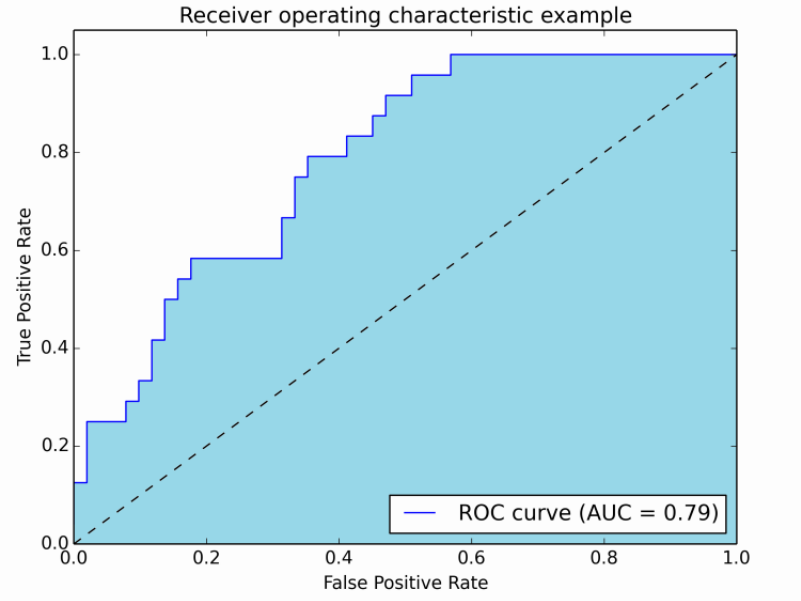

Рис. 2: Пример ROC-кривой и площади под ней.

AUC-ROC позволяет оценить качество работы алгоритма не на конкретном пороге выбора (который, вообще говоря, может быть установлен позже), а в целом.

29 Что такое PR-кривая? Что такое AUC-PRC? Для чего он используется?

PR-кривая аналогична ROC-кривой, только на осях координат откладываются precision и recall соответственно. *AUC-PRC* – это площадь под данной кривой. Этот показатель

также позволяет оценивать качество алгоритмов, причём он ведёт себя более адекватно в случае несбалансированности классов в обучающей выборке.

30 Как можно свести задачу многоклассовой классификации к серии задач бинарной классификации?

Пусть необходимо определять принадлежность объекта одному из k классов.
Существует несколько подходов, вот два из них:

- **Один против всех**. Необходимо обучить k линейных классификаторов, каждый из которых оценивает вероятность принадлежности объекта одному из классов. При классификации нового объекта мы отправляем его в тот класс, вероятность принадлежности (ответ соответствующего классификатора) к которому наибольшая. Настраивать каждый классификатор нужно по всей тренировочной выборке с той лишь разницей, что для классификатора, соответствующему классу i, ответ будет равен 1, если объект принадлежит классу i, и -1 иначе.
- **Все против всех**. Необходимо обучить $C_k^2 = \frac{k(k-1)}{2}$ (столько, сколько есть различных пар классов) линейных классификаторов. Каждый из таких классификаторов обучим на выборке, содержащей только объекты из двух классов j и i (для каждого классификатора своя пара значений i, j) – соответственно, он будет выдавать лишь класс i или j. Теперь, когда нужно классифицировать новый объект, его нужно пропустить через все классификаторы и выбрать тот класс, за который подано больше всего «голосов».

31 Что такое гиперпараметр? Чем гиперпараметры отличаются от обычных параметров алгоритмов? Приведите примеры параметров и гиперпараметров в линейных моделях.

Гиперпараметры – это параметры модели, которые определяются не из обучающей выборки, а из других соображений.

Например, в линейной регрессии обычные параметры (те, которые определяются путём оптимизации функционала ошибки) – это вектор весов w, а гиперпараметр – коэффициент λ у регуляризатора. Мы хотим штрафовать алгоритм за большие веса, но если бы параметр λ определялся в процессе обучения, методы оптимизации всегда утверждали бы, что оптимум достигается при $\lambda = 0$.

32 Что такое решающее дерево? Как оно строит прогноз для объекта? Как обучаются решающие деревья в задачах классификации и регрессии (критерии информативности)?

Решающее дерево – это бинарное (то есть из каждого узла, который не является листом, выходит ровно два ребра) дерево, в каждой нелистовой вершине которого записан неко-

торое условие (вопрос), а в листах – конечное решение алгоритма.

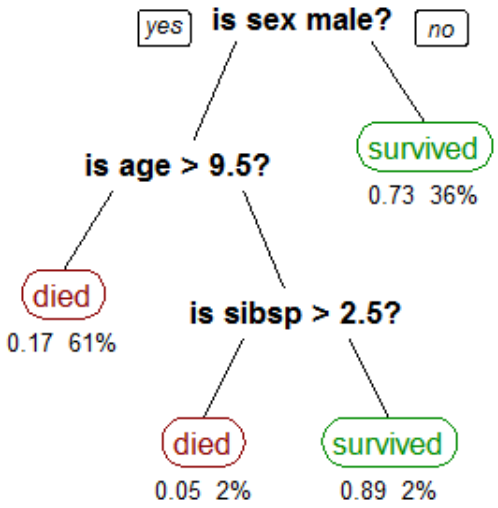

Рис. 3: Картинка с Википедии, по которой становится понятно, что такое решающее дерево.

Прогноз для объекта строится простым проходом по этому дереву до листа путём проверки условий в узлах.

Обучение решающих деревьев. Будем жадно растить дерево от корня до листьев. Допустим, мы находимся в некоторой вершине m с выборкой X_m и хотим записать в вершину некоторое условие $x^j \leq t$ (j – номер некоторого признака, t – числовой параметр), которое разделит выборку по двум потомкам данной вершины. Пусть есть некоторый критерий ошибки такого критерия на данной выборке – обозначим его $Q(X_m, j, t)$. Таким образом, для того, чтобы выбрать j, t, нужно минимизировать Q (можно сделать это простым перебором всех возможных значений j и t). После того, как мы выбрали признак j и порог t, можно разбить выборку на две части – для которой $x^j \leq t$ и для которой $x^j > t$. Первая пойдет в одно поддерево, вторая – в другое.

Если в текущей вершине выполняется некоторый критерий останова, она объявляется листом, нам необходимо указать класс для объекта или выдать предсказанное значение целевой переменной.

- Если мы обучаем дерево, решающее задачу классификации, мы рассматриваем все элементы обучающей выборки, попавшие в данный лист и выдаем класс, к которому принадлежит больше всего таких объектов:

$$a_m = \arg\max_{y \in \mathbb{Y}} \sum_{i \in X_m} [y_i = y]$$

- Если мы обучаем дерево, решающее задачу регрессии, мы можем вернуть средний

ответ для таких объектов:

$$a_m = \frac{1}{|X_m|} \sum_{i \in X_m} y_i$$

В качестве критерия ошибки $Q(X_m, j, t)$ можно взять следующий функционал:

$$Q(X_m, j, t) = H(X_m) - \frac{|X_l|}{|X_m|} H(X_l) - \frac{|X_r|}{|X_m|} H(X_r)$$

Здесь X_l, X_r – подвыборки, отправляющиеся в левое и правое поддерево соответственно, а функция H (критерий информативности) описывает разброс ответов (чем он меньше, тем меньше должно быть её значение).

Для задачи регрессии можно взять дисперсию:

$$H(X) = \frac{1}{|X|} \sum_{i \in X} \left(y_i - \overline{y}\left(X\right)\right)^2$$

где $\overline{y}(X)$ – среднее значение целевой переменной на подвыборке:

$$\overline{y}(X) = \frac{1}{|X|} \sum_{i \in X} y_i$$

Для задачи классификации можно взять энтропию (меру неопределённости распределения):

$$H(X) = -\sum_{i=1}^{n} p_i \log p_i$$

(здесь $p_1, ..., p_n$ – вероятности различных ответов (то есть просто доли объектов с таким ответом в выборке X), а $0 \log 0 = 0$)

33 Какие вы знаете критерии останова и способы выбора значений в листьях? Какие гиперпараметры имеются у деревьев?

Некоторые возможные критерии останова в вершинах:

- В вершине остались объекты одного класса
- Глубина дерева превысила заданный порог
- Число оставшихся в вершине объектов меньше некоторого заданного порога
- Ограничение максимального количества листьев в дереве
- Требование, что функционал качества при дроблении улучшался как минимум на s процентов.

Гиперпараметры деревьев зависят от используемых критериев останова. Например, если мы используем второй критерий из списка выше, то гиперпараметром будет, в частности, максимальная глубина дерева.

О способах выбора значений в листьях написано в ответе на предыдущий вопрос.

34 Что такое бэггинг и метод случайных подмножеств? Что такое случайный лес, как он обучается и как он строит прогнозы?

Предположим, у нас есть несколько базовых алгоритмов $b_1(x), b_2(x), ..., b_N(x)$ (которые решают задачу по-разному, но при этом лучше случайного угадывания) и мы хотим сконструировать один алгоритм (композицию) на их основе.

Если это задача классификации, можно использовать алгоритм, который возвращает тот класс, за который «проголосовало» больше всех базовых алгоритмов:

$$a(x) = \arg\max_{y \in \mathbb{Y}} \sum_{n=1}^{N} [b_n(x) = y]$$

Если это задача регрессии, то можно возвращать среднее по всем базовым алгоритмам:

$$a(x) = \frac{1}{N} \sum_{n=1}^{N} b_n(x)$$

Вопрос лишь в том, как по одной и той же обучающей выборке создать N базовых алгоритмов.

- В *бэггинге* обучение каждого базового алгоритма происходит по случайно генерируемым подвыборкам тренировочной выборки.
- В *методе случайных подмножеств* обучение каждого базового алгоритма происходит по случайно генерируемому подмножеству признаков.

Размер каждой случайной подвыборки или подмножества признаков является гиперпараметром соответствующей модели.

Также можно добавить элемент случайности к самому алгоритму построения дерева. В ходе работы жадного алгоритма построения решающего дерева мы перебирали все возможные пары признаков и порогов (j, t) и искали те значения этих параметров, для которых при разбиении выборки на две части по условию $x^j < t$ получался наилучший результат с точки зрения некоторого критерия ошибки $Q(X_m, j, t)$. Можно выбирать признак j не из всего множества признаков, а из некоторого его случайного подмножества. Это приводит к алгоритму построения того, что называется *решающим лесом*.

for $n \leftarrow 1..N$ **do**
$\widetilde{X} \leftarrow$ выборка, генерируемая бутстрапом
$b_n(x) \leftarrow$ решающее дерево по выборке $\widetilde{X}$ (оно строится, пока в каждом листе окажется не более некоторого числа n_{min} объектов, а оптимальное разбиение на каждом шагу ищется среди q случайных признаков)
end for

Решающее дерево строит прогноз по принципу аггрегации базовых алгоритмов, о котором написано в начале (большинство голосов в случае классификации или среднее по базовым алгоритмам в случае регрессии).