Машинное обучение

DS-поток

Лекция 9

Работа с признаками

Работа с признаками

Числовые признаки

Дата/время и координаты

Порядковые и категориальные признаки

Масштабирование

Отбор признаков

Ранг

Идея: заменить значения $X_1, ..., X_n$ на ранги $R_1, ..., R_n$,
где ранг R_i — порядковый номер X_i в упорядоченном наборе.

X_i	7.3	2.2	0.3	6.2	1.6	6.2	9.6
R_i	6	3	1	4.5	2	4.5	7

Для нового объекта x

- ранг ближайшего объекта из train.
- средний ранг по ближайшим объектам из выборки.
- взвешенный средний ранг по ближайшим объектам из выборки.

Смысл:

Подвигаем выборсы к остальным объектам,
$\Rightarrow$ они перестают вносить большой вклад в модель.

Применение:

Иногда хорошо работает для KNN, лин. моделей, нейросетей,
особенно есть нет времени разбираться с выбросами.

Трансформации

- Логарифмическая

  $\widetilde{x} = \ln(x)$

- Возведение в степень

  $\widetilde{x} = \sqrt{x+1}$

- Пребразование Бокса-Кокса

$$\widetilde{x} = \begin{cases} \dfrac{x^\lambda - 1}{\lambda}, & \lambda \neq 0 \\ \ln(x), & \lambda = 0 \end{cases}$$

Смысл: Объекты с бОльшими значениями признаков становятся ближе к остальным объектам.
Особенно хорошо работают для нейростей.

Замечание: при отрицательных значениях нужно произвести сдвиг.

Применение логарифмирования

Мультипликативные признаки

После логарифмирования такой признак станет аддитивным.

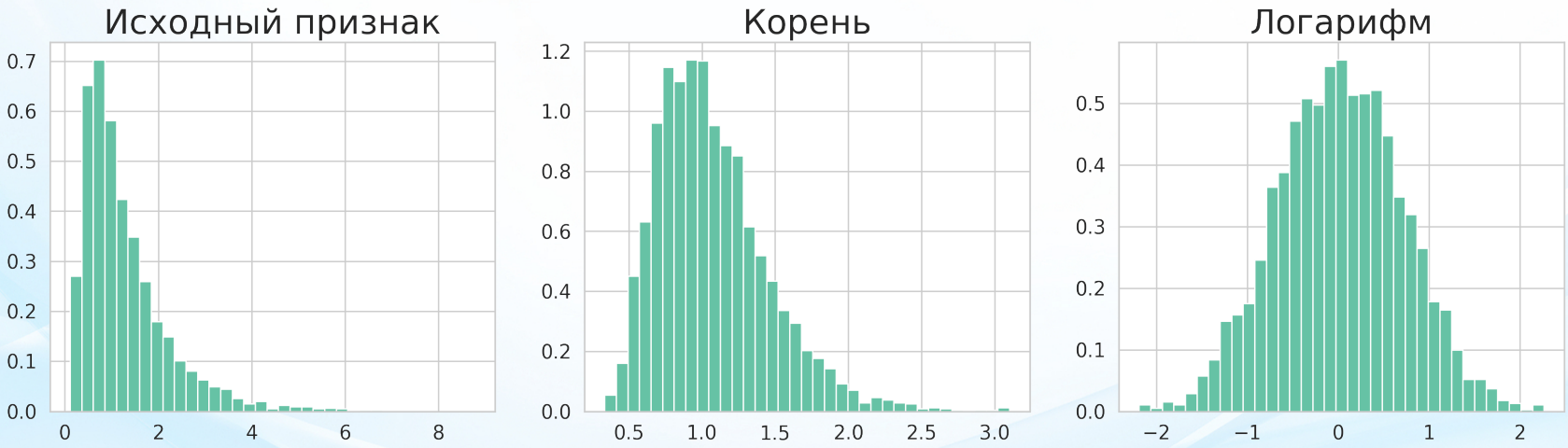
Рассмотрим лин. регрессию с одним признаком: $\widehat{y}(x) = \theta x + \theta_0$

- Пусть x показывает *во сколько раз* увеличились цены.
  $\Rightarrow x$ — *мультипликативный признак*.
  Тогда $\widehat{y_1} = \theta x_1 + \theta_0 \qquad \widehat{y_2} = \theta x_2 + \theta_0$
  $$\Rightarrow \widehat{y_1} - \widehat{y_2} = \theta(x_1 - x_2)$$
  Но рассматривать $x_1 - x_2$ нелогично, лучше x_1/x_2.
- Пусть x показывает *на сколько* увеличились цены.
  $\Rightarrow x$ — *аддитивный признак*.
  Здесь рассматривать $x_1 - x_2$ вполне логично.

Применение логарифмирования

Искаженное распределение

Преобразует искаженное распределение ближе к нормальному.



Генерация признаков

Рассмотрим значения признака

Цена	0.99	2.49	1	9.99

Разделим на целую и вещественную часть.

Целая часть	0	2	1	9
Дробная часть	0.99	0.49	0	0.99

Такие преобразования очень часто делают для цены,

так как это отражает восприятние цены человеком.

Генерация признаков

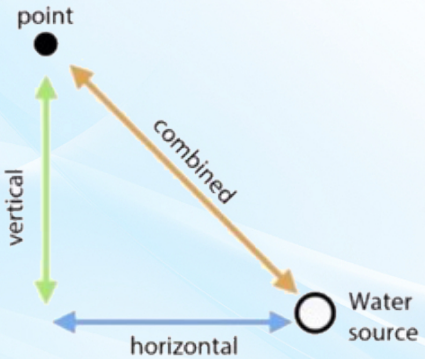
Имеем:

Квадратная площадь: $55m^2$

Цена: 107000 $

Получаем:

Цена за 1 m^2: 10700 $ / 55 m^2



Имеем:

Расстояние по вертикали: 3 м

Расстояние по горизонтали: 2 м

Получаем:

Полное расстояние: 3.60 м

Такие признаки помогут многим моделям,
т.к. модели плохо умеют умножать/делить фичи друг на друга.

Квантование (Binning)

Разбиение множества значений признака на интервалы (бины) и замена признака на категориальную переменную.

- **Fixed-Width Binning**

  Выбор точек разбиения вручную или равномерно

- **Adaptive Binning**

  Выбор точек разбиения в зависимости от выборки.

Работа с признаками

Числовые признаки

Дата/время и координаты

Порядковые и категориальные признаки

Масштабирование

Отбор признаков

Дата и время

Дата/timestamp в чистом виде не достаточно информативны.

Полезно добавить дополнительные признаки.

- **Характеристики даты**
  Год, сезон, день месяца, день недели, час, минуты, секунды, праздничный ли день и какой праздник.
- **Время с момента некоторого события**
  Кол-во дней с прошлых выходных.
  Кол-во дней с последней покупки.
  Кол-во дней от фиксированной даты X.
- **Синус от времени с периодом кратным длине сезона/дня.**

Замечание:
Такие признаки как день недели являются категориальными!

Координаты

Местоположение обычно характеризуется широтой и долготой.
В чистом виде они недостаточно информативны.
Рассмотрим разные случаи.

- **Есть дополнительные данные**
  Можно добавить расстояния до ближайшего магазина, больницы, школы и прочего.

Координаты

- **Дополнительных данных нет**

  Придумаем местоположения сами по имеющимся данным.

  Пример для анализа цен на квартиры.

  - Разделим карту на квадраты.

    В каждом квадрате найдем самую дорогую квартиру.

    Для объектов в квадрате добавим расстояние до этой квартиры.

  - Организуем имеющиеся точки в кластеры.

    Найдем центры кластеров.

    Будем использовать их как важные местоположения.

  - Найдем район с старыми строениями.

    Посчитаем расстояние до него.

Координаты

- **Агрегирующие статистики**

  Посчитаем статистики по объектам, находящимся рядом.

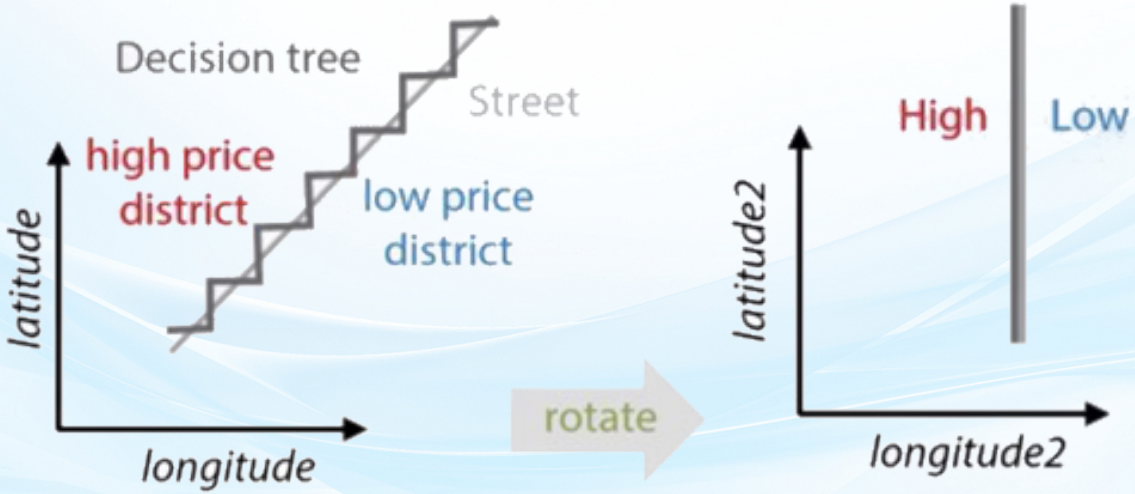
  Примеры: кол-во квартир, средняя цена квартиры.

- **Поворот координат**

  Повернем координаты и возьмем это как новые признаки.

  Можно сделать несколько разных поворотов.

  Полезно для tree-based методов.



Работа с признаками

Числовые признаки

Дата/время и координаты

Порядковые и категориальные признаки

Масштабирование

Отбор признаков

Порядковые признаки

Для таких признаков значения сравнимы между собой,

но *расстояния между ними не определены*.

Примеры:

- Класс билета
  A, B, C
- Образование
  школа, бакалавриат, магистратура, аспирантура
- Оценка
  1, 2, 3, ..., 9, 10

Обработка: **Label encoding**

Каждому состоянию сопоставляем число.

Порядок состояний должен сохраниться.

Категориальные признаки

Значения категориальных признаков не сравнимы друг с другом.

Категория — одно значение данного признака.

Примеры:

- Город

  Москва, Краснодар, Калининград, Якутск, Анадырь

- Пол

  мужской, женский

Как работать с категориальными признаками?

- **Label encoding**

  Каждому состоянию сопоставляем число.

  sklearn.preprocessing.LabelEncoder — алфавитный порядок.

  pandas.factorize — в порядке встречаемости значения.

  Минусы:
  Линейные модели плохо работают с такими признаками.
  Деревья могут работать, но потребуется глубокое дерево.

- **Count encoding** и **Frequency encoding**

  Заменяем категорию на ее кол-во/частоту в обучении.

  Замечание:
  Если частоты у категорий похожи, то они будут неразличимы.
  Поэтому можно использовать ранги частот.

Как работать с категориальными признаками?

- **One-hot encoding**

  Создается $K - 1$ новых бинарных признаков,

  где K — кол-во категорий.

  Замечание 1:
  Если есть пара вещественных, очень значимых признаков,
  то деревьям и KNN будет трудно обращать на них внимание
  из-за большого кол-ва новых one-hot признаков.

  Замечание 2:
  Если у кат. фичи много уникальных значений,
  то добавим много новых признаков, в которых,
  возможно, только пара ненулевых элементов.
  Тогда обычно хранят только ненулевые элементы — **sparse matrix**.

Как работать с категориальными признаками?

- **Binary encoding**

  Применяется Label encoding.

  Полученные номера переводятся в *двоичную систему* счисления и двоичные числа разбиваются на столбцы.

  Минусы: Полученные признаки могут коррелировать.

- **Mean encoding** (Target encoding)

  Заменяем категорию на *ср. значение* или другую статистику таргета у объектов, имеющих данную категорию.

  Является очень мощным методом работы с кат. признаками.

  Разберем подробнее.

Mean Encoding: какие статистики выбирать?

Бинарная классификация

- Частота класса 1
- Логиты
- Кол-во объектов класса 1
- Разница кол-ва объектов между классами

Многоклассовая классификация

Для кат. признака введем K признаков, где K — число классов.

k-ый признак строится по таргету вида $I\{Y_i = k\}$.

Регрессия

- Среднее
- Дисперсия
- Квантили
- Максимум
- Распределение по бинам

Mean Encoding: пример

Label encoding

id	job	job_label	target
1	Doctor	1	1
2	Doctor	1	0
3	Doctor	1	1
4	Doctor	1	0
5	Teacher	2	1
6	Teacher	2	1
7	Engineer	3	0
8	Engineer	3	1
9	Waiter	4	1
10	Driver	5	0

Mean encoding

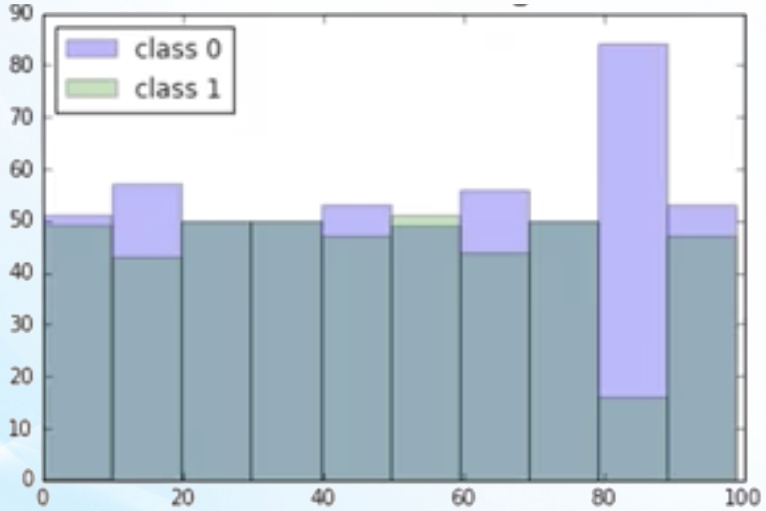
id	job	job_mean	target
1	Doctor	0,50	1
2	Doctor	0,50	0
3	Doctor	0,50	1
4	Doctor	0,50	0
5	Teacher	1	1
6	Teacher	1	1
7	Engineer	0,50	0
8	Engineer	0,50	1
9	Waiter	1	1
10	Driver	0	0

- `job` — категориальный признак
- `target` — целевой признак
- `job_label` — преобразование с помощью Label encoding
- `job_mean` — преобразование с помощью Mean encoding

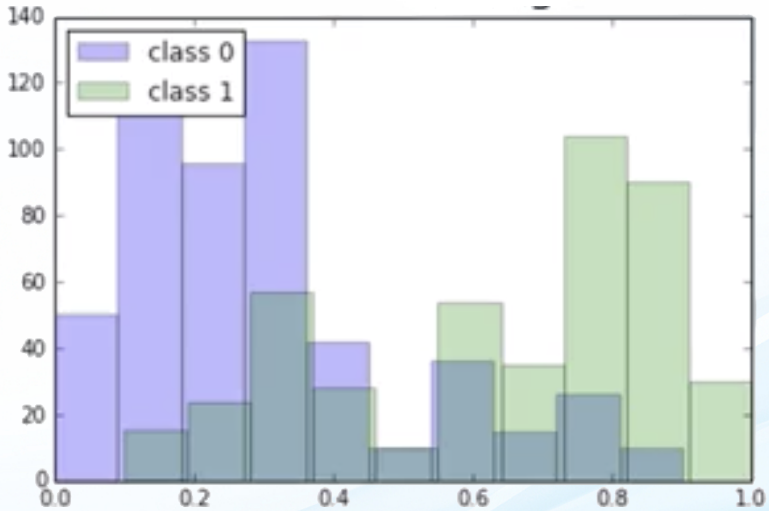
Mean Encoding: почему он работает

Рассмотрим случай бинарной классификации.

Label encoding



Mean encoding



Посмотрим на гистограммы признака для класса 0 и класса 1.

В случае Label encoding получаем равномерное распределение.

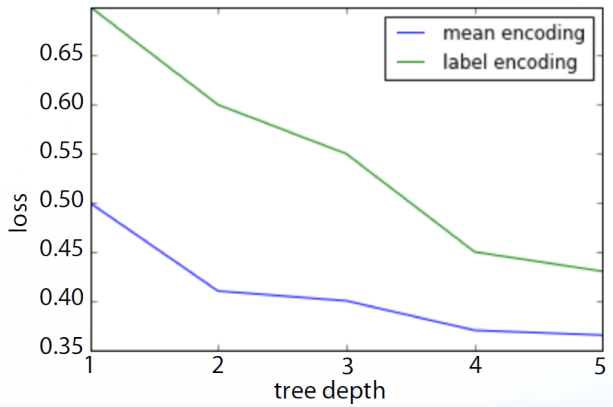
В случае Mean encoding классы выглядят более разделимыми.

Mean Encoding

Модели на основе решающих деревьев

Деревьям трудно работать с кат. признаками с большим кол-вом уникальных значений: нужна большая глубина.

Mean encoding решает эту проблему: меньшая ошибка при меньшей глубине.



Проблемы

- Статистики по обучающей выборке, не всегда верны для теста.

  Например, если количество объектов в категории мало, то оценка статистики будет очень шумной.

- При подсчете статистики используем таргет

  $\Rightarrow$ при обучении на объекте x_i у модели есть информация о Y_i.

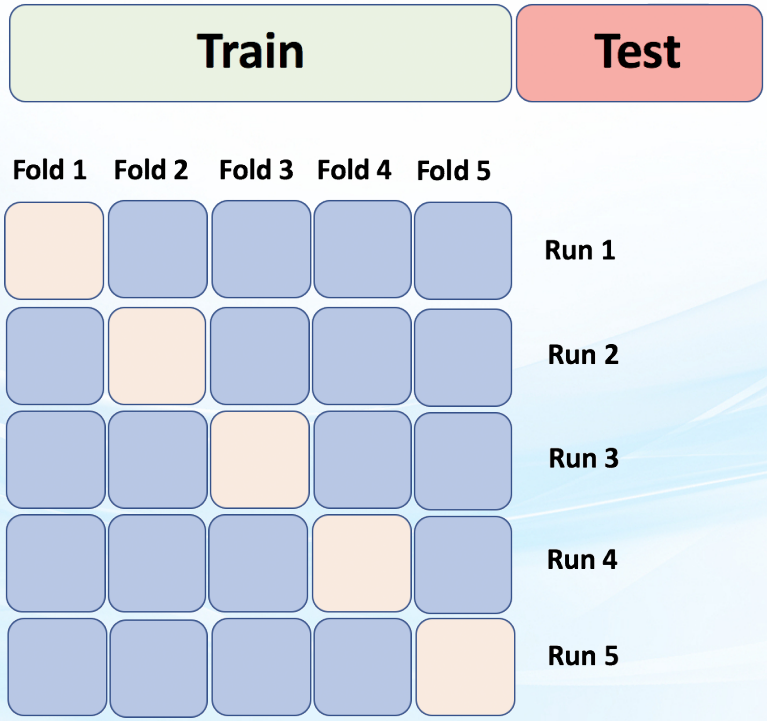
Эти проблемы вызывают **переобучение** модели.

Регуляризация

I. CV loop

1. Разбиваем данные на k фолдов.
2. Для получения статистики для k-ого фолда
   используем таргеты всех фолдов, кроме k-ого.
   Для подсчета статистики для теста используется весь train.

Вывод: для объекта x_i не используем его таргет y_i в статистике.



Регуляризация

В некоторых случаях все равно может произойти переобучение.

Пример.

- Рассмотрим категорию.
- Применим Leave-One-Out
  случай $k = n$
- По новому признаку можно однозначно восстановить таргет.

	feature	feature_mean	target
0	Moscow	0.50	0
1	Moscow	0.25	1
2	Moscow	0.25	1
3	Moscow	0.50	0
4	Moscow	0.50	0

Однако для теста значение статистики для всех объектов этой категории $= 0.4$, т.е. восстановить ответ уже нельзя.

Вывод: произошла утечка таргета.

Регуляризация

II. Сглаживание

Заменим значение кат. признака на

$$S_c = \frac{n_c \overline{y}_c + \alpha \overline{y}}{n_c + \alpha},$$

где $\overline{y}_c$ — среднее значение таргета в категории c в обучении.
n_c — количество объектов категории c в обучающей выборке.

Свойства:
$\alpha = 0 \Rightarrow$ нет регуляризации.
$\alpha \to \infty \Rightarrow$ статистика стремится к глобальному среднему.

III. Expanding mean

Зафиксируем некоторый порядок объектов в трейне.

Для подсчета статистики для x_i используем только $y_1, \ldots, y_{i-1}$.

Плюсы:

- Самое маленькая утечка таргета среди всех методов.
- Нет гиперпараметров.

Используется в CatBoost для обработки кат. признаков.

Работа с признаками

Числовые признаки

Дата/время и координаты

Порядковые и категориальные признаки

Масштабирование

Отбор признаков

Масштабирование

Масштабирование — приведение признаков к единому масштабу.

Важно для

- линейных моделей с регуляризацией
- метрических моделей
- градиентного спуска (как следствие для нейросетей)

Не важно для решающих деревьев.

Почему?

- Регуляризация имеет тенденцию штрафовать параметры при признаках меньшего масштаба.
- Начинают учитываться только крупномасштабные признаки.

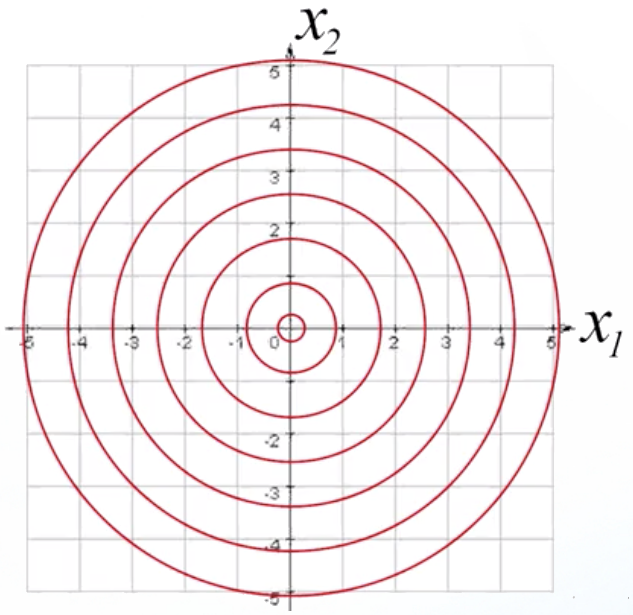
Масштабирование

Пример 1

$$\mathcal{L}(x_1, x_2) = x_1^2 + x_2^2 \to \min_{x_1, x_2}$$

$$-\frac{\partial \mathcal{L}}{\partial x}(1, 1) = (-2, -2)$$

Вектор антиградиента проходит через точку минимума.



Пример 2

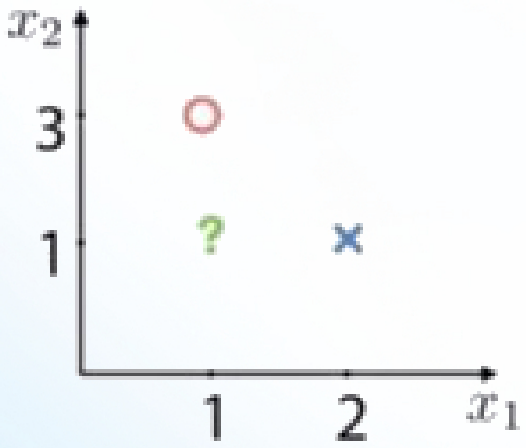
$$\mathcal{L}(x_1, x_2) = x_1^2 + 100x_2^2 \to \min_{x_1, x_2}$$

$$-\frac{\partial \mathcal{L}}{\partial x}(1, 1) = (-2, -2)$$

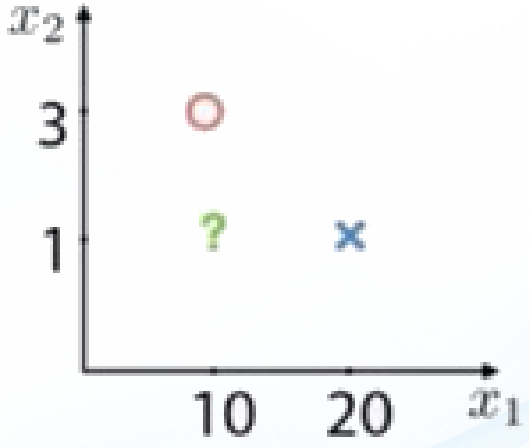
Вектор антиградиента направлен практически вниз и проходит мимо точки минимума.

Масштабирование

Как классифицируется "**?**" методом ближайшего соседа?



Как "**x**"



Как "**o**"

Однако поменялся только масштаб!

Масштабирование

- **Стандартизация**

$$\mu_j = \frac{1}{n}\sum_{i=1}^{n} x_{ij} \qquad \sigma_j = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \mu_j)^2} \qquad \widetilde{x}_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j}$$

Чаще применяется для линейных моделей и нейросетей

- **Min-max нормализация**

Масштабируем на отрезок $[0, 1]$:

$$m_j = \min(x_{1j}, ..., x_{nj}) \qquad M_j = \max(x_{1j}, ..., x_{nj})$$

$$\widetilde{x}_{ij} = \frac{x_{ij} - m_j}{M_j - m_j}$$

Чаще применяется для метрических моделей

Масштабирование

- **Нормализация средним**

$$m_j = \min(x_{1j}, ..., x_{nj})$$

$$M_j = \max(x_{1j}, ..., x_{nj})$$

$$\mu_j = \frac{1}{n} \sum_{i=1}^{n} x_{ij}$$

$$\widetilde{x}_{ij} = \frac{x_{ij} - \mu_j}{M_j - m_j}$$

- **Нормализация**

$$\widetilde{x}_i = \frac{x_i}{\|x_i\|^2}$$

Замечание:

Масштабируем и тест, и трейн одинаково.

Все статистики подбираем по трейну!

Работа с признаками

Числовые признаки

Дата/время и координаты

Порядковые и категориальные признаки

Масштабирование

Отбор признаков

Зачем отбирать признаки?

Ситуация 1

- Добавим к нашим признакам 100 шумовых признаков
- С большой вероятностью хотя бы один немного коррелирует с таргетом на обуч. выборке.
- Модель может решить, что он важный, и использовать его.
- На других данных такой корреляции уже не будет. Качество будет страдать.

Ситуация 2

- Имеется 1000 признаков.
- Обучаем решающее дерево.
- Чтобы учесть каждый признак хотя бы по одному разу, нужно дерево глубины ≥ 10. У такого дерева более 1000 листьев.
- В каждый лист должно попасть достаточное число объектов, иначе — риск переобучения.

Замечание.
Если хочется учесть каждый признак больше, чем один раз, то дерево должно быть еще глубже.

Зачем отбирать признаки?

Ускорение модели

- Чем больше признаков, тем сложнее модель.
- Чем сложнее модель, тем дольше она вычисляет прогнозы и обучается.
- В некоторых задачах могут быть жесткие ограничения на скорость работы и обучения.
  Например, в онлайн-моделях.

Одномерный отбор признаков

Принцип

- Измеряем связь (информативность) каждого признака с целевой переменной отдельно.
- Отбираем лучшие по информативности.

Как оценить информативность?

1. Важность признаков
2. Корреляции
3. Качество моделей, обученных по каждому из признаков отдельно

Важность признаков в общем случае

Permutation feature importance

1. Обучим модель и измерим метрику на валидации.
2. Для одного выбранного признака перемешаем все его значения в датасете, на котором до этого измерили метрику.
3. Измерим метрику на видоизмененном датасете.
4. Определим важность данного признака как *разницу между исходным и новым значением метрики*.
5. Сделаем пункты 2-4 для всех признаков.

Плюсы:

- Подходит для любых моделей.
- Требует одного обучения модели.
- Использует тестовое множество и является более надежным, чем MDI для деревьев.

Минусы:

- Более вычислительно затратно, чем MDI для деревьев.
- Переоценивает важность для скоррелированных признаков. (Strobl et al (2008))

Важность признаков в общем случае

Drop Column feature importance

Сравним 2 модели:

- Модель, обученная на датасете со всеми признаками
- Модель, обученная на данных без одного признака.

Важность этого признака — разница метрик
на тесте/валидации для этих моделей.

Плюсы:

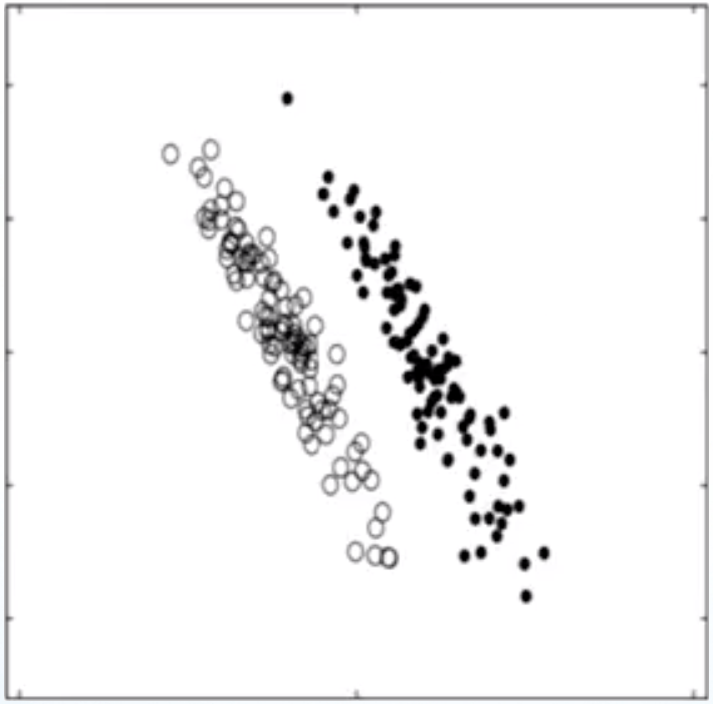
- Самая точная важность признаков.
- Подходит для любых моделей.

Минусы:

- Вычислительно сложно.
  Требует обучения большого количества разных моделей.

Одномерный отбор признаков

Проблема: сложные закономерности



По двум признакам можно идеально разделить классы.

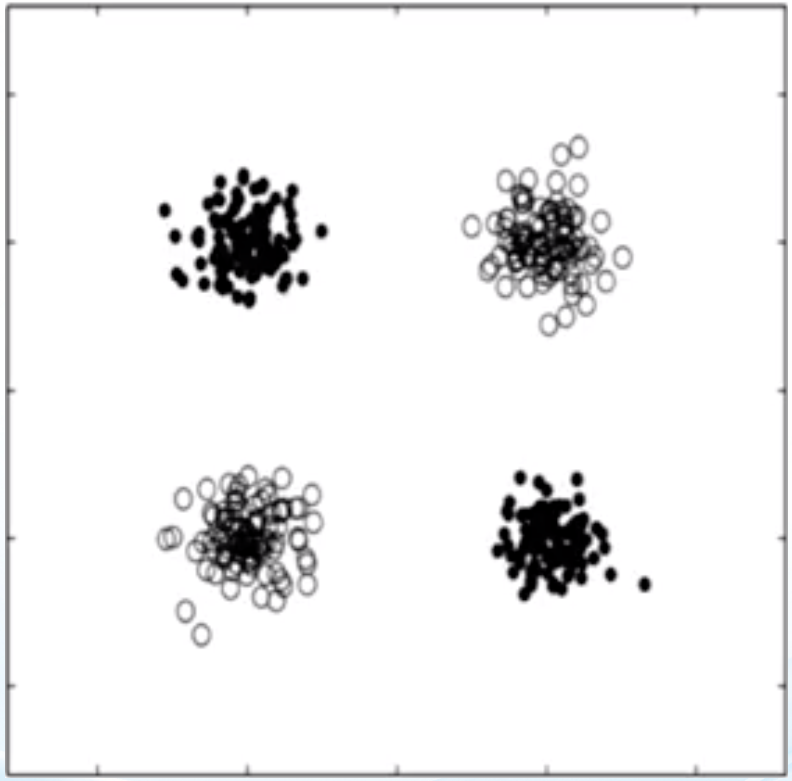
По x_1 данные можно как-то разделить $\Rightarrow$ x_1 — информативный.

По x_2 данные нельзя разделить $\Rightarrow$ x_2 будет неинформативным.

$\Rightarrow$ Останется только признак x_1, сильно теряем в качестве.

Одномерный отбор признаков

Проблема: сложные закономерности



По двум признакам можно идеально разделить классы.

При одномерном отборе и x_1, и x_2 будут неинформативными.

Перебор признаков

Принцип

- Каким-то методом перебираем комбинации признаков
- Для каждой комбинации обучаем модель
- Выбираем комбинацию, дающую лучшую модель

Полный перебор

Пробуем все подмножества признаков и выбираем лучшее.

Свойства:

- Находит точное решение.
- Перебирает 2^d вариантов.
  $\Rightarrow$ Подходит только для малого числа признаков.

Жадное добавление

Пусть F_t — множество информативных признаков на итерации t.

Принцип:

1. Сначала F_0 — пустое
2. Находим признак x_j, при добавлении которого к F_{t-1} получим наименьшую ошибку модели
3. Повторяем до тех пор, пока ошибка уменьшается

Плюсы:

- Работает достаточно быстро — требует d итераций.
  На каждой итерации t происходит обучение $(d - t)$ моделей.
  $\Rightarrow$ обучается всего $\frac{d(d-1)}{2}$ моделей.

Минусы:

- Слишком жадно.
  После добавления признака в F_t он там навсегда останется.
  Нет возможности убрать признак после добавления.

Add-Del

- Жадное добавление.

  Добавляем по одному признаку пока ошибка уменьшается.

- Жадное удаление.

  Удаляем по одному признаку пока ошибка уменьшается.

- Повторяем стадии добавления и удаления,

  пока ошибка уменьшается.

Может исправлять ошибки, сделанные в процессе перебора ранее.

Отбор на основе моделей

Линейные модели

$$y(x) = \sum_{j=1}^{d} \theta_j x_j$$

- Если признаки отмасштабированы:
  Веса можно использовать как показатели информативности.
  Чем больше $|\theta_j|$, тем больший вклад вносит признак x_j.

- Если признаки не отмасштабированы:
  Веса нельзя использовать как показатели информативности.

Для повышения числа нулевых весов — L_1-регуляризация


ВСЁ!