



Линейные регрессионные модели. Часть 2.

ОЦЕНКА ПРЕДСКАЗАТЕЛЬНОЙ СПОСОБНОСТИ АЛГОРИТМА

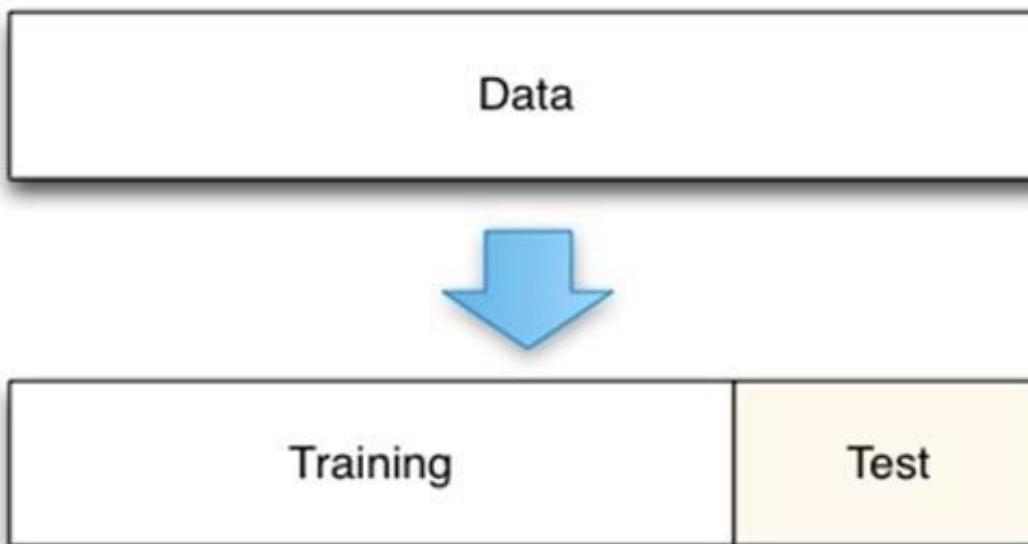
- Пусть мы решаем задачу *предсказания стоимости дома по его признакам.*



- В обучающей выборке 1000 домов.
- Мы обучаем алгоритм по имеющимся 1000 домам. На каких объектах будем проверять качество алгоритма?

ОЦЕНКА ПРЕДСКАЗАТЕЛЬНОЙ СПОСОБНОСТИ АЛГОРИТМА

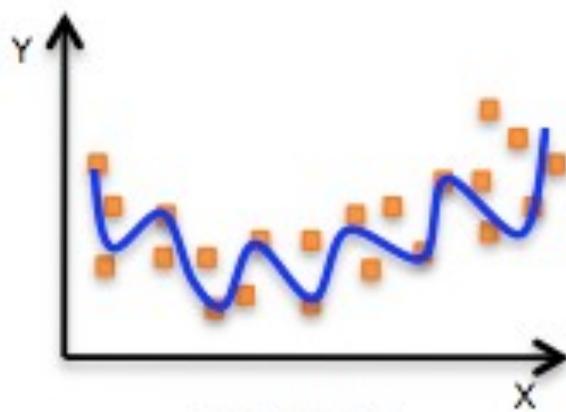
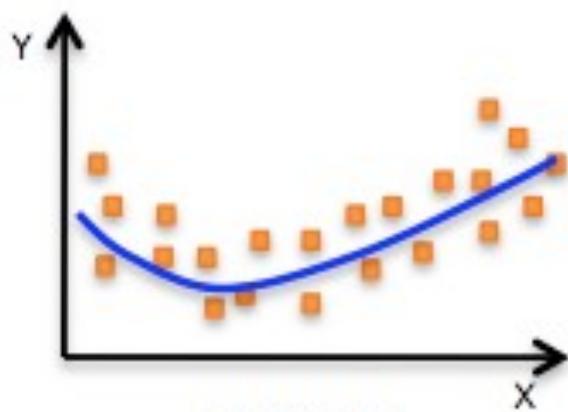
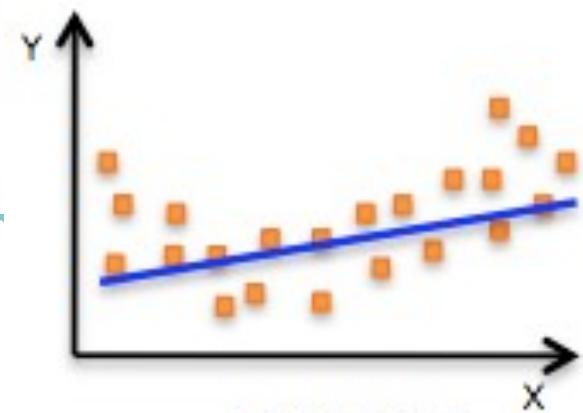
- Перед началом обучения отложим часть обучающих объектов и не будем использовать их для построения модели (отложенная выборка).



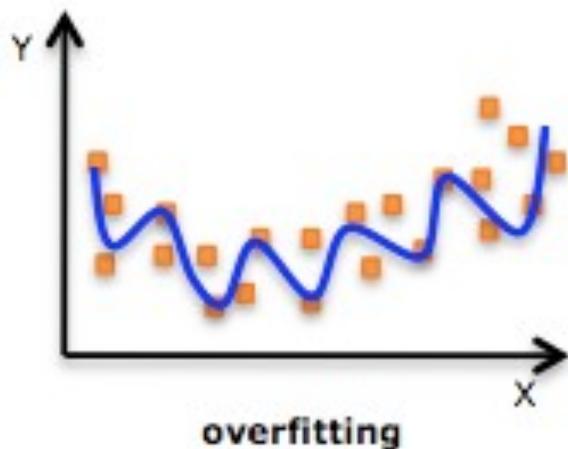
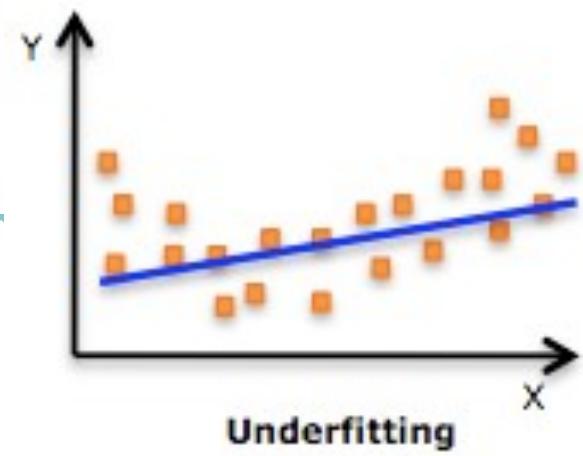
ОТЛОЖЕННАЯ ВЫБОРКА

- Перед началом обучения отложим часть обучающих объектов и не будем использовать их для построения модели (отложенная выборка).
- Тогда можно измерить качество построенной модели на отложенной выборке и оценить ее предсказательную силу.

ПЕРЕОБУЧЕНИЕ И НЕДООБУЧЕНИЕ



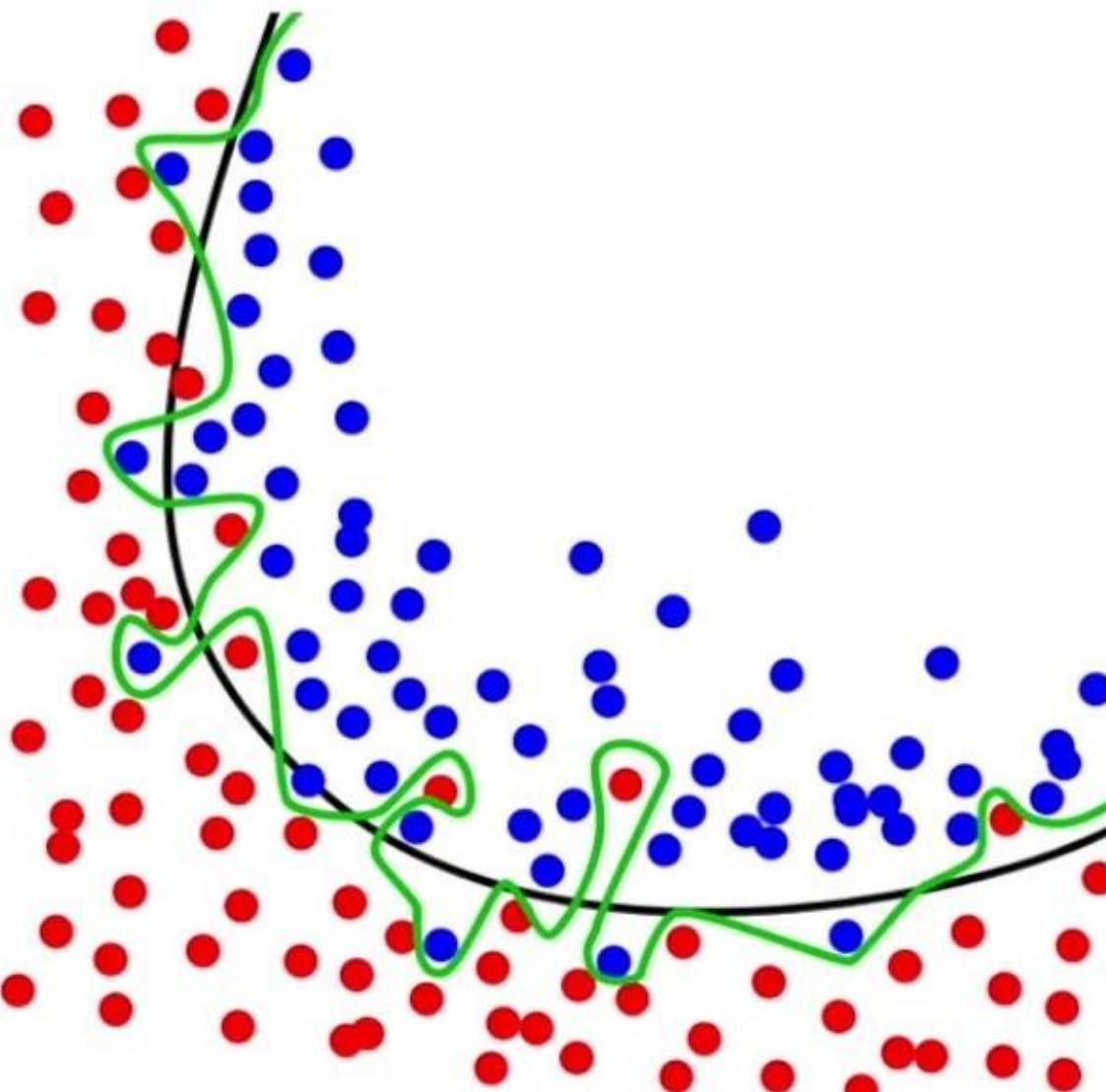
ПЕРЕОБУЧЕНИЕ И НЕДООБУЧЕНИЕ



ИЗ-ЗА ЧЕГО ВОЗНИКАЕТ ПЕРЕОБУЧЕНИЕ

- Избыточная сложность модели (большое количество весов). В этом случае лишние степени свободы в модели “тратятся” на чрезмерно точную подгонку под обучающую выборку.
- Переобучение есть всегда, когда есть оптимизация параметров по конечной (заведомо неполной) выборке.

ПРИМЕР ПЕРЕОБУЧЕНИЯ В ЗАДАЧЕ КЛАССИФИКАЦИИ



ПРИЗНАК ПЕРЕОБУЧЕНИЯ

- *Если качество на отложенной выборке сильно ниже качества на обучающих данных, то происходит переобучение*

ЛИНЕЙНАЯ РЕГРЕССИЯ

Пример (напоминание):

Предположим, что мы хотим предсказать *стоимость дома* y по его *площади (x_1)* и *количество комнат (x_2)*.

Линейная модель для предсказания стоимости:

$$a(x) = w_0 + w_1x_1 + w_2x_2,$$

где w_0, w_1, w_2 -

параметры модели (*веса*).

ЛИНЕЙНАЯ РЕГРЕССИЯ

Пример (напоминание):

Предположим, что мы хотим предсказать *стоимость дома* y по его *площади* (x_1) и *количество комната* (x_2).

Линейная модель для предсказания стоимости:

$$a(x) = w_0 + w_1x_1 + w_2x_2,$$

где w_0, w_1, w_2 -

параметры модели (веса).



Общий вид (линейная регрессия):

$$a(x) = w_0 + w_1x_1 + w_2x_2 + \dots + w_nx_n,$$

где x_1, \dots, x_n - признаки объекта x .

ЛИНЕЙНАЯ РЕГРЕССИЯ

Линейная регрессия:

$$a(x) = w_0 + w_1x_1 + w_2x_2 + \cdots + w_n x_n$$

ЛИНЕЙНАЯ РЕГРЕССИЯ

Линейная регрессия:

$$a(x) = w_0 + w_1x_1 + w_2x_2 + \cdots + w_nx_n$$

- сокращенная запись:

$$a(x) = w_0 + \sum_{j=1}^n w_jx_j$$

- запись через скалярное произведение (с добавлением признака $x_0 = 1$):

$$a(x) = w_0 \cdot 1 + \sum_{j=1}^n w_jx_j = \sum_{j=0}^n w_jx_j = (w, x)$$

ЛИНЕЙНАЯ РЕГРЕССИЯ

Линейная регрессия:

$$a(x) = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n$$

- сокращенная запись:

$$a(x) = w_0 + \sum_{j=1}^n w_j x_j$$

- запись через скалярное произведение (с добавлением признака $x_0 = 1$):

$$a(x) = w_0 \cdot 1 + \sum_{j=1}^n w_j x_j = \sum_{j=0}^n w_j x_j = (w, x) \leftrightarrow a(x) = (w, x)$$

ЛИНЕЙНАЯ РЕГРЕССИЯ

Линейная регрессия:

$$a(x) = w_0 + \sum_{j=1}^n w_j x_j = (w, x)$$

Обучение линейной регрессии - минимизация
среднеквадратичной ошибки:

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

(здесь l – количество объектов)

О ПРИМЕНИМОСТИ ЛИНЕЙНОЙ МОДЕЛИ

Пример:

Предположим, что мы хотим предсказать *стоимость дома* у него *площади (x_1)* и *количество комнат (x_2)*, *району (x_3)* и *удаленности от МКАД (x_4)*.

$$a(x) = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4.$$



О ПРИМЕНИМОСТИ ЛИНЕЙНОЙ МОДЕЛИ

Пример:

Предположим, что мы хотим предсказать *стоимость дома* y по его *площади* (x_1) и *количество комнат* (x_2), *району* (x_3) и *удаленности от МКАД* (x_4).

$$a(x) = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4.$$

Проблема №1: район (x_3) - это не число, а название района. Например, Мамыри, Дудкино, Барвиха... Что с этим делать?



О ПРИМЕНИМОСТИ ЛИНЕЙНОЙ МОДЕЛИ

Пример:

Предположим, что мы хотим предсказать *стоимость дома* y по его *площади* (x_1) и *количество комнат* (x_2), *району* (x_3) и *удаленности от МКАД* (x_4).

$$a(x) = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4.$$

Проблема №1: район (x_3) - это не число, а название района. Например, Мамыри, Дудкино, Барвиха... Что с этим делать?

Решение - one-hot encoding (ОНЕ): создаем новые числовые столбцы, каждый из которых является индикатором района.



ONE-HOT ENCODING



Район
Дудкино
Барвиха
Мамыри
...
Барвиха



Мамыри	Дудкино	Барвиха
0	1	0
0	0	1
1	0	0
...
0	0	1

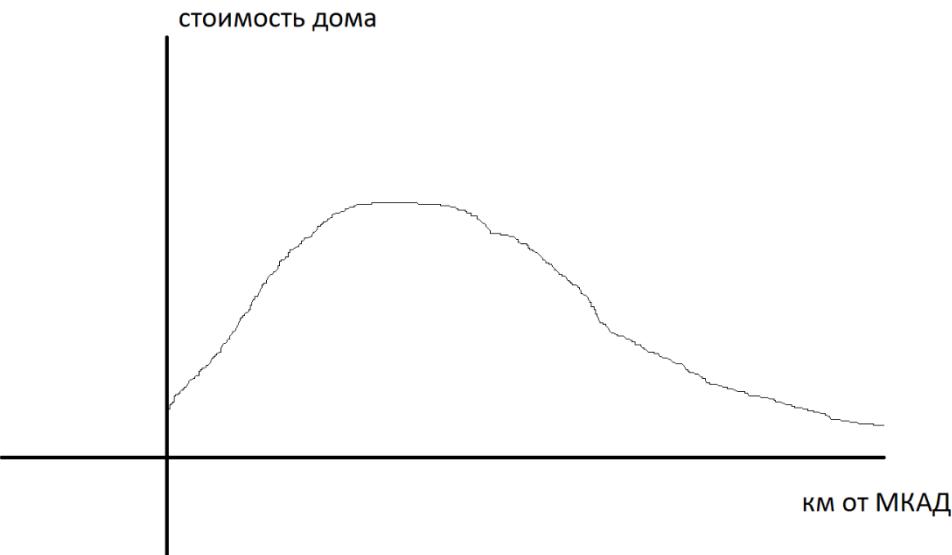
$$a(x) = w_0 + w_1x_1 + w_2x_2 + w_{31}x_{\text{Мамыри}} + w_{32}x_{\text{Дудкино}} + w_{33}x_{\text{Барвиха}} + w_4x_4.$$

О ПРИМЕНИМОСТИ ЛИНЕЙНОЙ МОДЕЛИ

Пример:

Предположим, что мы хотим предсказать *стоимость дома* y по его *площади* (x_1) и *количество комнат* (x_2), *району* (x_3) и *удаленности от МКАД* (x_4).

Проблема №2: удаленность от МКАД (x_4) не монотонно влияет на стоимость дома.



О ПРИМЕНИМОСТИ ЛИНЕЙНОЙ МОДЕЛИ

Проблема №2: удаленность от МКАД (x_4) не монотонно влияет на стоимость дома.

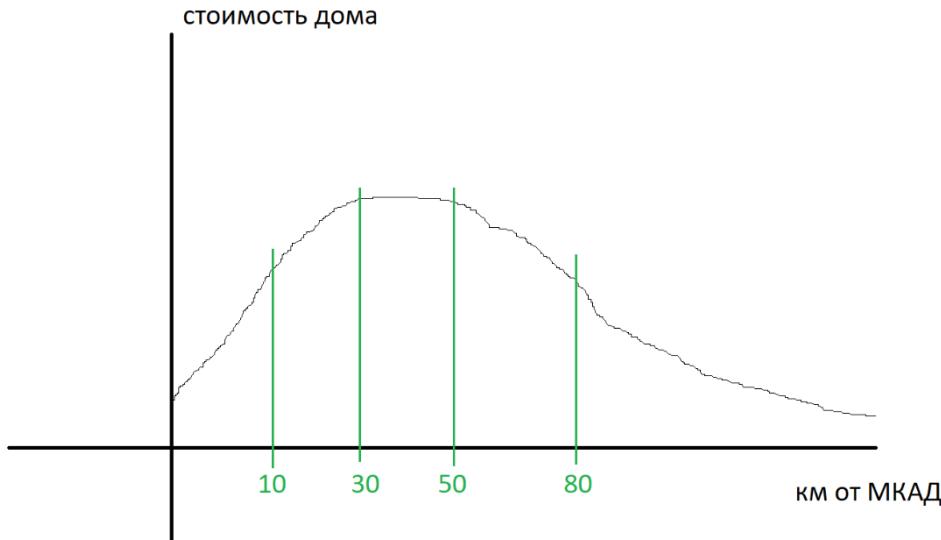
Решение - бинаризация (разбиение на бины).

Новые признаки:

- $x_{[0;10)}$ - равен 1, если дом находится в пределах 10 км от МКАД, и 0 иначе

- $x_{[10;30)}$ - равен 1, если

дом находится в пределах от 10 км до 30 км МКАД, и 0 иначе. И т.д.



О ПРИМЕНИМОСТИ ЛИНЕЙНОЙ МОДЕЛИ

Проблема №2: удаленность от МКАД (x_4) не монотонно влияет на стоимость дома.

Решение - бинаризация (разбиение на бины).

Новые признаки:

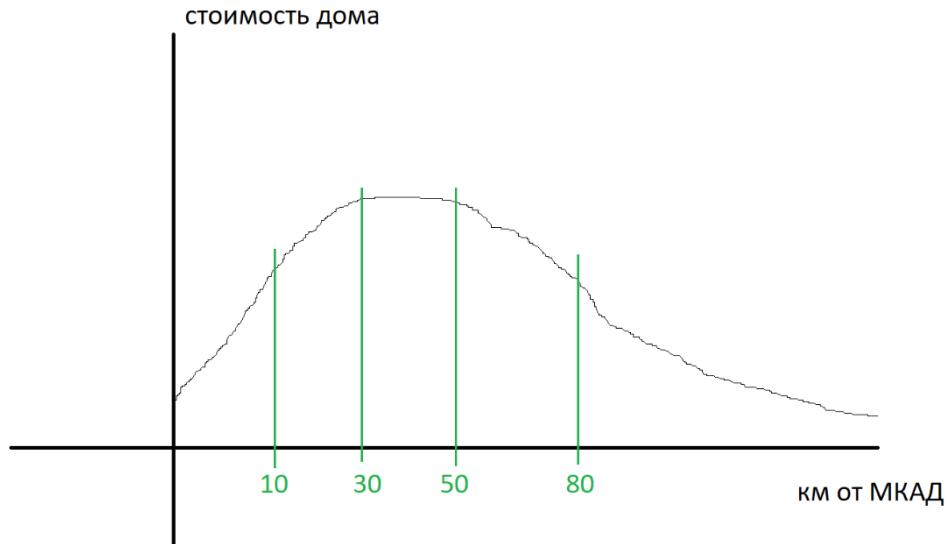
- $x_{[0;10)}$ - равен 1, если

дом находится в пределах

10 км от МКАД, и 0 иначе

- $x_{[10;30)}$ - равен 1, если

дом находится в пределах от 10 км до 30 км МКАД, и 0 иначе. И т.д.



$$a(x) =$$

$$= w_0 + w_1 x_1 + w_2 x_2 + \dots + w_{41} x_{[0;10)} + w_{42} x_{[10;30)} + w_{43} x_{[30;50)} + w_{44} x_{\geq 50}$$

АНАЛИТИЧЕСКОЕ РЕШЕНИЕ ЗАДАЧИ МЕТОДА НАИМЕНЬШИХ КВАДРАТОВ (МНК)

Задача обучения линейной регрессии (в матричной форме):

$$\frac{1}{\ell} \|Xw - y\|^2 \rightarrow \min_w$$

Точное (аналитическое) решение:

$$w = (X^T X)^{-1} X^T y$$

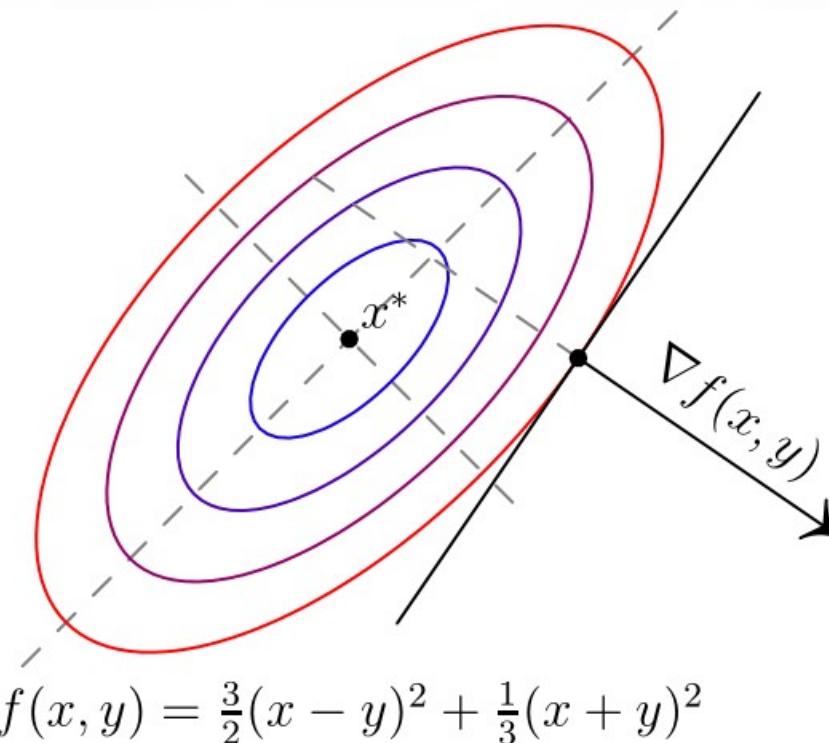
НЕДОСТАТКИ АНАЛИТИЧЕСКОЙ ФОРМУЛЫ

- Обращение матрицы - сложная операция ($O(N^3)$ от числа признаков)
- Матрица $X^T X$ может быть вырожденной или плохо обусловленной
- Если заменить среднеквадратичный функционал ошибки на другой, то скорее всего не найдем аналитическое решение

ТЕОРЕМА О ГРАДИЕНТЕ

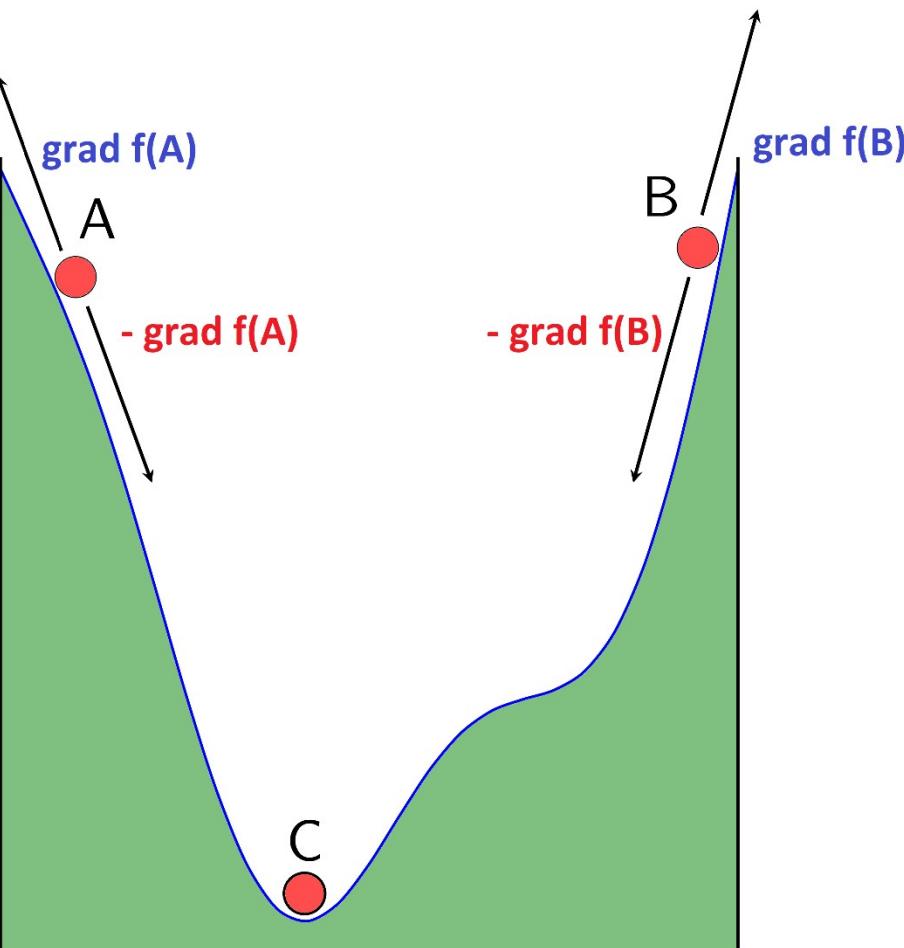
Теорема. Градиент – это вектор, в направлении которого функция быстрее всего растёт.

Антиградиент (вектор, противоположный градиенту) –
вектор, в направлении которого функция быстрее всего
убывает.



ТЕОРЕМА О ГРАДИЕНТЕ

Антиградиент (вектор, противоположный градиенту) – вектор, в направлении которого функция быстрее всего убывает.



МЕТОД ГРАДИЕНТНОГО СПУСКА

- Наша задача при обучении модели - найти такие веса w , на которых достигается **минимум функции ошибки**.

МЕТОД ГРАДИЕНТНОГО СПУСКА

- Наша задача при обучении модели - найти такие веса w , на которых достигается минимум функции ошибки.
- В простейшем случае, если ошибка среднеквадратичная, то её график - это парабола.

МЕТОД ГРАДИЕНТНОГО СПУСКА

- Наша задача при обучении модели - найти такие веса w , на которых достигается минимум функции ошибки.
- В простейшем случае, если ошибка среднеквадратичная, то её график - это парабола.
- Идея метода градиентного спуска:

На каждом шаге (на каждой итерации метода) движемся в сторону антиградиента функции потерь!

То есть на каждом шаге движемся в направлении уменьшения ошибки.

МЕТОД ГРАДИЕНТНОГО СПУСКА

- Наша задача при обучении модели - найти такие веса w , на которых достигается минимум функции ошибки.
- В простейшем случае, если ошибка среднеквадратичная, то её график - это парабола.
- Идея метода градиентного спуска:

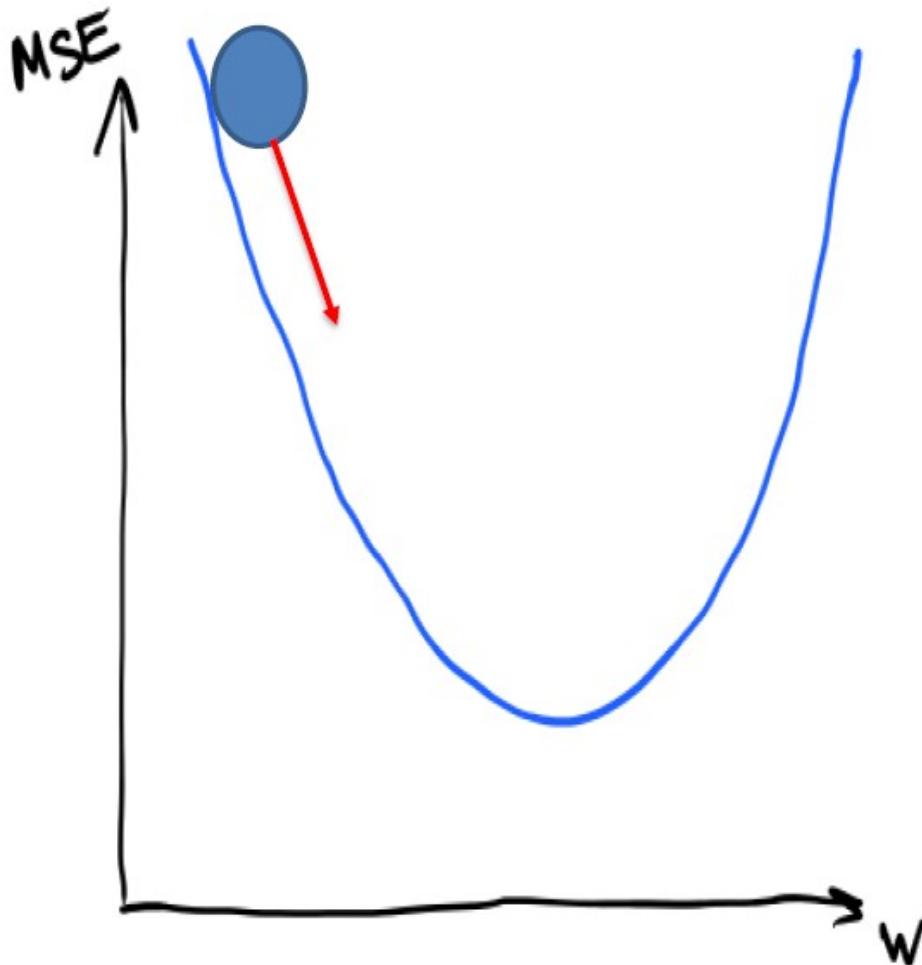
На каждом шаге (на каждой итерации метода) движемся в сторону антиградиента функции потерь!

То есть на каждом шаге движемся в направлении уменьшения ошибки.

Вектор градиента функции потерь обозначают ***grad Q*** или **∇Q** .

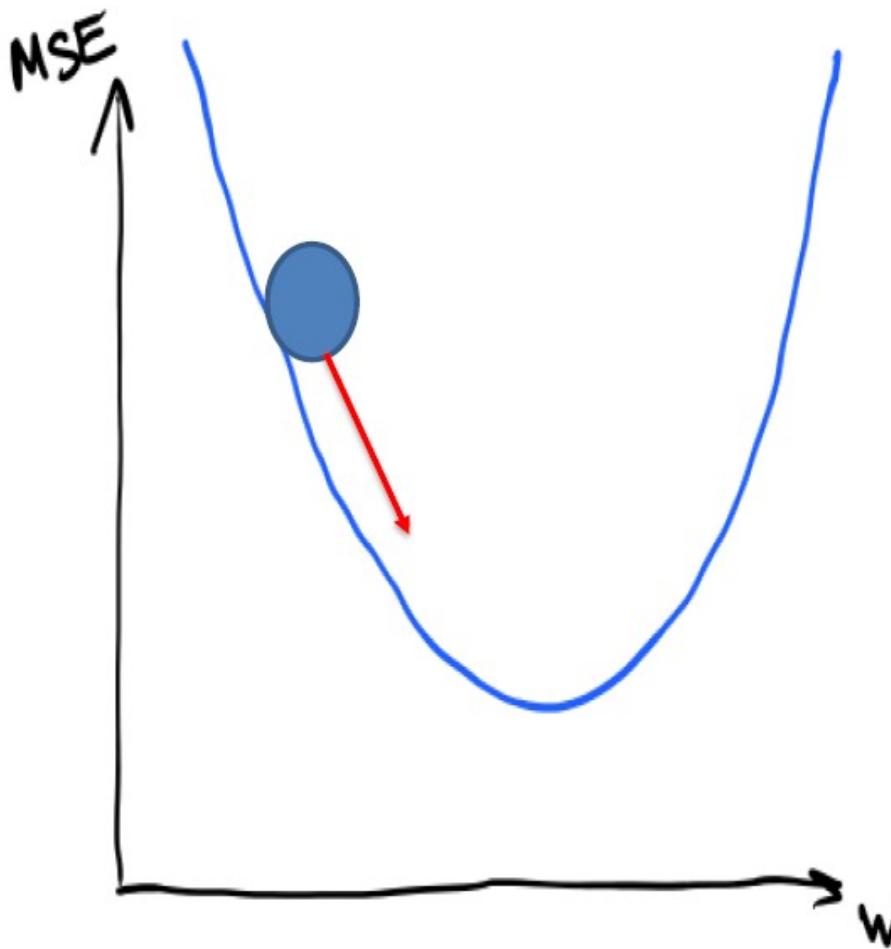
МЕТОД ГРАДИЕНТНОГО СПУСКА

На каждом шаге (на каждой итерации метода) движемся в сторону антиградиента функции потерь!



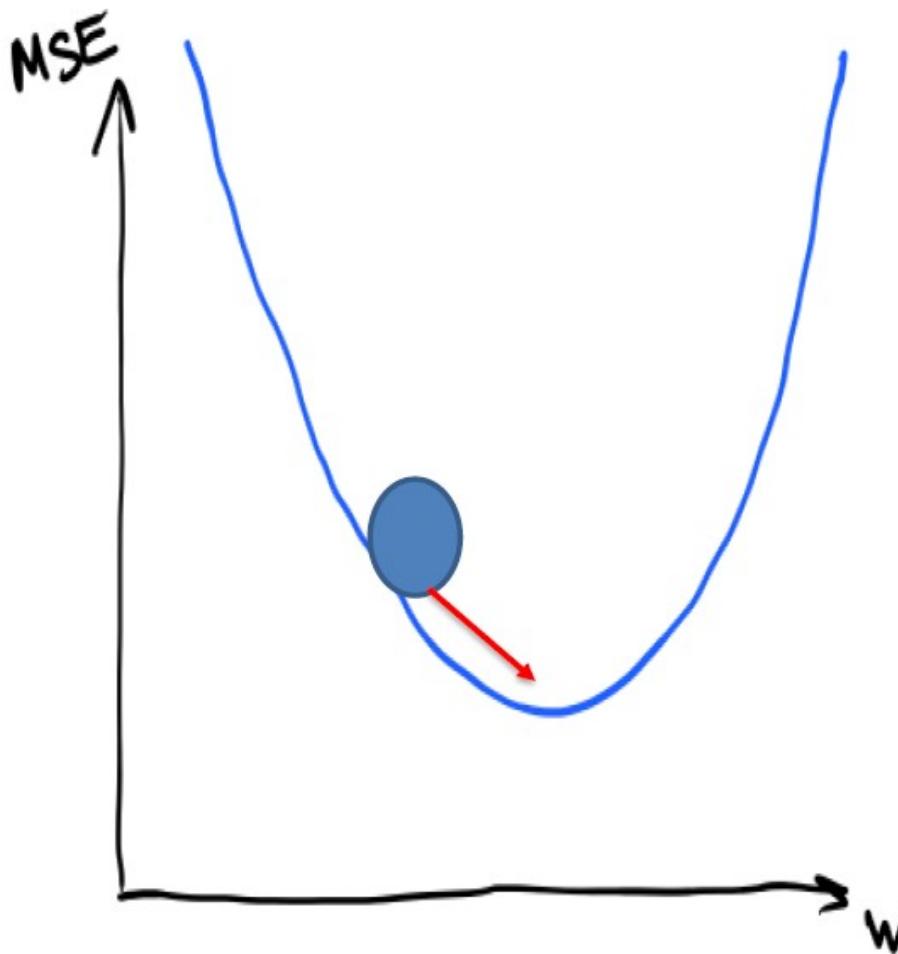
МЕТОД ГРАДИЕНТНОГО СПУСКА

На каждом шаге (на каждой итерации метода) движемся в сторону антиградиента функции потерь!



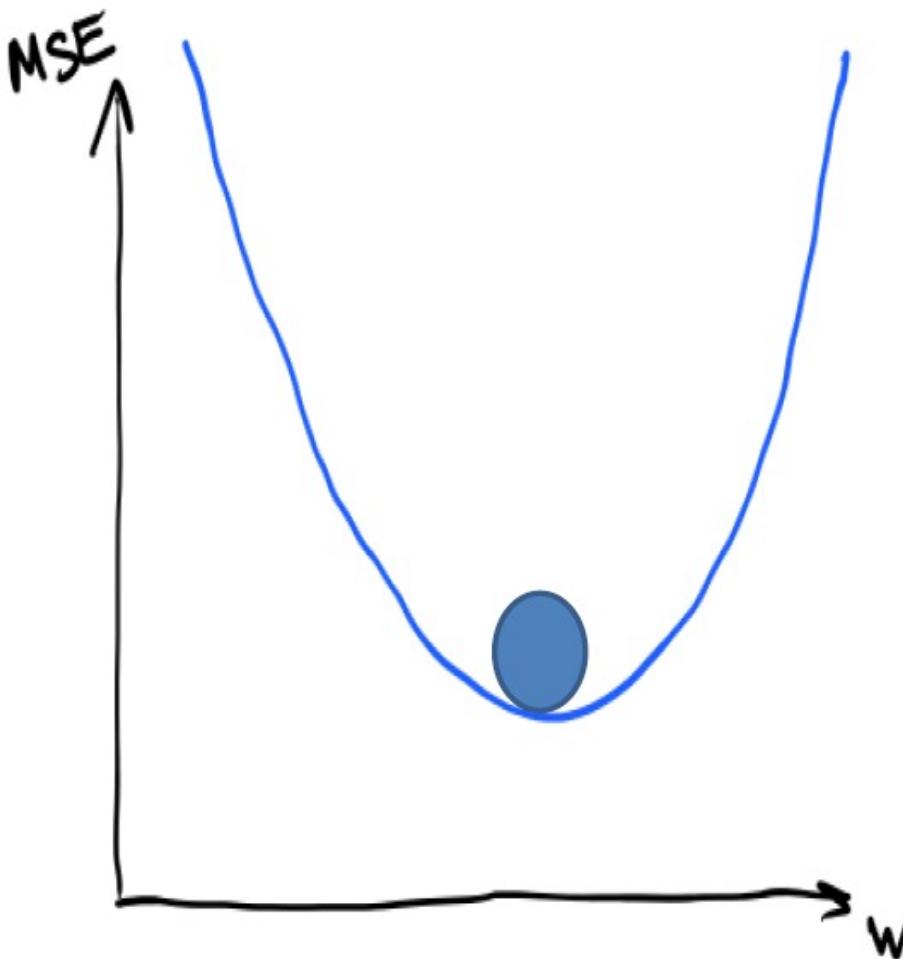
МЕТОД ГРАДИЕНТНОГО СПУСКА

На каждом шаге (на каждой итерации метода) движемся в сторону антиградиента функции потерь!



МЕТОД ГРАДИЕНТНОГО СПУСКА

На каждом шаге (на каждой итерации метода) движемся в сторону антиградиента функции потерь!



МЕТОД ГРАДИЕНТНОГО СПУСКА

Метод градиентного спуска (одномерный случай):

Пусть у нас только один вес - w .

Тогда при добавлении к весу w слагаемого $-\frac{\partial Q}{\partial w}$ функция $Q(w)$ убывает.

- Инициализируем вес $w^{(0)}$.
- На каждом следующем шаге обновляем вес, добавляя $-\frac{\partial Q}{\partial w}(w^{(k-1)})$:

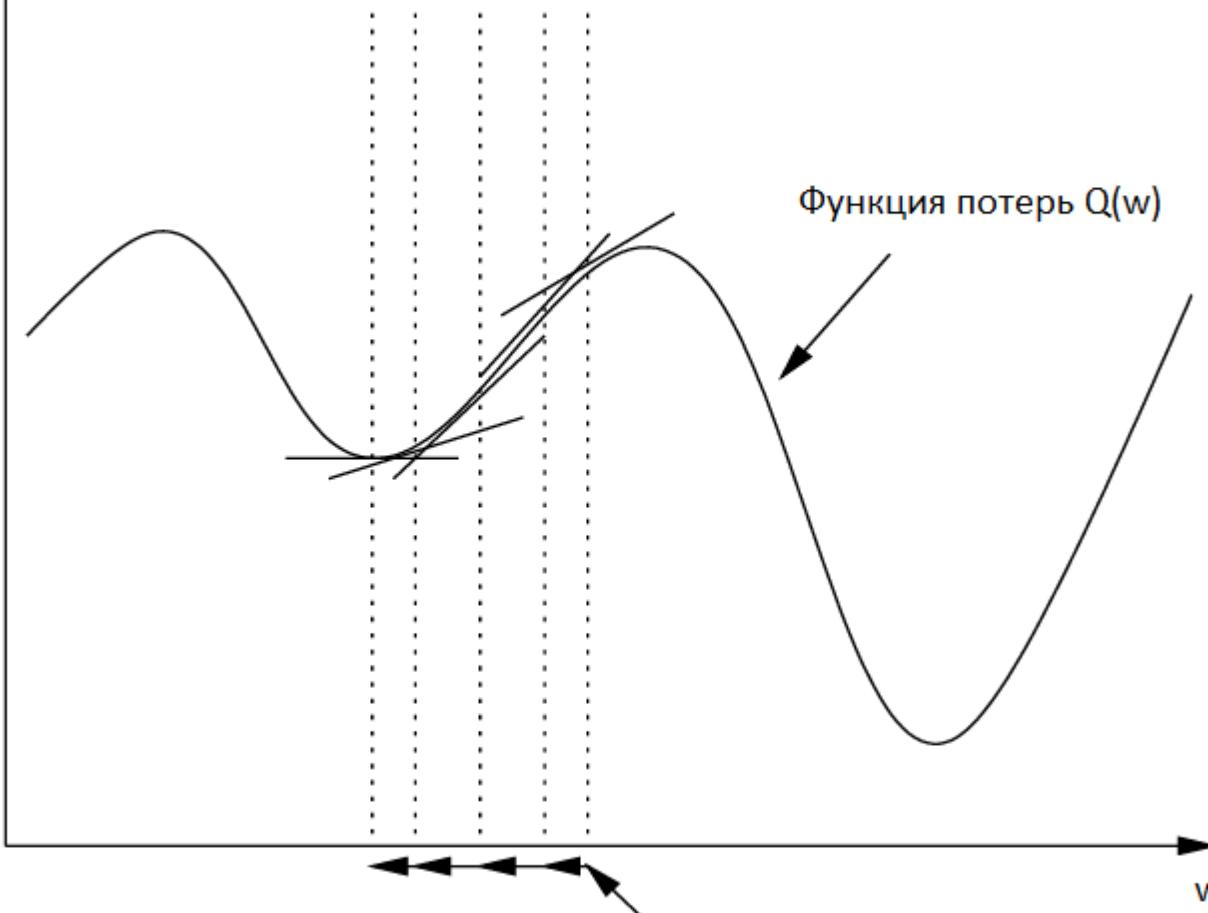
$$w^{(k)} = w^{(k-1)} - \frac{\partial Q}{\partial w}(w^{(k-1)})$$

МЕТОД ГРАДИЕНТНОГО СПУСКА

$Q(w)$

Функция потерь $Q(w)$

Random Initial point $w^{(0)}$



МЕТОД ГРАДИЕНТНОГО СПУСКА

Метод градиентного спуска (общий случай случай):

Пусть w_0, w_1, \dots, w_n - веса, которые мы ищем.

Тогда $\nabla Q(w) = \left\{ \frac{\partial Q}{\partial w_0}, \frac{\partial Q}{\partial w_1}, \dots, \frac{\partial Q}{\partial w_n} \right\}$

МЕТОД ГРАДИЕНТНОГО СПУСКА

Метод градиентного спуска (общий случай случай):

Пусть w_0, w_1, \dots, w_n - веса, которые мы ищем.

Тогда $\nabla Q(w) = \left\{ \frac{\partial Q}{\partial w_0}, \frac{\partial Q}{\partial w_1}, \dots, \frac{\partial Q}{\partial w_n} \right\}$

- Инициализируем веса $w_0^{(0)}, w_1^{(0)}, w_2^{(0)}, \dots, w_n^{(0)}$.
- На каждом следующем шаге обновляем веса:

$$w_0^{(k)} = w_0^{(k-1)} - \frac{\partial Q}{\partial w_0} (w_0^{(k-1)}),$$

...

$$w_n^{(k)} = w_n^{(k-1)} - \frac{\partial Q}{\partial w_n} (w_n^{(k-1)}).$$

МЕТОД ГРАДИЕНТНОГО СПУСКА

Формулу для обновления весов можно записать в векторном виде:

- Инициализируем веса $w^{(0)}$.
- На каждом следующем шаге обновляем веса по формуле:

$$w^{(k)} = w^{(k-1)} - \nabla Q(w^{(k-1)})$$

МЕТОД ГРАДИЕНТНОГО СПУСКА

Формулу для обновления весов можно записать в векторном виде:

- Инициализируем веса $w^{(0)}$.
- На каждом следующем шаге обновляем веса по формуле:

$$w^{(k)} = w^{(k-1)} - \nabla Q(w^{(k-1)})$$

В формулу обычно добавляют параметр η – величина градиентного шага (learning rate). Он отвечает за скорость движения в сторону антиградиента:

$$w^{(k)} = w^{(k-1)} - \eta \nabla Q(w^{(k-1)})$$

МЕТОД ГРАДИЕНТНОГО СПУСКА

Формулу для обновления весов можно записать в векторном виде:

- Инициализируем веса $w^{(0)}$.
- На каждом следующем шаге обновляем веса по формуле:

$$w^{(k)} = w^{(k-1)} - \nabla Q(w^{(k-1)})$$

В формулу обычно добавляют параметр η – величина градиентного шага (learning rate). Он отвечает за скорость движения в сторону антиградиента:

$$w^{(k)} = w^{(k-1)} - \eta \nabla Q(w^{(k-1)})$$

Если функция $Q(w)$ выпуклая и гладкая, а также имеет минимум в точке w^* , то метод градиентного спуска при аккуратно подобранном η через некоторое число шагов гарантированно попадет в малую окрестность точки w^* .

МЕТОД ГРАДИЕНТНОГО СПУСКА

Пример:

формула обновления весов методом
градиентного спуска.

$$y = w_0 + w_1 x$$

$$Q(w) = \sum_{i=1}^l (w_0 + w_1 x_i - y_i)^2$$

ВАРИАНТЫ ИНИЦИАЛИЗАЦИИ ВЕСОВ

- $w_j = 0, j = 1, \dots, n$

- Небольшие случайные значения:

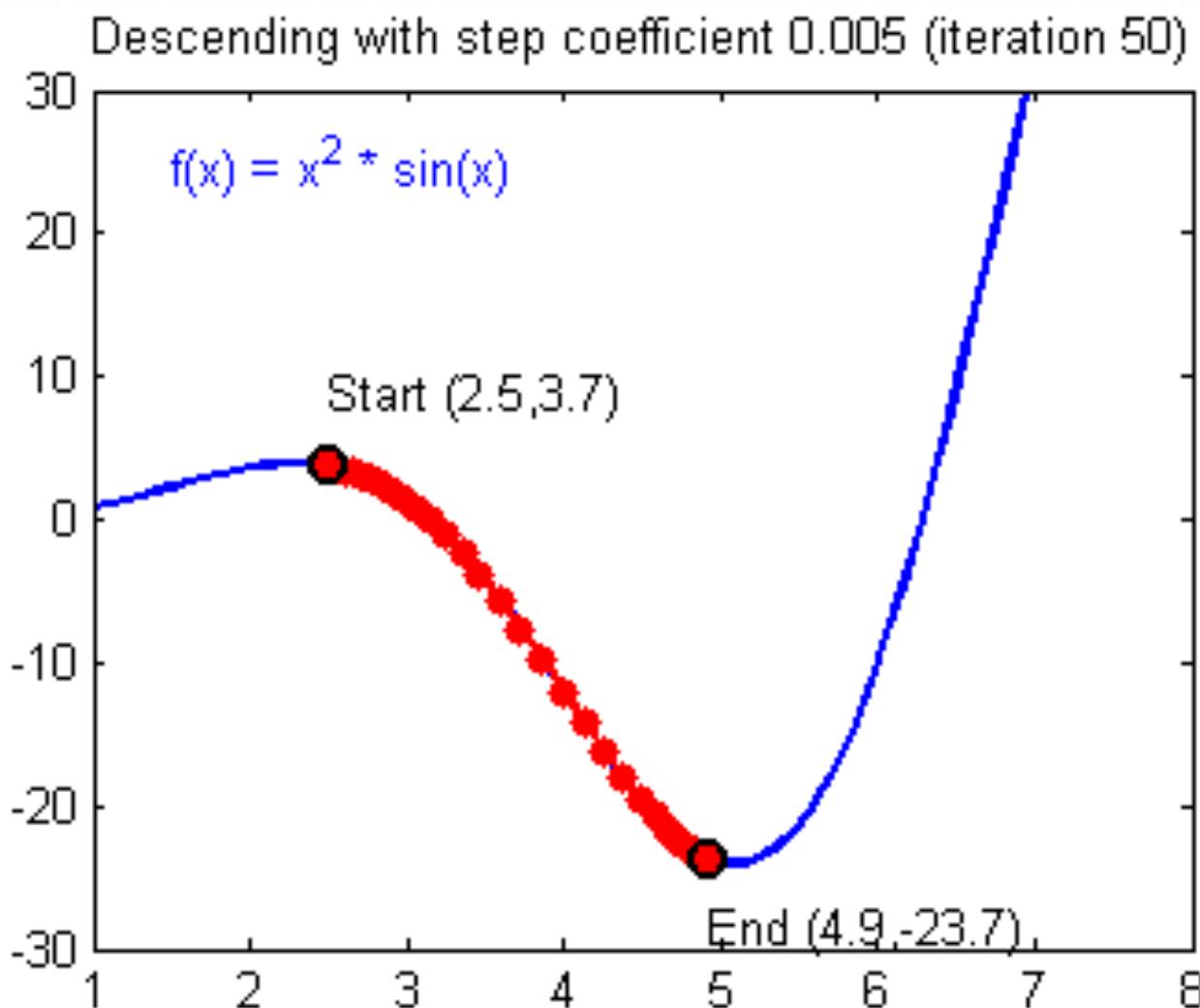
$$w_j := \text{random}(-\varepsilon, \varepsilon)$$

- Обучение по небольшой случайной подвыборке объектов
- Мультистарт: многократный запуск из разных случайных начальных приближений и выбор лучшего решения

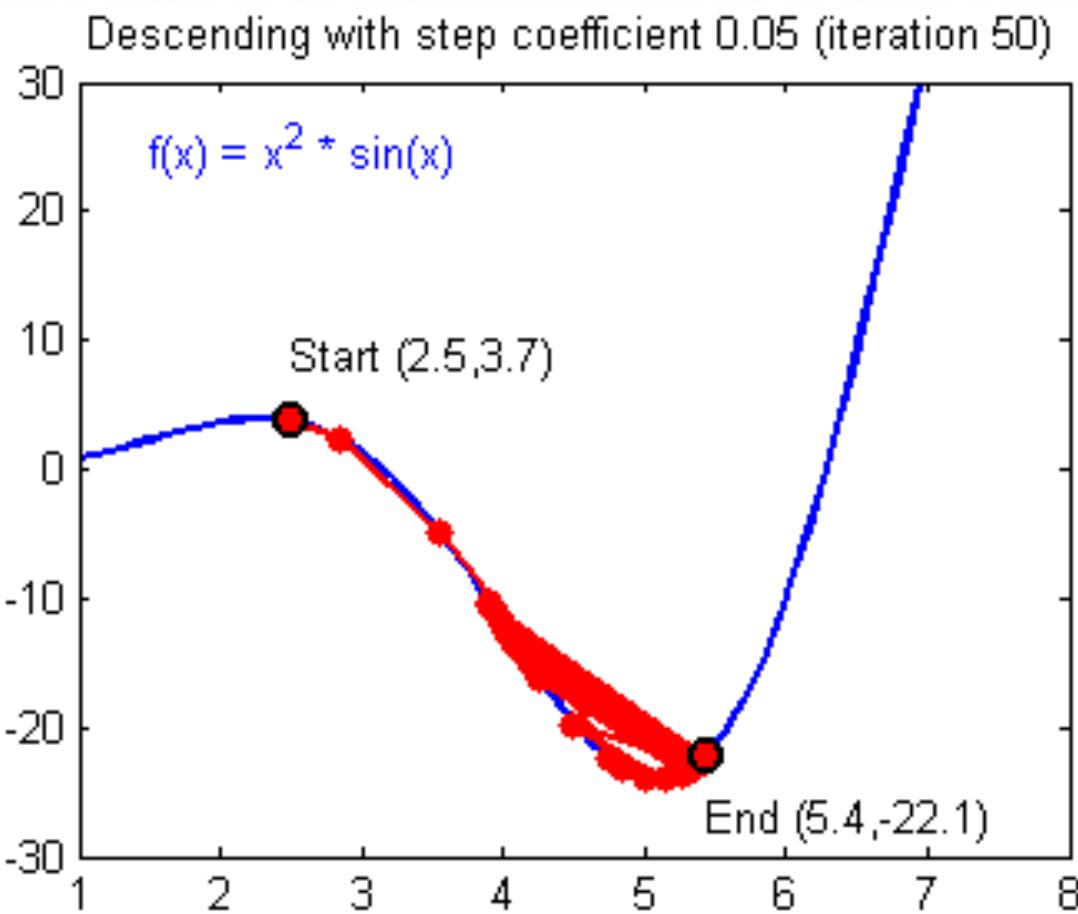
КРИТЕРИИ ОСТАНОВА

- $|Q(w^{(k)}) - Q(w^{(k-1)})| < \varepsilon$
- $\|w^{(k)} - w^{(k-1)}\| < \varepsilon$

ГРАДИЕНТНЫЙ СПУСК



ПРОБЛЕМА ВЫБОРА ГРАДИЕНТНОГО ШАГА



ГРАДИЕНТНЫЙ ШАГ

В общем случае градиентный шаг может зависеть от номера итерации, тогда будем писать не η , а η_k .

- $\eta_k = c$
- $\eta_k = \frac{1}{k}$
- $\eta_k = \lambda \left(\frac{s_0}{s_0+k} \right)^p$, λ, s_0, p - параметры

ОДИН ИЗ НЕДОСТАТКОВ ГРАДИЕНТНОГО СПУСКА

(с точки зрения реализации)

- На каждом шаге для вычисления $\nabla Q(w)$ мы вычисляем производную по каждому весу от каждого объекта. То есть вычисляем целую матрицу производных – это затратно и по времени, и по памяти.

СТОХАСТИЧЕСКИЙ ГРАДИЕНТНЫЙ СПУСК

Stochastic gradient descent (SGD):

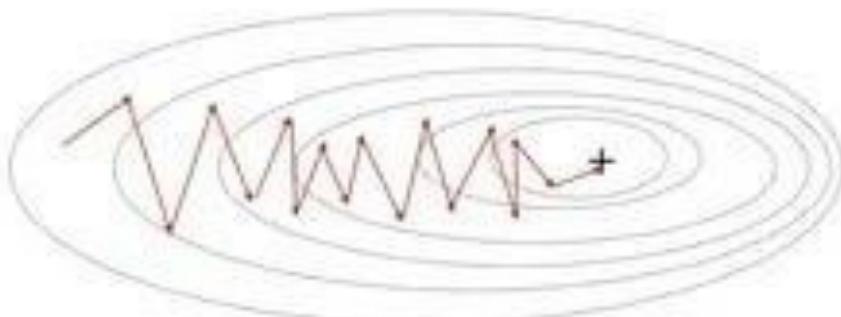
- на каждом шаге выбираем **один случайный объект** и сдвигаемся в сторону антиградиента по этому объекту:

$$w^{(k)} = w^{(k-1)} - \eta_k \cdot \nabla q_{i_k}(w^{(k-1)}),$$

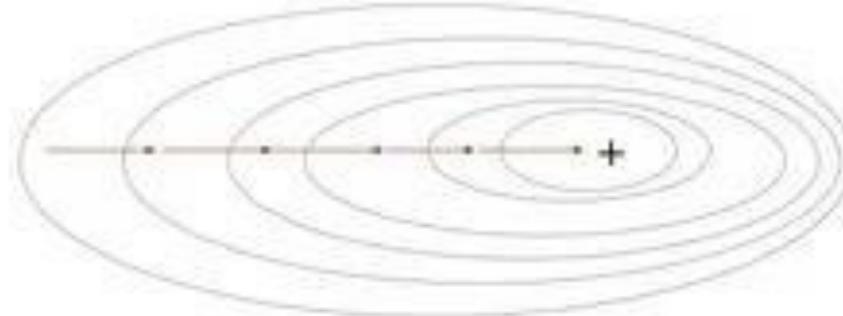
где $\nabla q_{i_k}(w^{(k-1)})$ - градиент функции потерь, вычисленный только по объекту с номером i_k (а не по всей обучающей выборке).

СТОХАСТИЧЕСКИЙ ГРАДИЕНТНЫЙ СПУСК

Stochastic Gradient Descent



Gradient Descent



Если функция $Q(w)$ выпуклая и гладкая, а также имеет минимум в точке w^* , то метод стохастического градиентного спуска при аккуратно подобранном η через некоторое число шагов гарантированно попадет в малую окрестность точки w^* . Однако, сходится метод медленнее, чем обычный градиентный спуск

MINI-BATCH GRADIENT DESCENT

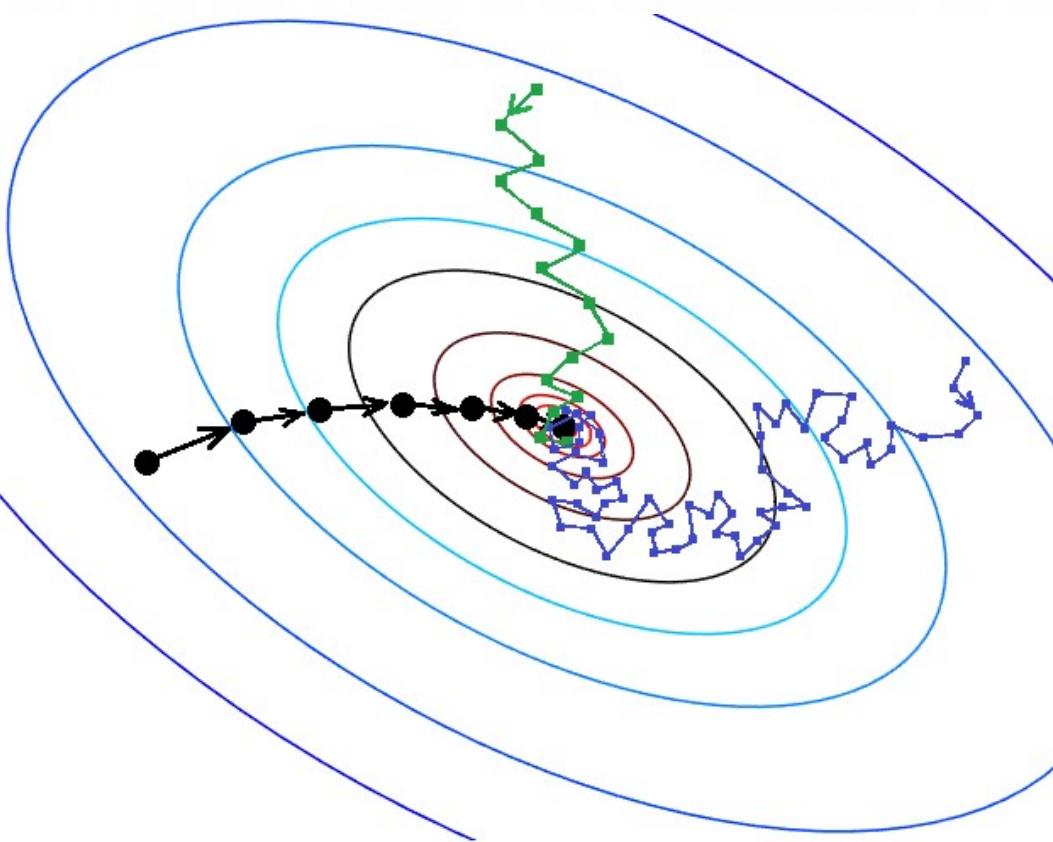
Промежуточное решение между классическим градиентным спуском и стохастическим вариантом.

- Выбираем batch size (например, 32, 64 и т.д.). Разбиваем все пары объект-ответ на группы размера batch size.
- На i -й итерации градиентного спуска вычисляем $\nabla Q(w)$ только по объектам i -го батча:

$$w^{(k)} = w^{(k-1)} - \eta_k \cdot \nabla Q_i(w^{(k-1)}),$$

где $\nabla Q_i(w^{(k-1)})$ - градиент функции потерь, вычисленный по объектам из i -го батча.

ВАРИАНТЫ ГРАДИЕНТНОГО СПУСКА



Batch GD

- Slowest
- Perfect gradient

Stochastic GD

- Fastest
- Rough-estimate grad

Mini-batch GD

- Compromise

1. МЕТРИКИ КАЧЕСТВА И ФУНКЦИОНАЛЫ ОШИБКИ В ЗАДАЧАХ РЕГРЕССИИ

МЕТРИКИ КАЧЕСТВА И ФУНКЦИИ ОШИБКИ

- Функционал (функция) ошибки – функция, которую минимизируют в процессе обучения модели для нахождения неизвестных параметров (весов).
- Метрика качества – функция, которую используют для оценки качества построенной (уже обученной) модели.

МЕТРИКИ КАЧЕСТВА И ФУНКЦИИ ОШИБКИ

- **Функционал (функция) ошибки** – функция, которую минимизируют в процессе обучения модели для нахождения неизвестных параметров (весов).
- **Метрика качества** – функция, которую используют для оценки качества построенной (уже обученной) модели.

Иногда одна и та же функция может использоваться и для обучения модели (функция ошибки), и для оценки качества модели (метрика качества).

ЛИНЕЙНАЯ РЕГРЕССИЯ

Линейная регрессия:

$$a(x) = w_0 + \sum_{j=1}^d w_j x_j$$

Обучение линейной регрессии - минимизация
среднеквадратичной ошибки:

$$\frac{1}{l} \sum_{i=1}^l (a(x_i) - y_i)^2 \rightarrow \min_w$$

СРЕДНЕКВАДРАТИЧНОЕ ОТКЛОНЕНИЕ: MSE (MEAN SQUARED ERROR)

Среднеквадратичное отклонение:

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

СРЕДНЕКВАДРАТИЧНОЕ ОТКЛОНЕНИЕ: MSE (MEAN SQUARED ERROR)

Среднеквадратичное отклонение:

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

Плюсы:

- Позволяет сравнивать модели
- Подходит для контроля качества во время обучения

СРЕДНЕКВАДРАТИЧНОЕ ОТКЛОНЕНИЕ: MSE

Среднеквадратичное отклонение:

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

Плюсы:

- Позволяет сравнивать модели
- Подходит для контроля качества во время обучения

Минусы:

- Плохо интерпретируется, т.к. не сохраняет единицы измерения (если целевая переменная – кг, то MSE измеряется в кг в квадрате)
- Тяжело понять, насколько хорошо данная модель решает задачу, так как MSE не ограничена сверху.

RMSE (ROOT MEAN SQUARED ERROR)

Корень из среднеквадратичной ошибки:

$$RMSE(a, X) = \sqrt{\frac{1}{l} \sum_{i=1}^l (a(x_i) - y_i)^2}$$

Плюсы:

- Все плюсы MSE
- Сохраняет единицы измерения (в отличие от MSE)

Минусы:

- Тяжело понять, насколько хорошо данная модель решает задачу, так как RMSE не ограничена сверху.

КОЭФФИЦИЕНТ ДЕТЕРМИНАЦИИ (R^2)

Коэффициент детерминации:

$$R^2(a, X) = 1 - \frac{\sum_{i=1}^l (a(x_i) - y_i)^2}{\sum_{i=1}^l (y_i - \bar{y})^2},$$

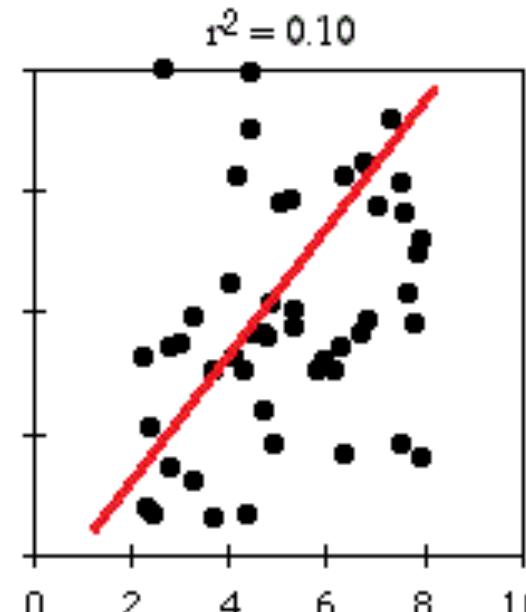
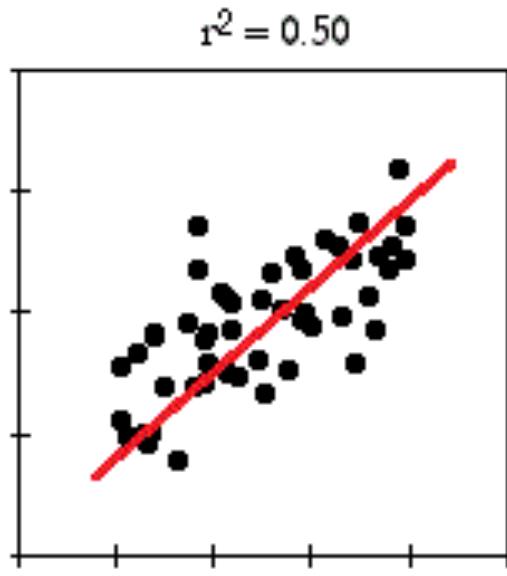
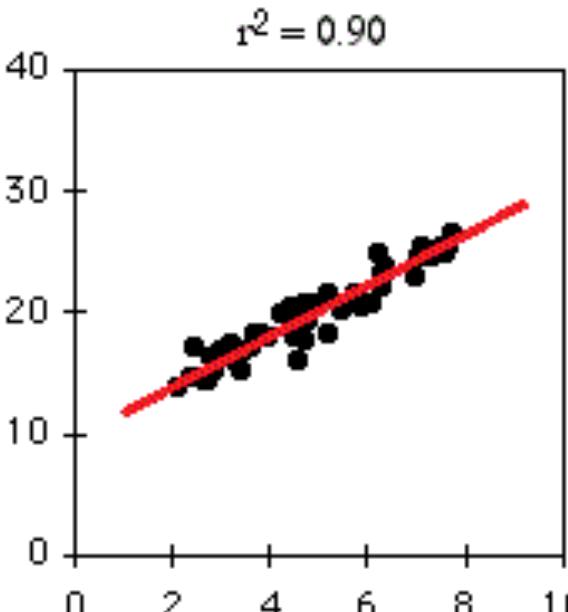
где $\bar{y} = \frac{1}{l} \sum_{i=1}^l y_i$.

Коэффициент детерминации это доля дисперсии целевой переменной, объясняемая моделью.

- Чем ближе R^2 к 1, тем лучше модель объясняет данные
- Чем ближе R^2 к 0, тем ближе модель к константному предсказанию
- Отрицательный R^2 говорит о том, что модель плохо решает задачу

КОЭФФИЦИЕНТ ДЕТЕРМИНАЦИИ (R^2)

$$R^2 \leq 1$$



MAE (MEAN ABSOLUTE ERROR)

Средняя абсолютная ошибка:

$$MAE(a, X) = \frac{1}{l} \sum_{i=1}^l |\mathbf{a}(\mathbf{x}_i) - y_i|$$

MAE (MEAN ABSOLUTE ERROR)

Средняя абсолютная ошибка:

$$MAE(a, X) = \frac{1}{l} \sum_{i=1}^l |a(x_i) - y_i|$$

Плюсы:

- Менее чувствителен к выбросам, чем MSE

MAE (MEAN ABSOLUTE ERROR)

Средняя абсолютная ошибка:

$$MAE(a, X) = \frac{1}{l} \sum_{i=1}^l |a(x_i) - y_i|$$

Плюсы:

- Менее чувствителен к выбросам, чем MSE

Минусы:

- MAE - не дифференцируемый функционал

ОПТИМУМЫ MSE И MAE

Рассмотрим вероятностную постановку задачи.

Предположим, что на объектах с одинаковым признаковым описанием могут быть разные ответы. В этом случае на всех таких объектах MSE (или MAE) должна выдать один и тот же ответ.

Теорема. Пусть даны l объектов с одинаковым признаковым описанием и значениями целевой переменной y_1, \dots, y_l .

Тогда:

1. Оптимум MSE достигается на среднем значении ответов:

$$\alpha_{MSE} = \sum_{i=1}^l y_i$$

2. Оптимум MAE достигается на медиане ответов:

$$\alpha_{MAE} = median\{y_1, \dots, y_l\}$$

MSLE (MEAN SQUARED LOGARITHMIC ERROR)

Среднеквадратичная логарифмическая ошибка:

$$MSLE(a, X) = \frac{1}{l} \sum_{i=1}^l (\log(a(x_i) + 1) - \log(y + 1))^2$$

- Подходит для задач с неотрицательной целевой переменной ($y \geq 0$)
- Штрафует за отклонения в порядке величин
- Штрафует заниженные прогнозы сильнее, чем завышенные

MAPE

MAPE – Mean Absolute Percentage Error:

$$MAPE(a, X) = \frac{1}{l} \sum_{i=1}^l \frac{|y_i - a(x_i)|}{|y_i|}$$

MAPE измеряет относительную ошибку.

MAPE

$$MAPE(a, X) = \frac{1}{l} \sum_{i=1}^l \frac{|y_i - a(x_i)|}{|y_i|}$$

Плюсы:

- Ограничена: $0 \leq MAPE \leq 1$
- Хорошо интерпретируема: например, $MAPE=0.16$ означает, что ошибка модели в среднем составляет 16% от фактических значений.

MAPE

$$MAPE(a, X) = \frac{1}{l} \sum_{i=1}^l \frac{|y_i - a(x_i)|}{|y_i|}$$

Плюсы:

- Ограничена: $0 \leq MAPE \leq 1$
- Хорошо интерпретируема: например, $MAPE=0.16$ означает, что ошибка модели в среднем составляет 16% от фактических значений.

Минусы:

- По-разному относится к недо- и перепрогнозу. Например, если правильный ответ $y = 10$, а прогноз $a(x) = 20$, то ошибка $\frac{|10-20|}{|10|} = 1$, а если ответ $y = 30$, то ошибка $\frac{|30-20|}{|30|} = \frac{1}{3} \approx 0.33$.

SMAPE

SMAPE – *Symmetric Mean Absolute Percentage Error*
(симметричный вариант MAPE):

$$SMAPE(a, X) = \frac{1}{l} \sum_{i=1}^l \frac{|y_i - a(x_i)|}{(|y_i| + |a(x_i)|)/2}$$

SMAPE – попытка сделать симметричным прогноз (то есть дать одинаковую ошибку для недо- и перепрогноза).

SMAPE

SMAPE – *Symmetric Mean Absolute Percentage Error*
(симметричный вариант MAPE):

$$SMAPE(a, X) = \frac{1}{l} \sum_{i=1}^l \frac{|y_i - a(x_i)|}{(|y_i| + |a(x_i)|)/2}$$

SMAPE – попытка сделать симметричным прогноз (то есть дать одинаковую ошибку для недо- и перепрогноза).

Проверим:

Пусть правильный ответ $y = 10$, а прогноз $a(x) = 20$, то

ошибка $\frac{|10-20|}{|10+20|/2} = \frac{2}{3} \approx 0.67$, а если ответ $y = 30$, то ошибка

$$\frac{|30-20|}{|30+20|/2} = \frac{2}{5} = 0.4.$$

SMAPE

SMAPE – попытка сделать симметричным прогноз (то есть дать одинаковую ошибку для недо- и перепрогноза).

Проверим:

Пусть правильный ответ $y = 10$, а прогноз $a(x) = 20$, то

ошибка $\frac{|10-20|}{|10+20|/2} = \frac{2}{3} \approx 0.67$, а если ответ $y = 30$, то ошибка

$\frac{|30-20|}{|30+20|/2} = \frac{2}{5} = 0.4$.

Ошибки стали меньше отличаться друг от друга, но всё-таки не равны.

SMAPE

SMAPE – попытка сделать симметричным прогноз (то есть дать одинаковую ошибку для недо- и перепрогноза).

“Сейчас уже в среде прогнозистов сложилось более-менее устойчивое понимание, что SMAPE не является хорошей ошибкой. Тут дело не только в завышении прогнозов, но и в том, что наличие прогноза в знаменателе позволяет манипулировать результатами оценки.” (см. [источник](#))

КВАНТИЛЬНАЯ РЕГРЕССИЯ

Квантильная функция потерь:

$$Q(a, X^\ell) = \sum_{i=1}^{\ell} \rho_\tau(y_i - a(x_i))$$

Здесь

$$\rho_\tau(z) = (\tau - 1)[z < 0]z + \tau[z \geq 0]z = (\tau - \frac{1}{2})z + \frac{1}{2}|z|$$

Параметр $\tau \in [0; 1]$.

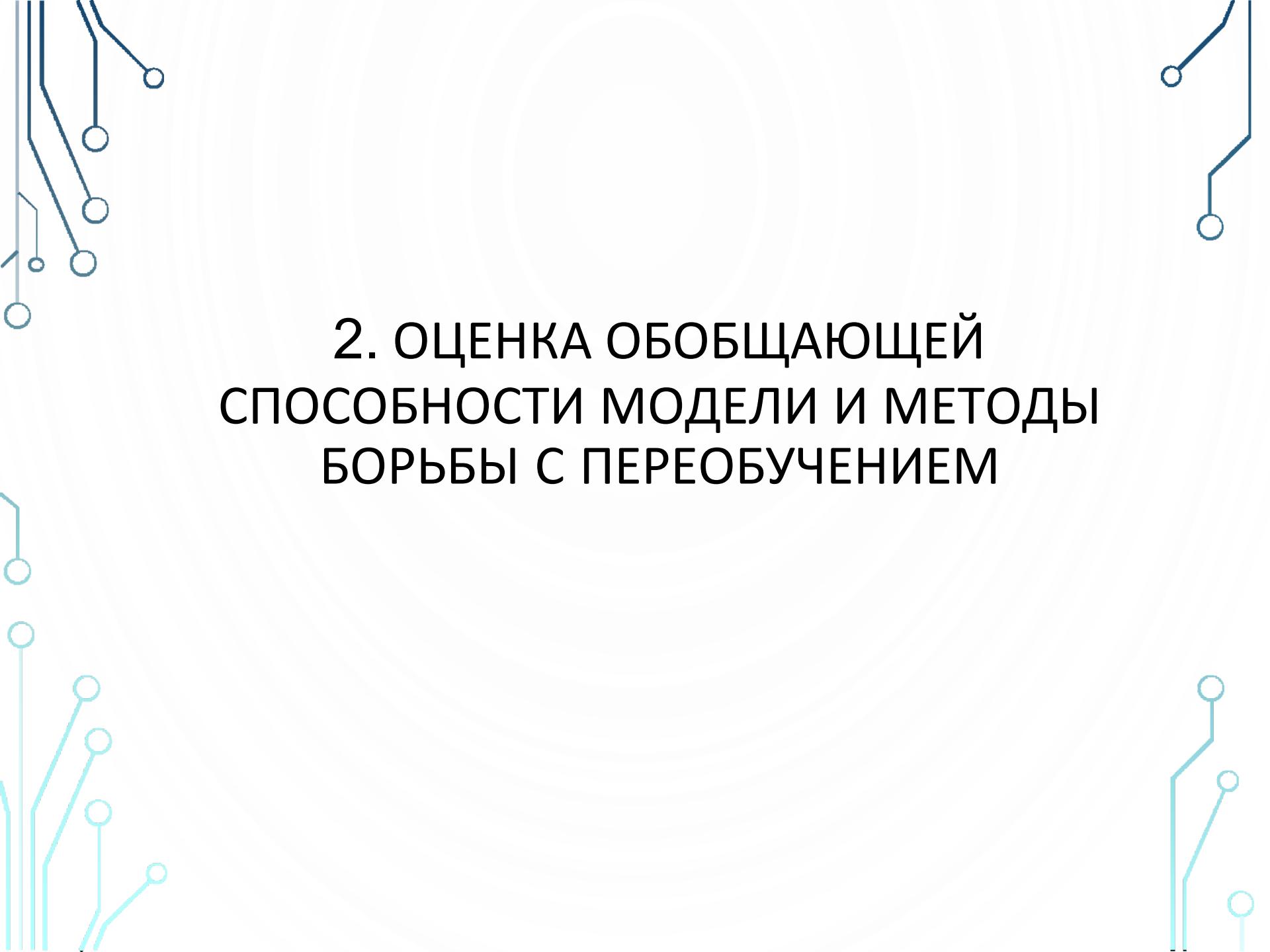
- Чем больше τ , тем больше штрафуем за занижение прогноза.

ВЕРОЯТНОСТНЫЙ СМЫСЛ КВАНТИЛЬНОЙ ФУНКЦИИ ПОТЕРЬ

Теорема.

Пусть в каждой точке $x \in X$ (пространство объектов) задано распределение $p(y|x)$ на ответах для данного объекта.

Тогда оптимизация функции потерь $\rho_\tau(z)$ дает алгоритм $a(x)$, приближающий τ -квантиль распределения ответов в каждой точке $x \in X$.

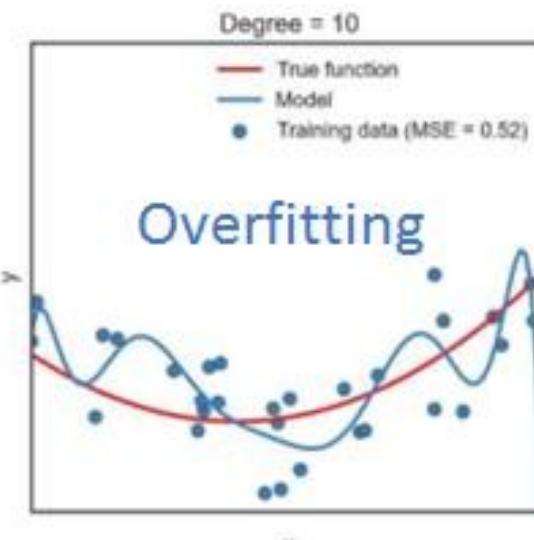
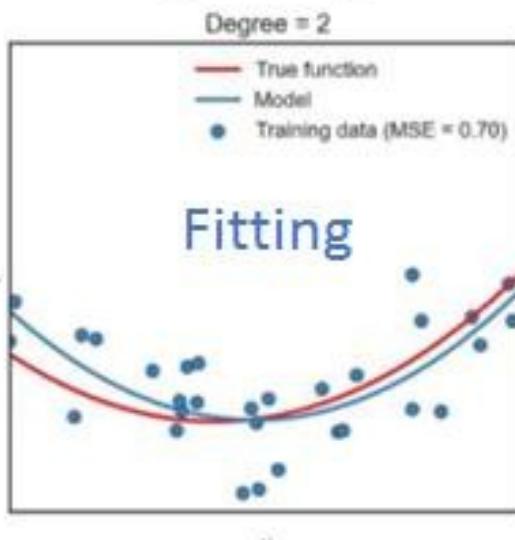
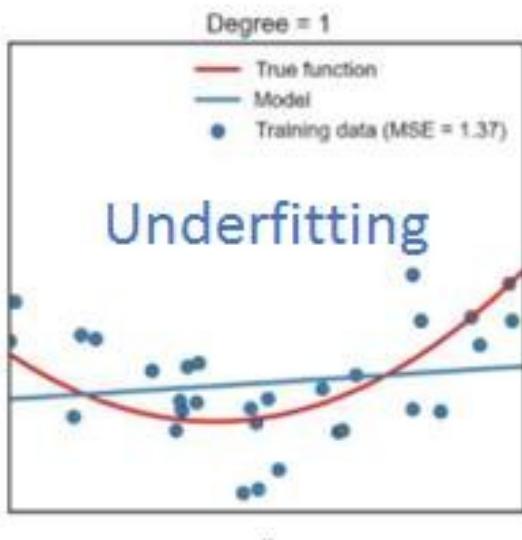


2. ОЦЕНКА ОБОБЩАЮЩЕЙ СПОСОБНОСТИ МОДЕЛИ И МЕТОДЫ БОРЬБЫ С ПЕРЕОБУЧЕНИЕМ

ОЦЕНКА ОБОБЩАЮЩЕЙ СПОСОБНОСТИ МОДЕЛИ

Переобучение (overfitting) – явление, при котором качество модели на новых данных сильно хуже, чем качество на тренировочных данных.

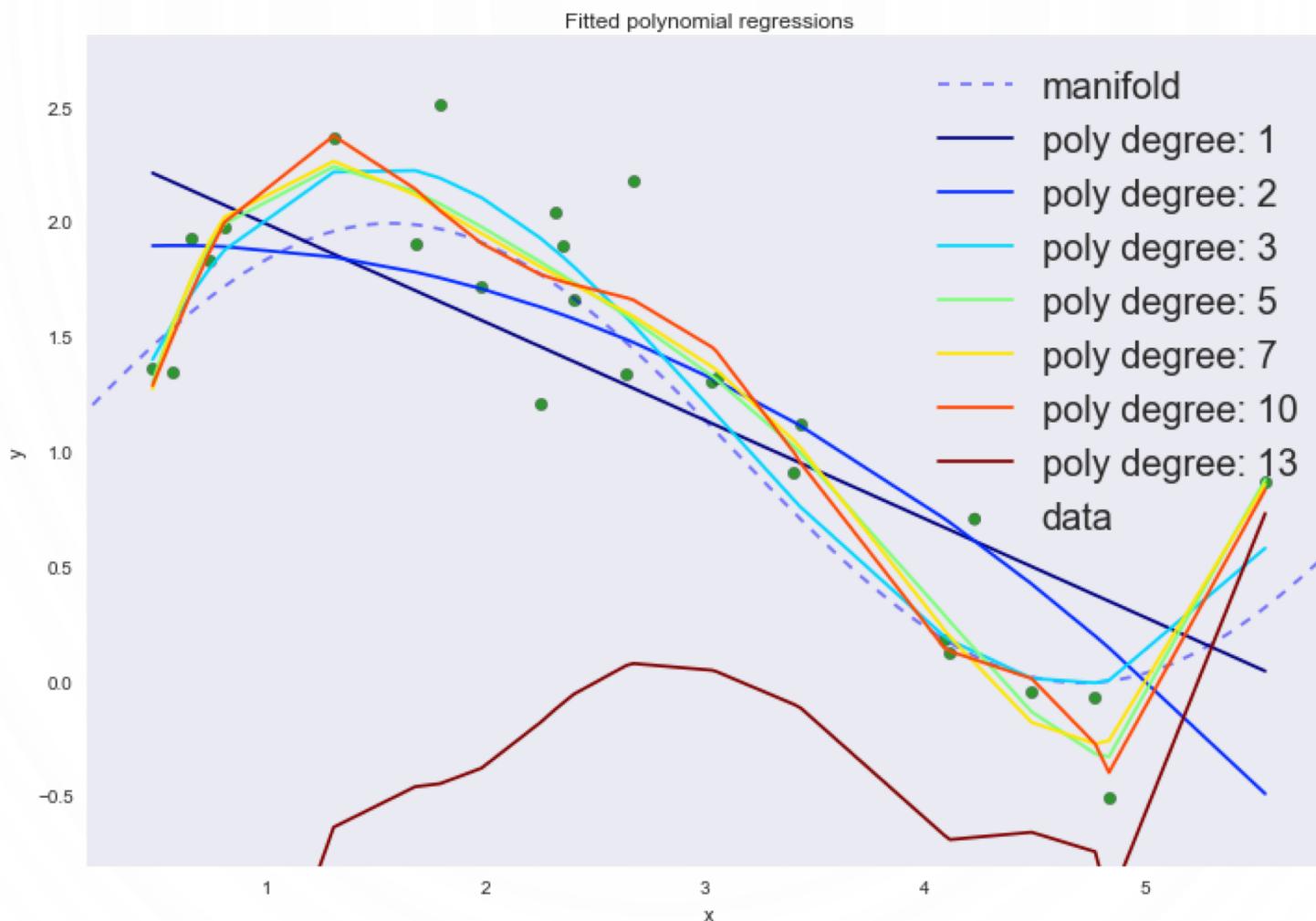
Fitting training data



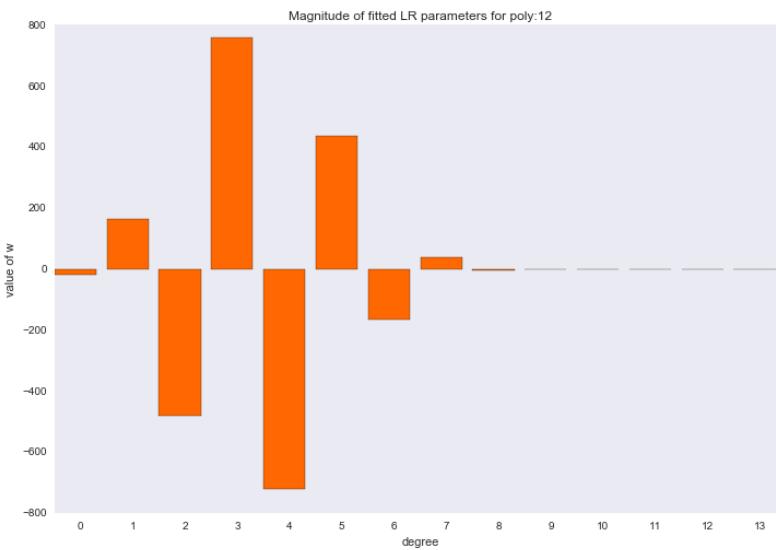
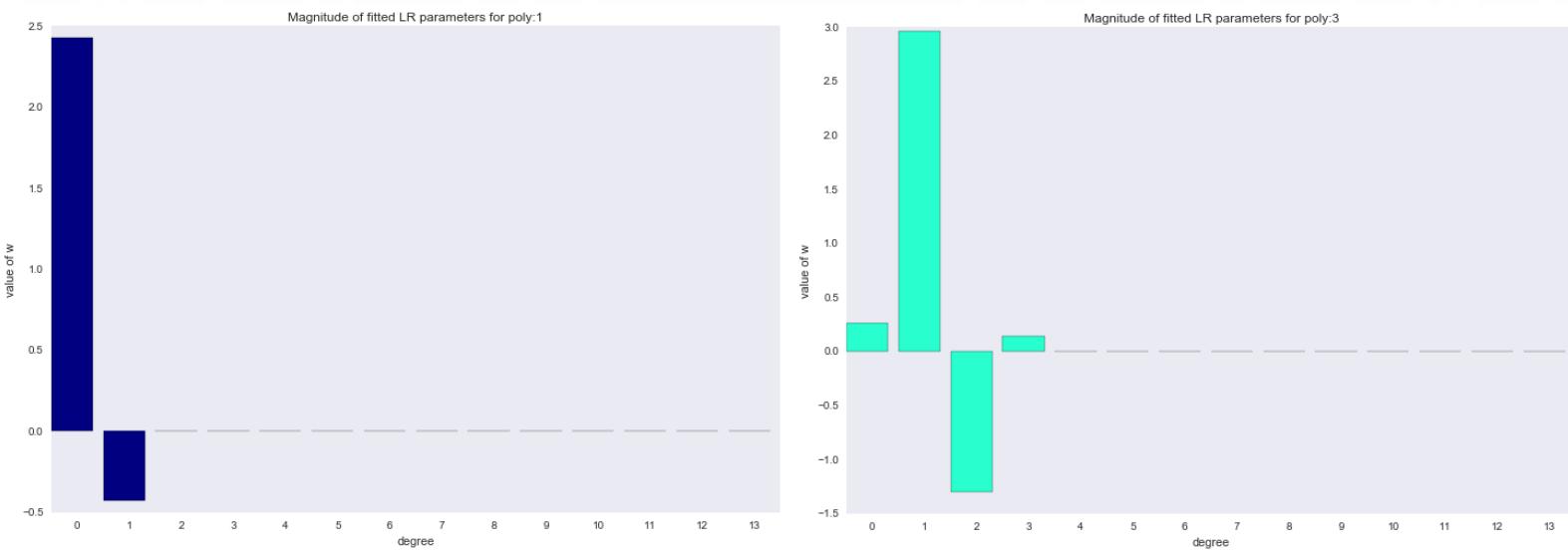
ПРИЗНАКИ ПЕРЕОБУЧЕННОЙ МОДЕЛИ

- Большая разница в качестве на тренировочных и тестовых данных (модель подгоняется под тренировочные данные и не может найти истинную зависимость)
- Большие значения параметров (весов) w_j модели
- Неустойчивость дискриминантной (разделяющей) функции (w, x).

ПЕРЕОБУЧЕНИЕ: ПРИМЕР



ПЕРЕОБУЧЕНИЕ: ПРИМЕР



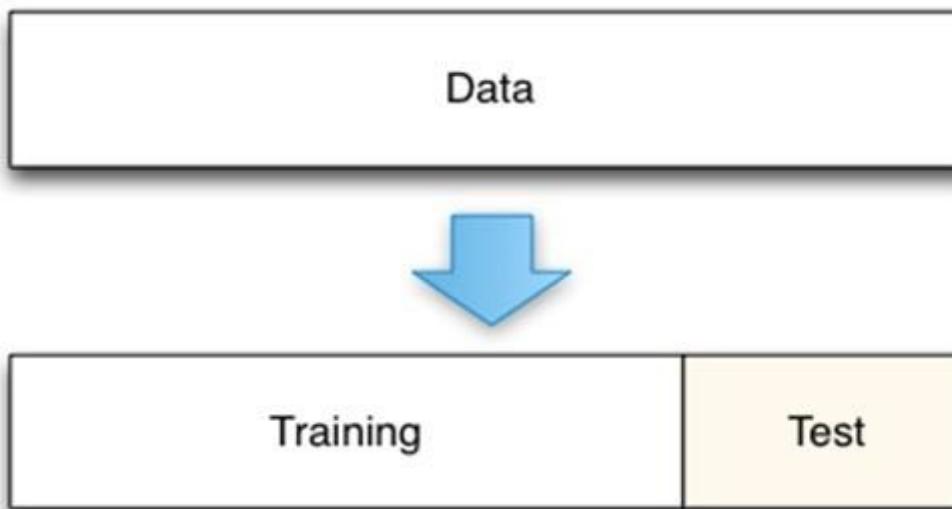
ОЦЕНИВАНИЕ КАЧЕСТВА МОДЕЛИ

- Отложенная выборка
- Кросс-валидация

ОТЛОЖЕННАЯ ВЫБОРКА

Делим тренировочную выборку на две части:

- По первой части обучаем модель (*train*)
- По оставшимся данным – оцениваем качество (*test*)

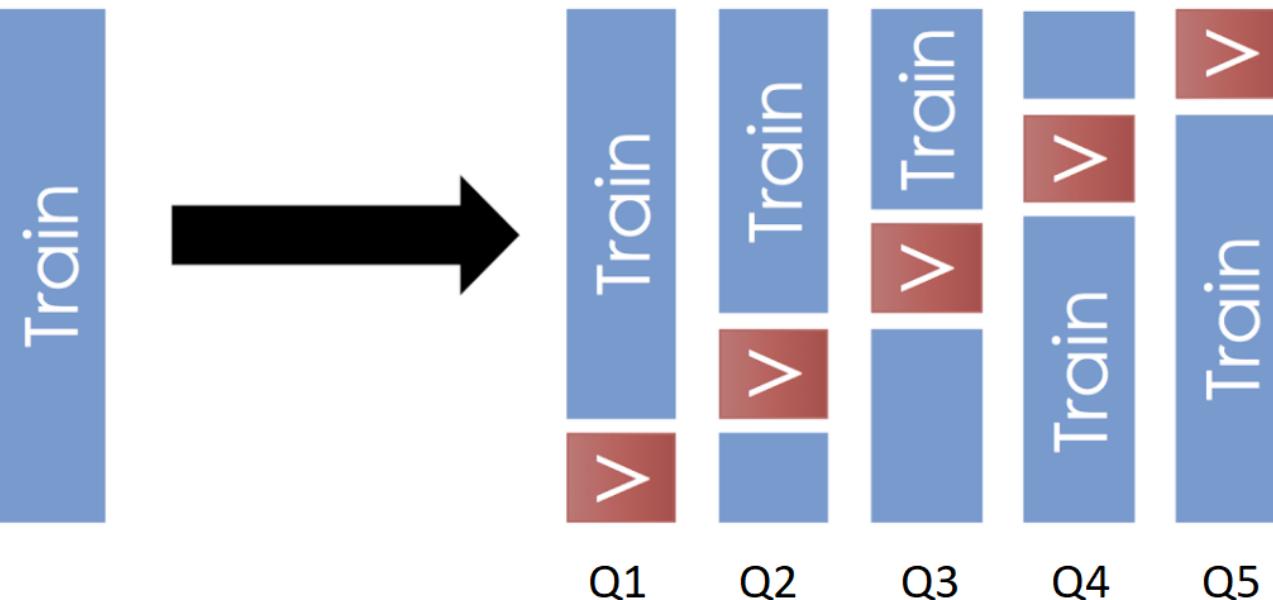


Недостаток:

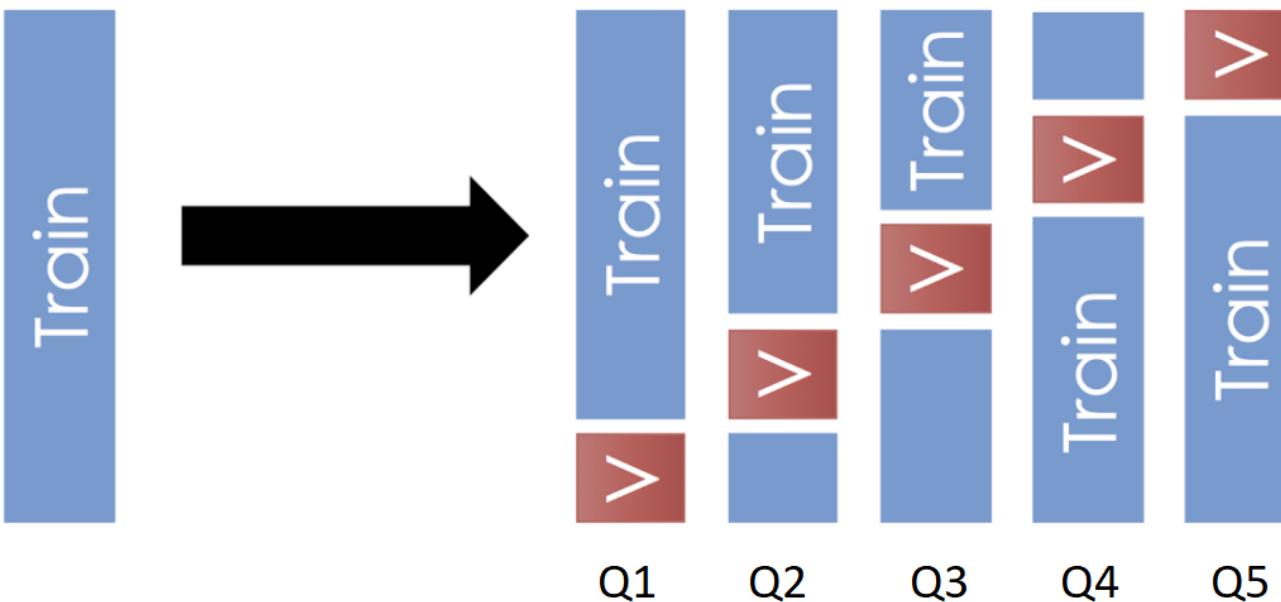
- Результат сильно зависит от разбиения на *train* и *test*

КРОСС-ВАЛИДАЦИЯ

- Разбиваем объекты на тренировку (train) и валидацию (validation) несколько раз (при разбиении k раз получаем k-fold кросс-валидацию)
- Для каждого разбиения вычисляем качество на валидационной части
- Усредняем полученные результаты



КРОСС-ВАЛИДАЦИЯ



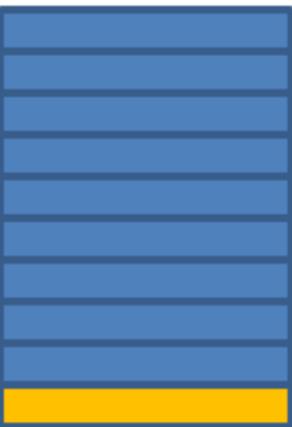
$$CV = \frac{1}{k} \sum_{i=1}^k Q_i(a_i(x), X_i) = \frac{1}{k} \sum_{i=1}^k Q_i$$

ВИДЫ КРОСС-ВАЛИДАЦИИ

- **k-fold cross-validation** – разбиваем данные на k блоков, каждый из которых по очереди становится контрольным (валидационным)
- **Complete cross-validation** – перебираем ВСЕ разбиения
- **Leave-one-out cross-validation** – каждый блок состоит из одного объекта (число блоков = числу объектов)

ВЫБОР КОЛИЧЕСТВА БЛОКОВ В K-FOLD КРОСС-ВАЛИДАЦИИ

 Validation Set
 Training Set



$k = 10$



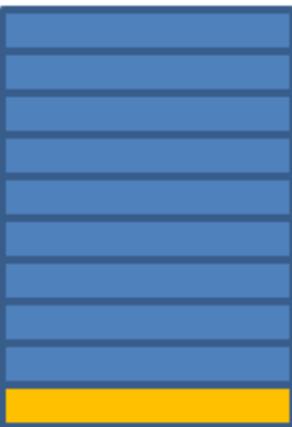
$k = 2$

• Проблемы при маленьком k ?

• Проблемы при большом k ?

ВЫБОР КОЛИЧЕСТВА БЛОКОВ В K-FOLD КРОСС-ВАЛИДАЦИИ

 Validation Set
 Training Set



$k = 10$



$k = 2$

- Маленькое k – оценка может быть пессимистично занижена из-за маленького размера тренировочной части
- Большое k – оценка может иметь большую дисперсию из-за маленького размера валидационной части

МЕТОД БОРЬБЫ С ПЕРЕОБУЧЕНИЕМ: РЕГУЛЯРИЗАЦИЯ

Утверждение. Если в выборке есть линейно-зависимые признаки, то задача оптимизации $Q(w) \rightarrow \min$ имеет бесконечное число решений.

- Большие значения параметров (весов) модели w – признак переобучения.

МЕТОД БОРЬБЫ С ПЕРЕОБУЧЕНИЕМ: РЕГУЛЯРИЗАЦИЯ

Утверждение. Если в выборке есть линейно-зависимые признаки, то задача оптимизации $Q(w) \rightarrow \min$ имеет бесконечное число решений.

- Большие значения параметров (весов) модели w – признак переобучения.

Решение проблемы – *регуляризация*.

Будем минимизировать регуляризованный функционал ошибки:

$$Q_{\alpha}(w) = Q(w) + \alpha \cdot R(w) \rightarrow \min_w ,$$

где $R(w)$ – регуляризатор.

РЕГУЛЯРИЗАЦИЯ

- Регуляризация штрафует за слишком большие веса.

Наиболее используемые регуляризаторы:

- L_2 -регуляризатор: $R(w) = \|w\|_2 = \sum_{i=1}^d w_i^2$
- L_1 -регуляризатор: $R(w) = \|w\|_1 = \sum_{i=1}^d |w_i|$

РЕГУЛЯРИЗАЦИЯ

- Регуляризация штрафует за слишком большие веса.

Наиболее используемые регуляризаторы:

- L_2 -регуляризатор: $R(w) = \|w\|_2 = \sum_{i=1}^d w_i^2$
- L_1 -регуляризатор: $R(w) = \|w\|_1 = \sum_{i=1}^d |w_i|$

Пример регуляризованного функционала:

$$Q(a(w), X) = \frac{1}{l} \sum_{i=1}^l ((w, x_i) - y_i)^2 + \alpha \sum_{i=1}^d w_i^2,$$

где α – коэффициент регуляризации.

АНАЛИТИЧЕСКОЕ РЕШЕНИЕ ЗАДАЧИ МНК С L_2 -РЕГУЛЯРИЗАТОРОМ

Задача оптимизации в матричном виде:

$$Q(w) = (y - Xw)^T(y - Xw) + \alpha w^T I w \rightarrow \min \quad (*)$$

где I – единичная матрица.

Эта задача имеет аналитическое решение:

$$\mathbf{w} = (X^T X + \alpha I)^{-1} X^T y$$

- Матрица $X^T X + \alpha I$ всегда положительно определена, поэтому её можно обратить. Следовательно, задача (*) имеет единственное решение.

ПОЛЕЗНОЕ СВОЙСТВО L1-РЕГУЛЯРИЗАЦИИ

Все ли признаки в задаче нужны?

- Некоторые признаки могут не иметь отношения к задаче, т.е. они не нужны.
- Если есть ограничения на скорость получения предсказаний, то чем меньше признаков, тем быстрее
- Если признаков больше, чем объектов, то решение задачи будет неоднозначным.

Поэтому в таких случаях надо делать отбор признаков, то есть убирать некоторые признаки.

L_1 -РЕГУЛЯРИЗАЦИЯ

Утверждение. В результате обучения модели с L_1 -регуляризатором происходит зануление некоторых весов, т.е. отбор признаков.

Можно показать, что задачи

$$(1) \quad Q(w) + \alpha \|w\|_1 \rightarrow \min_w$$

и

$$(2) \quad \begin{cases} Q(w) \rightarrow \min_w \\ \|w\|_1 \leq c \end{cases}$$

эквивалентны.

РАЗРЕЖЕННЫЕ МОДЕЛИ

Модели, в которых часть весов равна 0, называются ***разреженными моделями***.

- L1-регуляризация зануляет часть весов, то есть делает модель разреженной.

3. ПРАКТИЧЕСКАЯ ЧАСТЬ

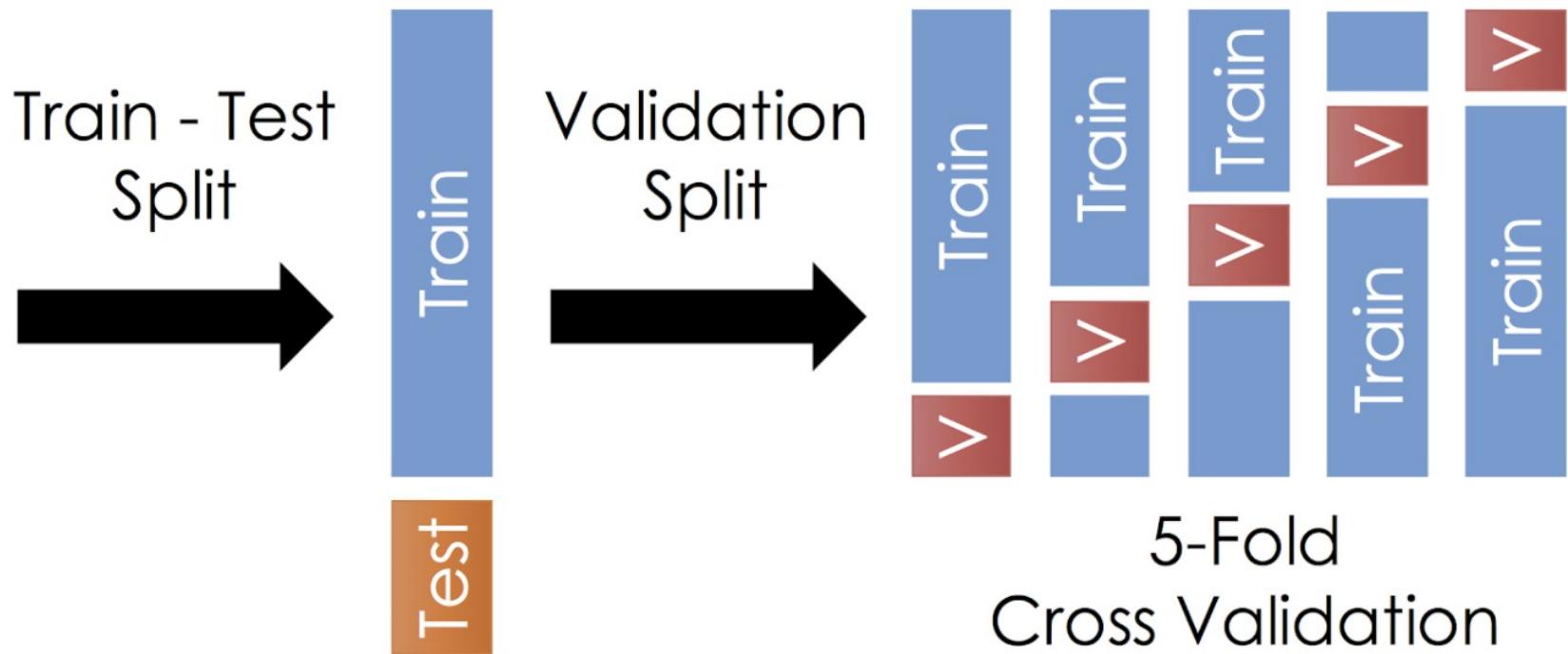
- Параметры и гиперпараметры модели
- Работа с признаками: способы кодирования категориальных признаков

ГИПЕРПАРАМЕТРЫ МОДЕЛИ

- **Параметры модели** – величины, настраивающиеся по обучающей выборке (например, веса w в линейной регрессии)
- **Гиперпараметры модели** – величины, контролирующие процесс обучения. Поэтому они не могут быть настроены по обучающей выборке (например, коэффициент регуляризации α).

Проблема: если подбирать гиперпараметры по кросс-валидации, то мы будем использовать отложенную (валидационную) выборку для поиска наилучших значений гиперпараметров. Т.е. отложенная выборка становится обучающей.

СХЕМА РАЗБИЕНИЯ ДАННЫХ ДЛЯ ПОДБОРА ПАРАМЕТРОВ И ГИПЕРПАРАМЕТРОВ МОДЕЛИ



КОДИРОВАНИЕ КАТЕГОРИАЛЬНЫХ ПРИЗНАКОВ: ONE-HOT ENCODING

- Предположим, категориальный признак $f_j(x)$ принимает m различных значений: C_1, C_2, \dots, C_m .

Пример: еда может быть *горькой*, *сладкой*, *солёной* или *кислой* (4 возможных значения признака).

КОДИРОВАНИЕ КАТЕГОРИАЛЬНЫХ ПРИЗНАКОВ: ONE-HOT ENCODING

- Предположим, категориальный признак $f_j(x)$ принимает m различных значений: C_1, C_2, \dots, C_m .

Пример: еда может быть *горькой*, *сладкой*, *солёной* или *кислой* (4 возможных значения признака).

- Заменим категориальный признак на m бинарных признаков: $b_i(x) = [f_j(x) = C_i]$ (индикатор события).

Тогда One-Hot кодировка для нашего примера будет следующей:

горький = $(1, 0, 0, 0)$, *сладкий* = $(0, 1, 0, 0)$,

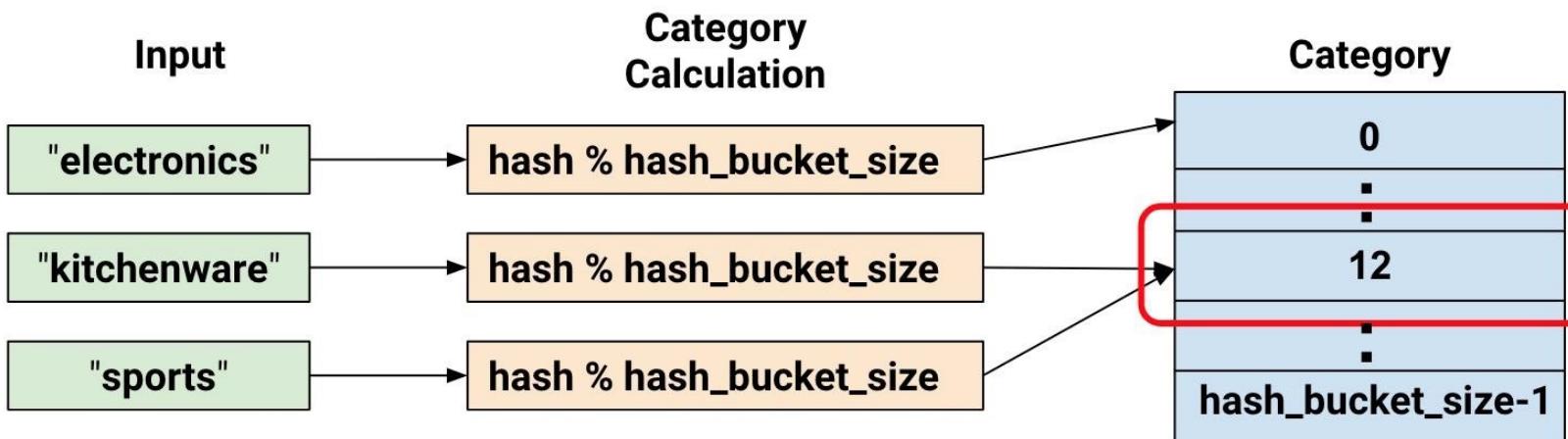
солёный = $(0, 0, 1, 0)$, *кислый* = $(0, 0, 0, 1)$.

ХЭШИРОВАНИЕ ПРИЗНАКОВ

- Если у категориального признака слишком много значений, скажем, миллион, то после применения one-hot кодировки мы получим миллион новых столбцов. С такой огромной матрицей тяжело работать.
- Хэширование развивает идею one-hot кодирования, но позволяет получать любое заранее заданное число новых числовых столбцов после кодировки.

АЛГОРИТМ ХЭШИРОВАНИЯ

- 1) Для каждого значения признака вычисляем значение некоторой функции – хэш-функции (hash)
- 2) Задаем `hash_bucket_size` – итоговое количество различных значений категориального признака.
- 3) Берем остаток: $\text{hash} \% \text{hash_bucket_size}$ – тем самым кодируем каждое значение признака числом от 0 до `hash_bucket_size-1`.
- 4) Дальше к полученным числам применяем ОНЕ.

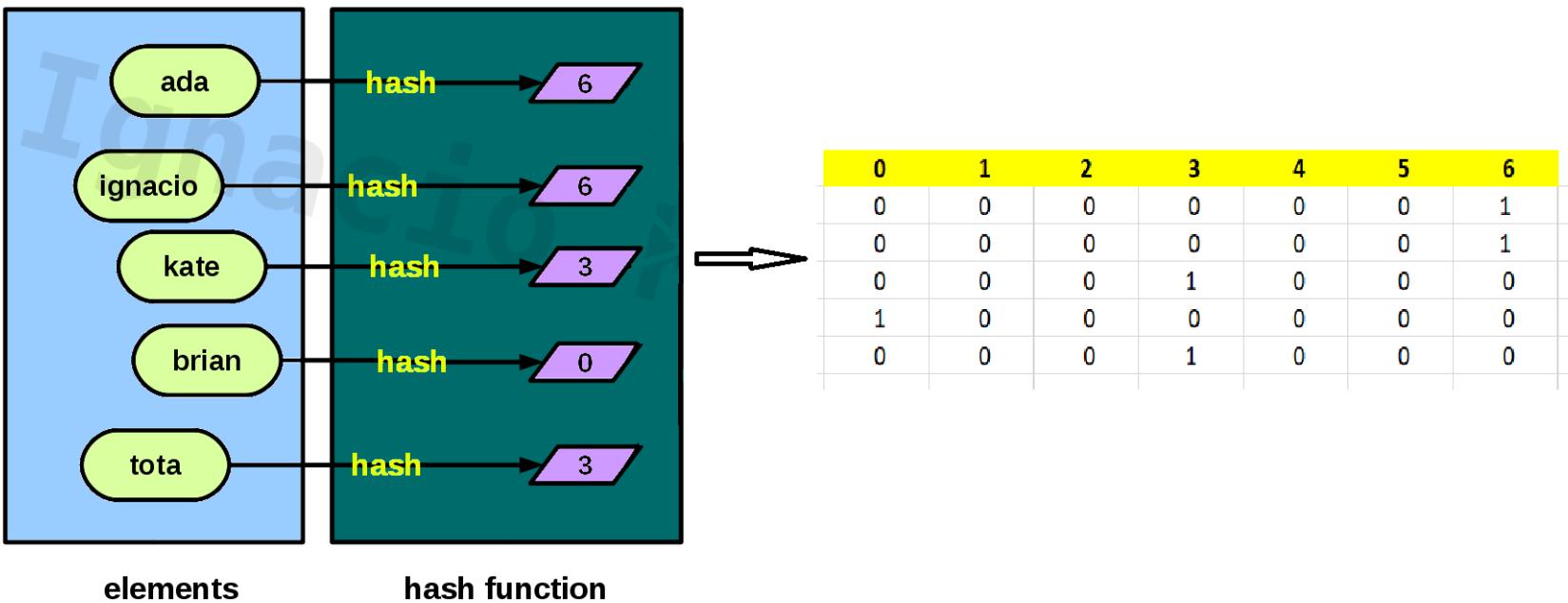


ЧТО ДЕЛАЕТ ХЭШ-ФУНКЦИЯ

Идея: хэш-функция группирует значения категориального признака:

- часто встречающиеся значения признака формируют отдельные группы
- редко встречающиеся значения попадают в одну группу при группировке

ХЭШИРОВАНИЕ ПРИЗНАКОВ: ПРИМЕР



ХЭШИРОВАНИЕ

- Хэширование – это способ кодирования категориальных данных, принимающих множество различных значений, показывающий хорошие результаты на практике.
- Хэширование позволяет закодировать любое значение категориального признака (в том числе то, которого не было в тренировочной выборке).

Статья про хэширование:

<https://arxiv.org/abs/1509.05472>