



# Лекция 3 Линейные методы регрессии. Часть 2.

Блуменау М.И.

На основе материалов Кантоnistовой Е.О.

ВШЭ, 2025

# 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

Среднеквадратичное отклонение:

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

Плюсы:

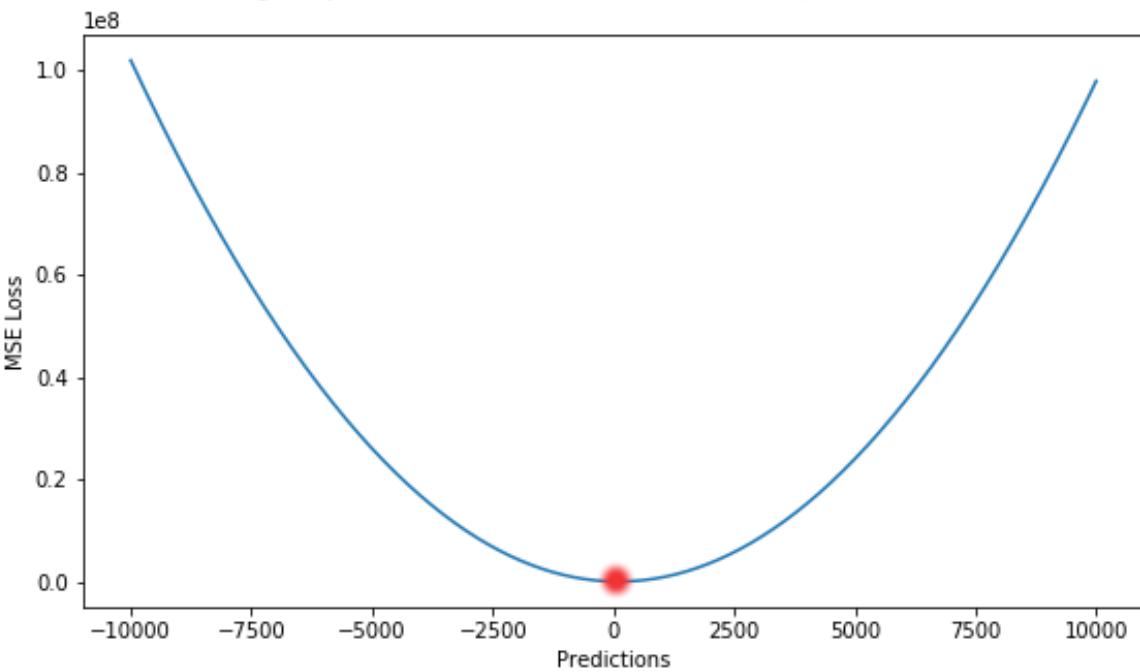
- Позволяет сравнивать модели
- Подходит для контроля качества во время обучения

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.

# MSE

Range of predicted values: (-10,000 to 10,000) | True value: 100



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

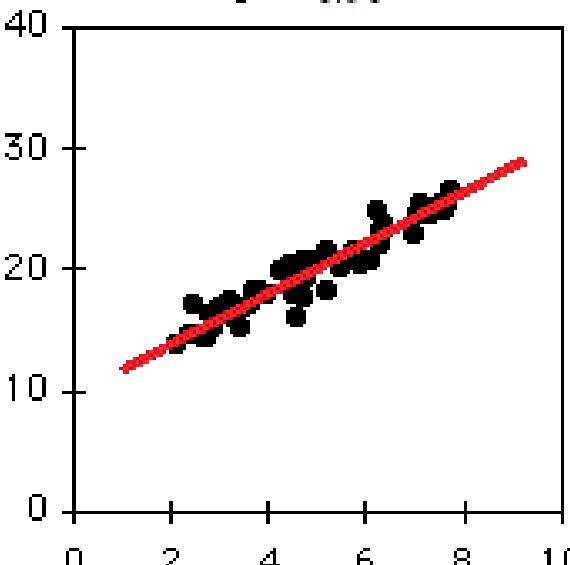
Коэффициент детерминации это доля дисперсии целевой переменной, объясняемая моделью.

- 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

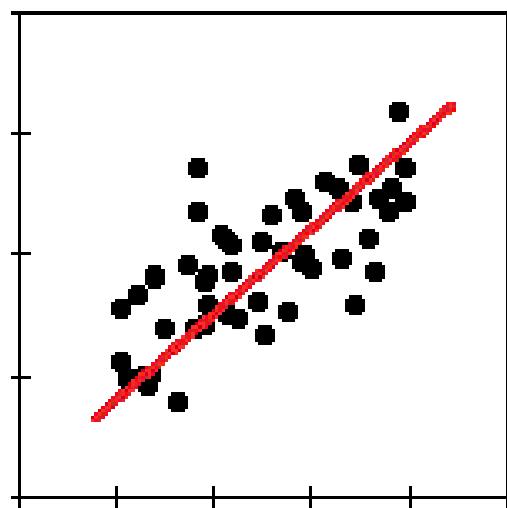
# КОЭФФИЦИЕНТ ДЕТЕРМИНАЦИИ ( $R^2$ )

$$R^2 \leq 1$$

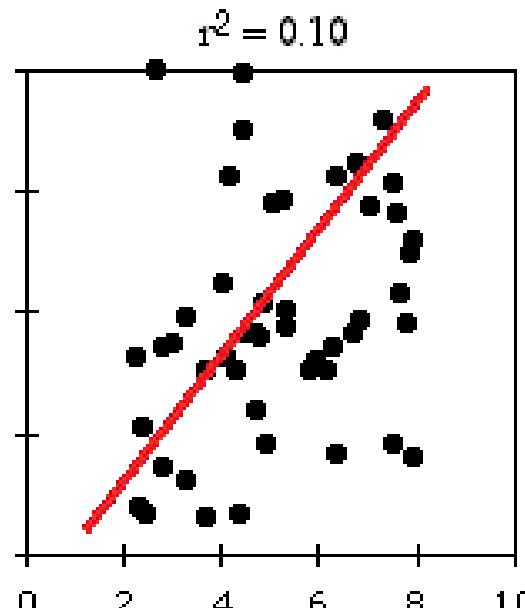
$r^2 = 0.90$



$r^2 = 0.50$



$r^2 = 0.10$



# 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

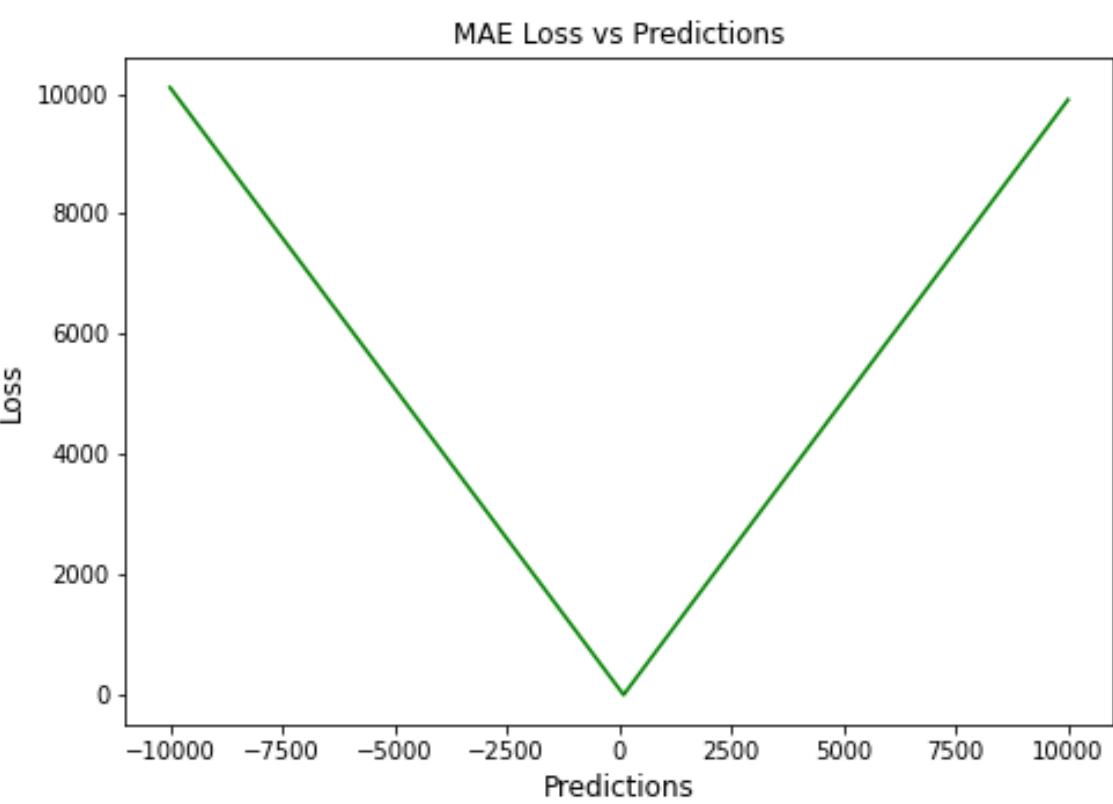
Минусы:

- MAE - не дифференцируемый функционал

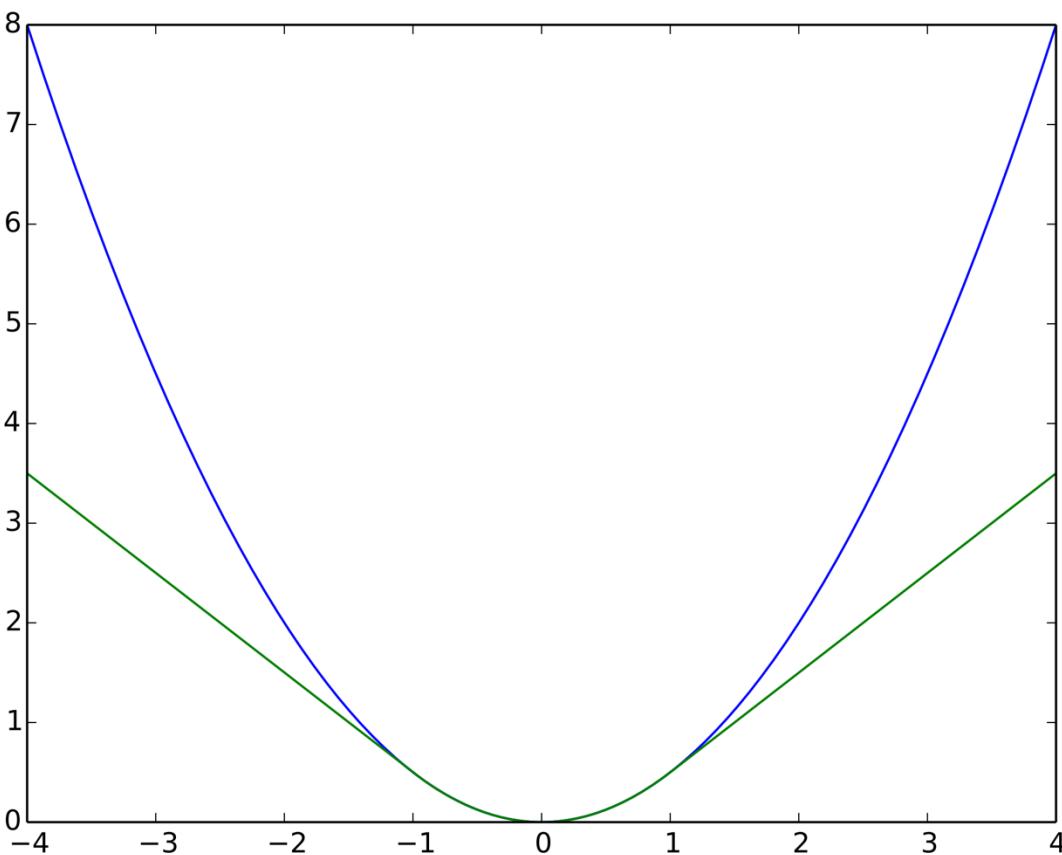
Сомнительные плюсы:

- Минус можно подлечить (Huber Loss?)

# MAE



# HUBER LOSS



# 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:

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

2. Оптимум MAE достигается на медиане ответов:

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

# MSLE (MEAN SQUARED LOGARITHMIC ERROR)

Среднеквадратичная логарифмическая ошибка:

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

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

# MAPE

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

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).

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).

*“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.”*

# 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  $\tau$ , 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  $\tau$ -квантиль распределения ответов в каждой точке  $x \in X$ .

# МЕТРИКИ: ОНЛАЙН, ОФЛАЙН, БИЗНЕС

Бизнес

Показатели бизнеса

Например:

- Lifetime value
- Прибыль
- Расходы
- Доля аудитории
- Цена акций

Мы хотели бы смотреть, как модель влияет на них, но не можем

Измеряются месяцами

Метрики

Онлайн

Оффлайн

Связаны с показателями бизнеса  
Можно сделать быстрый тест

Например:

- Конверсия в клик
- Оценка сервиса
- Средний чек
- MAU, DAU, WAU

Мы можем оценить эти метрики, проведя А/В-тест

Измеряются неделями

Являются приближением  
онлайн-метрик

Считываются на исторических  
данных

Например:

- Precision, recall
- Accuracy

Считываются минуты-часы

Можем почти бесплатно  
проверить наши модели

# СВОЙСТВА МЕТРИК

- Чувствительность
- Шум
- Интерпретация
- Иерархия

# ПРИМЕР ИЕРАРХИИ МЕТРИК

- Хотим внедрить новое ML-ранжирование рекомендаций товаров
- Находимся в ситуации, когда этот элемент уже есть на сайте

Ваша подборка для покупок у нас



Что измеряем?

# ИЕРАРХИЯ МЕТРИК

Бизнес-  
метрика

Онлайн-  
метрики

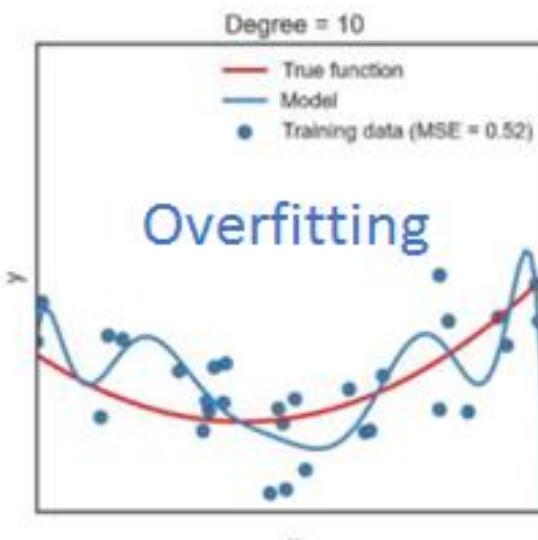
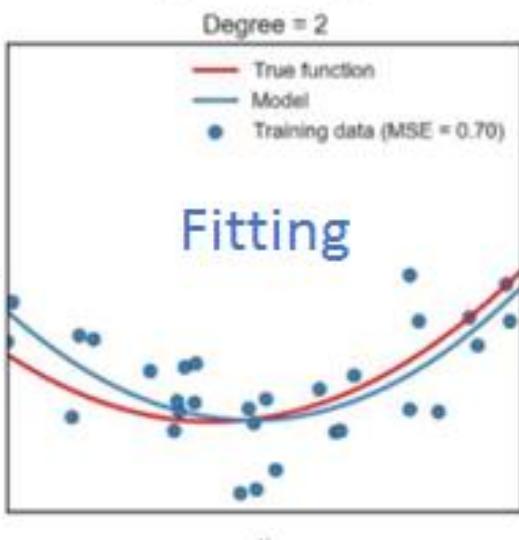
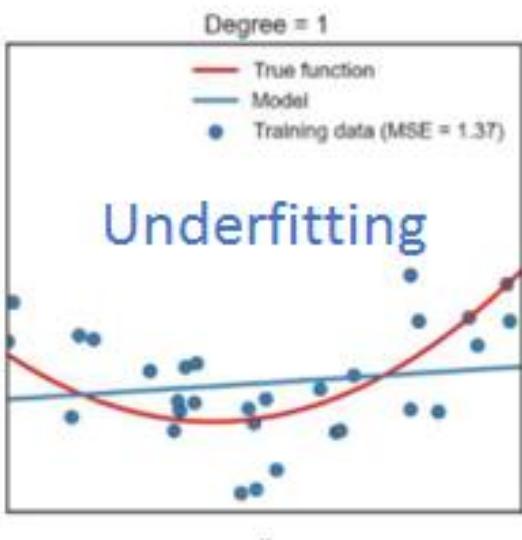
Оффлайн-  
метрики

- Выручка
- Средний чек / Число купивших пользователей
- Выручка проданных товаров, через наш элемент
- CTR элемента
- Оффлайн метрики ранжирования
- Accuracy на валидационной выборке

# ОЦЕНКА ОБОБЩАЮЩЕЙ СПОСОБНОСТИ МОДЕЛИ

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

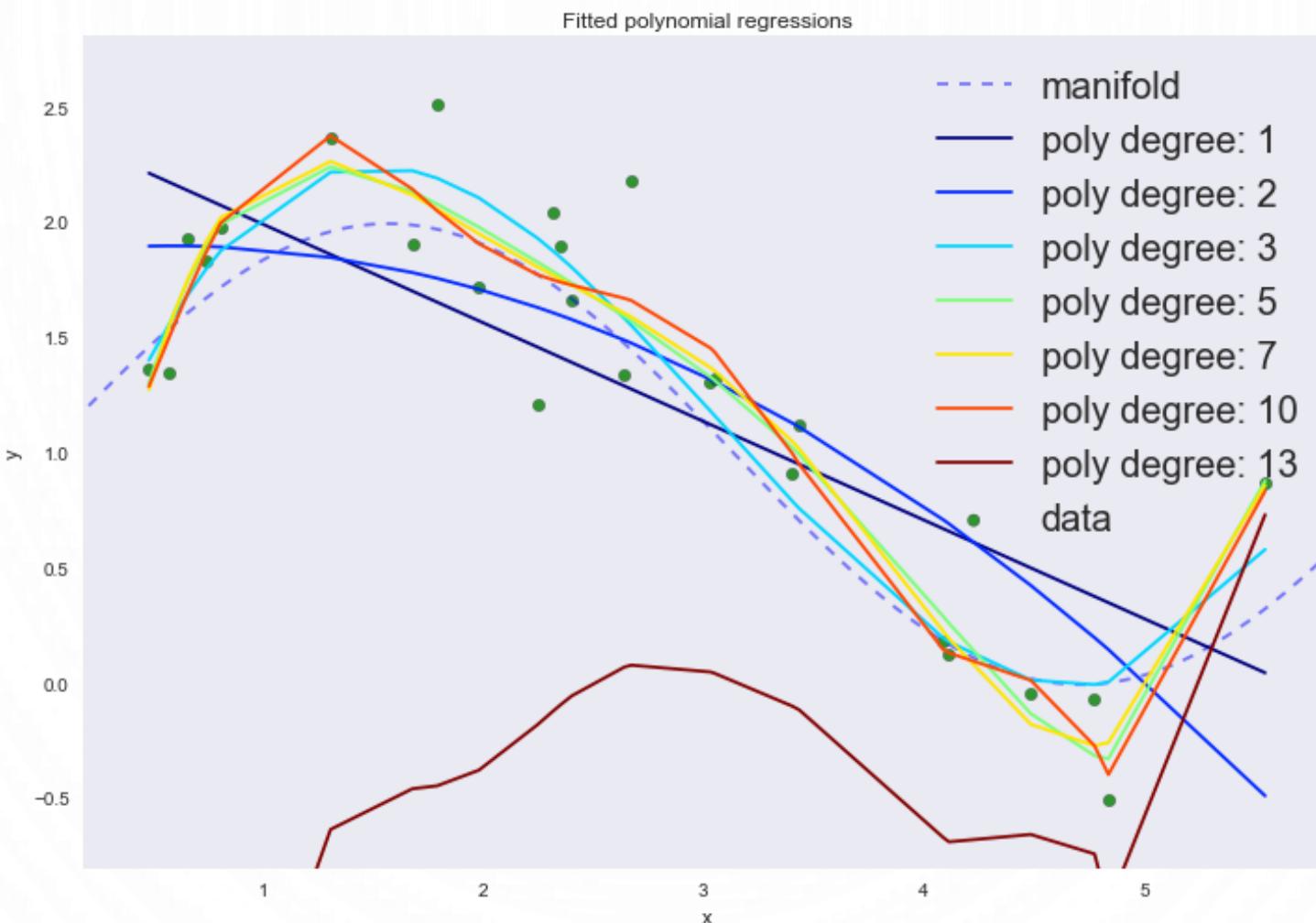
Fitting training data



# 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



# ОТЛОЖЕННАЯ ВЫБОРКА

Делим тренировочную выборку на две части:

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

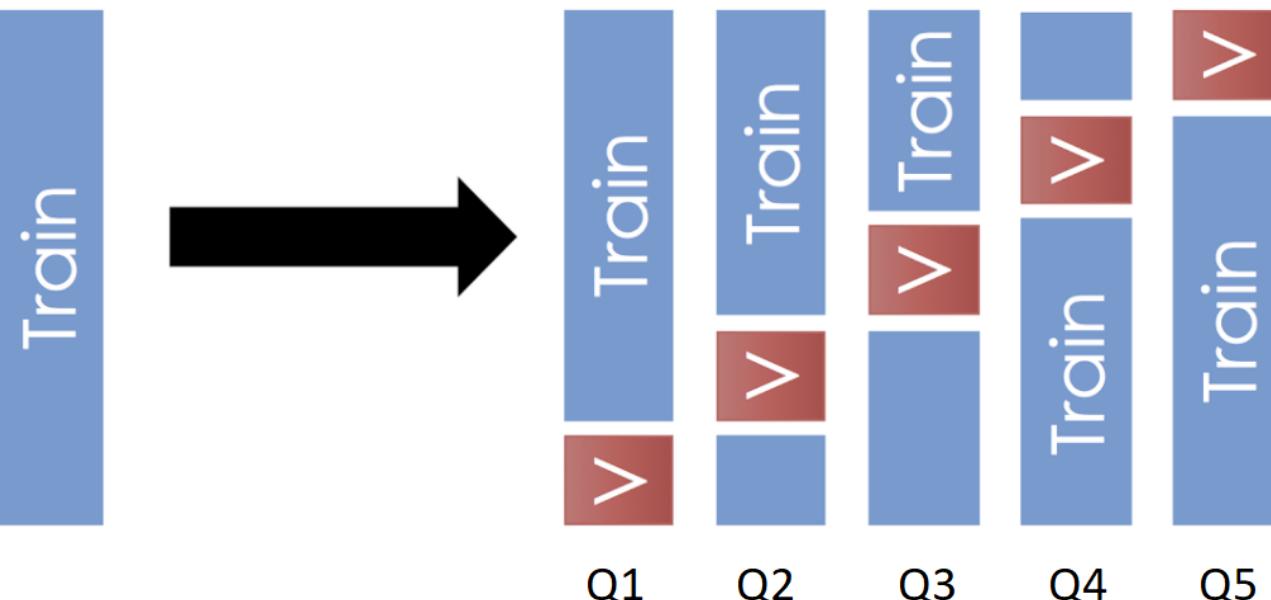


Недостаток:

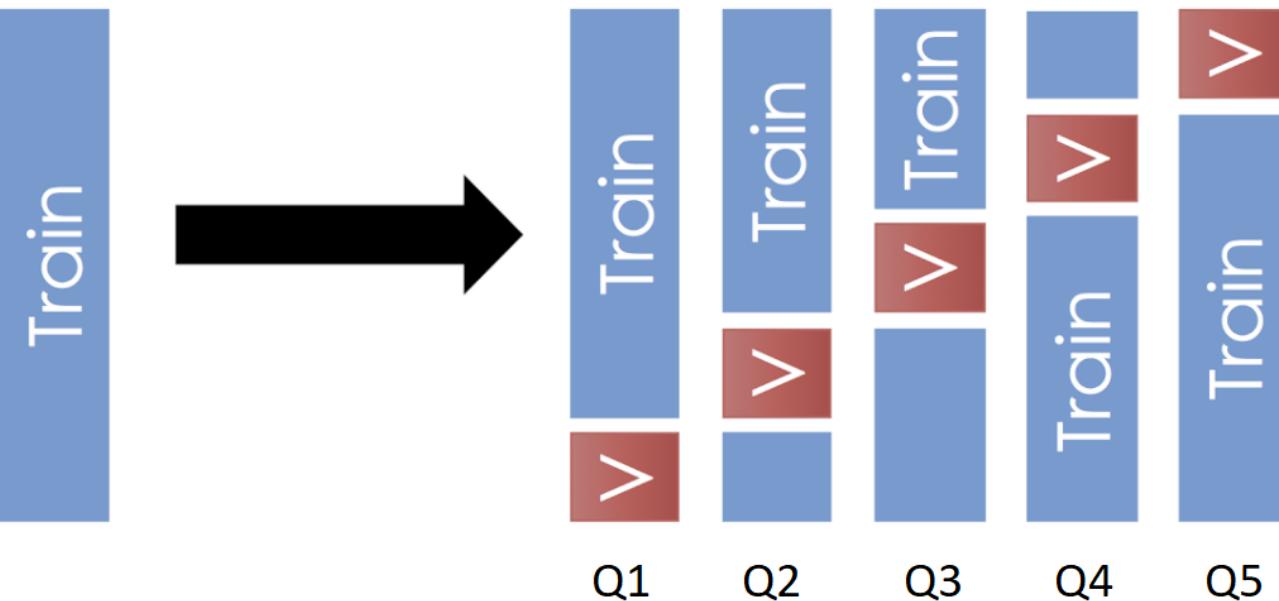
- Результат сильно зависит от разбиения на *train* и *test*

# 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



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

# 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)

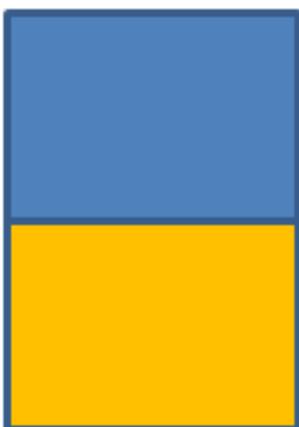
$$C_n^k = \frac{n!}{(n-k)! k!}$$

# 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:  $R(w) = \|w\|_1 = \sum_{i=1}^d |w_i|$

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  $\alpha$  – 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:

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

- 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

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

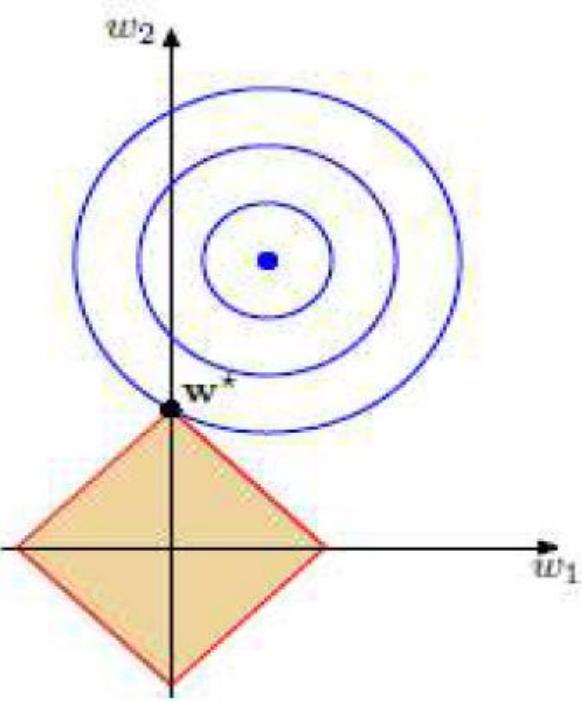
и

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

эквивалентны.

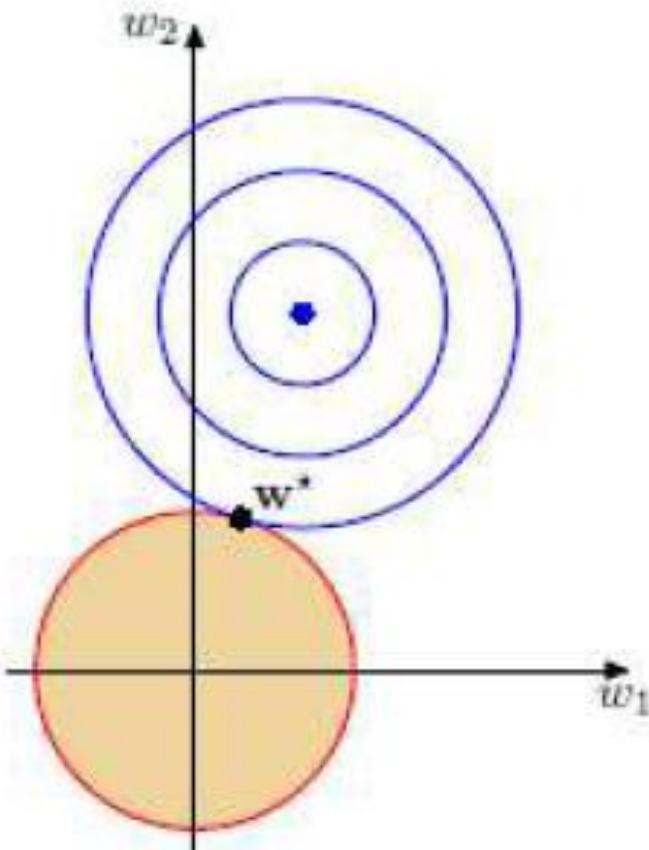
# ОТБОР ПРИЗНАКОВ ПО L1-РЕГУЛЯРИЗАЦИИ

Нарисуем линии уровня  $Q(w)$  и область  $\|w\|_1 \leq C$ :



Если признак незначимый, то соответствующий вес близок к 0. Отсюда получим, что в большинстве случаев решение нашей задачи попадает в вершину ромба, т.е. обнуляет незначимый признак.

# L2-РЕГУЛЯРИЗАЦИЯ НЕ ОБНУЛЯЕТ ПРИЗНАКИ



# РАЗРЕЖЕННЫЕ МОДЕЛИ

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

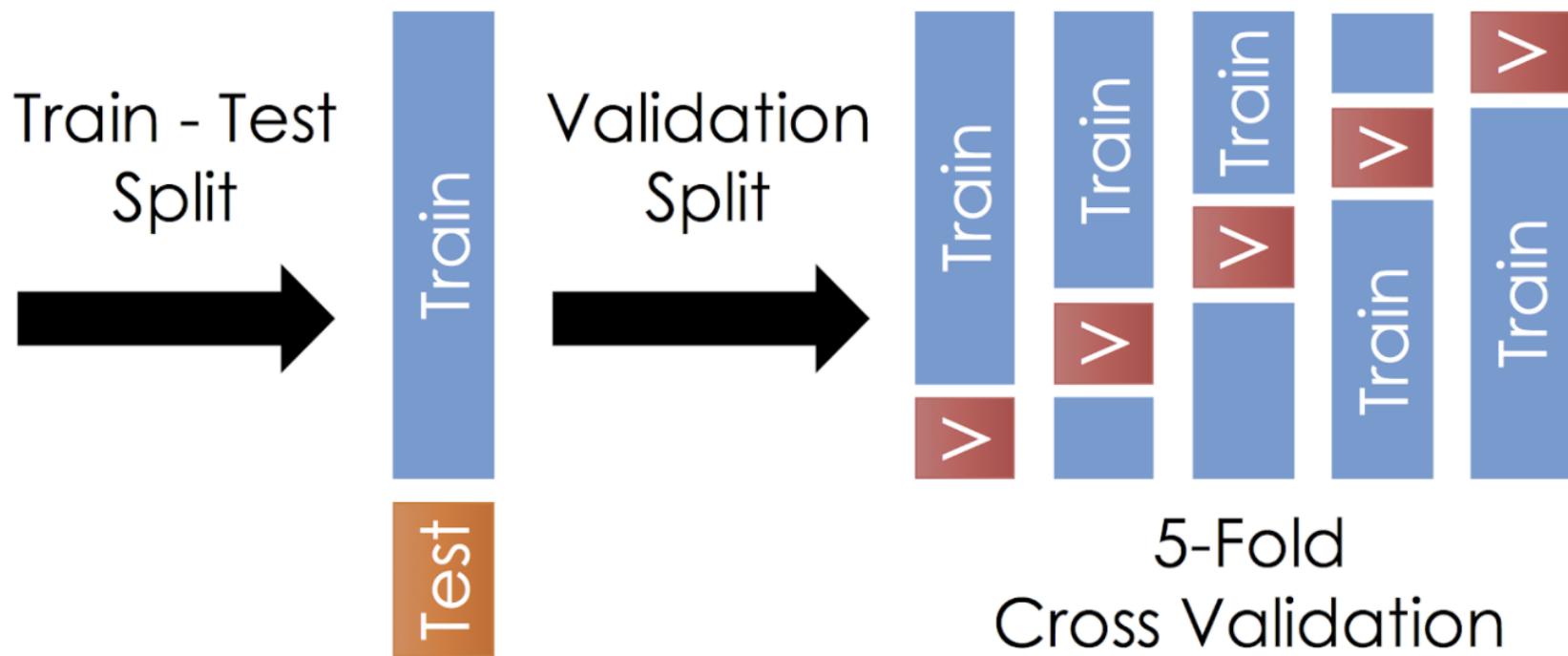
- L1-регуляризация зануляет часть весов, то есть делает модель разреженной.

# 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  $\alpha$ ).

*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  $t$  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  $t$  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* =  $(0, 1, 0, 0)$ ,

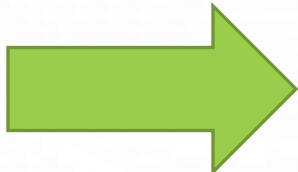
*солёный* =  $(0, 0, 1, 0)$ , *кислый* =  $(0, 0, 0, 1)$ .

# СЧЁТЧИКИ

**Счётчик (*mean target encoding*)** – это вероятность получить значение целевой переменной для данного значения категориального признака.

# СЧЁТЧИКИ (ПРИМЕР)

	feature	target
0	Moscow	0
1	Moscow	1
2	Moscow	1
3	Moscow	0
4	Moscow	0
5	Tver	1
6	Tver	1
7	Tver	1
8	Tver	0
9	Klin	0
10	Klin	0
11	Tver	1



	feature	feature_mean	target
0	Moscow	0.4	0
1	Moscow	0.4	1
2	Moscow	0.4	1
3	Moscow	0.4	0
4	Moscow	0.4	0
5	Tver	0.8	1
6	Tver	0.8	1
7	Tver	0.8	1
8	Tver	0.8	0
9	Klin	0.0	0
10	Klin	0.0	0
11	Tver	0.8	1

# СЧЁТЧИКИ В ЗАДАЧЕ БИНАРНОЙ КЛАССИФИКАЦИИ

В случае бинарной классификации счётчики можно задать формулой:

$$Likelihood = \frac{Goods}{Goods + Bads} = mean(target),$$

где *Goods* – число единиц в столбце *target*,  
*Bads* – число нулей в столбце *target*.

# СЧЁТЧИКИ (ОБЩАЯ ФОРМУЛА)

$$counts(u, X) = \sum_{(x,y) \in X} [f(x) = u]$$

$$successes_k(u, X) = \sum_{(x,y) \in X} [f(x) = u][y = k], k = 1, \dots, K$$

Тогда кодировка:

$$mean\_target_k(x, X) = \frac{successes_k(f(x), X)}{counts(f(x), X)}$$

*Недостаток? Когда такой способ кодирования  
переобучит наш алгоритм?*

*Ответ: если в данных много редких категорий.*

# СЧЁТЧИКИ + СГЛАЖИВАНИЕ

Используем счётчики (mean target encoding) со  
сглаживанием:

$$\frac{\text{mean}(\text{target}) \cdot n_{\text{rows}} + \text{global mean} \cdot \alpha}{n_{\text{rows}} + \alpha},$$

$n_{\text{rows}}$  - количество строк в категории,

$\alpha$  – параметр регуляризации.

# СЧЁТЧИКИ: ОПАСНОСТЬ ПЕРЕОБУЧЕНИЯ

*Вычисляя счётчики, мы закладываем в признаки информацию о целевой переменной и, тем самым, переобучаемся!*

# СЧЁТЧИКИ: КАК ВЫЧИСЛЯТЬ

- Можно вычислять счётчики так:

city	target	
Moscow	1	Вычисляем счетчики по этой части
London	0	
London	2	
Kiev	1	
Moscow	1	
Moscow	0	Кодируем признак вычисленными счётчиками и обучаемся по этой части
Kiev	0	
Moscow	2	

# СЧЁТЧИКИ: КАК ВЫЧИСЛЯТЬ

Более продвинутый способ (по кросс-валидации):

1) Разбиваем выборку

на  $m$  частей  $X_1, \dots, X_m$

2) На каждой части  $X_i$

значения признаков

вычисляются по

оставшимся частям:

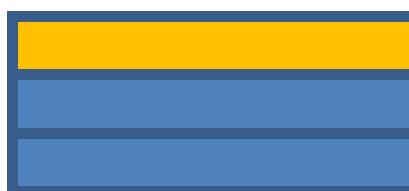
$$x \in X_i \Rightarrow g_k(x) = g_k(x, X \setminus X_i)$$



`clf1`



`clf2`



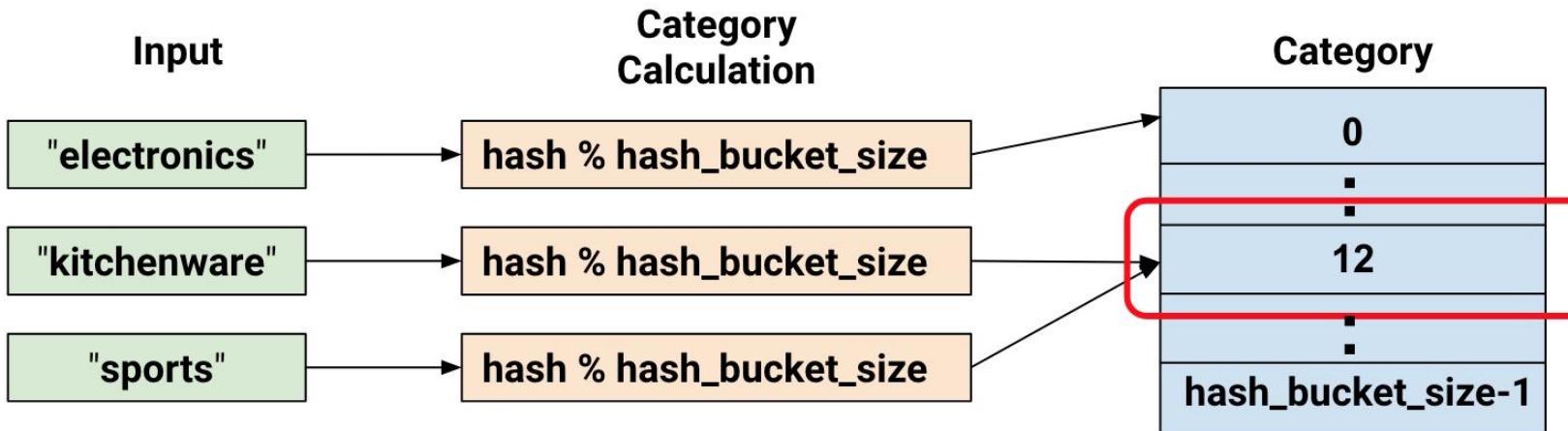
`clf3`

# 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.

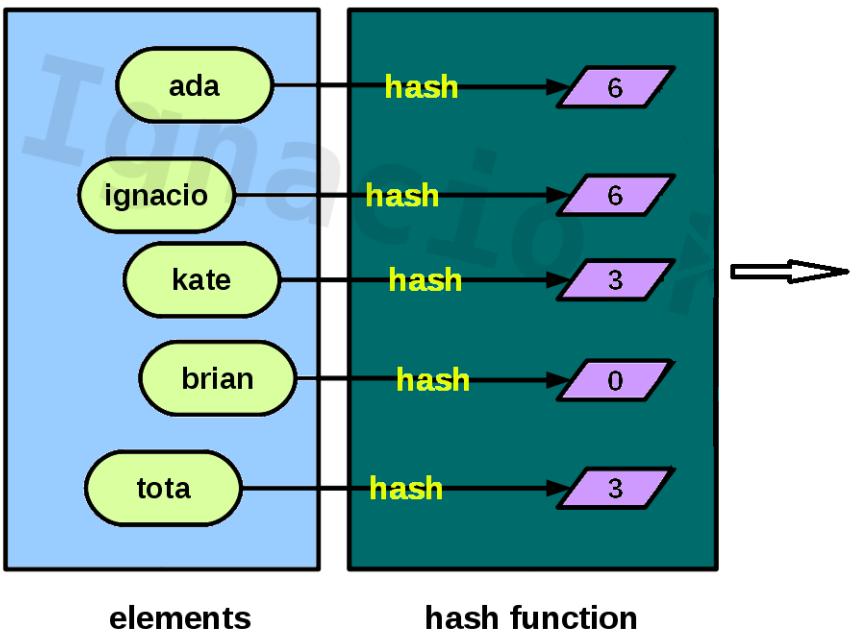


# ЧТО ДЕЛАЕТ ХЭШ-ФУНКЦИЯ

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



0	1	2	3	4	5	6
0	0	0	0	0	0	1
0	0	0	0	0	0	1
0	0	0	1	0	0	0
1	0	0	0	0	0	0
0	0	0	1	0	0	0

# 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>