Переобучение и регуляризация. Кодирование категориальных признаков

Елена Кантонистова

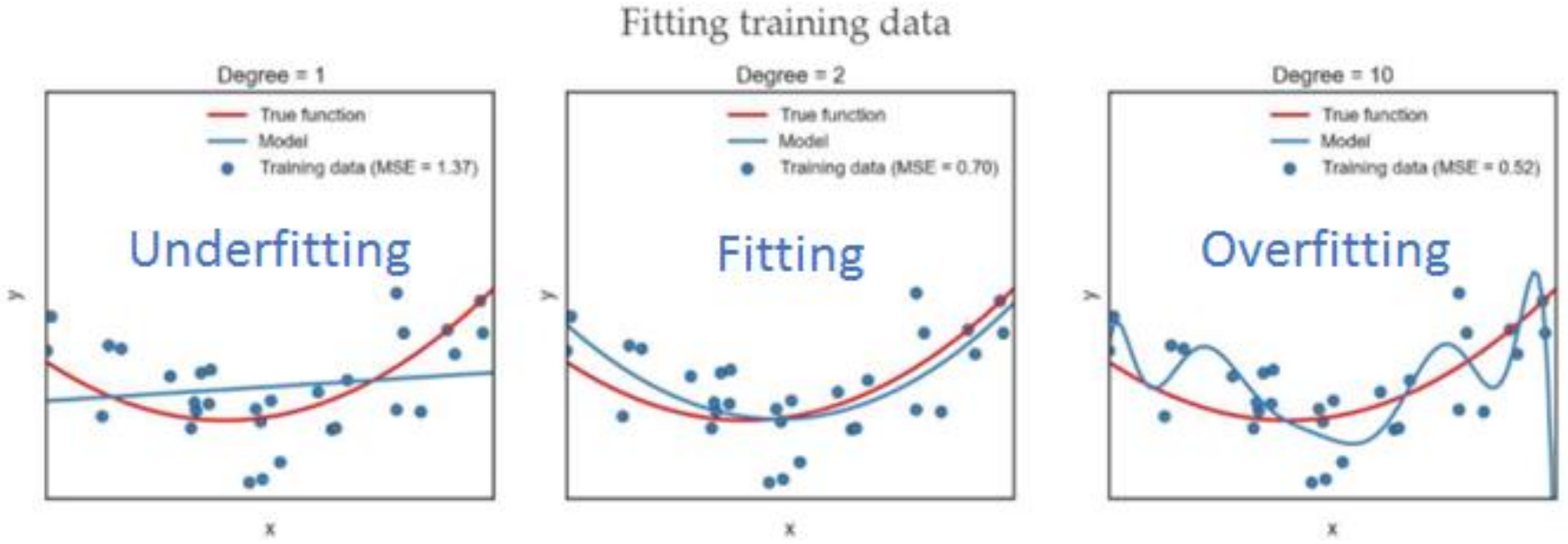
ПЛАН ЛЕКЦИИ

- Переобучение и регуляризация
- Методы кодирования категориальных признаков

ПЕРЕОБУЧЕНИЕ И РЕГУЛЯРИЗАЦИЯ

ОЦЕНКА ОБОБЩАЮЩЕЙ СПОСОБНОСТИ МОДЕЛИ

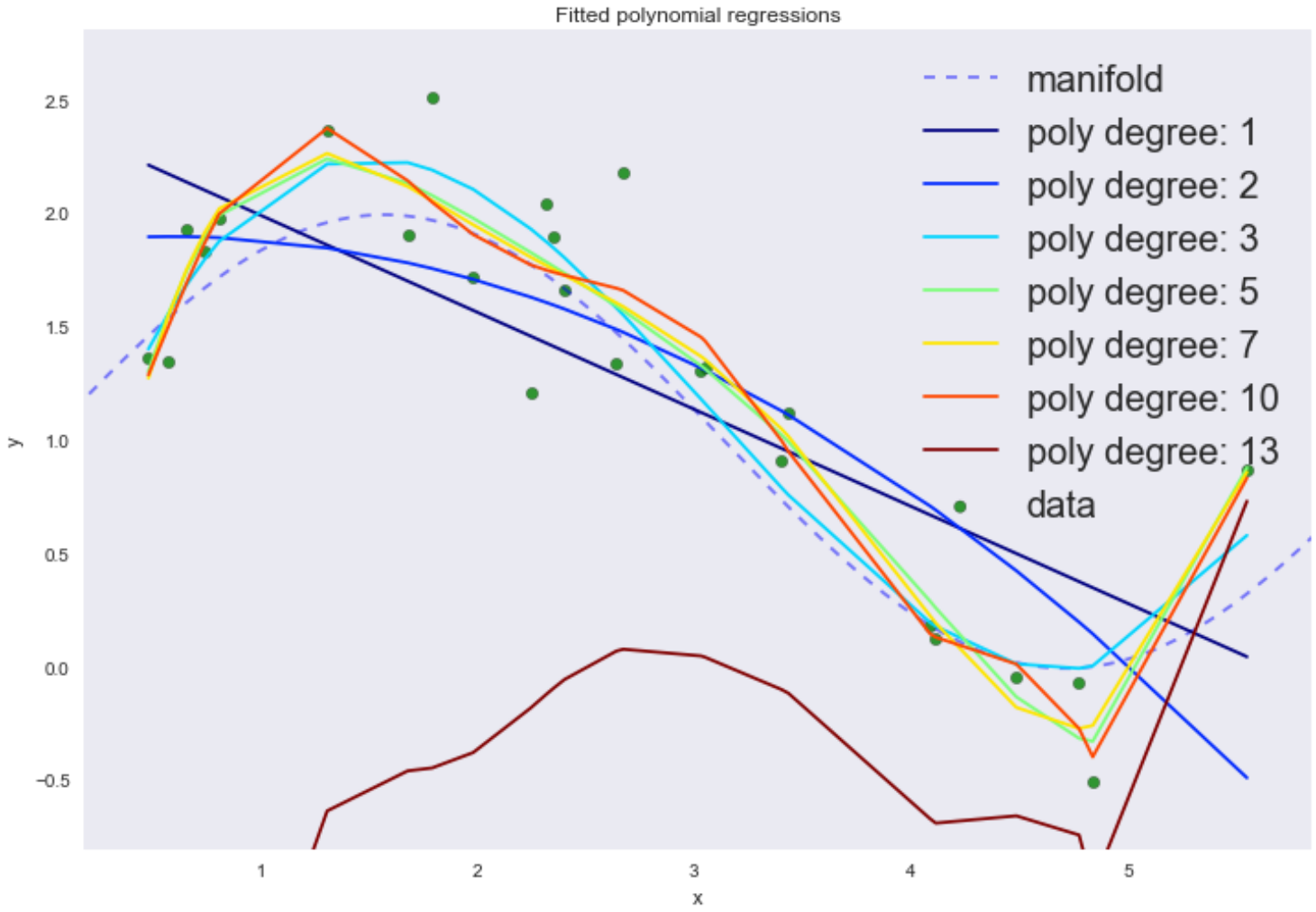
Переобучение (overfitting) – явление, при котором качество модели на новых данных сильно хуже, чем качество на тренировочных данных.



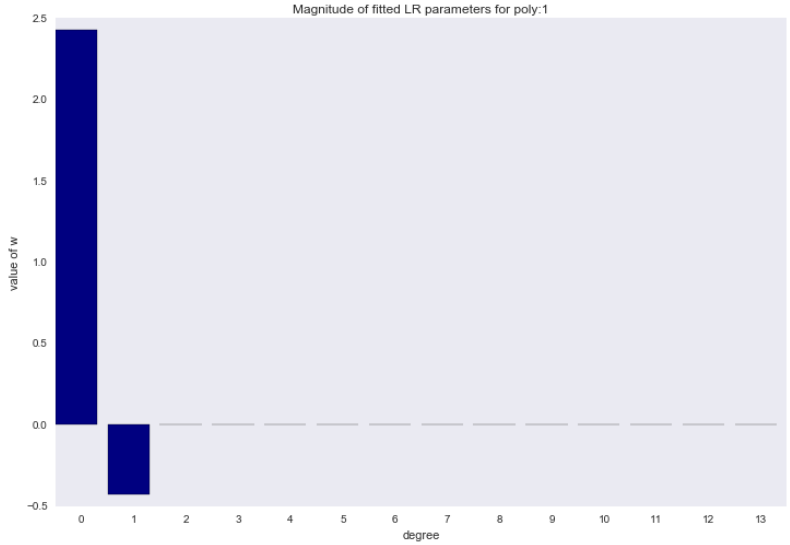
ПРИЗНАКИ ПЕРЕОБУЧЕННОЙ МОДЕЛИ

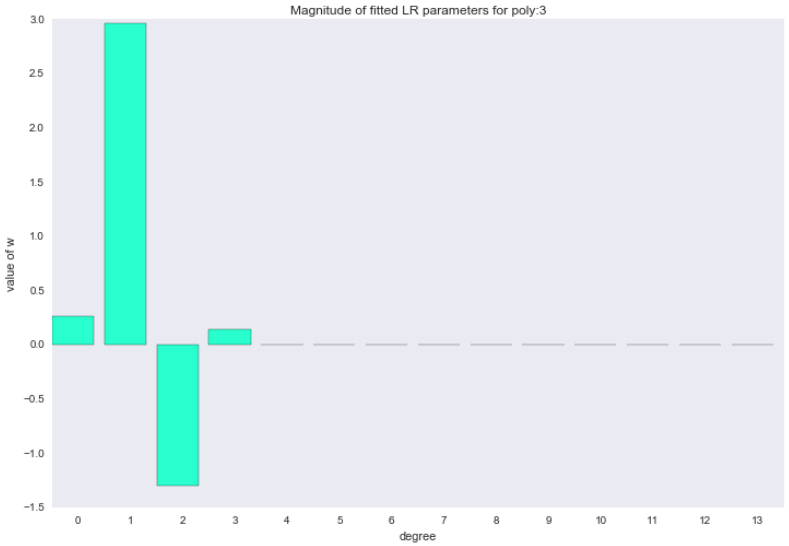
- Большая разница в качестве на тренировочных и тестовых данных (модель подгоняется под тренировочные данные и не может найти истинную зависимость)
- Большие значения параметров (весов) w_j модели
- Неустойчивость дискриминантной (разделяющей) функции (w, x).

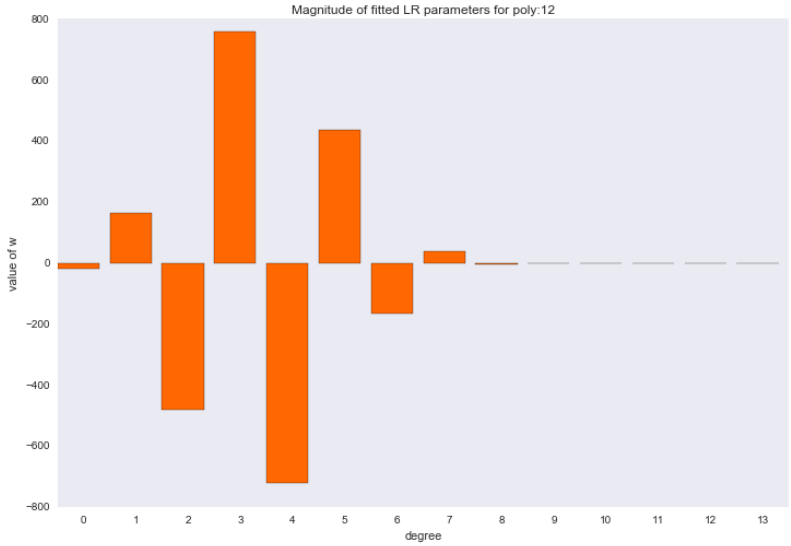
ПЕРЕОБУЧЕНИЕ: ПРИМЕР



ПЕРЕОБУЧЕНИЕ: ПРИМЕР







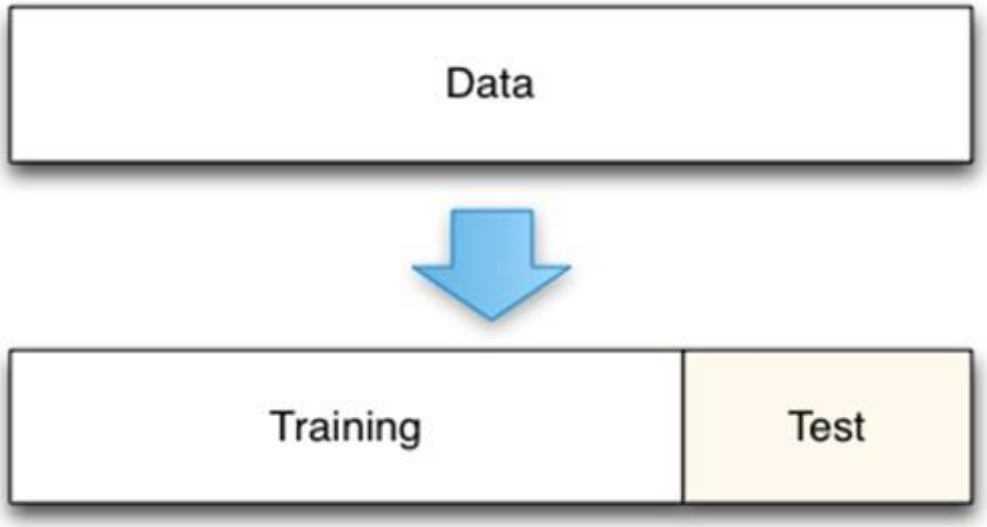
ОЦЕНИВАНИЕ КАЧЕСТВА МОДЕЛИ

- Отложенная выборка
- Кросс-валидация

ОТЛОЖЕННАЯ ВЫБОРКА

Делим тренировочную выборку на две части:

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

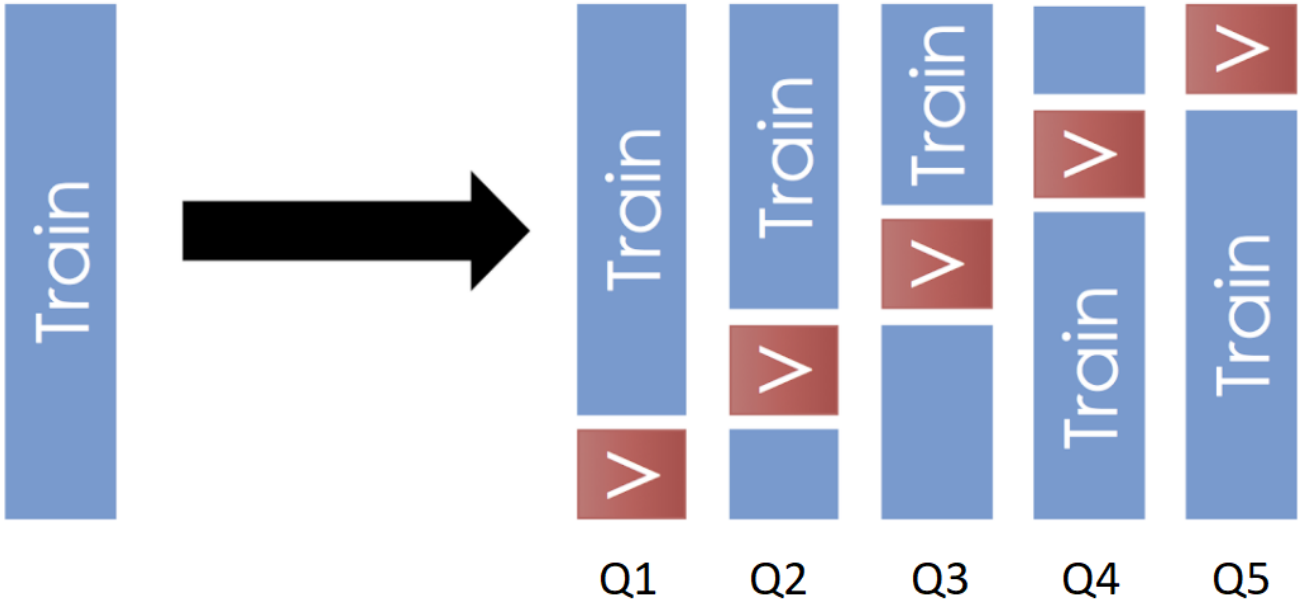


Недостаток:

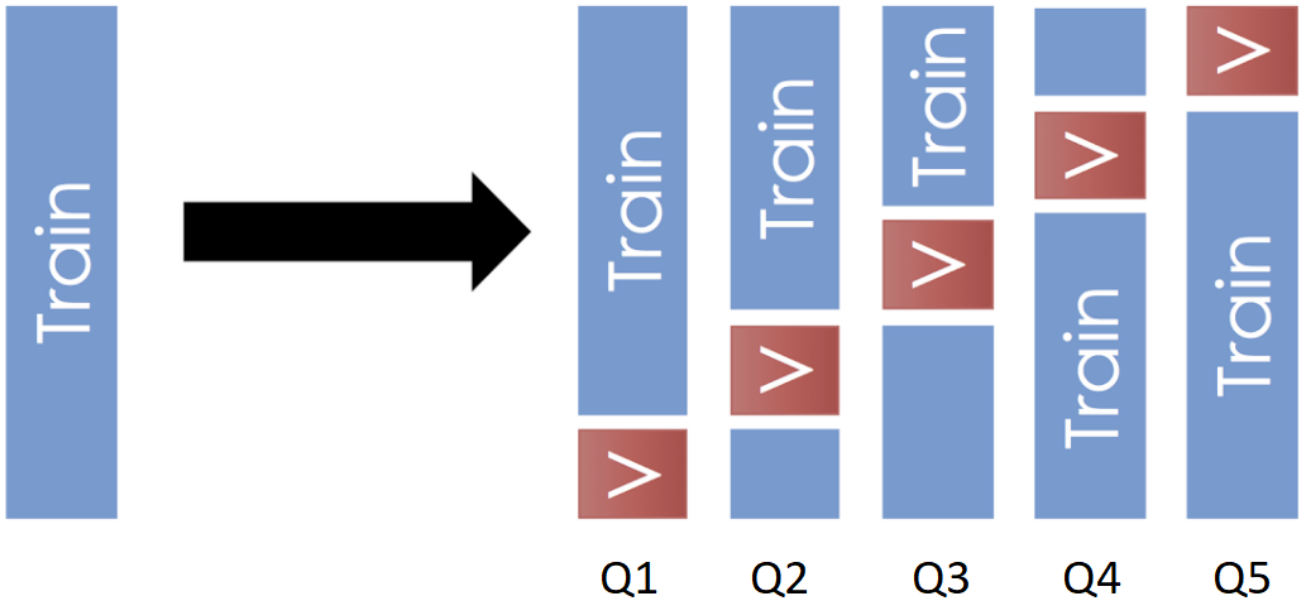
- Результат сильно зависит от разбиения на train и test

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



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

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

- *Проблемы при маленьком k?*
- *Проблемы при большом k?*

ВЫБОР КОЛИЧЕСТВА БЛОКОВ В K-FOLD КРОСС-ВАЛИДАЦИИ

- 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 + долго обучать много моделей

МЕТОД БОРЬБЫ С ПЕРЕОБУЧЕНИЕМ: РЕГУЛЯРИЗАЦИЯ

Утверждение 1. Большие значения параметров (весов) модели w – признак переобучения.

Утверждение 2. Если в выборке есть линейно-зависимые признаки, то задача оптимизации для обучения линейной модели $Q(w) \to min$ имеет бесконечное число решений.

МЕТОД БОРЬБЫ С ПЕРЕОБУЧЕНИЕМ: РЕГУЛЯРИЗАЦИЯ

Решение описанных проблем – ***регуляризация***.

Будем минимизировать регуляризованный функционал ошибки:

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

где $R(w)$ - регуляризатор.

РЕГУЛЯРИЗАЦИЯ

- **Регуляризация штрафует за слишком большие веса.**

Наиболее используемые регуляризаторы:

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

РЕГУЛЯРИЗАЦИЯ

- **Регуляризация штрафует за слишком большие веса.**

Наиболее используемые регуляризаторы:

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

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:

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

где I – единичная матрица.

Эта задача имеет аналитическое решение:

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

- 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

$$(1) \quad Q(w) + \alpha \left|\left|w\right|\right|_1 \to \min_w$$

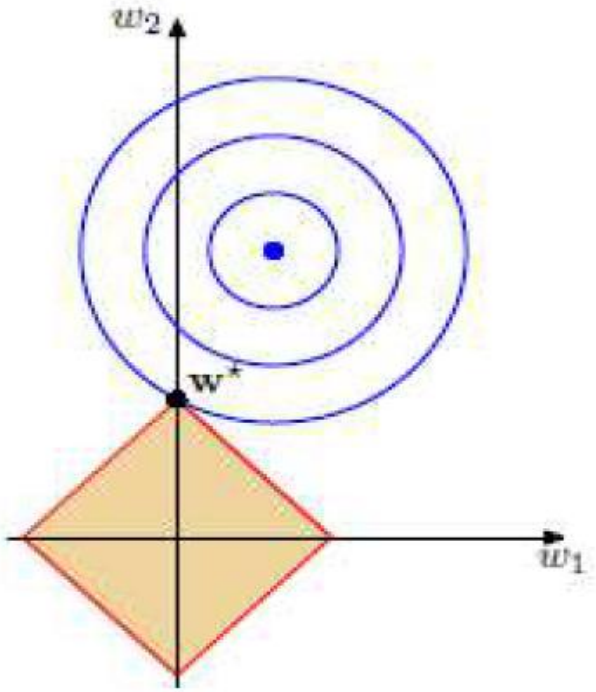
и

$$(2) \quad \begin{cases} Q(w) \to \min\limits_w \\ \left|\left|w\right|\right|_1 \leq C \end{cases}$$

эквивалентны.

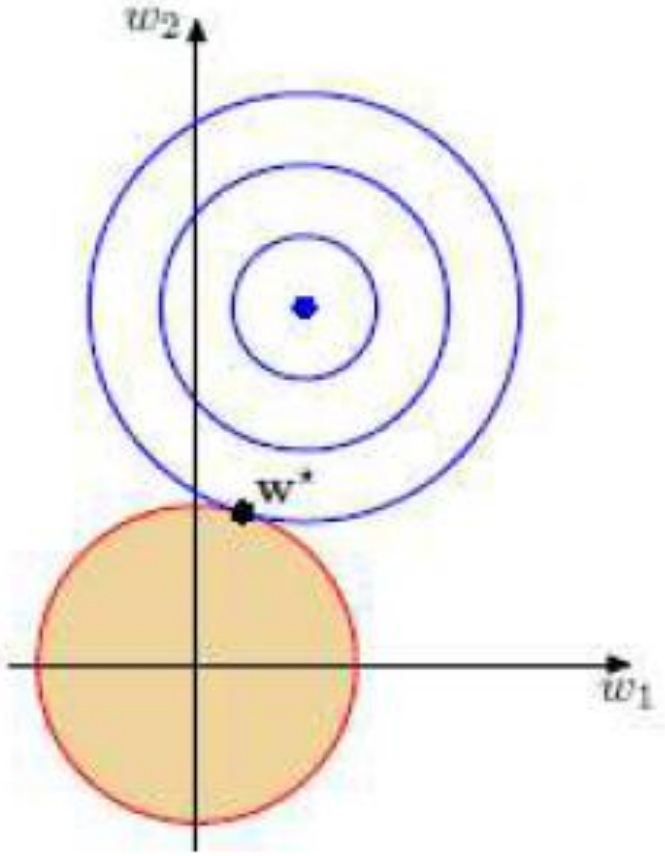
ОТБОР ПРИЗНАКОВ ПО L1-РЕГУЛЯРИЗАЦИИ

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



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***.

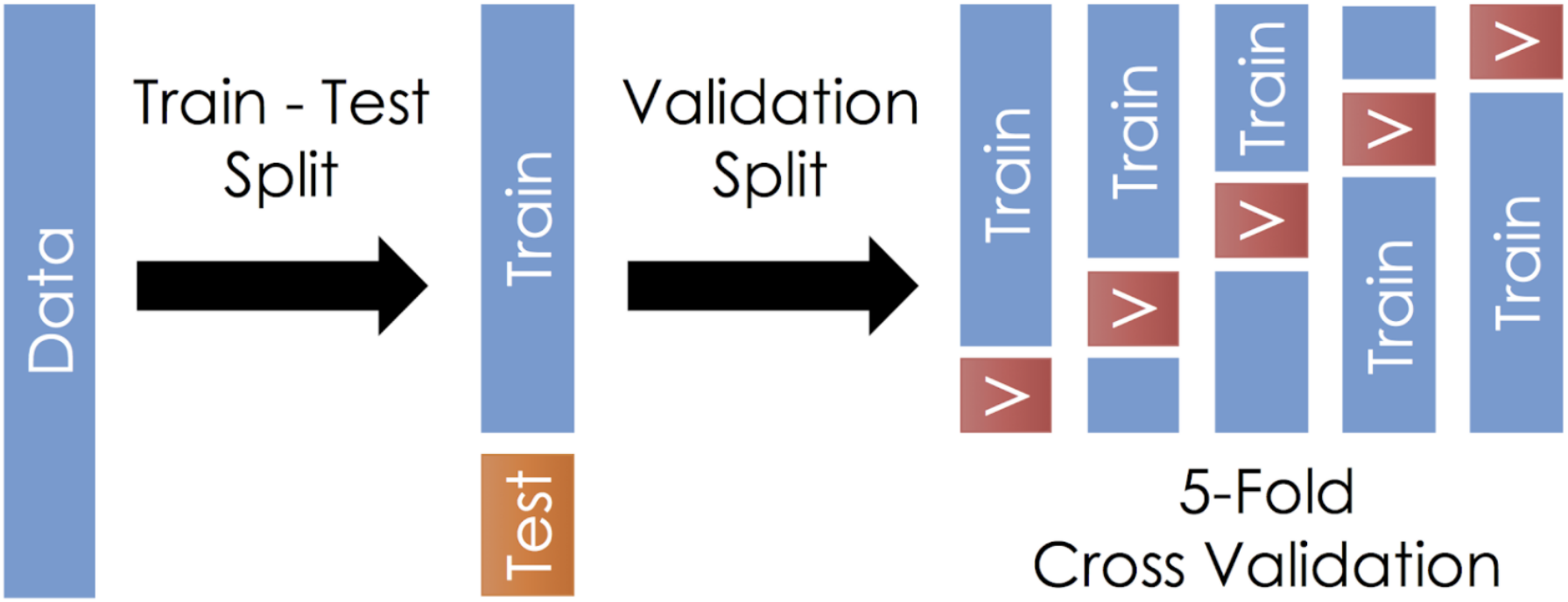
- L1-регуляризация зануляет часть весов, то есть делает модель разреженной.

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 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



МЕТОДЫ КОДИРОВАНИЯ КАТЕГОРИАЛЬНЫХ ПРИЗНАКОВ

КОДИРОВАНИЕ КАТЕГОРИАЛЬНЫХ ПРИЗНАКОВ: ONE-HOT ENCODING

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

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: $C_1, C_2, \ldots, C_m$.

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).

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

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: ПРИМЕР

city	target	0	1	2
Moscow	1	1/4	1/2	1/4
London	0	1/2	0	1/2
London	2	1/2	0	1/2
Kiev	1	1/2	1/2	0
Moscow	1	1/4	1/2	1/4
Moscow	0	1/4	1/2	1/4
Kiev	0	1/2	1/2	0
Moscow	2	1/4	1/2	1/4

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)

- Пусть целевая переменная y принимает значения от 1 до K.
- Закодируем категориальную переменную $f(x)$ следующим способом:

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

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

Тогда кодировка:

$$\boldsymbol{mean\_target_k(x, X) = \frac{successes_k(f(x), X)}{counts(f(x), X)} \approx p(y = k|f(x))}$$

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:

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

Недостатки? Когда такой способ кодирования может переобучить наш алгоритм?

СЧЁТЧИКИ: ОПАСНОСТИ

- *Вычисляя счётчики, мы закладываем в признаки информацию о целевой переменной и, тем самым, переобучаемся*
- *Если в данных есть редкие категории, то счетчики на них переобучатся*

РЕШЕНИЕ 1: СЧЁТЧИКИ + СГЛАЖИВАНИЕ

Используем счётчики (mean target encoding) со сглаживанием:

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

n_{rows} - количество строк в категории,

α – параметр регуляризации.

РЕШЕНИЕ 2: ОТЛОЖЕННАЯ ВЫБОРКА ИЛИ КРОСС-ВАЛИДАЦИЯ

- Можно вычислять счётчики так:

city	target
Moscow	1
London	0
London	2
Kiev	1
Moscow	1
Moscow	0
Kiev	0
Moscow	2

Вычисляем счетчики по этой части

Кодируем признак вычисленными счётчиками и обучаемся по этой части

РЕШЕНИЕ 2: ОТЛОЖЕННАЯ ВЫБОРКА ИЛИ КРОСС-ВАЛИДАЦИЯ

Более продвинутый способ (по кросс-валидации):

1) Разбиваем выборку

на m частей $X_1, \dots, X_m$

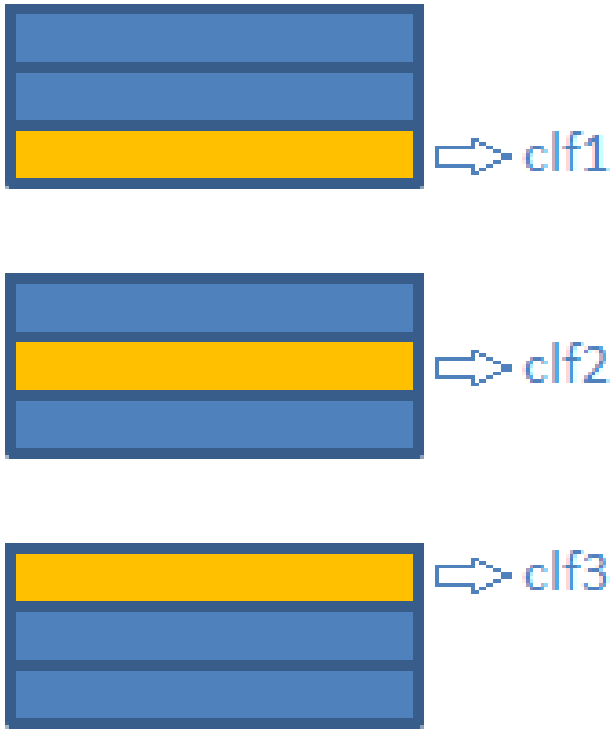
2) На каждой части X_i

значения признаков

вычисляются по

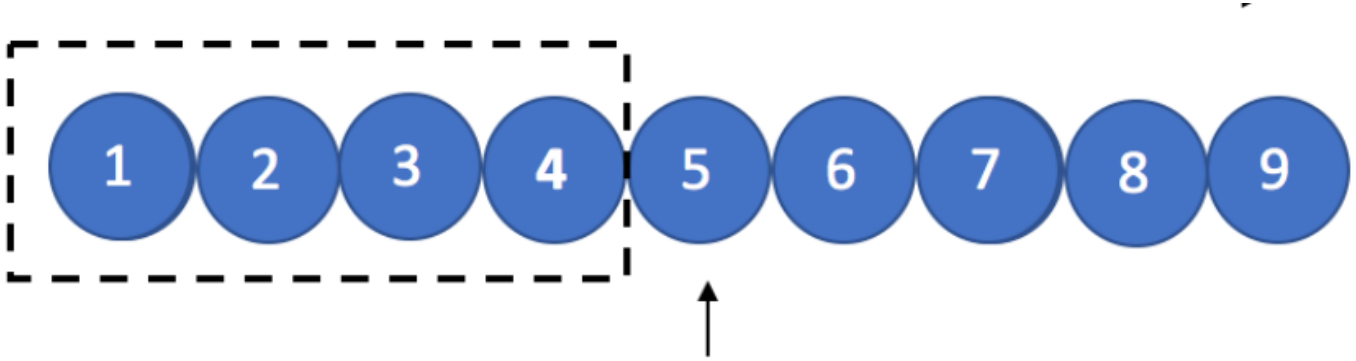
оставшимся частям:

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



РЕШЕНИЕ 3: ВЫЧИСЛЕНИЕ СЧЕТЧИКОВ С ПОМОЩЬЮ СХЕМЫ EXPANDING MEAN

Суть схемы заключается в том, чтобы пройти по отсортированному в определенном порядке датасету и для подсчета счетчика для строки *m* использовать строки от *0* до *m-1*.



Running mean calculation.
Numbers are assigned randomly to each observation. Only 1-4 are used to find encoding for 5

БОРЬБА С ПЕРЕОБУЧЕНИЕМ В СЧЁТЧИКАХ

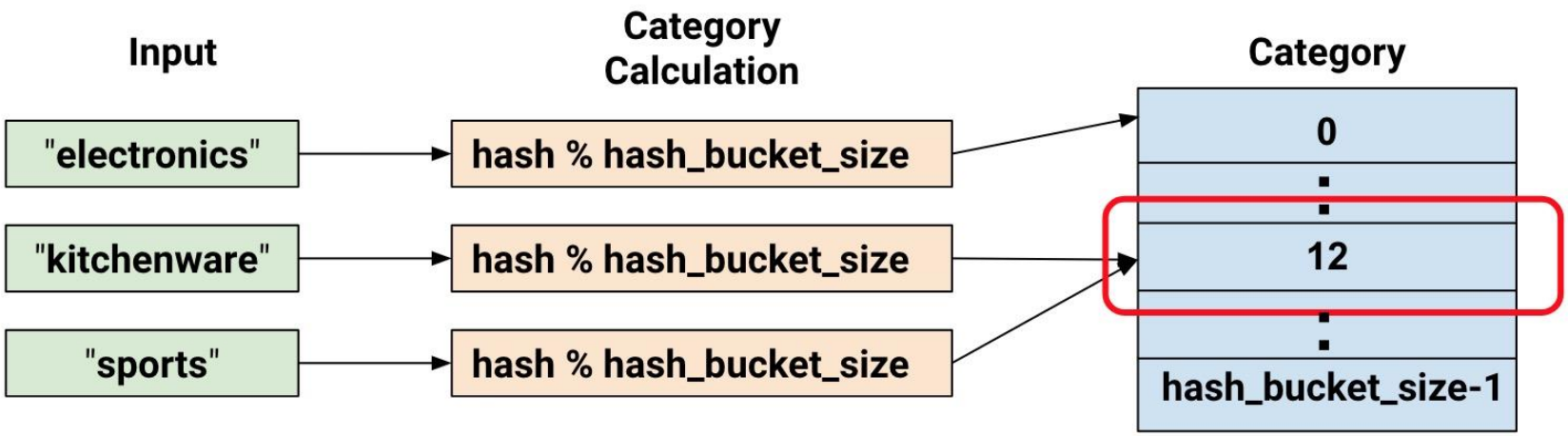
- Вычисление счётчиков по кросс-валидации
- Сглаживание
- Добавление случайных шумов
- Expanding mean

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: hash % hash_bucket_size – тем самым кодируем каждое значение признака числом от 0 до hash_bucket_size-1.
4) Дальше к полученным числам применяем OHE.

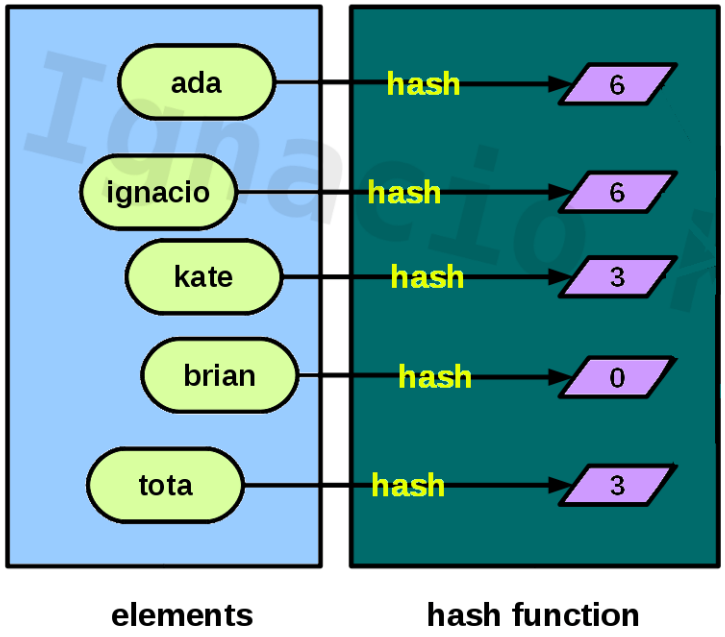


ЧТО ДЕЛАЕТ ХЭШ-ФУНКЦИЯ

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

ЧТО ПОЧИТАТЬ ПРО КОДИРОВАНИЕ КАТЕГОРИАЛЬНЫХ ПРИЗНАКОВ

- Лекция Жени Соколова
- Блог Александра Дьяконова
- Кусочек статьи с Хабра про хеширование