Линейная регрессия

Лекция 2

План лекции

- Понятие линейных моделей
- Измерение ошибки в задачах регрессии
- Обучение линейной регрессии
- Градиентный спуск
- Стохастический градиентный спуск
- Модификации градиентного спуска
- Предобработка данных
- Переобучение
- Оценка качества моделей
- Регуляризация

Понятие линейных моделей

- Линейная регрессионная модель:

$$a(\vec{x}_i) = w_0 + \sum_{j=1}^{d} w_j x_{ij} ,$$

где w_j – веса или весовые коэффициенты,
w_0 – свободный коэффициент или сдвиг (bias).

- В векторном виде:

$$a(\vec{x}_i) = w_0 + \langle \vec{w}, \vec{x}_i \rangle,$$

где $\vec{w} = (w_1, \dots, w_d)$, $\vec{x}_i = (x_{i1}, \dots, x_{id})$.

- В сокращенном векторном виде:

$$a(\vec{x}_i) = \langle \vec{w}, \vec{x}_i \rangle$$

Измерение ошибки в задачах регрессии

- Функция потерь:

$$L\left(y, y_{pred}\right) = L(y, a)$$

- Среднеквадратичная ошибка (mean squared error, MSE):

$$L(y, a) = (a - y)^2$$

$$MSE(a, X) = \frac{1}{l}\sum_{i=1}^{l} L(y_i, a_i) = \frac{1}{l}\sum_{i=1}^{l} (a(\vec{x}_i) - y_i)^2$$

- Root mean squared error (RMSE):

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

Измерение ошибки в задачах регрессии

- Коэффициент детерминации:

$$R^2(a, X) = 1 - \frac{\sum_{i=1}^{l}(a(\vec{x}_i) - y_i)^2}{\sum_{i=1}^{l}(y_i - \bar{y})^2} = 1 - \frac{\sigma^2}{\sigma_y^2}$$

где σ_y^2 – дисперсия y, σ^2 – дисперсия ошибки модели

- Среднее абсолютное отклонение (mean absolute error, MAE):

$$L(y, a) = |a - y|$$

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

Измерение ошибки в задачах регрессии

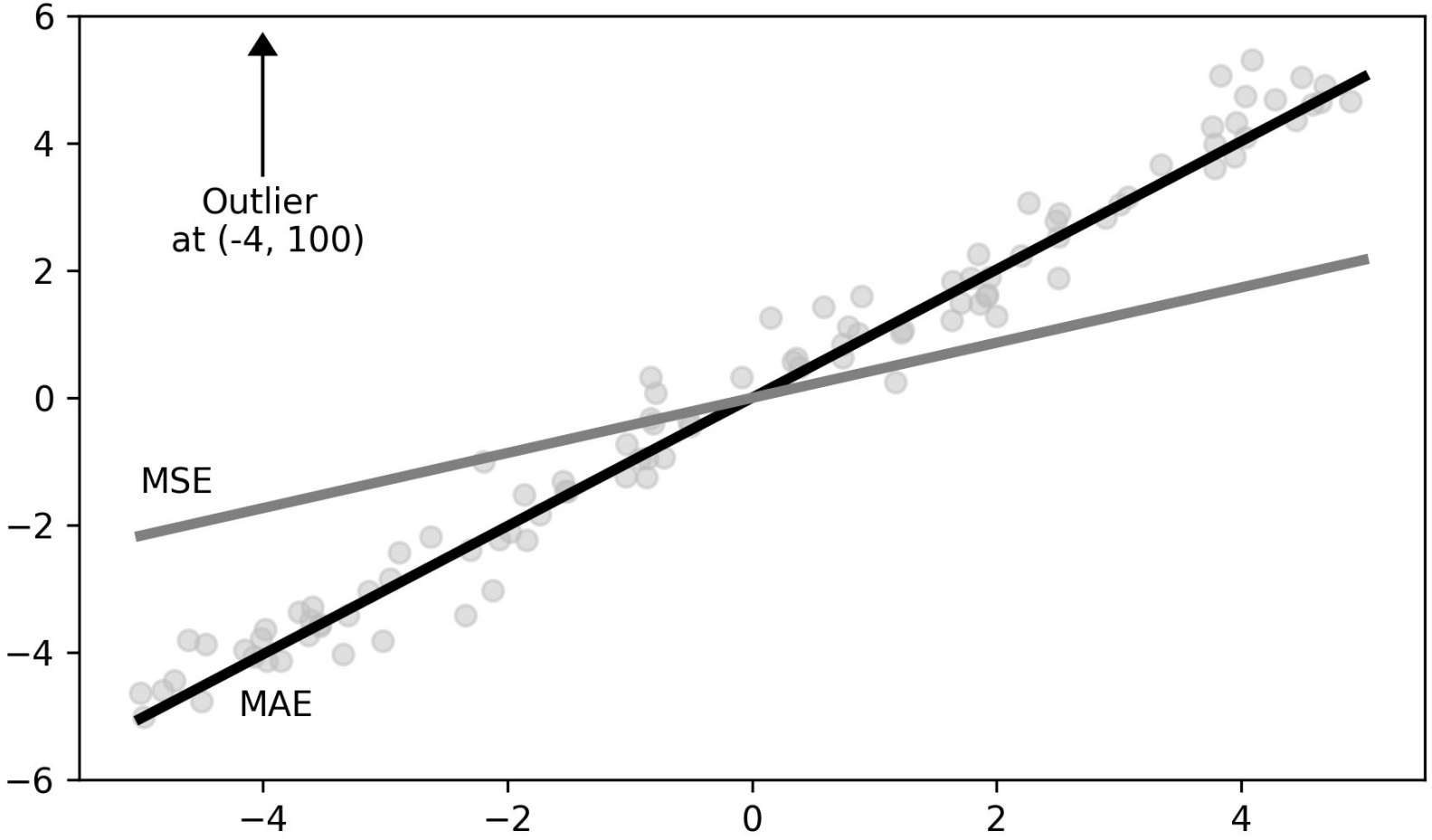
- Среднеквадратичная логарифмическая ошибка (mean squared logarithmic error, MSLE):

$$L(y, a) = \big(log(a + 1) - log(y + 1)\big)^2$$

- Средняя абсолютная процентная ошибка (mean absolute percentage error, MAPE):

$$L(y, a) = \left|\frac{y - a}{y}\right|$$

Измерение ошибки в задачах регрессии



Обучение линейной регрессии

- В случае использования среднеквадратичной ошибки (MSE):

$$\frac{1}{l}\sum_{i=1}^{l}(\langle\vec{w},\vec{x}_i\rangle - y_i)^2 \to \min_{\vec{w}}$$

- В матричном виде:

$$\frac{1}{l}\|X\vec{w} - \vec{y}\|^2 \to \min_{\vec{w}},$$

где $X \in \mathbb{R}^{l\times d}$, $\vec{w} \in \mathbb{R}^d$, $\vec{y} \in \mathbb{R}^l$

Обучение линейной регрессии

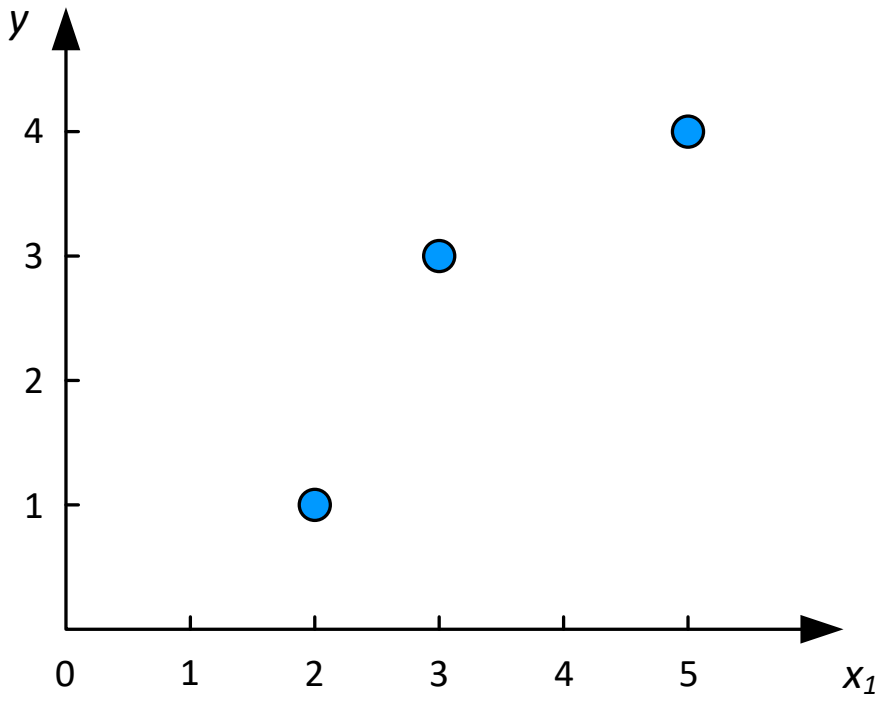
- После дифференцирования данного функционала по вектору $\vec{w}$, приравнивания к нулю и решения уравнения, получаем:

$$\frac{\partial}{\partial \vec{w}}\left(\frac{1}{l}\|X\vec{w}-\vec{y}\|^2\right)=0 \;\rightarrow \vec{w}_{opt}=(X^TX)^{-1}X^T\vec{y}$$

– *нормальное уравнение* (normal equation)

Обучение линейной регрессии

- Пример: пусть даны три точки $(2, 1)$, $(3, 3)$, $(5, 4)$
- Требуется построить линейную регрессионную модель на основе нормального уравнения



Обучение линейной регрессии

$$X = \begin{bmatrix} 1 & 2 \\ 1 & 3 \\ 1 & 5 \end{bmatrix}, \qquad \vec{y} = \begin{bmatrix} 1 \\ 3 \\ 4 \end{bmatrix}$$

$$x_0 \quad x_1$$

$$\vec{w}_{opt} = (X^T X)^{-1} X^T \vec{y}$$

Обучение линейной регрессии

$$X = \begin{bmatrix} 1 & 2 \\ 1 & 3 \\ 1 & 5 \end{bmatrix}, \qquad \vec{y} = \begin{bmatrix} 1 \\ 3 \\ 4 \end{bmatrix}$$

$$x_0 \quad x_1$$

$$\vec{w}_{opt} = (X^T X)^{-1} X^T \vec{y}$$

$$X^T X = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 3 & 5 \end{bmatrix} \times \begin{bmatrix} 1 & 2 \\ 1 & 3 \\ 1 & 5 \end{bmatrix} =$$

Обучение линейной регрессии

$$X = \begin{bmatrix} 1 & 2 \\ 1 & 3 \\ 1 & 5 \end{bmatrix}, \qquad \vec{y} = \begin{bmatrix} 1 \\ 3 \\ 4 \end{bmatrix}$$

$$x_0 \quad x_1$$

$$\vec{w}_{opt} = (X^T X)^{-1} X^T \vec{y}$$

$$X^T X = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 3 & 5 \end{bmatrix} \times \begin{bmatrix} 1 & 2 \\ 1 & 3 \\ 1 & 5 \end{bmatrix} = \begin{bmatrix} 3 & 10 \\ 10 & 38 \end{bmatrix}$$

$$2 \times 3 \qquad 3 \times 2 \qquad 2 \times 2$$

Обучение линейной регрессии

$$\vec{w}_{opt} = (X^T X)^{-1} X^T \vec{y}$$

Обучение линейной регрессии

$$\vec{w}_{opt} = (X^T X)^{-1} X^T \vec{y}$$

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

Обучение линейной регрессии

$$\vec{w}_{opt} = (X^T X)^{-1} X^T \vec{y}$$

$$(X^T X)^{-1} = \begin{bmatrix} 2.7 & -0.7 \\ -0.7 & 0.2 \end{bmatrix}$$

Обучение линейной регрессии

$$\vec{w}_{opt} = (X^T X)^{-1} X^T \vec{y}$$

$$(X^T X)^{-1} = \begin{bmatrix} 2.7 & -0.7 \\ -0.7 & 0.2 \end{bmatrix}$$

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

Обучение линейной регрессии

$$\vec{w}_{opt} = (X^T X)^{-1} X^T \vec{y}$$

$$(X^T X)^{-1} = \begin{bmatrix} 2.7 & -0.7 \\ -0.7 & 0.2 \end{bmatrix}$$

$$(X^T X)^{-1} X^T = \begin{bmatrix} 1.29 & 0.57 & -0.85 \\ -0.28 & -0.07 & 0.36 \end{bmatrix}$$

Обучение линейной регрессии

$$\vec{w}_{opt} = (X^T X)^{-1} X^T \vec{y}$$

$$(X^T X)^{-1} = \begin{bmatrix} 2.7 & -0.7 \\ -0.7 & 0.2 \end{bmatrix}$$

$$(X^T X)^{-1} X^T = \begin{bmatrix} 1.29 & 0.57 & -0.85 \\ -0.28 & -0.07 & 0.36 \end{bmatrix}$$

$$\vec{w}_{opt} = (X^T X)^{-1} X^T \vec{y} =$$

Обучение линейной регрессии

$$\vec{w}_{opt} = (X^T X)^{-1} X^T \vec{y}$$

$$(X^T X)^{-1} = \begin{bmatrix} 2.7 & -0.7 \\ -0.7 & 0.2 \end{bmatrix}$$

$$(X^T X)^{-1} X^T = \begin{bmatrix} 1.29 & 0.57 & -0.85 \\ -0.28 & -0.07 & 0.36 \end{bmatrix}$$

$$\vec{w}_{opt} = (X^T X)^{-1} X^T \vec{y} = \begin{bmatrix} -0.43 \\ 0.93 \end{bmatrix} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix}$$

Обучение линейной регрессии

$$\vec{w}_{opt} = (X^T X)^{-1} X^T \vec{y}$$

$$(X^T X)^{-1} = \begin{bmatrix} 2.7 & -0.7 \\ -0.7 & 0.2 \end{bmatrix}$$

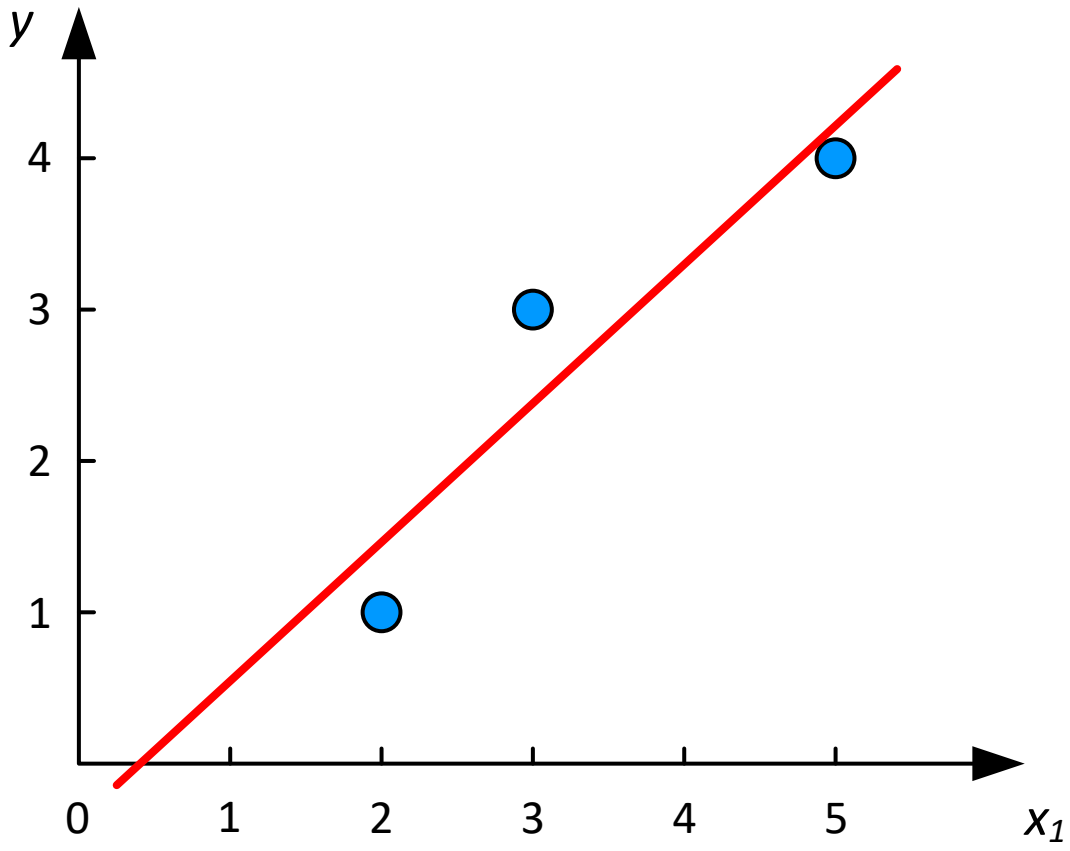
$$(X^T X)^{-1} X^T = \begin{bmatrix} 1.29 & 0.57 & -0.85 \\ -0.28 & -0.07 & 0.36 \end{bmatrix}$$

$$\vec{w}_{opt} = (X^T X)^{-1} X^T \vec{y} = \begin{bmatrix} -0.43 \\ 0.93 \end{bmatrix} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix}$$

$$a(\vec{x}) = w_0 + w_1 x_1 = -0.43 + 0.93 x_1$$

Обучение линейной регрессии

$$a(\vec{x}) = -0.43 + 0.93x_1, \qquad a(1) = 0.5, \qquad a(5) = 4.2$$



Градиентный спуск

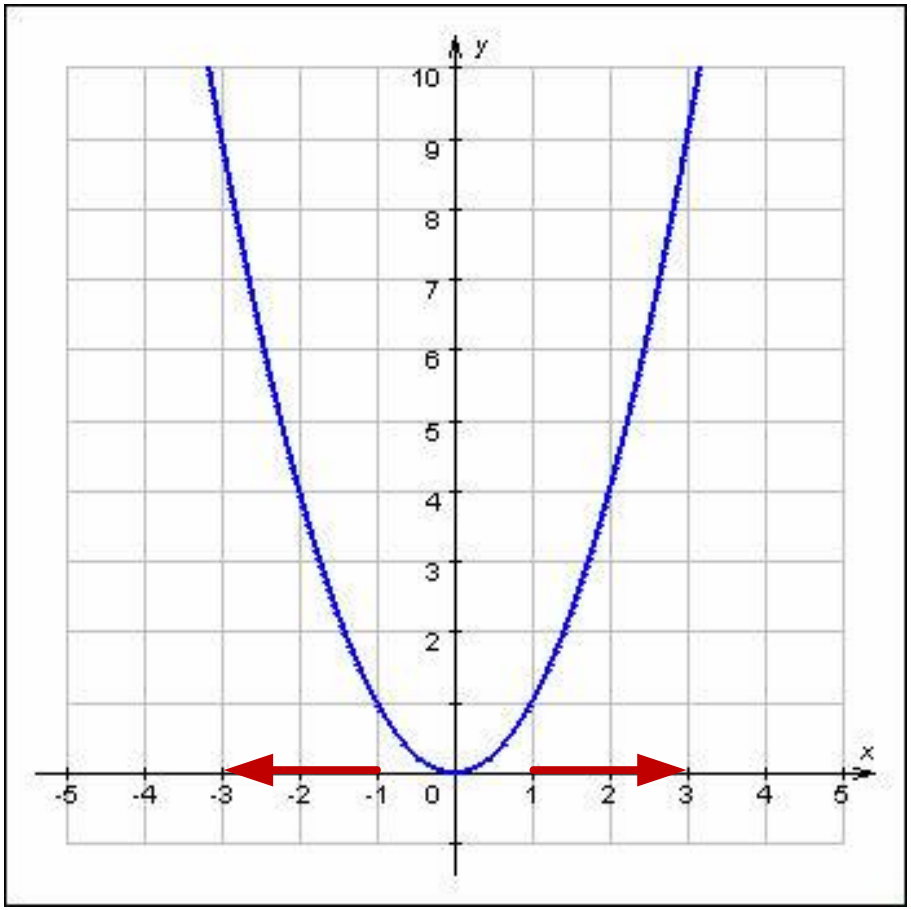
- *Градиентом* функции $f: \mathbb{R}^d \to \mathbb{R}^d$ называется вектор её частных производных (∇ – оператор набла, оператор Гамильтона):

$$\nabla f(x_1, \dots, x_d) = \left(\frac{\partial f}{\partial x_j}\right)_{j=1}^{d} = \left(\frac{\partial f}{\partial x_1}, \dots, \frac{\partial f}{\partial x_d}\right)$$

- Градиент является направлением наискорейшего роста функции, а *антиградиент* ($-\nabla f$) – направлением наискорейшего убывания

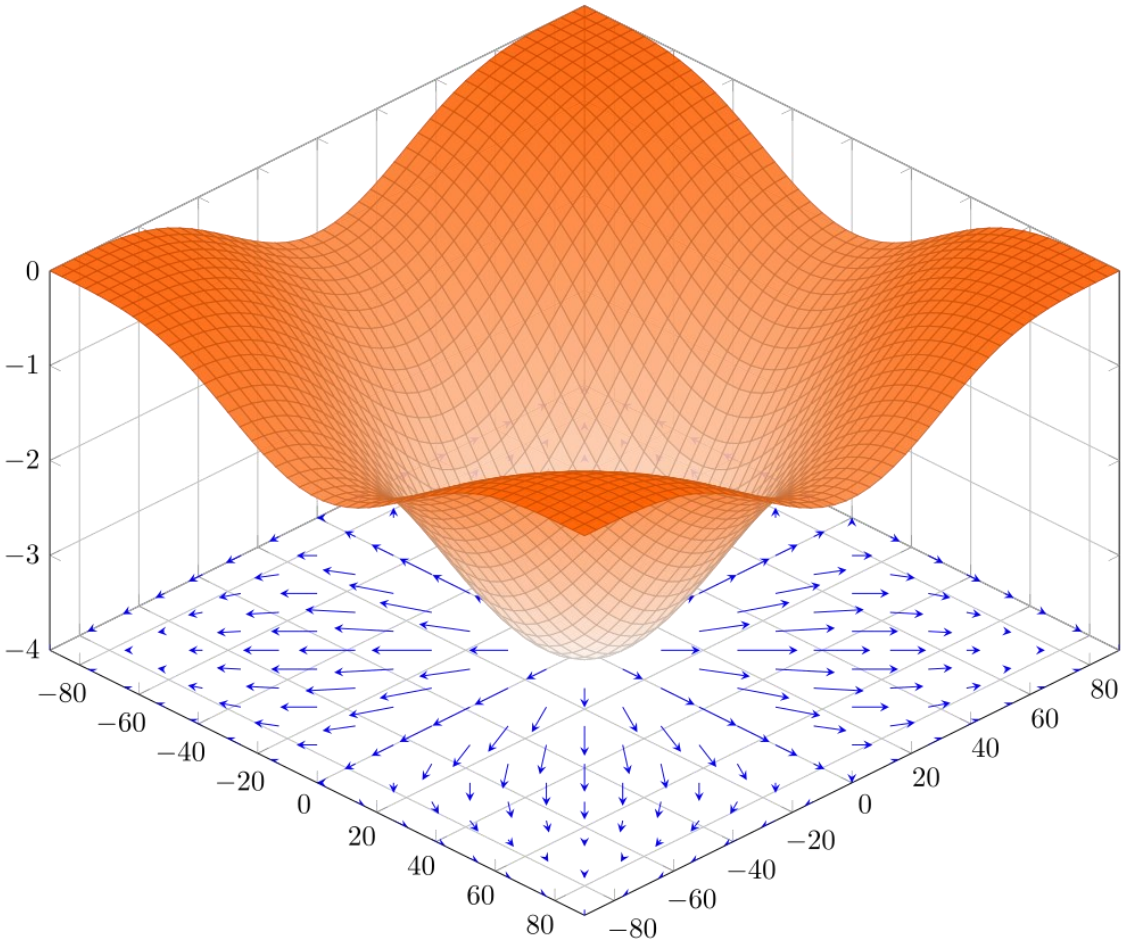
Градиентный спуск

- *Пример*: функция $y = x^2$, производная $\frac{dy}{dx} = 2x$



Градиентный спуск

- *Пример*: функция $y = -(\cos^2 x_1 + \cos^2 x_2)^2$



Градиентный спуск

Алгоритм градиентного спуска:

1. Выбрать начальную точку $\vec{w}^{(0)}$
2. Повторять до сходимости:

$$\vec{w}^{(k)} = \vec{w}^{(k-1)} - \eta_k \nabla Q\left(\vec{w}^{(k-1)}\right),$$

где k – номер шага,

$Q(\vec{w})$ – функция ошибки для набора параметров $\vec{w}$,

η_k – скорость спуска (длина k-го шага).

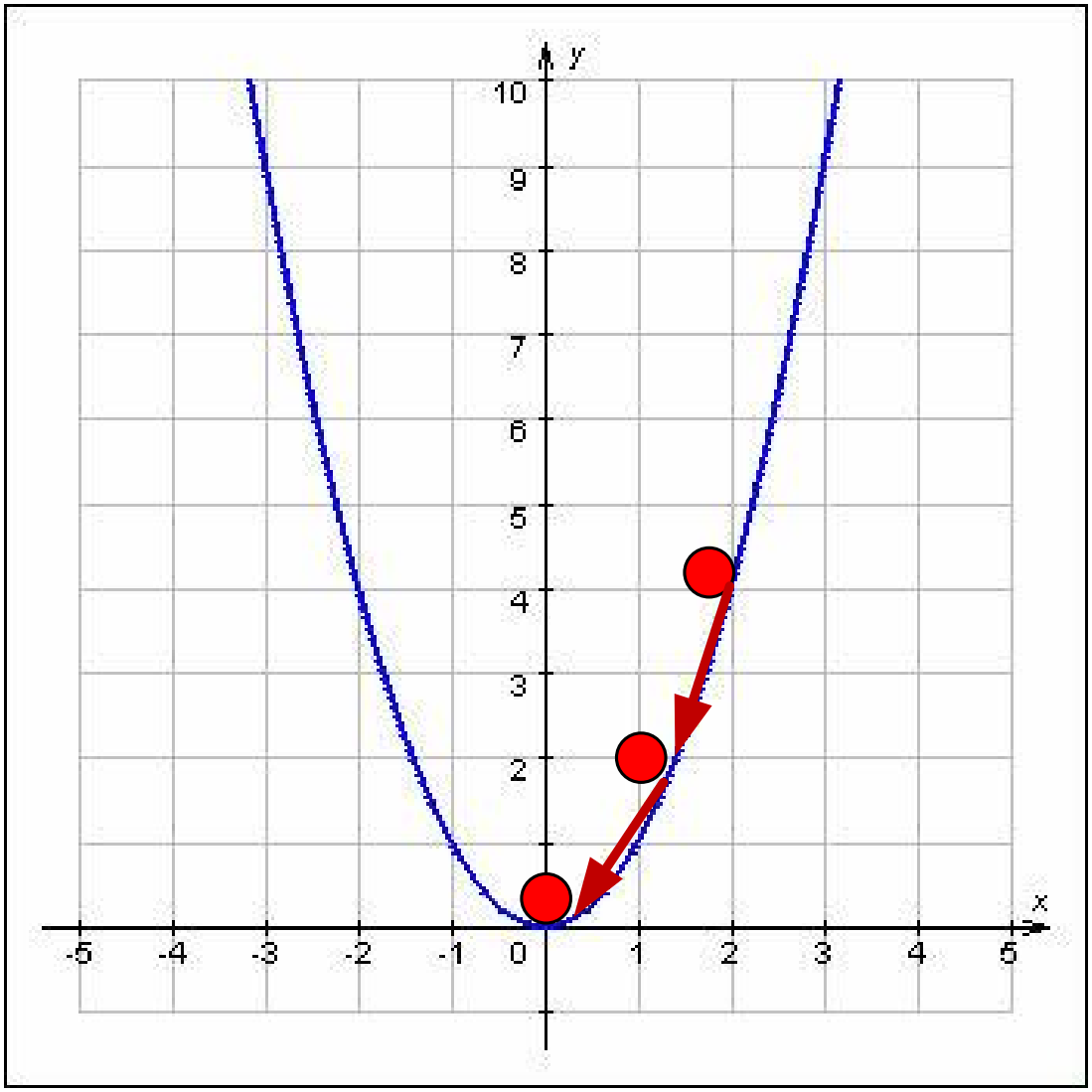
- Условия останова:
  - ошибка не уменьшается в течение нескольких итераций
  - вектор весов *почти* перестает изменяться
  - достигнуто максимальное число итераций

Градиентный спуск

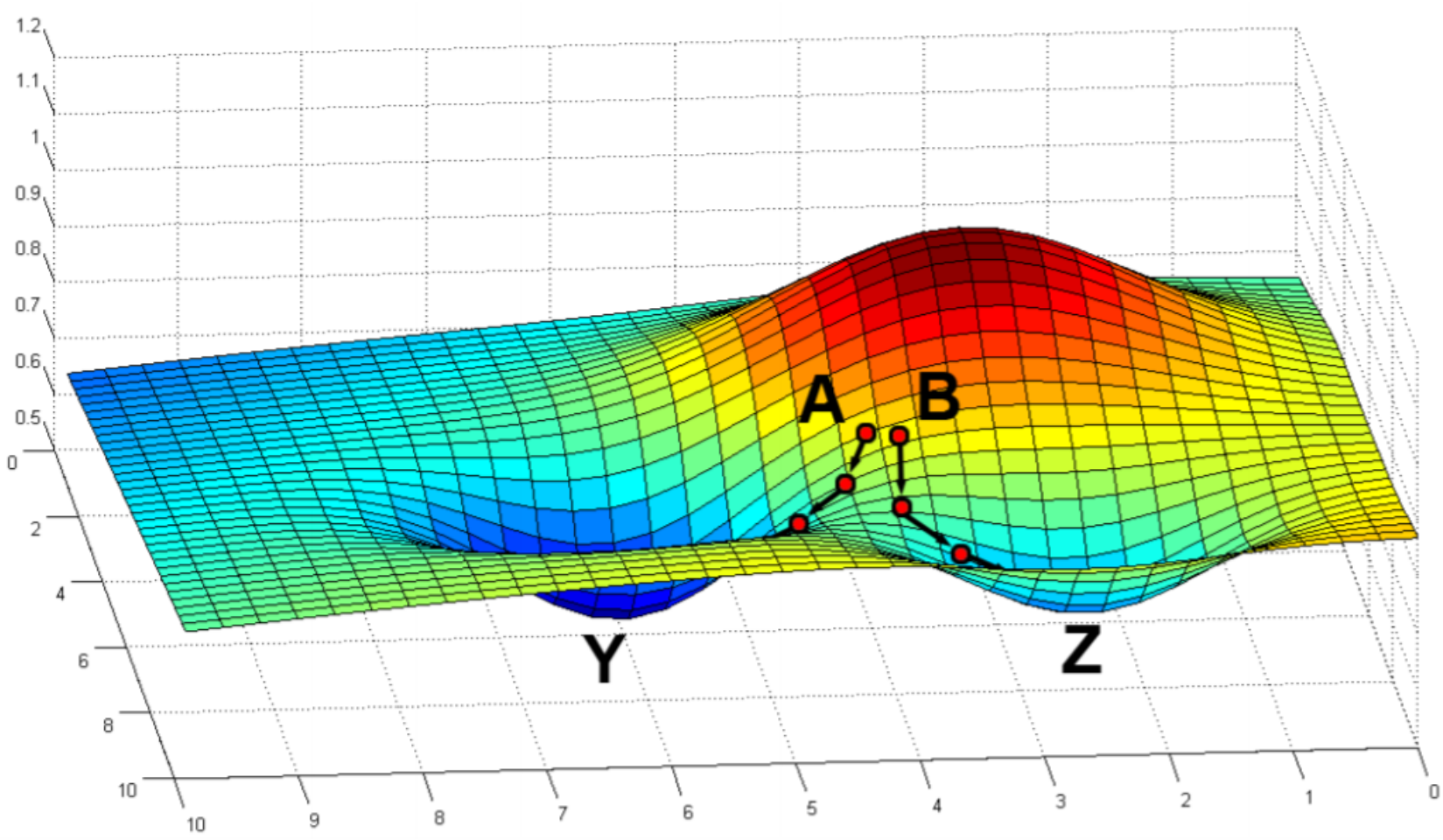
- Скорость спуска:
  - слишком высокая $\rightarrow$ переход через минимум
  - слишком низкая $\rightarrow$ медленная сходимость
- Варианты вычисления скорости спуска:
  - константная: $\eta_k = const$
  - линейное уменьшение (linear decay): $\eta_k = \eta_0 \left(1 - \frac{k}{K}\right)$
  - экспоненциальное уменьшение (exponential decay):

$$\eta_k = \eta_0 e^{-\frac{k}{K}}$$

Градиентный спуск



Градиентный спуск



Градиентный спуск

- Для среднеквадратичной ошибки:

$$\nabla Q(\vec{w}) = \nabla_{\vec{w}} \left(\frac{1}{l} \|X\vec{w} - \vec{y}\|^2 \right) = \frac{2}{l} X^T (X\vec{w} - \vec{y})$$

Стохастический градиентный спуск

- Функционал ошибки представим в виде суммы l функций ошибок:

$$Q(\vec{w}) = \frac{1}{l}\sum_{i=1}^{l} L_i(\vec{w})$$

- При градиентном спуске необходимо вычислять градиент всей суммы:

$$\nabla Q(\vec{w}) = \frac{1}{l}\sum_{i=1}^{l} \nabla L_i(\vec{w})$$

- Если выборка большая, вычисление градиента трудоемко

Стохастический градиентный спуск

- Оценить градиент суммы функций можно градиентом одного случайно взятого i_k слагаемого:

$$\nabla Q(\vec{w}) \approx \nabla L_{i_k}(\vec{w})$$

- Метод стохастического градиентного спуска (stochastic gradient descent, SGD):

$$\vec{w}^{(k)} = \vec{w}^{(k-1)} - \eta_k \nabla L_{i_k}\left(\vec{w}^{(k-1)}\right)$$

- Градиентный спуск по мини-батчам (mini-batch gradient descent):

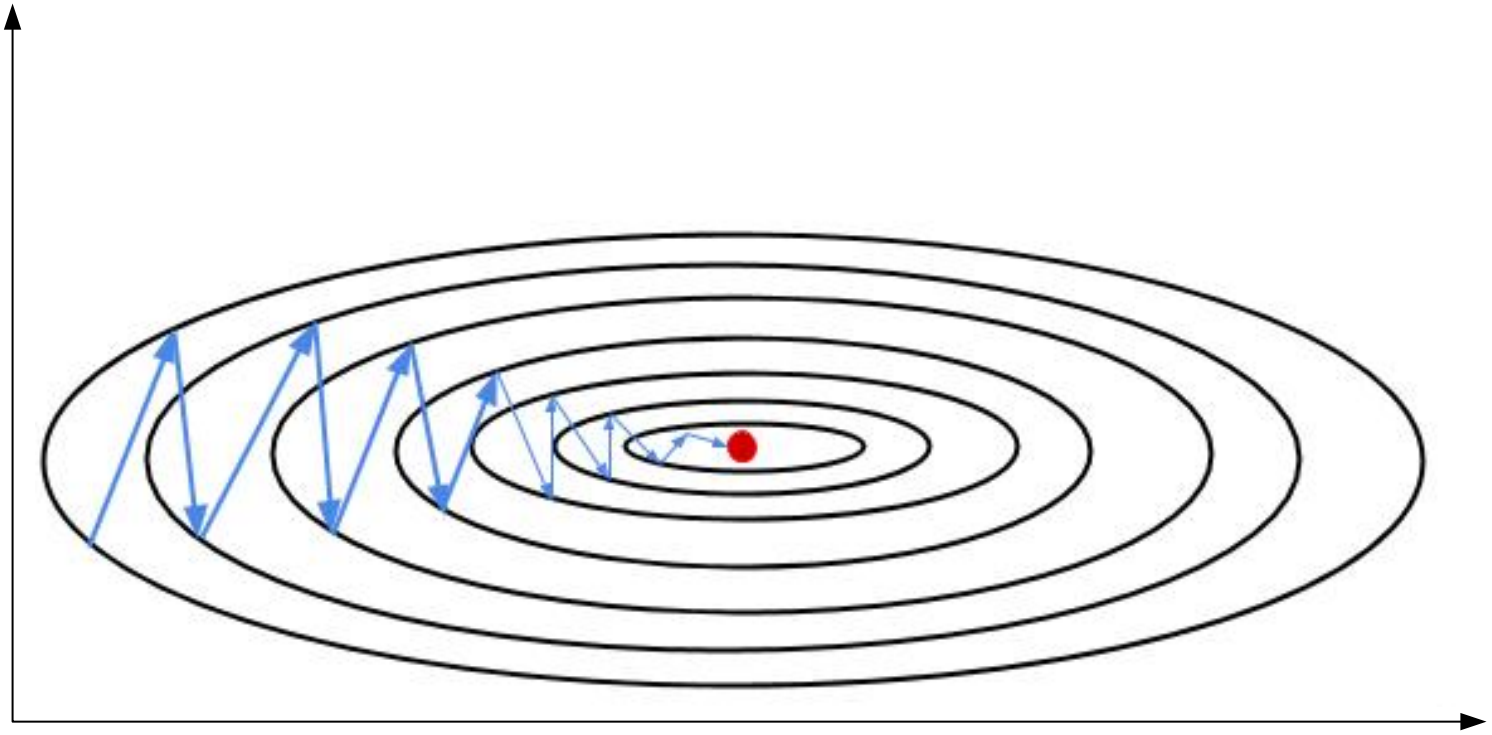
$$\nabla Q(\vec{w}) \approx \frac{1}{n}\sum_{j=1}^{n} \nabla L_{i_{kj}}(\vec{w}),$$

где i_{kj} – случайно выбранные номера слагаемых

Модификации градиентного спуска

Метод моментов

- Направление антиградиента может меняться на каждом шаге:



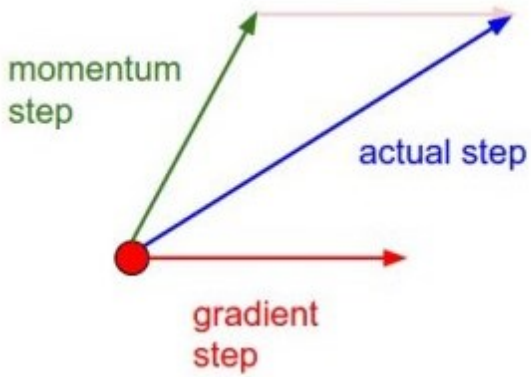
Модификации градиентного спуска

Метод моментов

- Можно усреднять векторы антиградиента с нескольких предыдущих шагов с помощью вектора инерции:

$$\vec{h}_0 = 0,$$
$$\vec{h}_k = \alpha\vec{h}_{k-1} + \eta_k \nabla Q\left(\vec{w}^{(k-1)}\right),$$

где α – коэффициент момента.



- Тогда обновление весов:

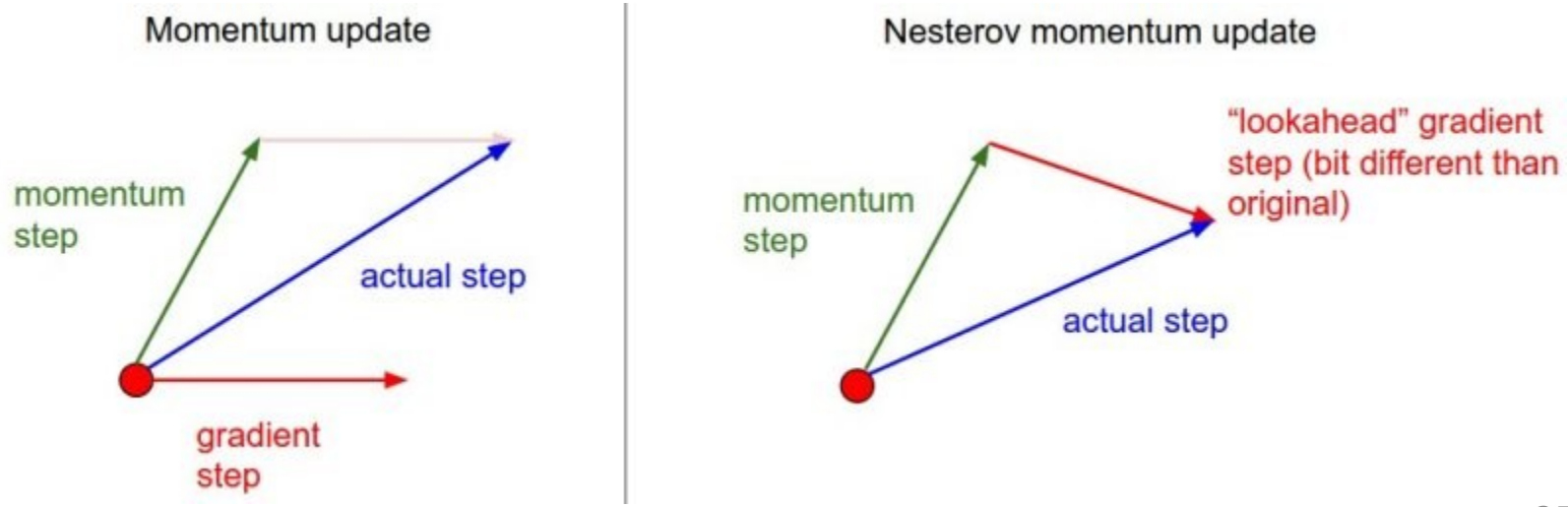
$$\vec{w}^{(k)} = \vec{w}^{(k-1)} - \vec{h}_k$$

Модификации градиентного спуска

Метод Нестерова

- Можно вычислять градиент сразу в промежуточной точке:

$$\vec{w}^{(k)} = \vec{w}^{(k-1)} - \alpha\vec{h}_{k-1} - \eta_k \nabla Q\left(\vec{w}^{(k-1)} - \alpha\vec{h}_{k-1}\right)$$



Модификации градиентного спуска

Метод AdaGrad

- Разное изменение скорости для разных компонентов вектора весов:

$$\nabla Q\left(\vec{w}^{(k-1)}\right) = \left(\mathrm{g}_1^{k-1}, \ldots, \mathrm{g}_d^{k-1}\right)$$

$$G_{kj} = G_{k-1,j} + \left(\mathrm{g}_j^{k-1}\right)^2$$

$$w_j^{(k)} = w_j^{(k-1)} - \frac{\eta_k}{\sqrt{G_{kj} + \varepsilon}} \mathrm{g}_j^{k-1}$$

Модификации градиентного спуска

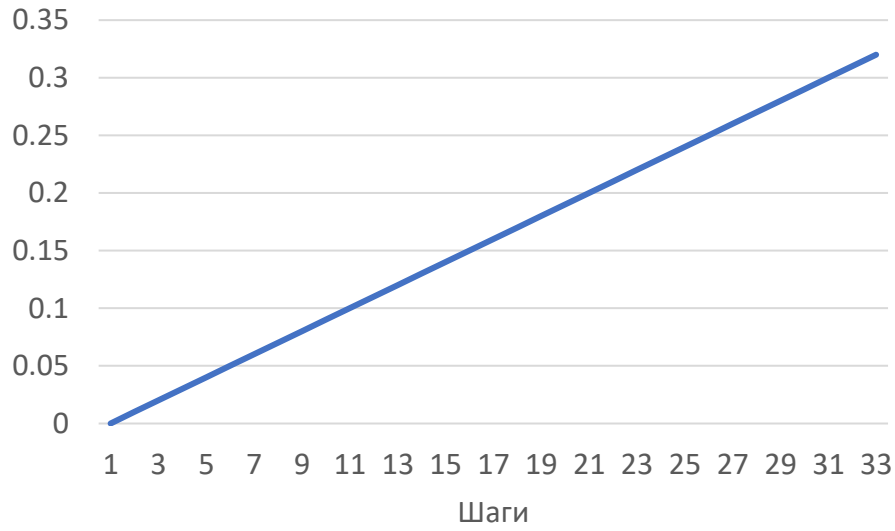
Метод RMSprop

- Проблема AdaGrad: переменная G_{kj} монотонно растёт, из-за чего шаги становятся всё медленнее и могут остановиться ещё до того, как достигнут минимум функционала

$$\nabla Q\left(\vec{w}^{(k-1)}\right) = \left(\mathrm{g}_1^{k-1}, \ldots, \mathrm{g}_d^{k-1}\right)$$

$$G_{kj} = G_{k-1,j} + \left(\mathrm{g}_j^{k-1}\right)^2$$

$$w_j^{(k)} = w_j^{(k-1)} - \frac{\eta_k}{\sqrt{G_{kj} + \varepsilon}} \mathrm{g}_j^{k-1}$$



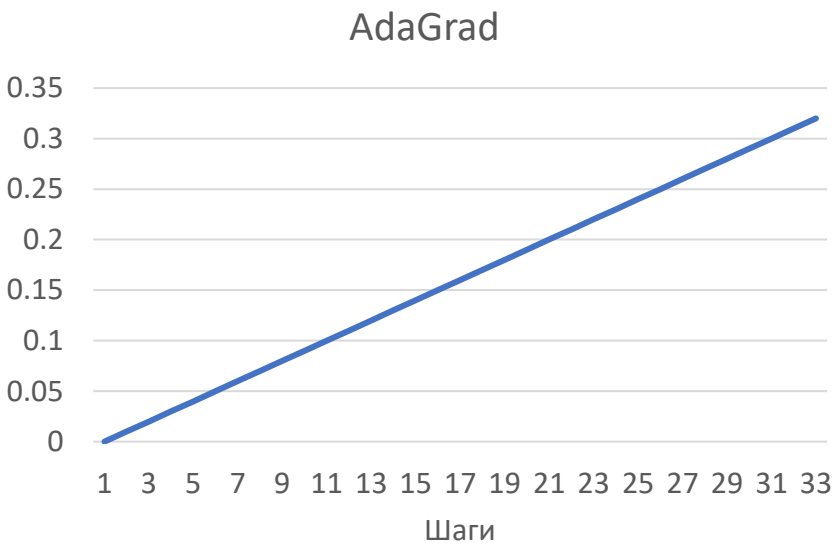
Модификации градиентного спуска

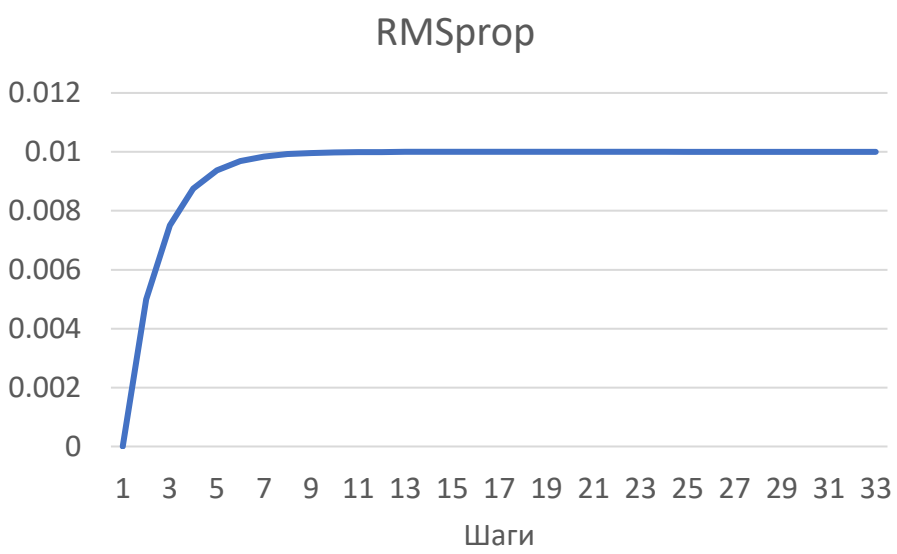
Метод RMSprop

- В методе RMSprop используется экспоненциальное затухание градиентов:

$$G_{kj} = \alpha G_{k-1,j} + (1-\alpha)\left(\mathrm{g}_j^{k-1}\right)^2,$$

где $\alpha \in [0,1]$ – сглаживающая константа, часто $\alpha = 0.9$





Предобработка данных

1. Кодирование категориальных признаков
2. Нелинейные признаки
3. Нормализация признаков
4. Исключение выбросов
5. Проверка на дубликаты
6. Исключение шумовых признаков
7. Обработка пропущенных значений

1. Кодирование категориальных признаков

Бинаризация или *one-hot encoding*

- Пусть категориальный признак $f_j(x)$ принимает значения из множества C $= \{c_1, \dots, c_m\}$
- Заменим его на m бинарных признаков $b_1(x), \dots, b_m(x)$, каждый из которых является индикатором одного из возможных категориальных значений:

$$b_i(x) = \left[f_j(x) = c_i\right]$$

1. Кодирование категориальных признаков

Бинаризация или *one-hot encoding*

id	color
1	red
2	blue
3	green
4	blue

id	color_red	color_blue	color_green
1	1	0	0
2	0	1	0
3	0	0	1
4	0	1	0

2. Нелинейные признаки

- С помощью линейной регрессии можно восстанавливать нелинейные зависимости, если провести преобразование признакового пространства (*полиномиальная регрессия*):

$$\vec{x} = (x_1, \dots, x_d) \to \varphi(\vec{x}) = \left(\varphi_1(\vec{x}), \dots, \varphi_m(\vec{x})\right)$$

- Полиномиальные признаки:
  - полином второй степени:

$$\vec{x} = (x_1, x_2) \to \varphi(\vec{x}) = (1, x_1, x_2, x_1^2, x_1 x_2, x_2^2)$$

  - полином третьей степени:

$$\vec{x} = (x_1, x_2) \to \varphi(\vec{x}) = (1, x_1, x_2, x_1^2, x_1 x_2, x_2^2, x_1^3, x_1^2 x_2, x_1 x_2^2, x_2^3)$$

- https://arachnoid.com/polysolve/
- sklearn.preprocessing.PolynomialFeatures

3. Нормализация признаков

- Нормализация признаков – приведение значений признаков к единой шкале
- Виды нормализации:
  1) Standard scaling (Z-score normalization)
  2) Минимаксная нормализация (Min-Max scaling)

3. Нормализация признаков

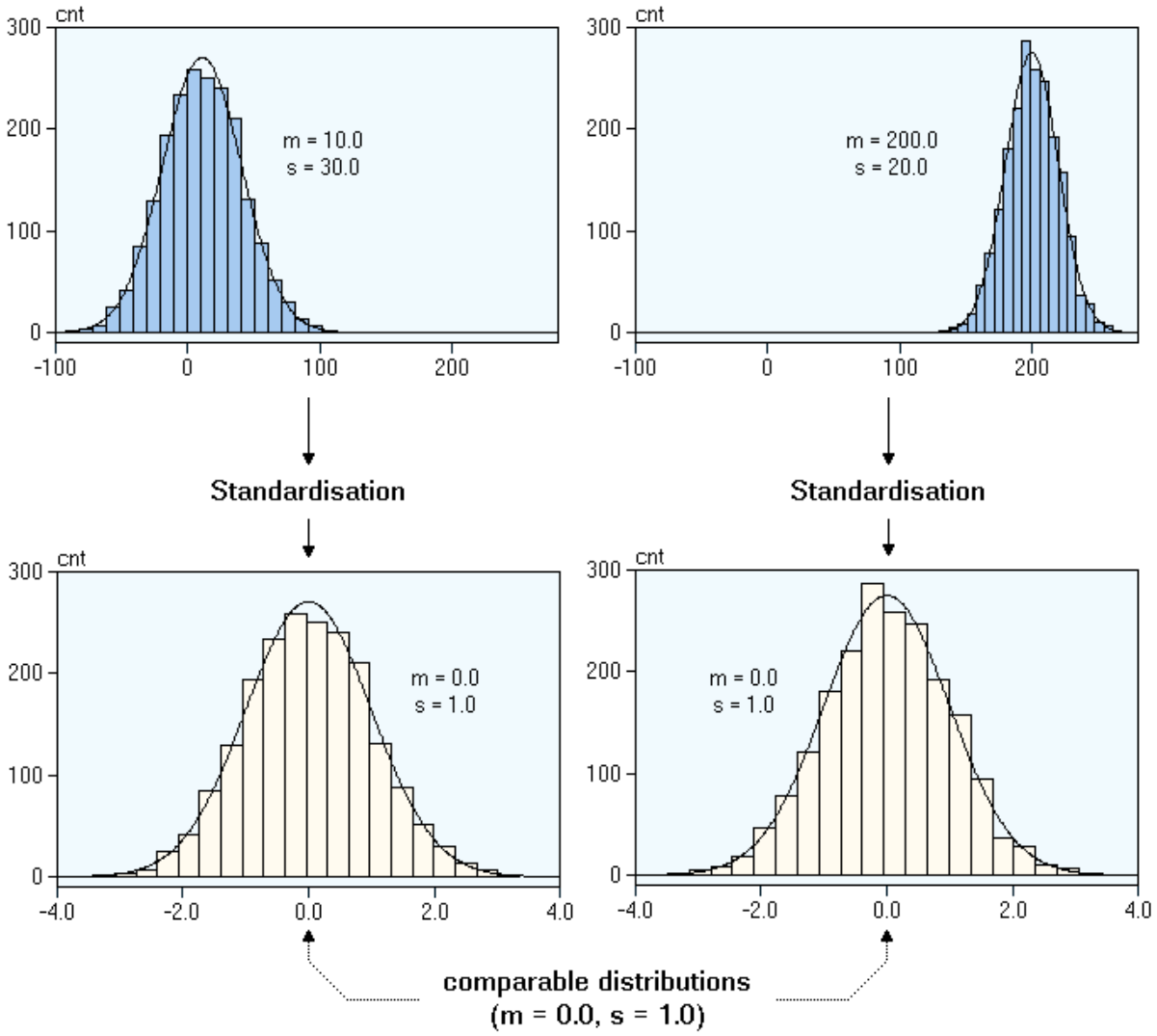
1) Standard scaling (Z-score normalization) – мера относительного разброса значения признака, которая показывает, сколько стандартных отклонений составляет его разброс относительного среднего значения признака:

$$z = \frac{x - \mu}{\sigma}, \qquad \mu = \frac{1}{N}\sum_{i}^{N} x_i\,, \qquad \sigma = \sqrt{\frac{1}{N}\sum_{i}^{N}(x_i - \mu)^2}$$

где μ – среднее значение признака,
σ – стандартное отклонение

- Standard scaling приводит произвольное распределение к распределению со средним значением 0 и дисперсией 1
- «Правило трёх сигм» – с вероятностью 0,9973 значение **нормально** распределённой случайной величины лежит в интервале $(\mu - 3\sigma, \mu + 3\sigma)$

3. Нормализация признаков

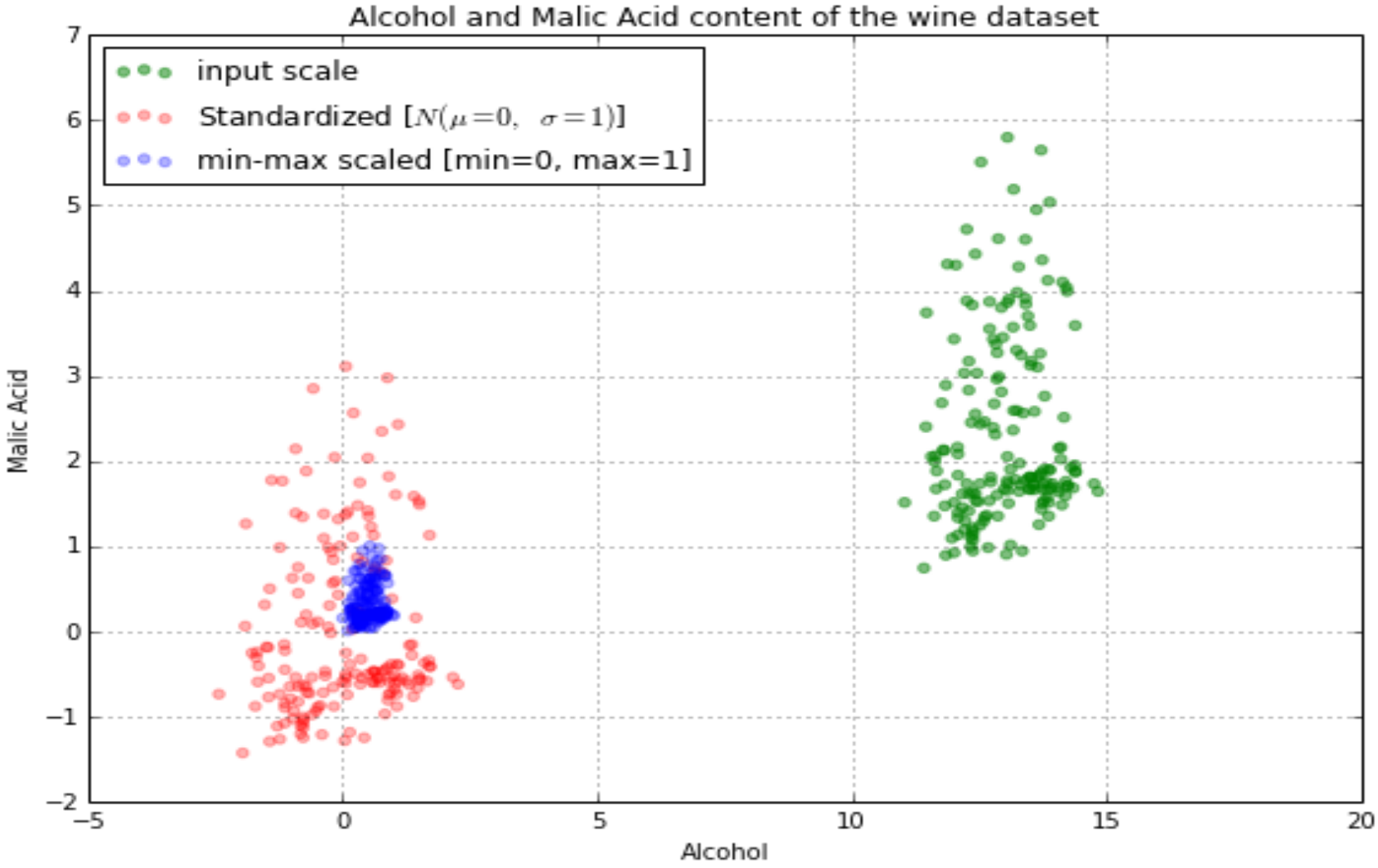


3. Нормализация признаков

2) Минимаксная нормализация (Min-Max scaling) – значения параметра приводятся к интервалу $[0,1]$:

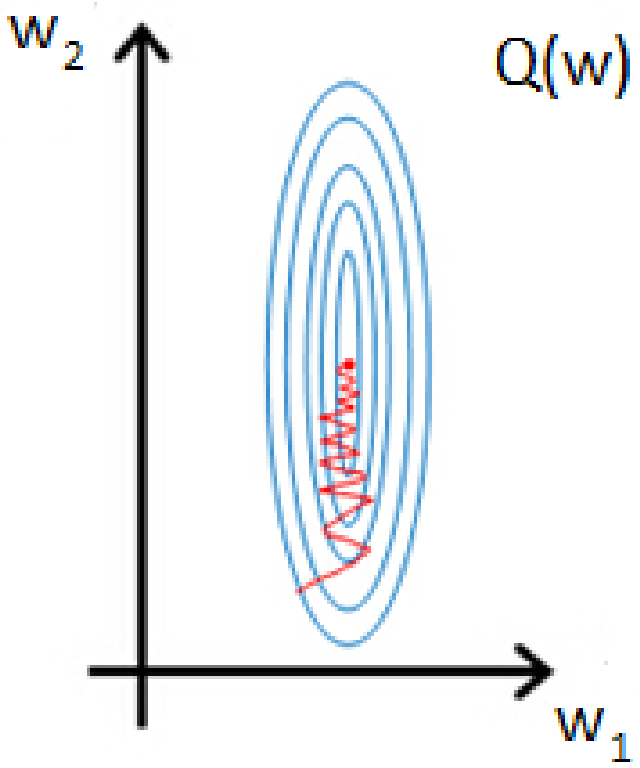
$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

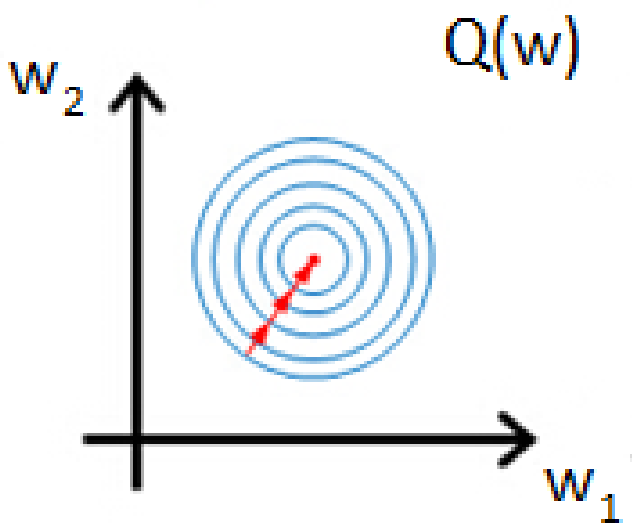
3. Нормализация признаков



3. Нормализация признаков

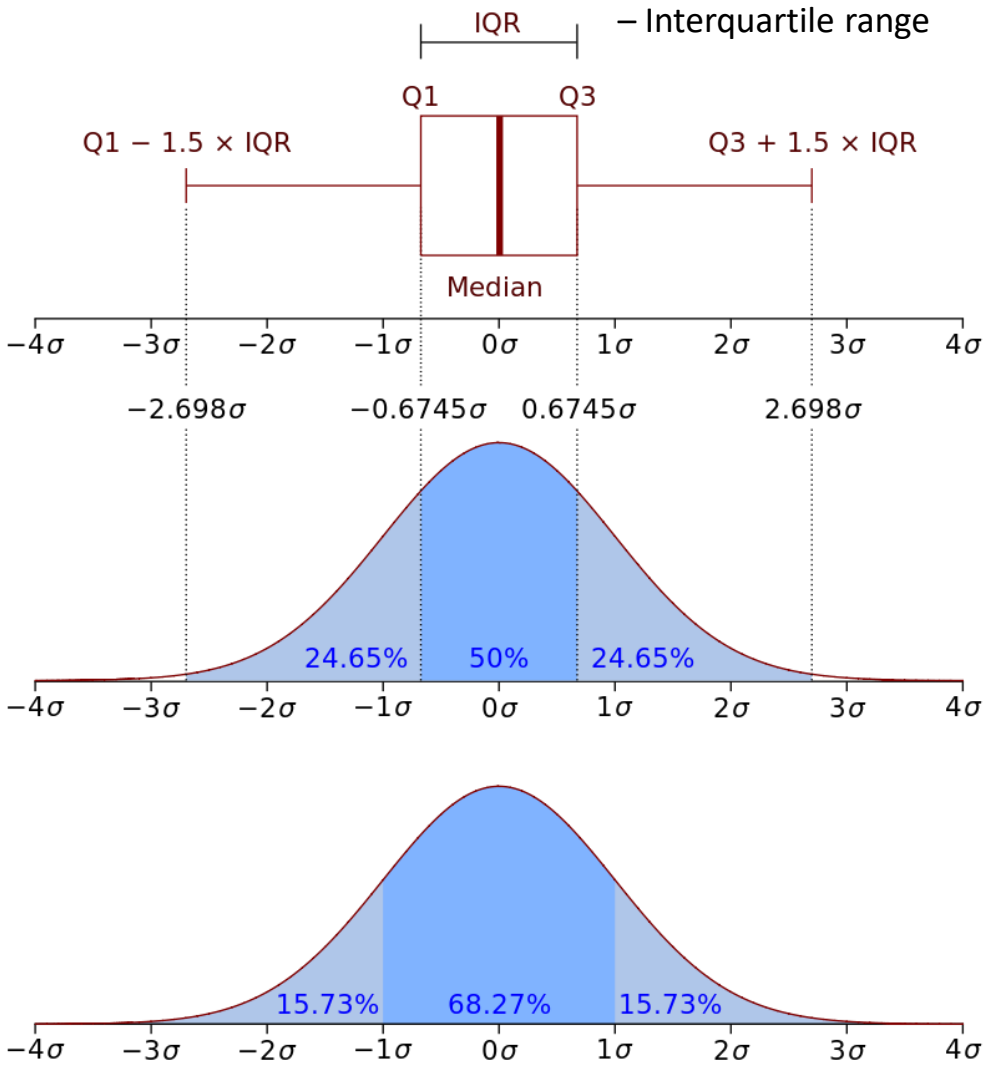
- Влияние на градиентный спуск:





4. Исключение выбросов

- *Выбросы* (outliers) – объекты, которые не являются корректными примерами, например,
  из-за неправильно посчитанных признаков, ошибки сбора данных и т. п.



5. Проверка на дубликаты

- Дубликаты:
  - объекты с одинаковыми признаками
    - иногда – это нормально
  - признаки с одинаковыми значениями (дублирование признаков)
    - корреляционный анализ
- Нечеткие дубликаты (near duplicates)

6. Исключение шумовых признаков

- *Шумовые признаки* (noisy features) – признаки, не имеющие отношения к целевой переменной и к решаемой задаче
- *Пример* – фаза луны в день первого экзамена

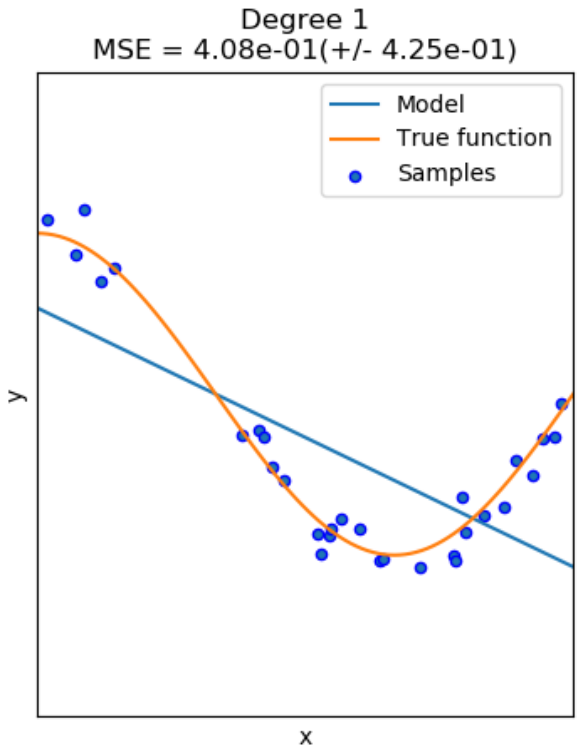
7. Обработка пропущенных значений

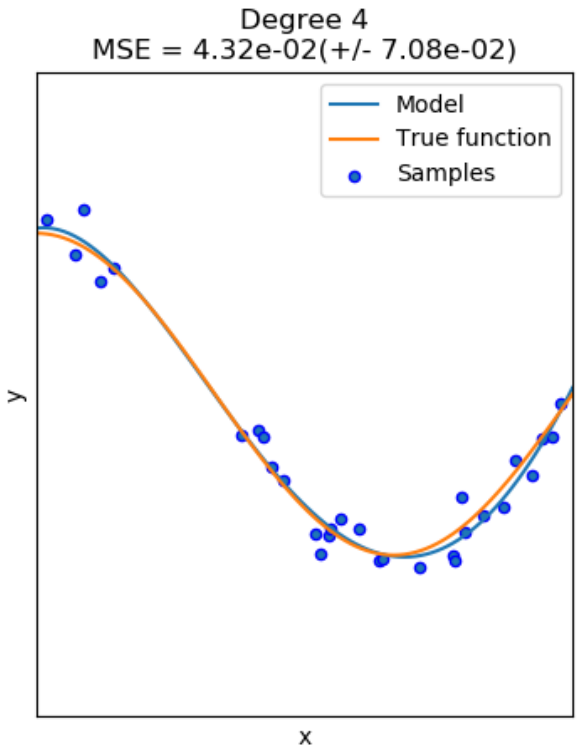
- Определить, почему данные пропущены
- Способы обработки:
  - Удалить все строки с пропущенными значениями
  - Проверить значимость признака
  - Подставить среднее (медиану)
  - Использовать модель машинного обучения для предсказания пропущенных значений
    - Линейная регрессия
    - Метод ближайшего соседа

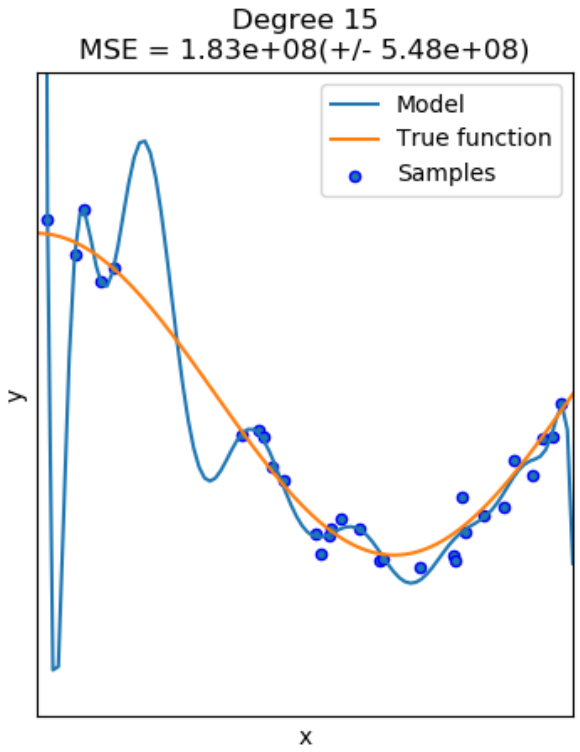
Переобучение

- Цель машинного обучения?
- Точная подстройка модели под обучающие данные не гарантирует, что модель будет хорошо обобщать результаты обучения на новые, неизвестные ей данные
- *Переобучение* (overfitting) – явление, при котором качество модели на новых, тестовых данных значительно хуже, чем на обучающих

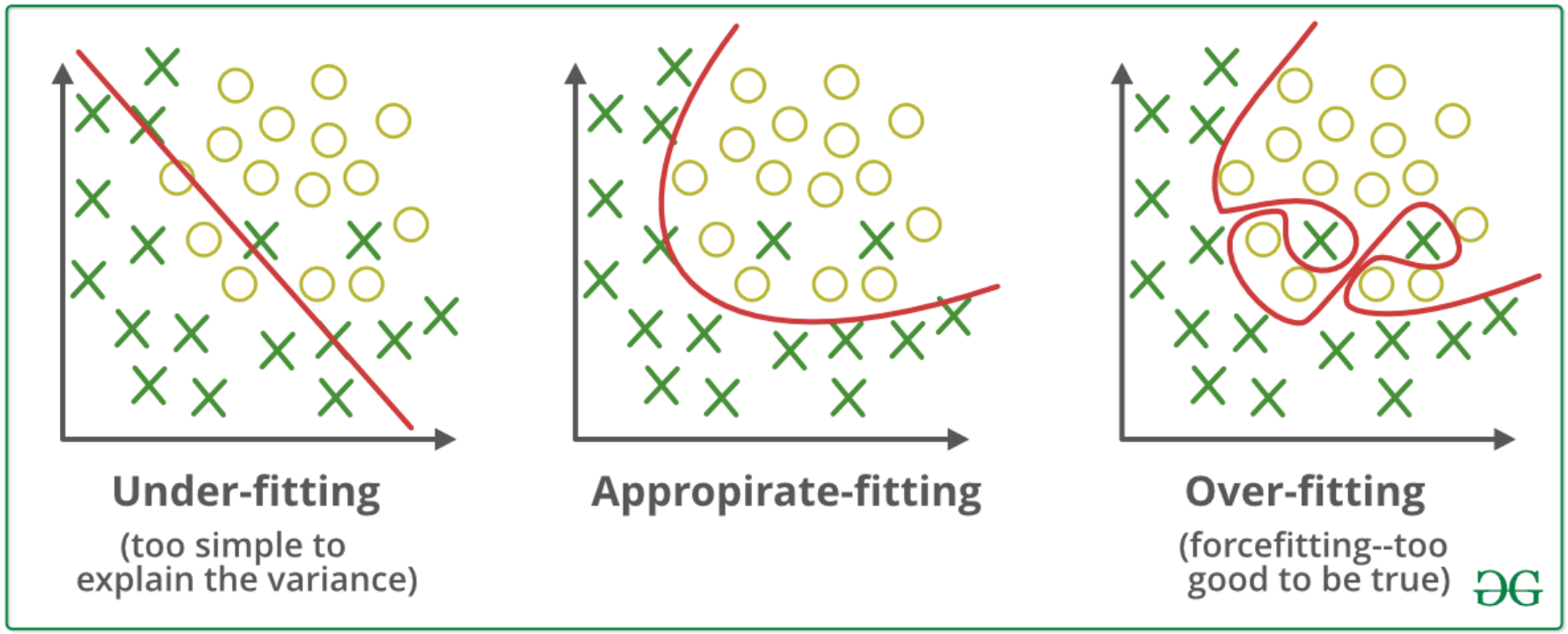
Переобучение







Переобучение



Оценка качества моделей

1. Отложенная выборка

- Имеющиеся размеченные данные (т. е. данные с известными ответами) разделяются на три части:
  - обучающую (тренировочную, training)
  - проверочную (validation, development, evaluation)
  - контрольную (тестовую, test, evaluation)
- Проблема: результат зависит от конкретного разбиения

Оценка качества моделей

2. Перекрестная проверка (cross-validation)

- Размеченные данные случайным образом разбиваются на k блоков $X_1, \dots, X_k$ примерно одинакового размера
- Обучается k моделей $a_1(x), \dots, a_k(x)$, причем i-ая модель обучается на всех блоках, кроме i-го, а оценивается на i-м блоке
- Для получения итоговой оценки все результаты усредняются:

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

Оценка качества моделей

Регуляризация

- Переобученные модели часто имеют большие значения весовых коэффициентов
- Для решения этой проблемы добавим к функционалу регуляризатор, который штрафует за слишком большую норму вектора весов:

$$Q_\alpha(\vec{w}) = Q(\vec{w}) + \alpha R(\vec{w}),$$

где α – параметр регуляризации

Регуляризация

Наиболее распространёнными являются L_2 и L_1-регуляризаторы:

- L_2-регуляризатор – гребневая регрессия (ridge regression):

$$R(\vec{w}) = \|\vec{w}\|_2^2 = \sum_{i=1}^{d} w_i^2$$

  - Перед обращением матрицы к ней добавляется диагональная матрица

- L_1-регуляризатор – Lasso (least absolute shrinkage and selection operator) regression:

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

Регуляризация

- Свободный коэффициент w_0 нет смысла регуляризовывать – если мы будем штрафовать за его величину, то получится, что мы учитываем некие априорные представления о близости целевой переменной к нулю и отсутствии необходимости в учёте её смещения

- Нормальное уравнение в случае с L_2-регуляризатором:

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

- Благодаря добавлению диагональной матрицы I к $X^T X$ данная матрица оказывается положительно определённой, и поэтому её можно обратить
- Таким образом, при использовании L_2-регуляризации решение всегда будет единственным