Занятие 2. Линейные методы регрессии. Часть 1.

Колмагоров Евгений
ml.hse.dpo@yandex.ru

14 октября 2024

План лекции

1. Алгоритм линейной регрессии
2. Функционал ошибки
3. Точные решение
4. Предобработка признаков
5. Градиентный спуск
6. Стохастическая вариация

Короткое напоминание

Машинное обучение – раздел науки о данных изучающий процесс, в результате которого компьютер способен показывать поведение на основе данных, которое в нём не было явно запрограммировано

Класс линейных моделей

Семейство линейных моделей A можно описать следующей формулой:

$$a(x, w) = w_0 + w_1 x_1 + \ldots w_d x_d$$

$w_0, w_1, \ldots, w_d$ – веса модели их $d + 1$ членов
$x_1, x_2, \ldots, x_d$ - признаки объекта

В частном случае для модельной задачи продажи дома формула выглядит, так:

$$a(x, w) = w_0 + w_1 x_1 + w_2 x_2$$

Добавление фиктивной переменной

Зачастую для представления модели используют сокращенную запись:

$$a(x, w) = w_0 + \sum_{i=1}^{d} w_i x_i$$

А если, добавить фиктивный признак $x_0 = 1$:

$$x : (x_1, \ldots, x_d) \to (1, x_1, \ldots, x_d)$$

То можно перейти к ещё более компактной записи:

$$a(x, w) = \sum_{i=0}^{d} w_i x_i$$

Представление через скалярное произведение

Если внимательней посмотреть на исходную формулу, то можно заметить, что формула суммы есть не что иное как скалярное произведение двух векторов:

- Вектора признаков объекта с фиктивной переменной $\boldsymbol{x} = (1, x_1, \ldots, x_d)$
- Вектора весов модели $\boldsymbol{w} = (w_0, \ldots, w_d)$

$$a(x, w) = \sum_{i=0}^{d} w_i x_i = (w, x)$$

Функционал ошибки

На прошлом занятии было введено понятие функционал ошибки, на основе которого происходит поиск весов модели:

$$Q(a, X) \to min$$

И был рассмотрен частный его случай с функционалом ошибки MSE:

$$Q(a, X) = \frac{1}{N} \sum_{i=1}^{N} ((\mathbf{w}, \mathbf{x_i}) - y_i)^2 \to min_{w_0, \ldots, w_d}$$

Поиск оптимального значения весов для MSE

Если в качестве функционала ошибки для задачи используется функция MSE, то можно получить точное аналитическое решение для оптимальных весов w^*

Так как предсказание для одного объекта: *a(**x**, **w**) = (**w**, **x**)*, то предсказание для N объектов, есть произведение матрицы объект-признаки ***X*** на вектор весов ***w*** размера d x 1

$$\hat{Y} = Xw$$

Тогда функционал *Q(a, X)* можно представить в матричном виде как

$$Q(a, X) = \frac{1}{N}\sum_{i=1}^{N}((\mathbf{w}, \mathbf{x_i}) - y_i)^2 = \frac{1}{N}\sum_{i=1}^{N}(\hat{y_i} - y_i)^2 = \frac{1}{N}(\hat{Y} - Y)^T \cdot (\hat{Y} - Y)$$

Матричное представление функционала

Что эквивалентно оптимизации следующего выражения:

$$Q(x, X) = \frac{1}{N}(\hat{Y} - Y)^T \cdot (\hat{Y} - Y) = \frac{1}{N}(Xw - Y)^T \cdot (Xw - Y) \to min_{w_0, \ldots, w_d}$$

Необходимое условие экстремума

Теорема: *Если функция имеет в точке локального экстремума производную, то эта производная равна нулю.*

Из которой следует, что $\boldsymbol{w}^*$ достигается там, где

$$\nabla_w Q(a, X) = 0$$

Напоминание!:

$$\nabla_w Q(a, X) = (\frac{\partial Q}{\partial w_0}, \ldots, \frac{\partial Q}{\partial w_d})$$

Достаточное условие экстремума

Теорема: *Если в стационарной точке* x^* *второй дифференциал В положительно определенная квадратичная форма, то* x^* *– точка локального минимума, если второй дифференциал отрицательно определенная квадратичная форма, то* x^* *– точка локального максимума.*

Из данной теоремы, следует, что если точка $\boldsymbol{w}^*$ – точка строгого минимума, то

$$\frac{\partial^2 Q(a,X)}{\partial w^2} > 0$$

Правила векторного/скалярного дифференцирования

Scalar derivative			Vector derivative		
$f(x)$	$\rightarrow$	$\frac{\mathrm{d}f}{\mathrm{d}x}$	$f(\mathbf{x})$	$\rightarrow$	$\frac{\mathrm{d}f}{\mathrm{d}\mathbf{x}}$
bx	$\rightarrow$	b	$\mathbf{x}^T\mathbf{B}$	$\rightarrow$	$\mathbf{B}$
bx	$\rightarrow$	b	$\mathbf{x}^T\mathbf{b}$	$\rightarrow$	$\mathbf{b}$
x^2	$\rightarrow$	$2x$	$\mathbf{x}^T\mathbf{x}$	$\rightarrow$	$2\mathbf{x}$
bx^2	$\rightarrow$	$2bx$	$\mathbf{x}^T\mathbf{B}\mathbf{x}$	$\rightarrow$	$2\mathbf{B}\mathbf{x}$

Векторная производная функционала

Посчитаем градиент функционала Q от вектора весов ***w***

$$\nabla_w Q(a, X) = \nabla_w \tfrac{1}{N}(Xw - Y)^T \cdot (Xw - Y) = \tfrac{1}{N}\nabla_w (w^T X^T Xw - w^T X^T Y - Y^T Xw + YY^T) =$$

$$= \{w^T X^T Y \text{ и } Y^T Xw \in \mathbb{R}^{1\times 1}\} = \frac{1}{N}\nabla_w (w^T X^T Xw - 2w^T X^T Y + Y^T Y)$$

Применим правила векторного дифференцирования 1 и 4 и теоремой о необходимом условии экстремума:

$$\nabla_w Q(a, X) = \tfrac{1}{N}(2X^T Xw - 2X^T Y) = 0$$

Аналитическое решение

Из полученного равенства выразим вектор весов $\boldsymbol{w}$

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

Чтобы убедиться, что $\boldsymbol{w}^*$ – точка минимума функционала достаточно взять второй раз градиент для $\boldsymbol{Q}$ по $\boldsymbol{w}$ и получить $\boldsymbol{X^T X}$ в ответе.

Подумайте, почему $\boldsymbol{X^T X} > 0$?

Вспомним модельную задачу

Предположим, что мы хотим предсказать стоимость дома - целевая переменная ***y*** - по двум его признакам:

- x_1 - его площади
- x_2 - количество комнат

Если применить линейную регрессию к данной задаче, то цена дома будет определяться по формуле:

$$\hat{y} = w_0 + w_1 x_1 + w_2 x_2$$

Добавим новый тип признака в исходную модель

Предположим, что теперь предсказания будут строиться по ещё одному признаку x_3 – район города, например, Барвиха, Дудкино, Мамыри и тд.

$$\hat{y} = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_3$$

Проблема: Признак район x_3 – это не число,

а название, строкового типа.

One-hot encoding

Одним из стандартных методов представления строковых данных является использование one-hot кодирования, в котором каждое уникальное значение категориального типа представляет собой бинарный признак в виде индикатора.

В данном случае создаем новые числовые столбцы, каждый из которых является индикатором района.

Кодирование района

Район
Дудкино
Барвиха
Мамыри
…
Барвиха

→

Мамыри	Дудкино	Барвиха
0	1	0
0	0	1
1	0	0
…	…	…
0	0	1

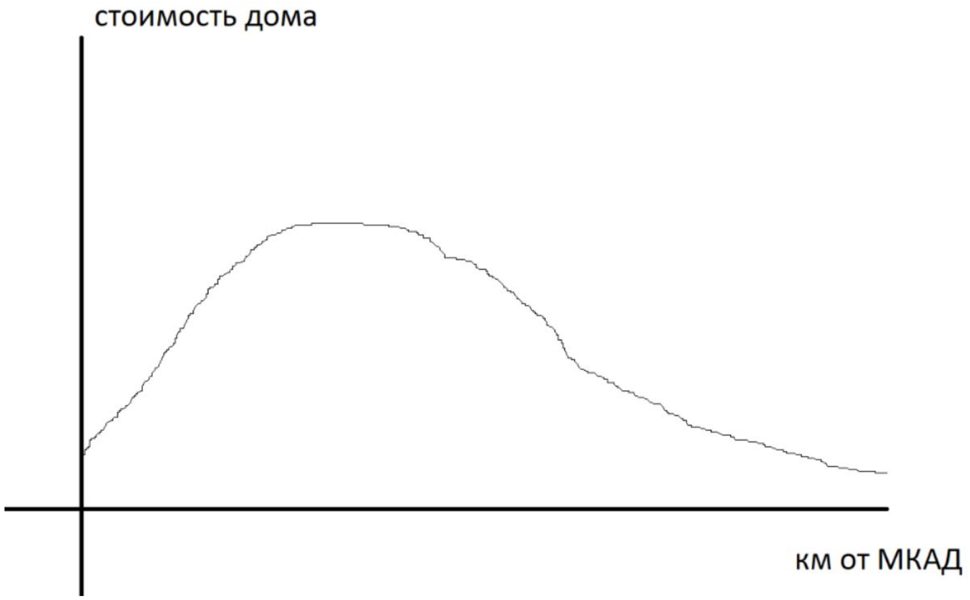
И формула приобретет следующий вид:

$$a(x, w) = w_0 + w_1 x_1 + w_2 x_2 + w_{31} x_{\text{Мамыри}} + w_{32} x_{\text{Дудкино}} + w_{33} x_{\text{Барвиха}}$$

Ещё один числовой признак

Предположим, что в качестве ещё одного фактора в модель будет добавлен признак x_4 – удалённость от МКАД.

Проблема: В отличии от первых двух числовых признаков, данный тип представляет собой не монотонную зависимость от целевой переменной ***y***

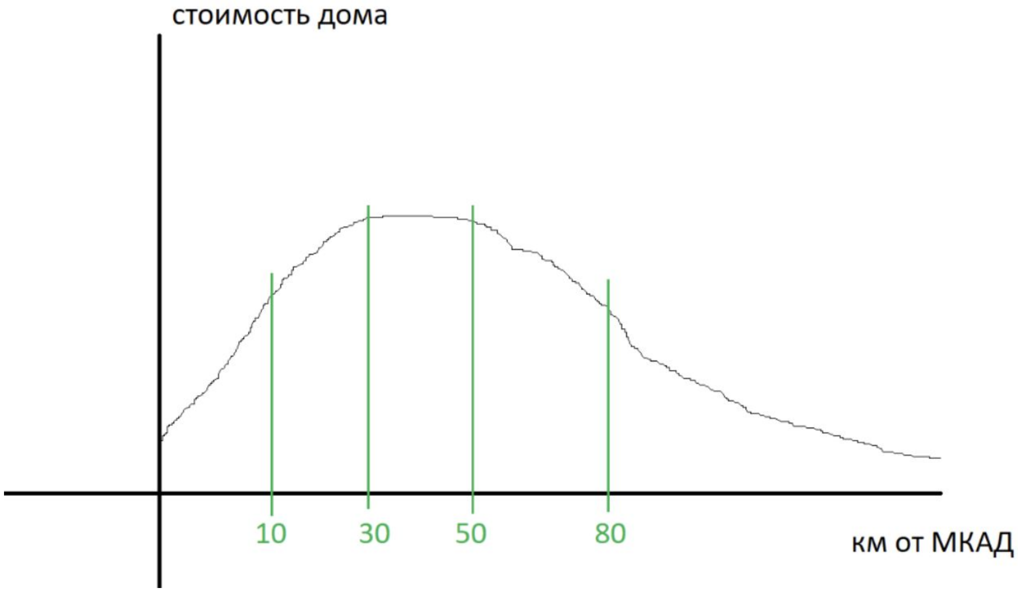


Бинаризация на участки монотонности

Решение: разбить область значений признака x_4 на бины.

И для каждого бина ввести свою переменную:

- $x_{[0,\ 10)}$ – равен 1, если дом находится в пределах 10 км от МКАД, иначе 0
- $x_{[10,\ 30)}$ - равен 1, если дом находится в пределах от 10 до 30 км МКАД, иначе 0



Формула линейной регрессии с бинами

Теперь полная формула приобретает следующий вид:

$$a(x, w) = w_0 + w_1 x_1 + w_2 x_2 + w_{31} x_{\text{Мамыри}} + w_{32} x_{\text{Дудкино}} + w_{33} x_{\text{Барвиха}}$$
$$+ w_{41} x_{[0,10)} + w_{42} x_{[10,30)} + w_{43} x_{[30,50)} + w_{44} x_{\geq 50}$$

Вернёмся к аналитическому решению

Посмотрим внимательней на полученную формулу аналитического решения

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

Какие есть негативные особенности у данной формулы?

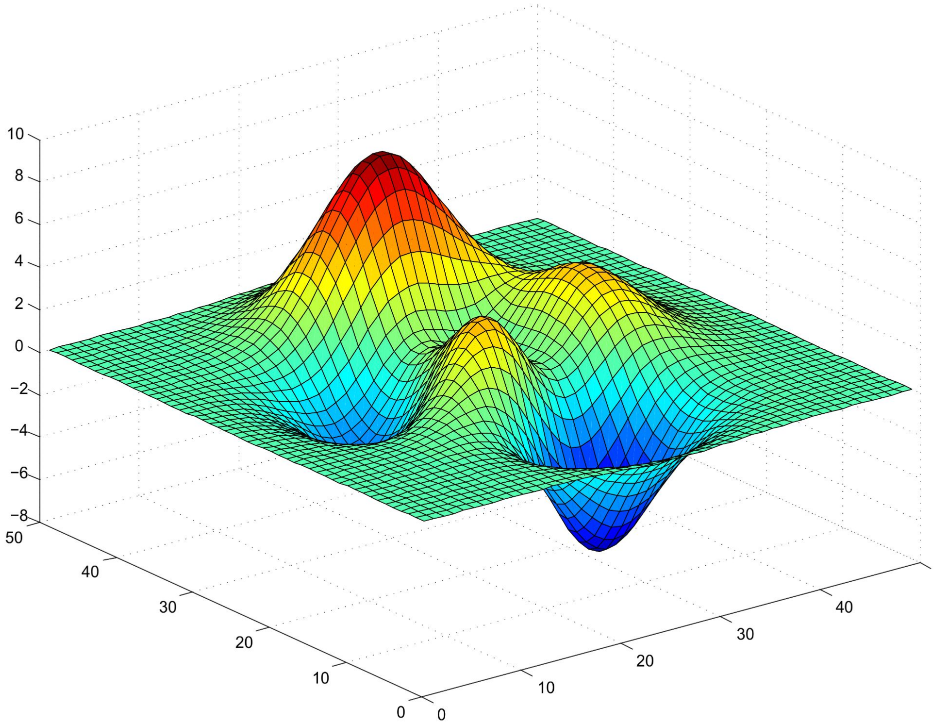
Минусы аналитического решения

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

- Обращение матрицы - вычислительно сложная операция ($O(d^3)$), где d - число признаков
- Матрица $X^T X$ - может быть вырожденной или плохо обусловленной
- Если будет другой функционал ошибки отличный от MSE, то нужно искать новую формулу, либо её может не быть в аналитическом виде

Поиск весов через оптимизации функционала ошибки

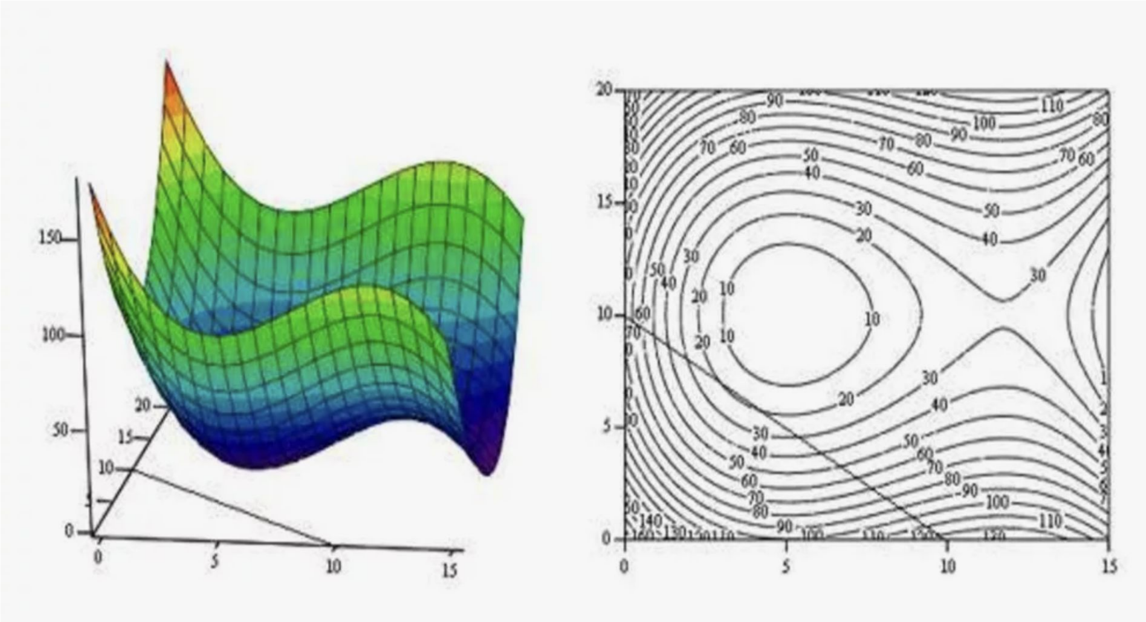
Зачастую задача поиск оптимального значения весов модели ***w*** может быть решена методами оптимизации математических функций, суть которых состоит в том, чтобы найти точку минимума функционала ошибки в многомерном пространстве весов.



Небольшое математическое напоминание

Линия уровня - множество точек D на координатной плоскости, в которых функция принимает одинаковые значения.

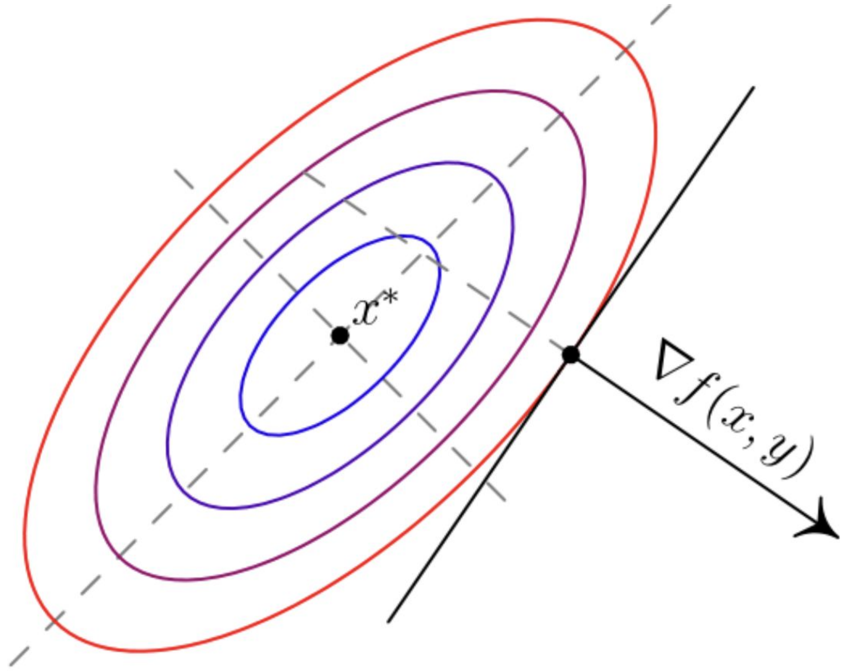
$$f(x_1, x_2, \ldots, x_d) = Const$$
$$\forall (x_1, x_2, \ldots, x_d) \in D$$



Небольшое математическое напоминание

Определение: Градиент функции – это вектор, в направлении которого функция **растёт** быстрее всего.

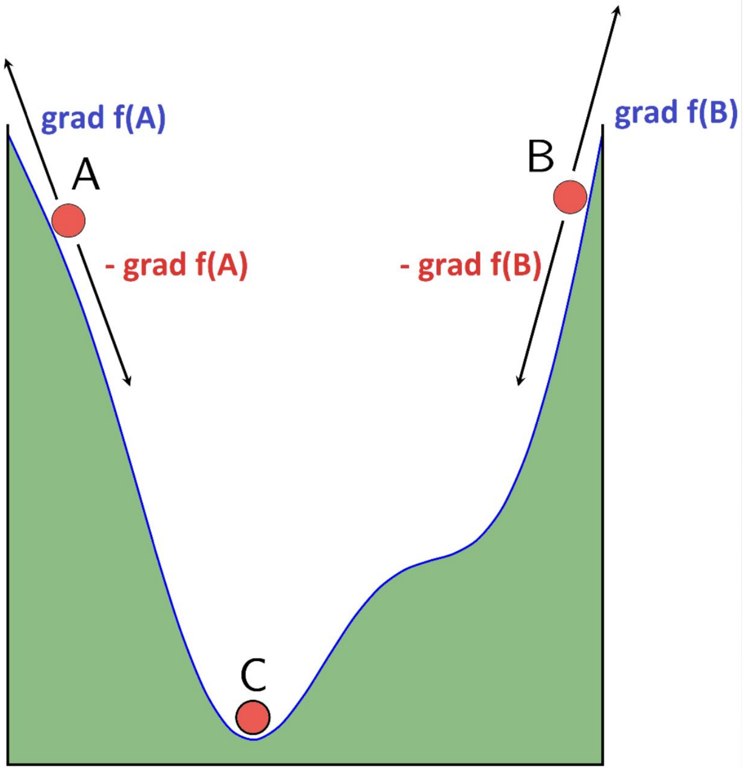
Вопрос: Почему градиент всегда перпендикулярен линиям уровня?



$$f(x, y) = \frac{3}{2}(x - y)^2 + \frac{1}{3}(x + y)^2$$

Зеркальное отражение градиента

Определение: Антиградиент (вектор противоположный градиенту) – это
вектор, в направлении которого функция
убывает быстрее всего



Метод градиентного спуска

Наша задача при обучении модели – найти такие веса ***w*,** на которых достигается минимум функции ошибки Q(a, X) .

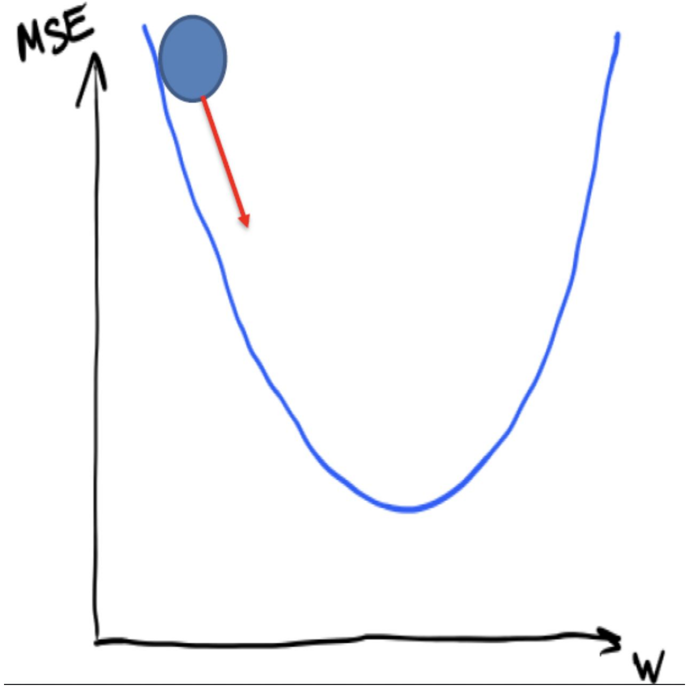
Идея метода градиентного спуска:

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

То есть на каждом шаге движемся в направлении уменьшения ошибки.

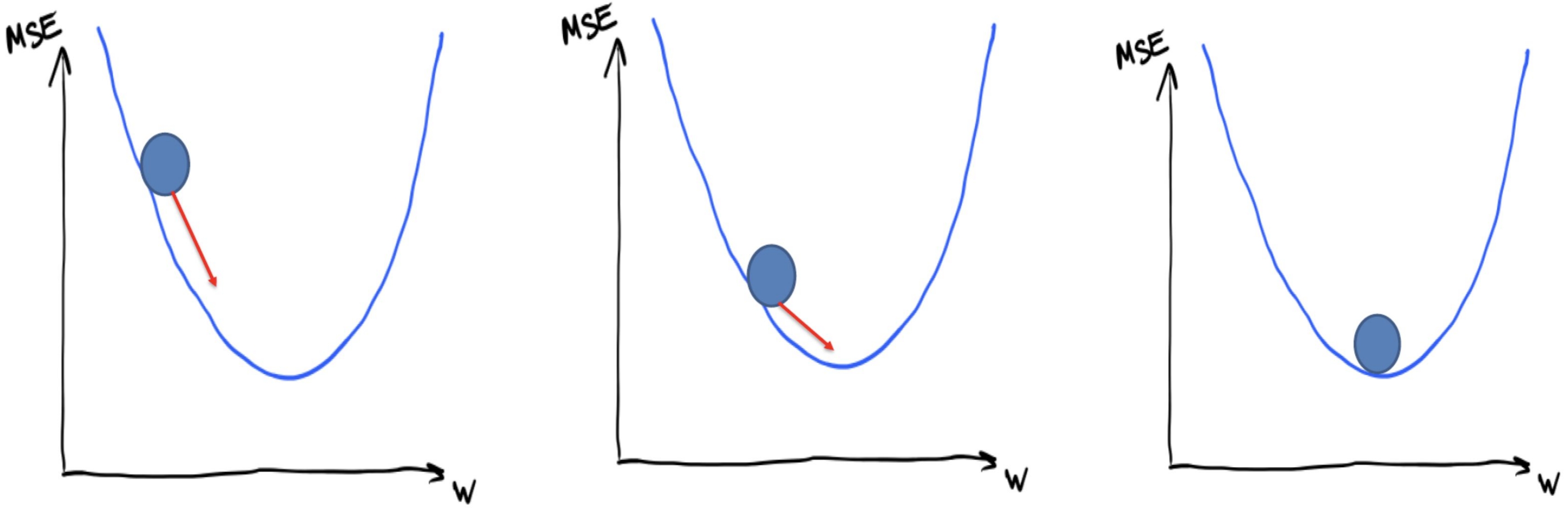
Средне квадратичная функция потерь

В простейшем случае, если ошибка среднеквадратичная, то её график – это парабола.



Итеративный поиск минимума

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



Обновление параметров *w*

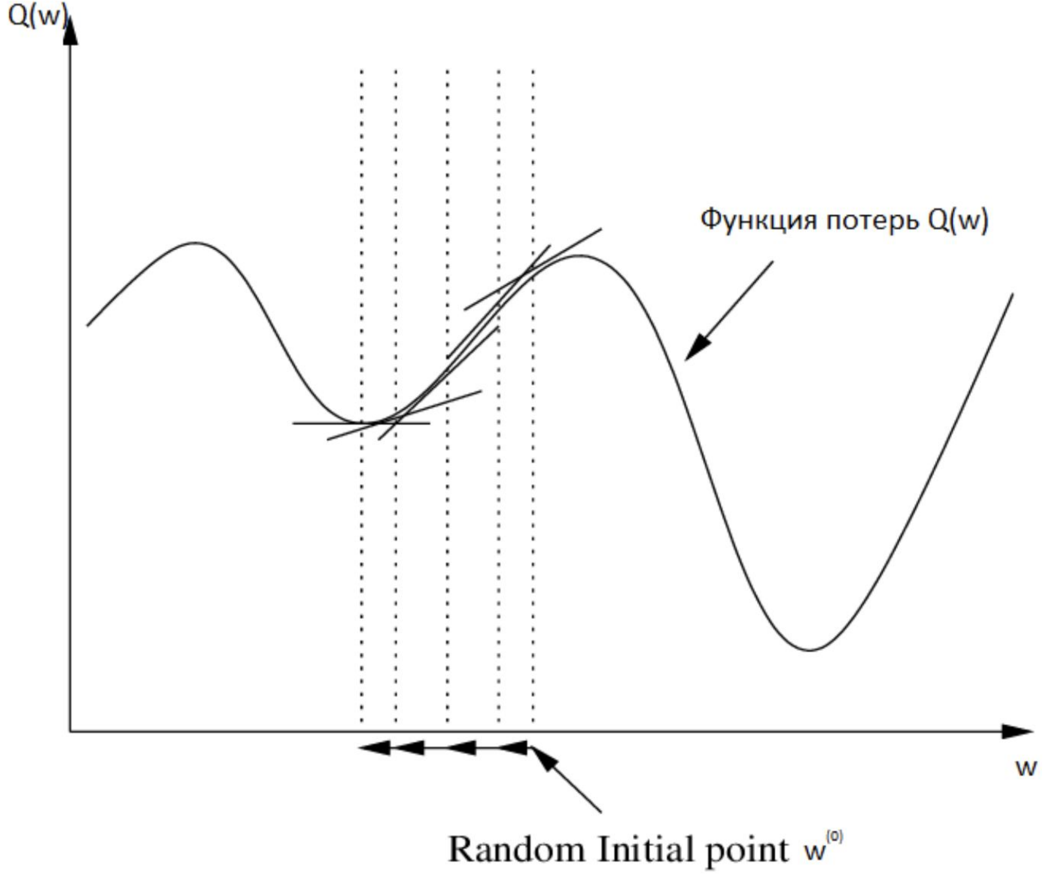
Так как смещение вдоль вектора антиградиента уменьшает функционал ошибки Q, то при добавлении его к весу ***w*** будем переходить в новую точку пространства весов ***w***, где функция ошибки меньше первоначальной.

Итеративный метод градиентного спуска:

- Начальная инициализация весов $\boldsymbol{w}^{(0)}$
- На каждом шаге обновляем вес, добавляя ***-grad Q(a, X)***

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

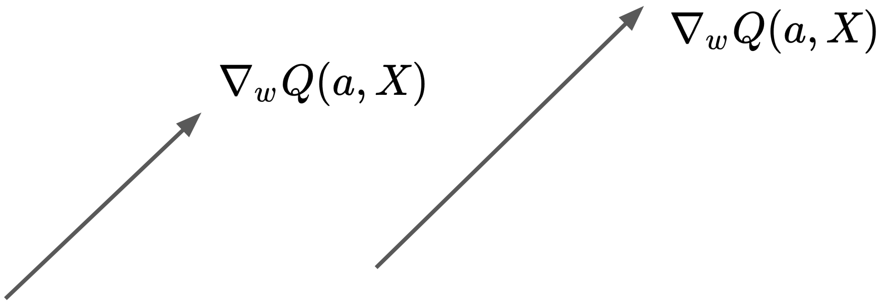
Итеративный метод градиентного спуска



Скорость спуска

Важно отметить, что градиент - это вектор, который лишь указывает **направление**, того куда необходимо сместить вектор весов, а не то насколько их абсолютно нужно **изменить**

В этом смысле градиент с векторов (1, 2, 3) абсолютно идентичен вектору (2, 4, 6), поскольку будут указывать на одно и тоже направление.



Скорость спуска

Для того, чтобы контролировать, каким должен быть шаг вдоль направления градиента, вводят параметр $\boldsymbol{\eta}$ - величина градиентного шага (learning rate)

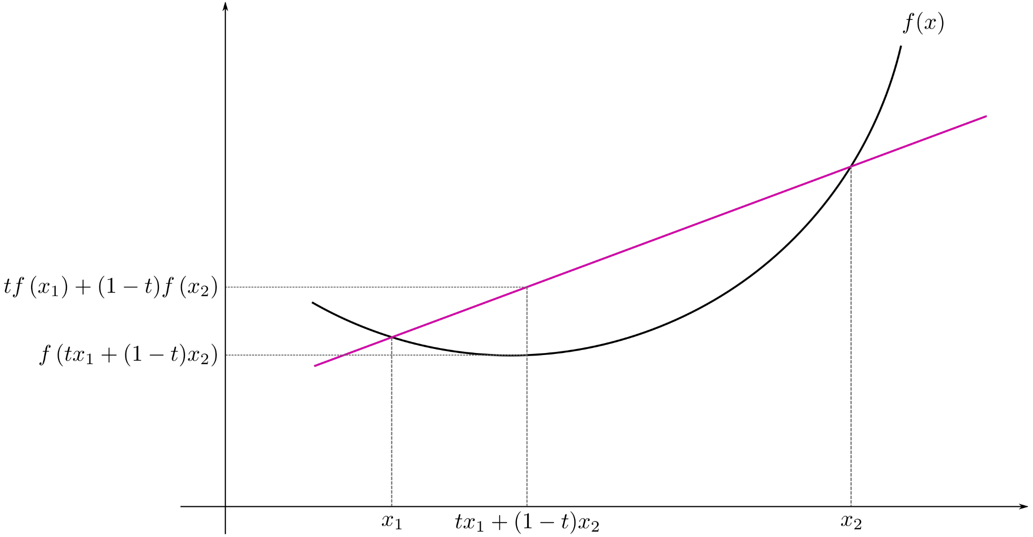
Градиентный спуск с шагом обучения $\boldsymbol{\eta}$:

$$w^k = w^(k-1) - \eta \nabla_w Q(a, X)|_{w=w^(k-1)}$$

Теорема о выпуклой функции

Теорема: Если функция Q 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****.

Пример выпуклой функции :



Градиентный спуск для линейной регрессии

При выводе аналитического решения вычислялся градиент в Q в матричном виде, попробуем теперь вычислить его в виде суммы:

$$\nabla_w Q(a, X) = \nabla_w[\frac{1}{N}\sum_{i=1}^{N}((\mathbf{x_i}, \mathbf{w}) - y_i)^2] = \frac{1}{N}\sum_{i=1}^{N}\nabla_w[((\mathbf{x_i}, \mathbf{w}) - y_i)((\mathbf{x_i}, \mathbf{w}) - y_i)] =$$

$$= \frac{1}{N}\sum_{i=1}^{N}[\nabla_w(\mathbf{w}^T\mathbf{x_i} - y_i)(\mathbf{w}^T\mathbf{x_i} - y_i)] = \{\text{дифференцирования произведения и 2 правило}\} =$$

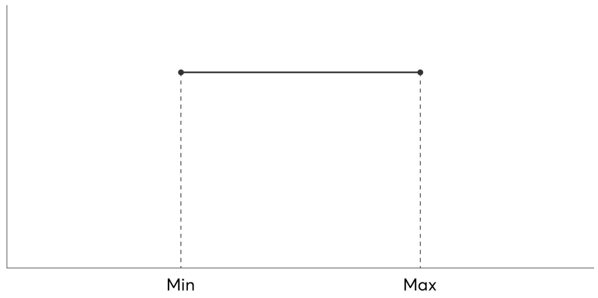
$$= \frac{2}{N}\sum_{i=1}^{N}[\mathbf{x_i}(\mathbf{w}^T\mathbf{x_i} - y_i)]$$

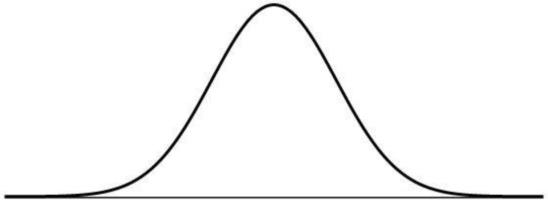
Шаг градиентного спуска:

$$\mathbf{w}^k = \mathbf{w}^{k-1} - \frac{2\eta}{N}\sum_{i=1}^{N}[\mathbf{x_i}((\mathbf{w}^{k-1}, \mathbf{x_i}) - y_i)]$$

Варианты инициализации весов

- Нулевая инициализация: $w_j = 0$, $j = 0, 1, \ldots, d$
- Небольшое случайное значение, из некоторого распределения
  - $w_j = N(0, \varepsilon)$, $j = 0, 1, \ldots, d$
  - $w_j = Uniform(-\varepsilon, \varepsilon)$, $j = 0, 1, \ldots, d$





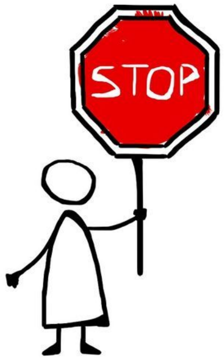
Критерии останова

- Остановка через заранее заданное число шагов M
- Остановка на основе изменения функционала

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

- Остановка через изменение весов

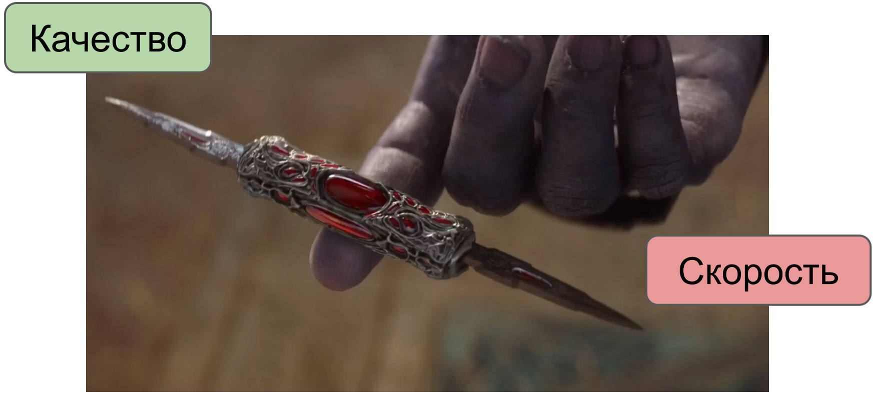
$$||w^k - w^{k-1}|| < \epsilon$$



Влияние градиентного шага

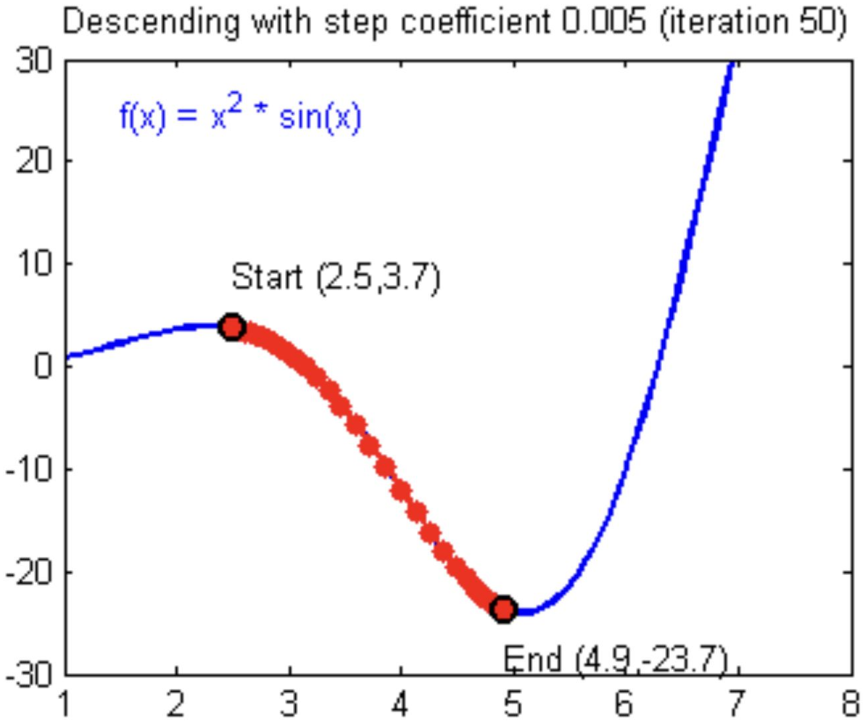
Значение шага η влияет на скорость сходимости алгоритма градиентного спуска, но и на точность получаемого решения.

Поэтому при выборе конкретного значения η необходимо выбирать компромисс между качеством и скоростью.



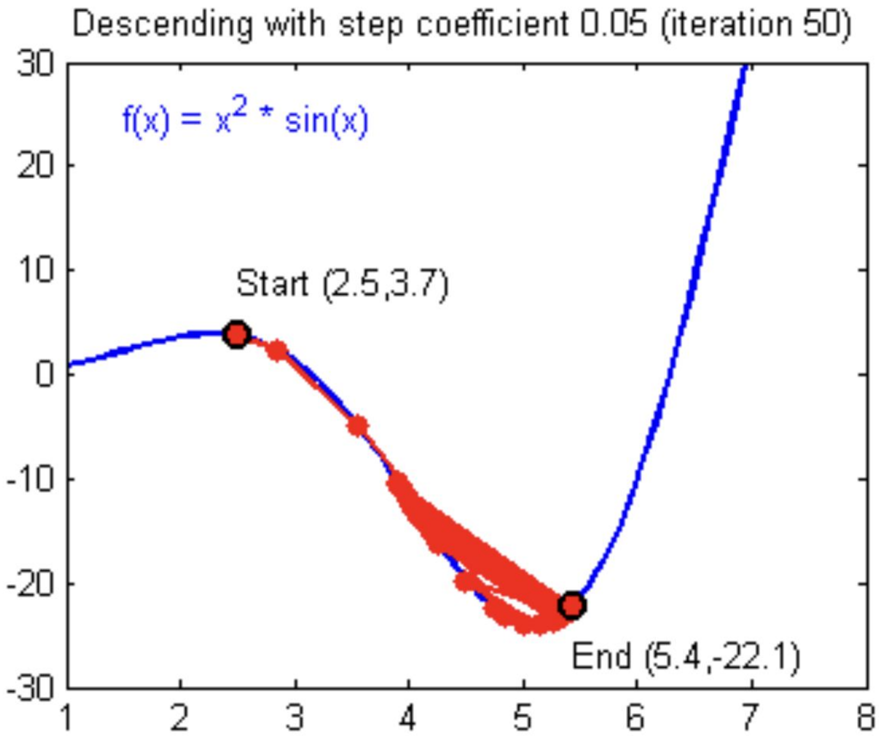
Инициализация в пользу качества

Для того чтобы получаемая точка минимума была ближе к её точному значению, необходимо ставить шаг обучения η как можно **меньше**



Инициализация в пользу скорость

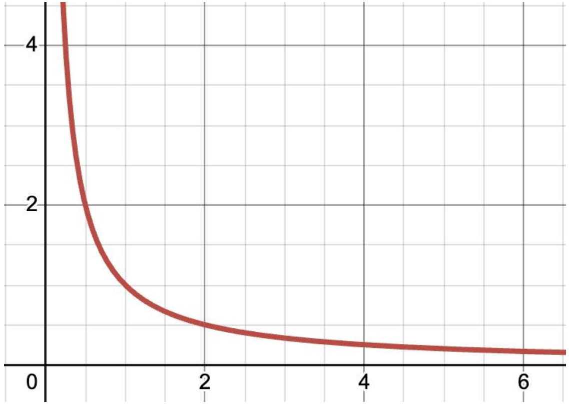
Для того чтобы уменьшить количество шагов алгоритма, необходимо ставить шаг обучения $\boldsymbol{\eta}$ как можно **больше**



Стратегии выбора темпа обучения

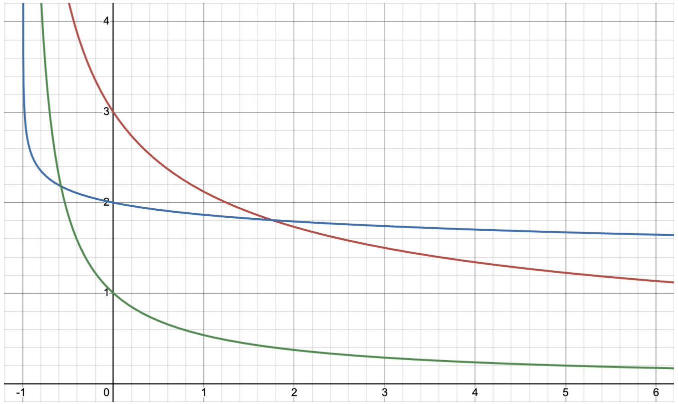
В зависимости от номера итерации k можно варьировать значение $\boldsymbol{\eta}$, т.е. теперь шаг обучения будет зависеть от итерации $\boldsymbol{\eta}_k$:

- $\eta_k = Const$
- $\eta_k = \frac{1}{k}$
- $\eta_k = \lambda \cdot (\frac{s_0}{s_0+k})^p$



2 подход

3 подход



Вычислительные особенности градиентного спуска

Если посмотреть на формулу градиента ниже

$$\nabla_w Q(a, X) = \sum_{i=1}^{N} [\mathbf{x_i}(\mathbf{w}^T \mathbf{x_i} - y_i)]$$

То можно заметить, что вычисление градиента по всем объектам тренировочной выборки, достаточно вычислительно сложная процедура - необходимо вычислять градиент функционала $Q(a, x_i)$ по каждому объекту и далее суммировать по всем объектам выборки

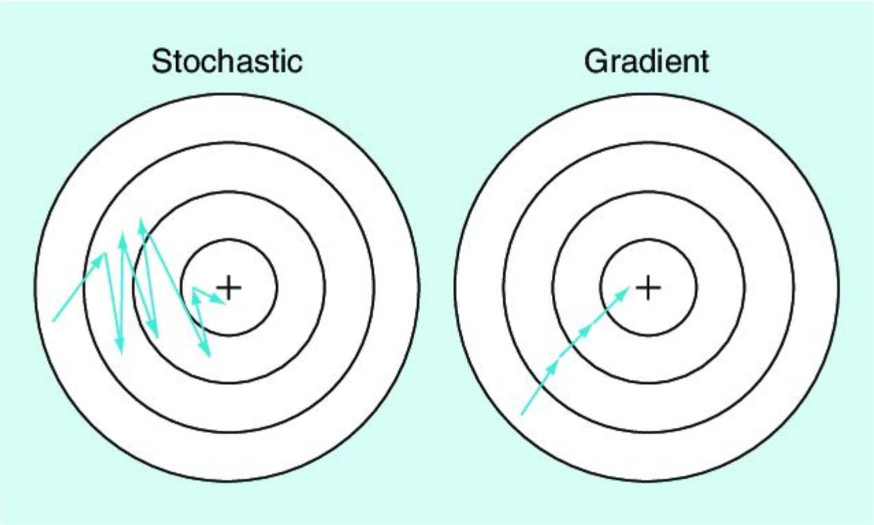
Стохастическая вариации

Одной из разновидностей градиентного спуска является стохастический градиентный спуск, в котором вектор градиента считается по одному объекту $\nabla_w Q(a, \mathbf{x_i})$, после чего вектор весов смещается в сторону его антиградиента

$$w^k = w^{k-1} - \eta_k \nabla_w Q(a, \mathbf{x_i})$$

Почему работает

Обычный градиент и стохастический градиентный спуск имеют одинаковое математическое ожидание.



Пусть математическое ожидание градиента на первом объекте

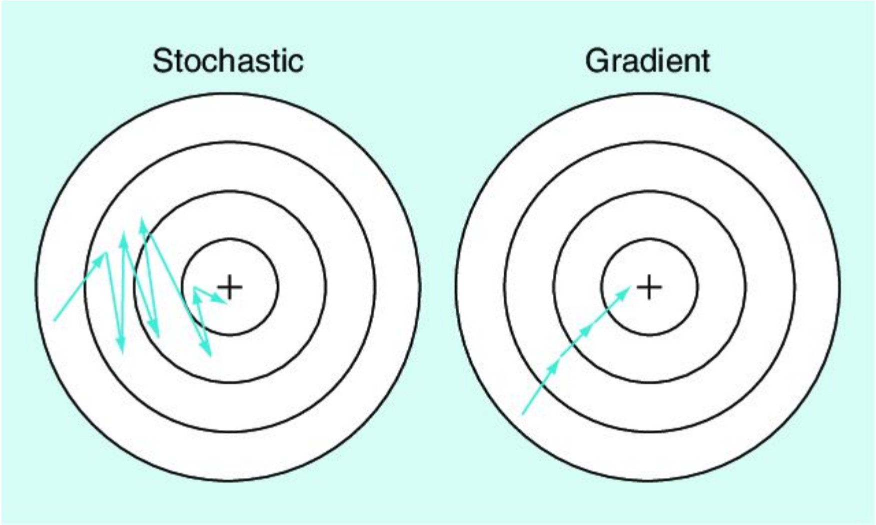
$$E[\nabla_w Q(a, \mathbf{x_1})] = q$$

Тогда мат.ожидание по всей выборке

$$E[\nabla_w Q(a, X)] = E[\tfrac{1}{N}\sum_{i=1}^{N} \nabla_w Q(a, \mathbf{x_i})] =$$
$$= \{\text{коммутативность} \sum \text{ и } E\} =$$
$$= \tfrac{1}{N}\sum_{i=1}^{N} E[\nabla_w Q(a, \mathbf{x_i})] = \{x_1, \ldots, x_N \sim i.i.d.\} = \tfrac{1}{N} \cdot N \cdot E[\nabla_w Q(a, \mathbf{x_1})] = q$$

Теорема о стохастическом градиентном спуске

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



Попробуйте сравнить дисперсии стохастического и полного градиента

Мини-батч

Промежуточное решение между классическим градиентным спуском и стохастическим вариантом:

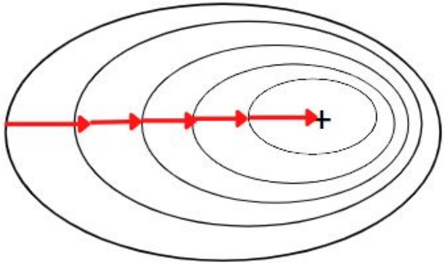
• Выбираем batch size (например, 32, 64 и т.д.). Разбиваем все пары объект-ответ на группы размера M.

• На i-й итерации градиентного спуска вычисляем $\nabla Q(w)$ только по объектам i-го батча:

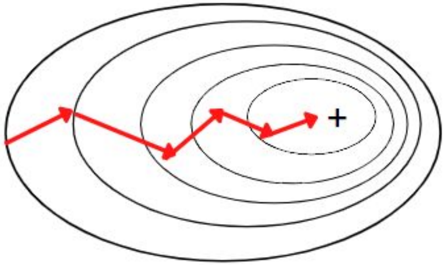
$$\mathbf{w}^k = \mathbf{w}^{k-1} - \frac{\eta}{M} \sum_{k=1}^{M} \nabla_w Q(a, x_{i_k})$$

Сравнение методов

Batch Gradient Descent



Mini-Batch Gradient Descent



Stochastic Gradient Descent