

# Занятие 3

## Градиентные методы

Елена Кантонистова

[elena.kantonistova@yandex.ru](mailto:elena.kantonistova@yandex.ru)

# ПЛАН ЗАНЯТИЯ

- Заканчиваем с задачей про Титаник
- Разбираем градиентный спуск
- Обучаем регрессию в Python

# ЛИНЕЙНАЯ РЕГРЕССИЯ

Линейная регрессия:

$$a(x) = w_0 + w_1x_1 + \dots + w_dx_d = (w, x)$$

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)

# ЛИНЕЙНАЯ РЕГРЕССИЯ

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)

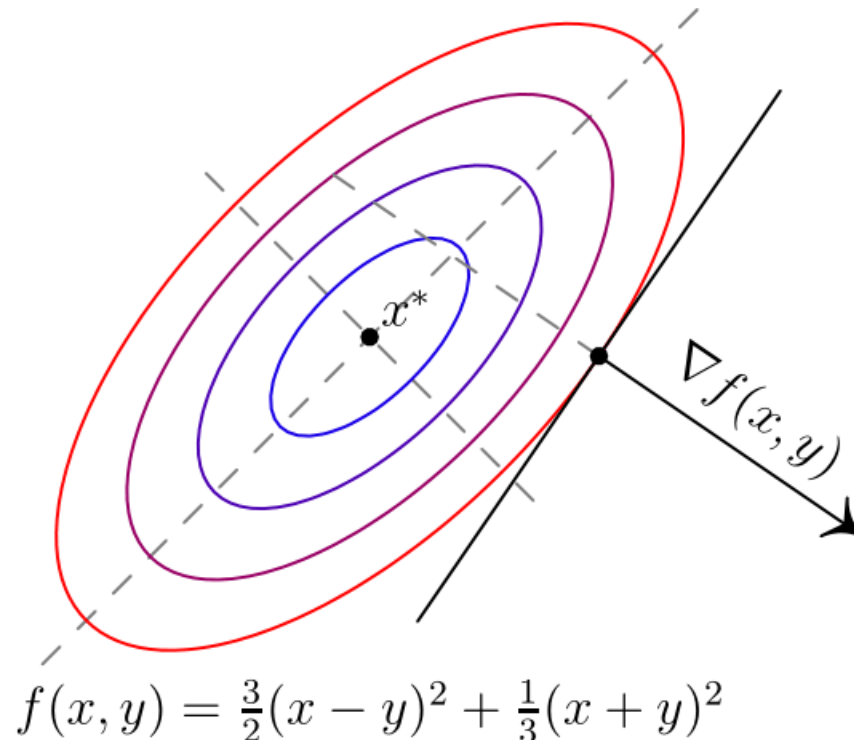
Подбираем веса  $w_0, w_1, \dots, w_d$  так, чтобы ошибка модели была минимальна.

**Как это сделать?**

# ТЕОРЕМА О ГРАДИЕНТЕ

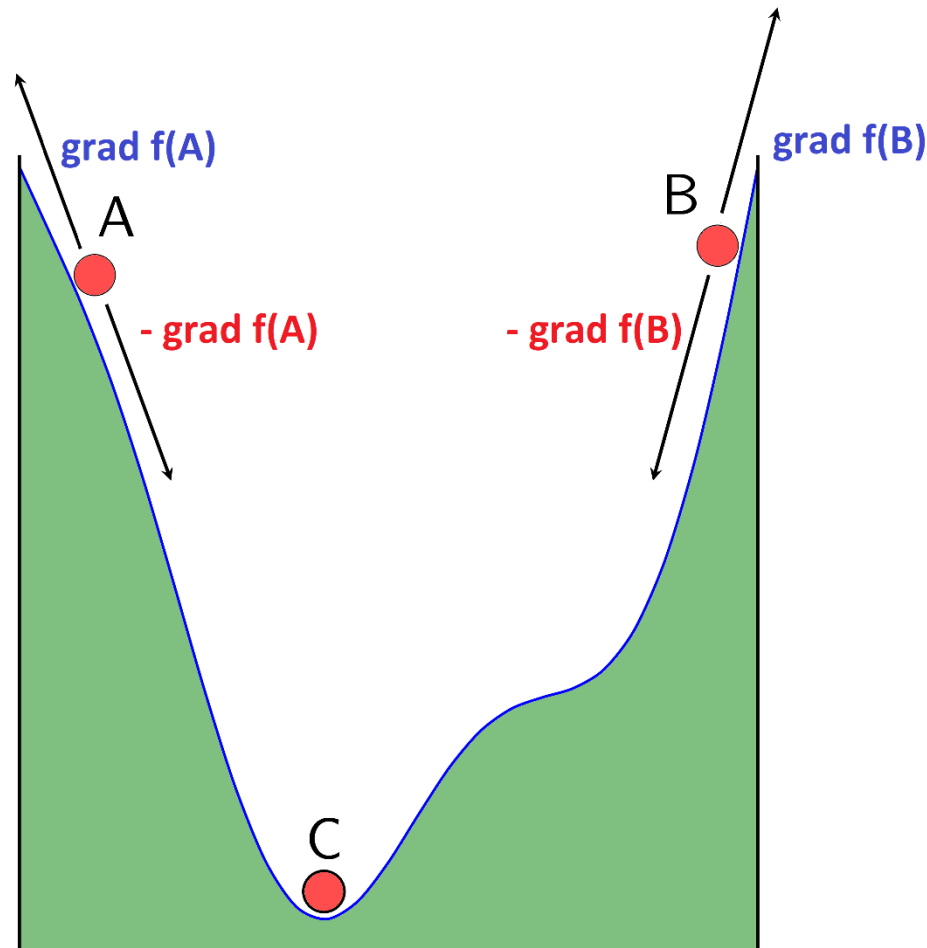
**Теорема.** Градиент – это вектор, в направлении которого функция быстрее всего растёт.

**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:**

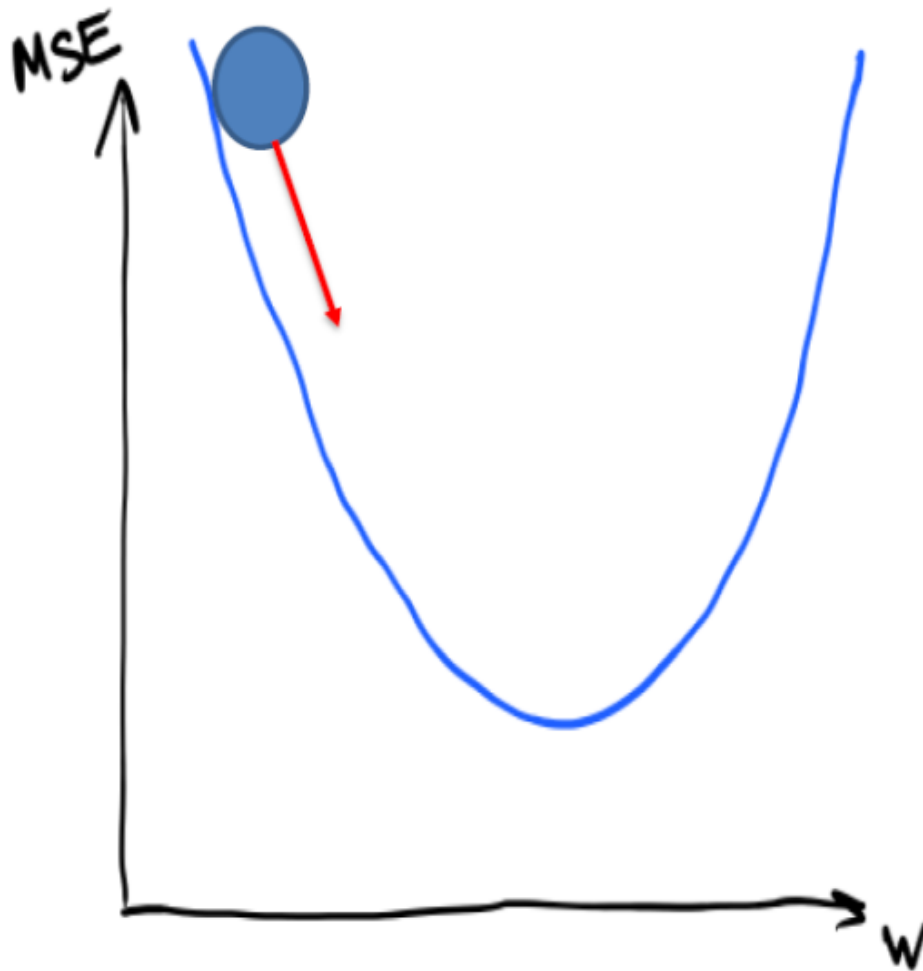
На каждом шаге (на каждой итерации метода) движемся в сторону антиградиента функции потерь!

То есть на каждом шаге движемся в направлении уменьшения ошибки.

Вектор градиента функции потерь обозначают ***grad Q*** или  **$\nabla Q$** .

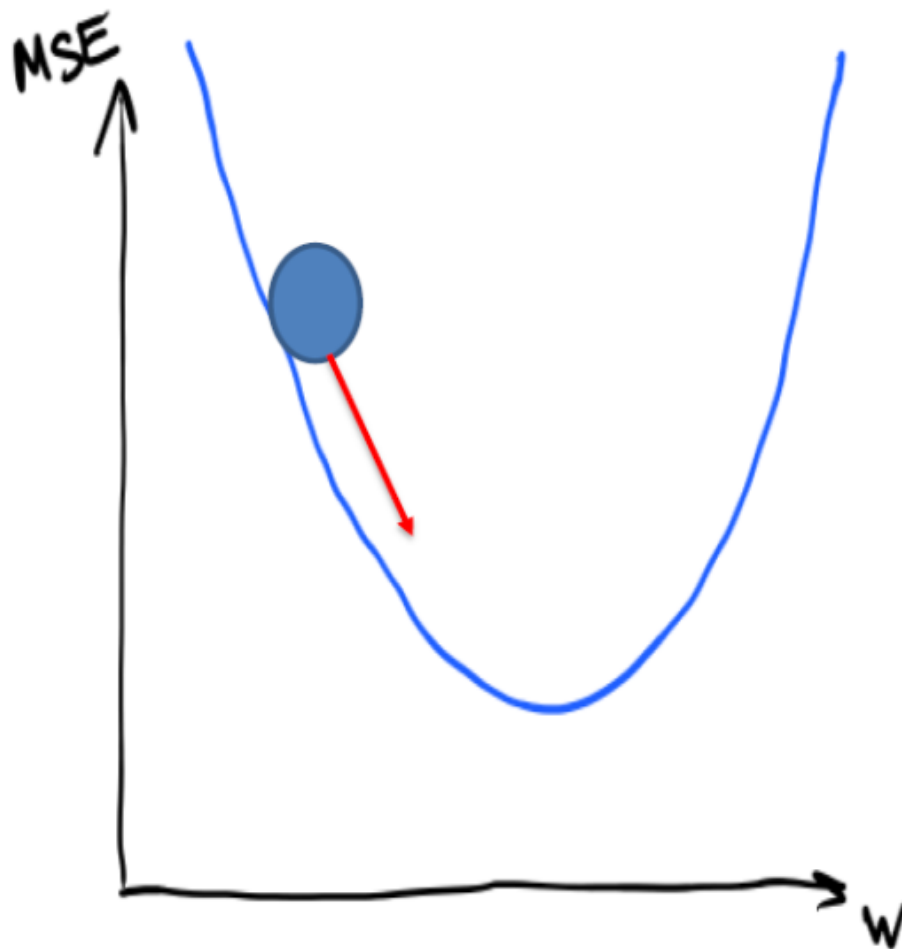
# МЕТОД ГРАДИЕНТНОГО СПУСКА

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



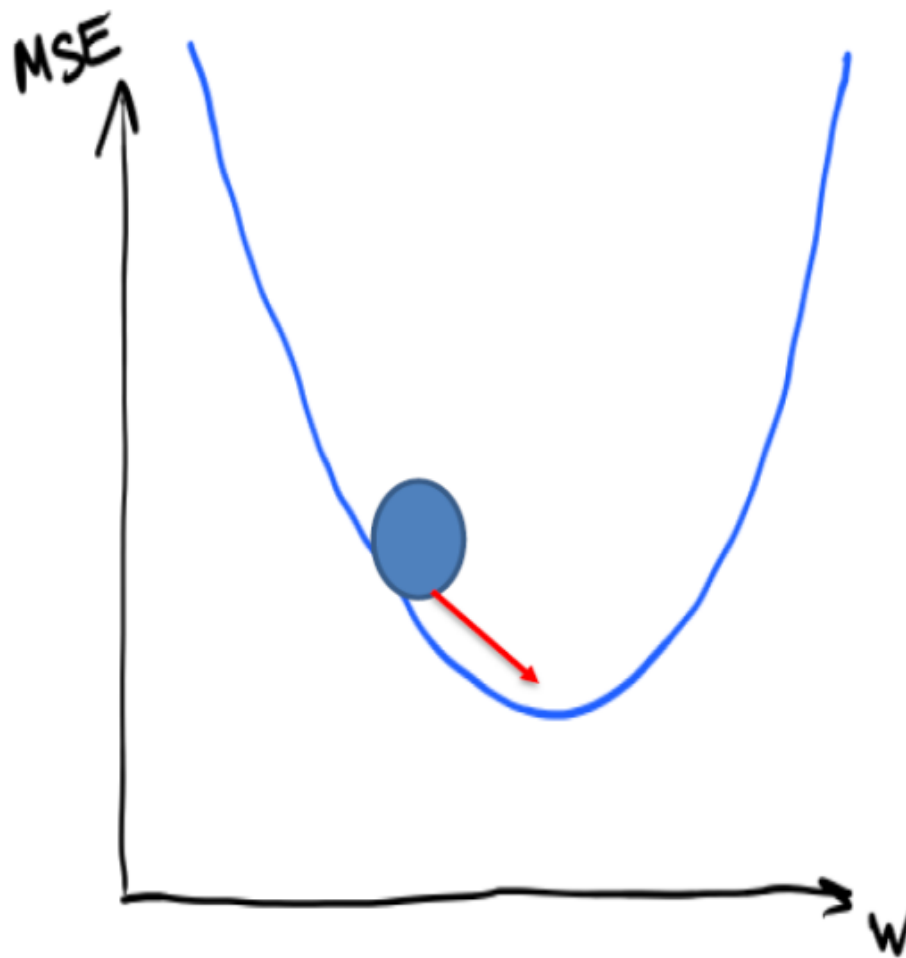
# МЕТОД ГРАДИЕНТНОГО СПУСКА

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



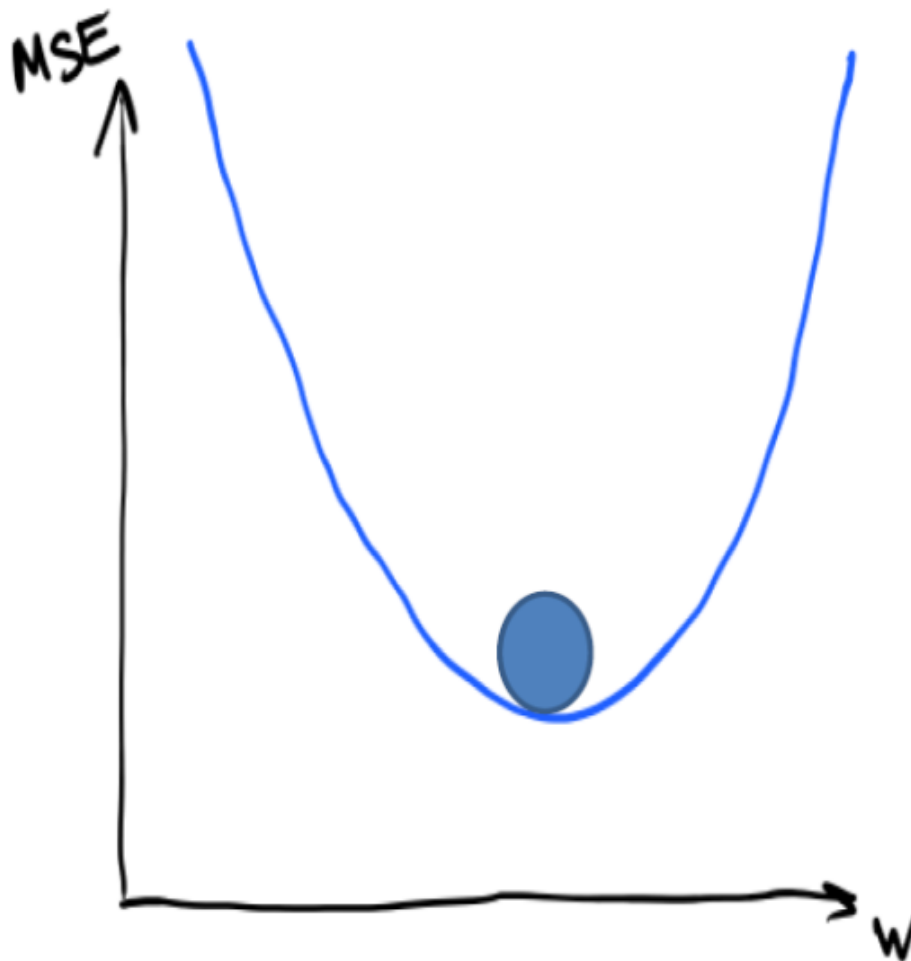
# МЕТОД ГРАДИЕНТНОГО СПУСКА

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.

# МЕТОД ГРАДИЕНТНОГО СПУСКА

## Метод градиентного спуска (одномерный случай):

Пусть у нас только один вес -  $w$ .

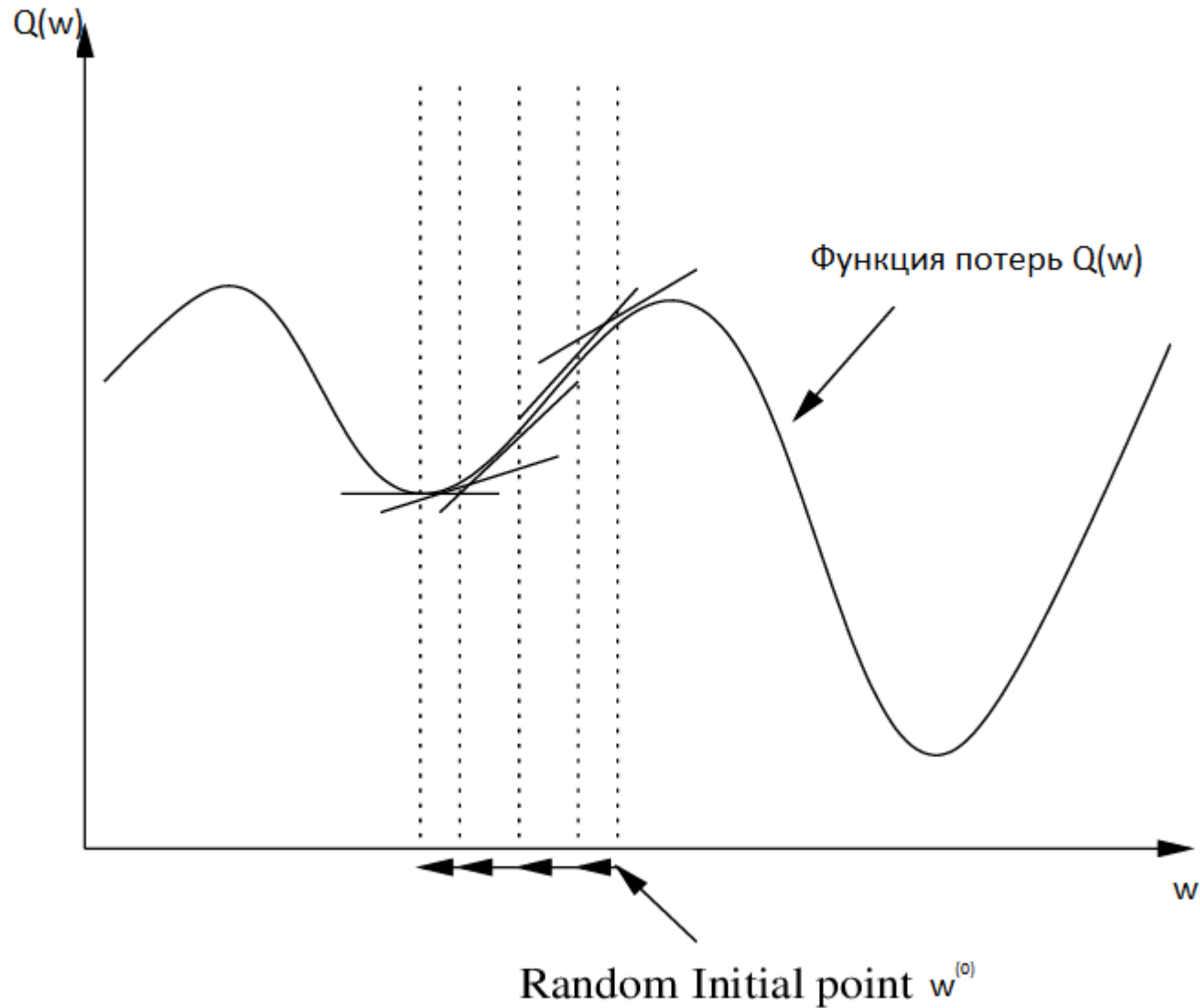
Тогда при добавлении к весу  $w$  слагаемого  $-\frac{\partial Q}{\partial w}$  функция  $Q(w)$  убывает.

- 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



# 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  $w_0^{(0)}, w_1^{(0)}, w_2^{(0)}, \dots, w_n^{(0)}$ .

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

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

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

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

# МЕТОД ГРАДИЕНТНОГО СПУСКА

Формулу для обновления весов можно записать в векторном виде:

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

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

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

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

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

# МЕТОД ГРАДИЕНТНОГО СПУСКА

**Пример (решение на доске):**

выписать формулы 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:

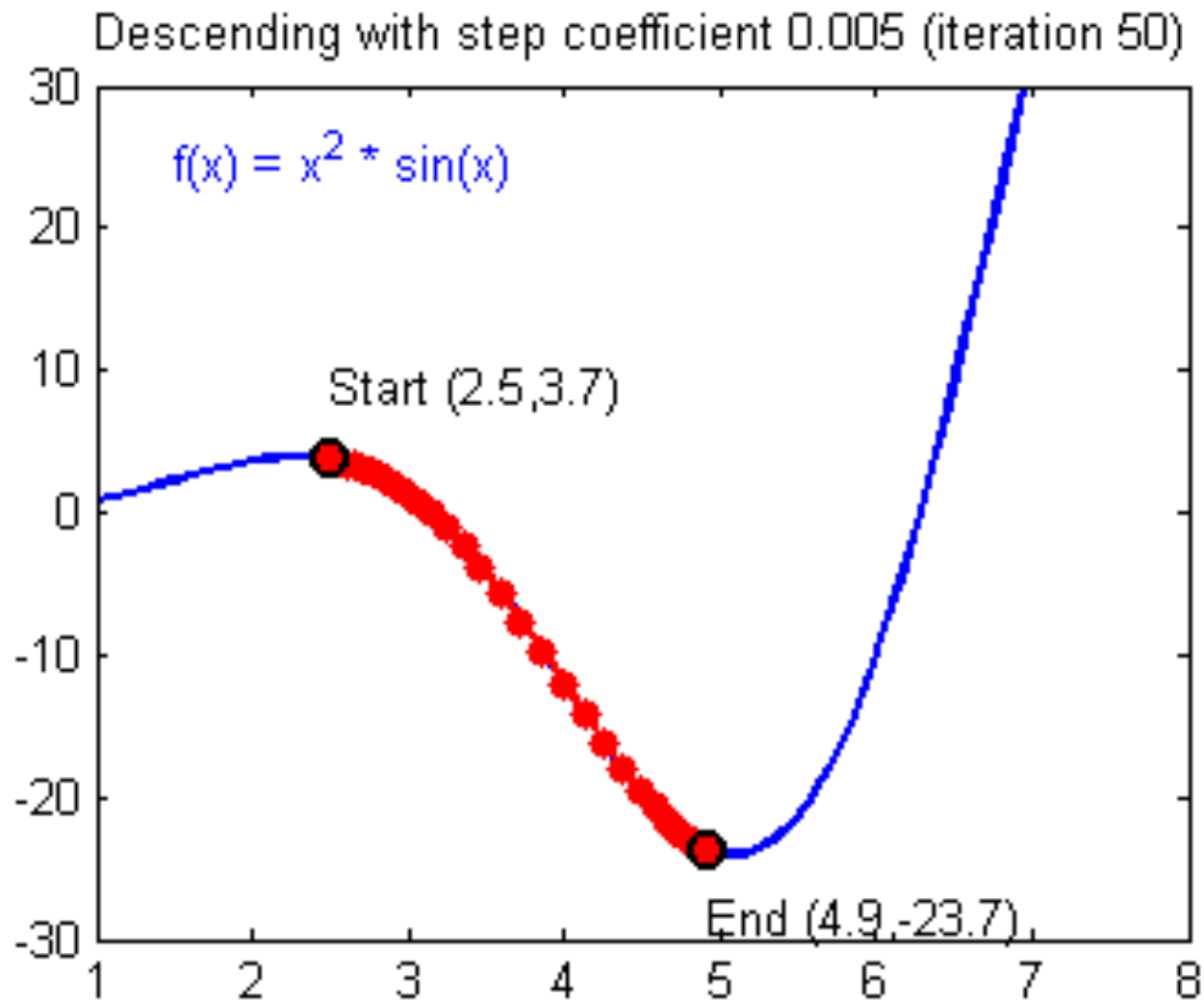
$$w_j := \text{random}(-\varepsilon, \varepsilon)$$

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

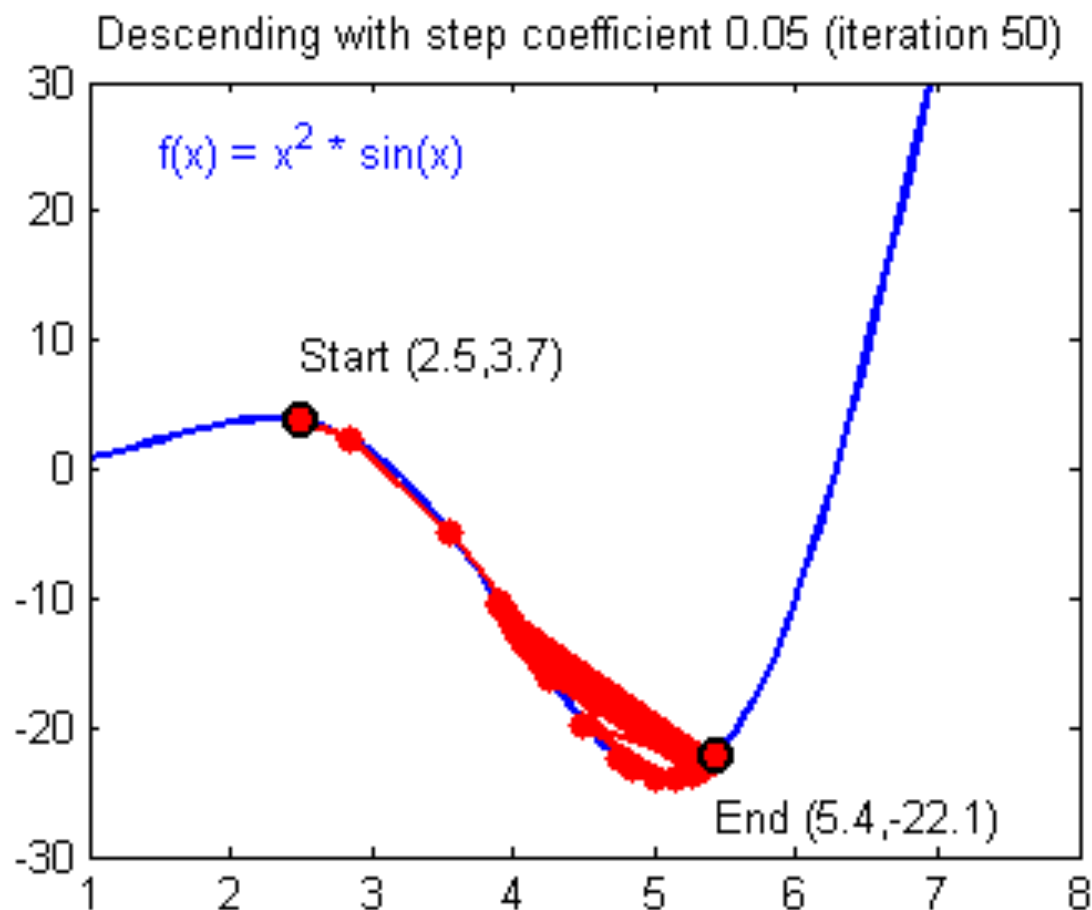
# КРИТЕРИИ ОСТАНОВА

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

# ГРАДИЕНТНЫЙ СПУСК



# ПРОБЛЕМА ВЫБОРА ГРАДИЕНТНОГО ШАГА



# ГРАДИЕНТНЫЙ ШАГ

В общем случае градиентный шаг может зависеть от номера итерации, тогда будем писать не  $\eta$ , а  $\eta_k$ .

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

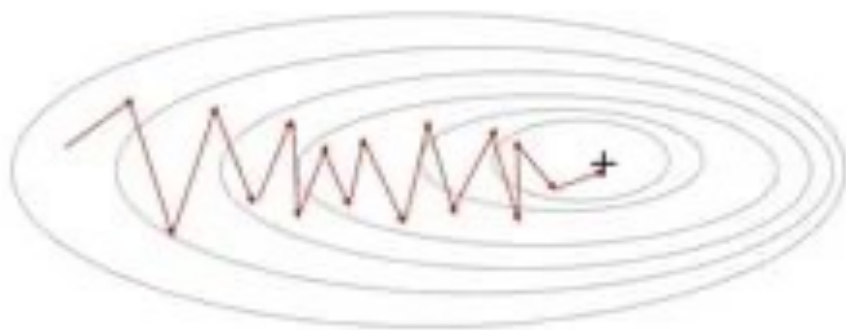
- на каждом шаге выбираем ***один случайный объект*** и сдвигаемся в сторону антиградиента по этому объекту:

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

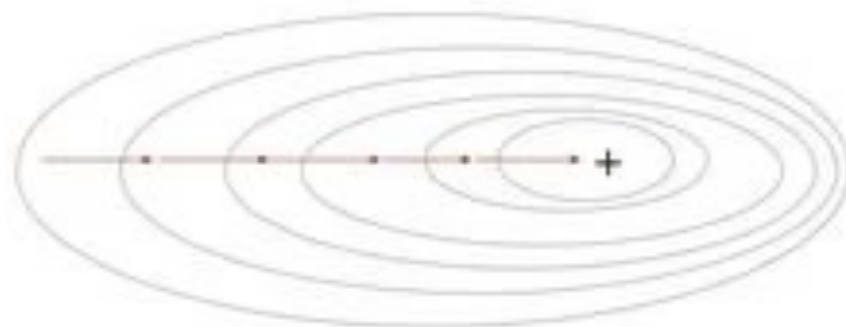
где  $\nabla q_{i_k}(w^{(k-1)})$  - градиент функции потерь, вычисленный только по объекту с номером  $i_k$  (а не по всей обучающей выборке).

# 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  $\eta$  через некоторое число шагов гарантированно попадет в малую окрестность точки  $w^*$ . Однако, сходится метод медленнее, чем обычный градиентный спуск



# MINI-BATCH GRADIENT DESCENT

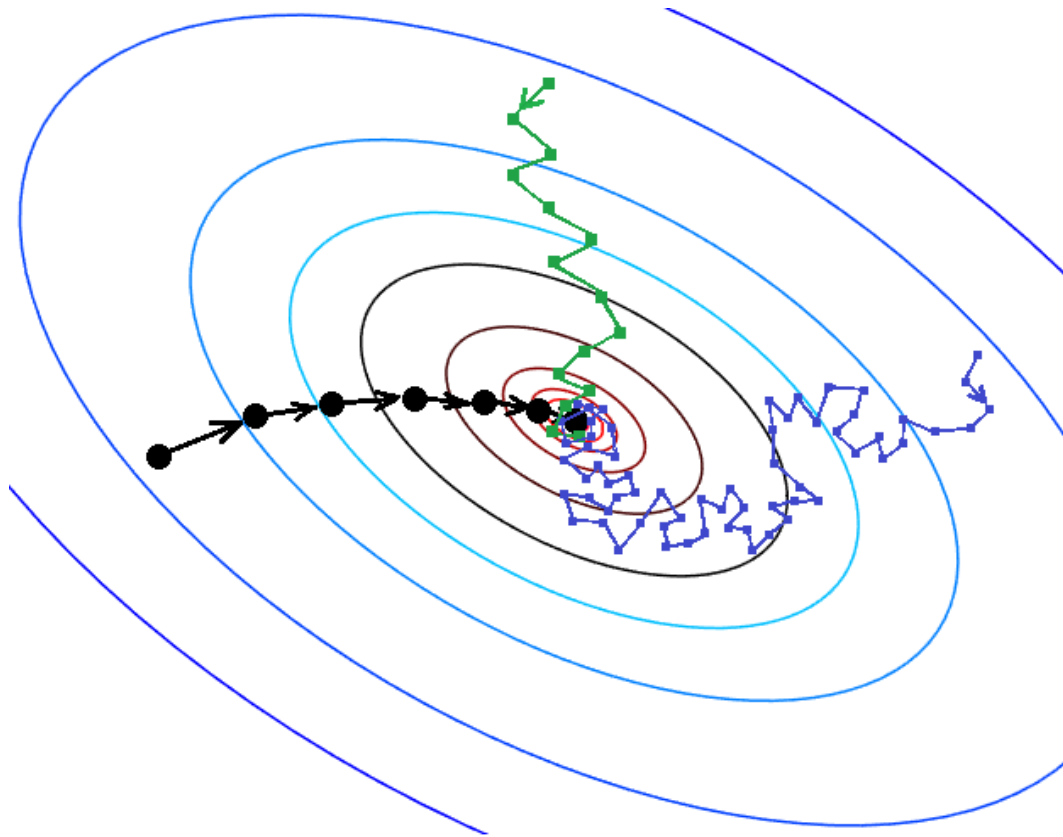
Промежуточное решение между классическим градиентным спуском и стохастическим вариантом.

- 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