Градиентный спуск с линейной регрессией

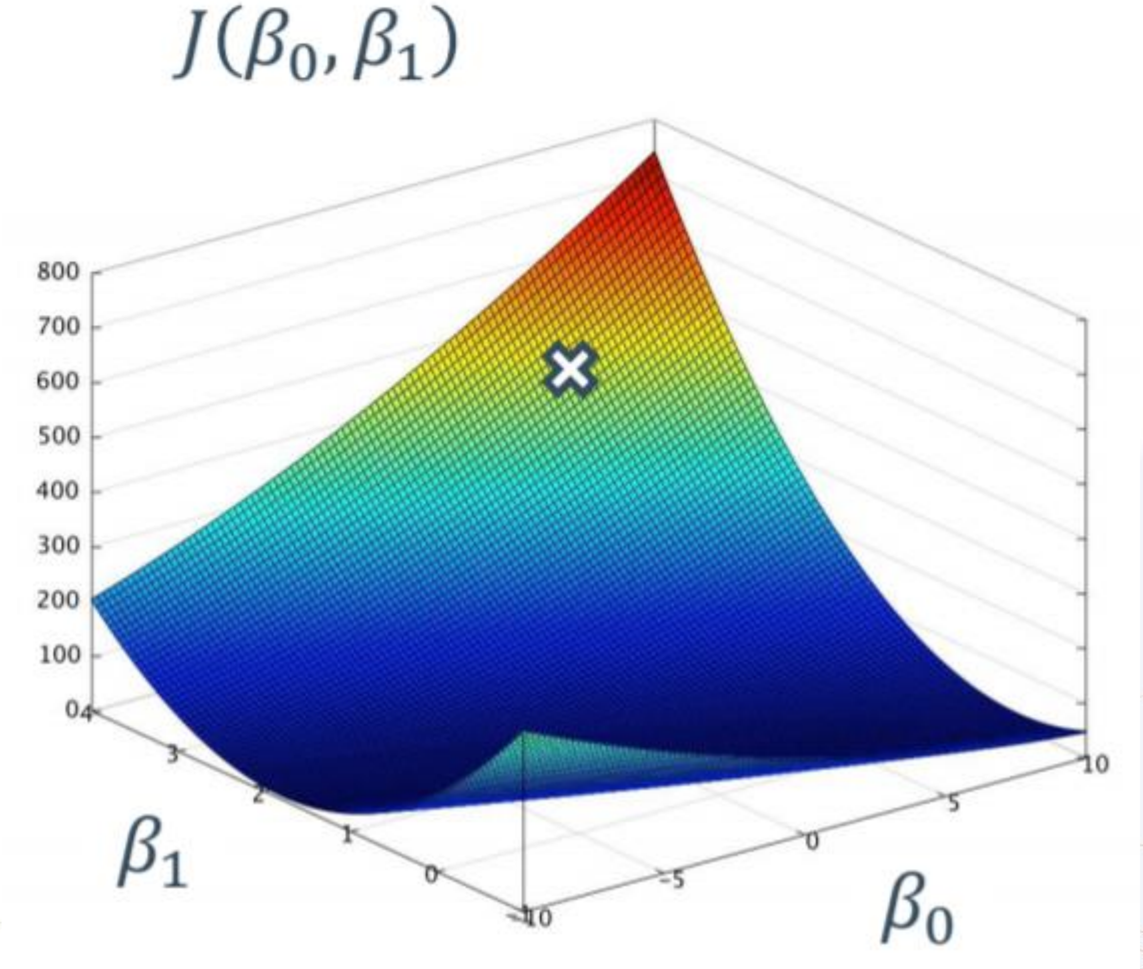
ФИНАНСОВЫЙ УНИВЕРСИТЕТ
ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ
1919

Градиентный спуск с линейной регрессией

- Теперь представьте, что есть два параметра (β_0, β_1)

- Это более сложная поверхность, на которой должен быть найден минимум

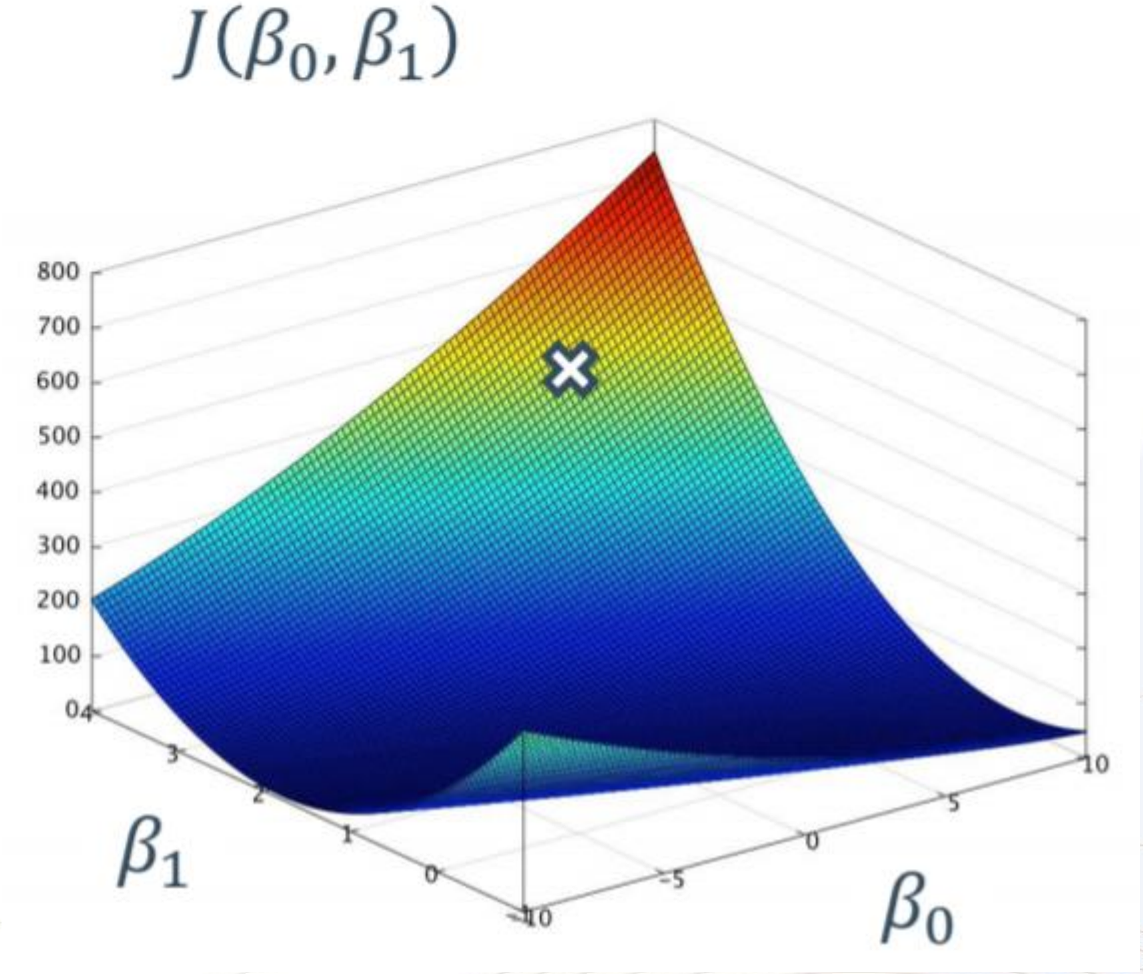
- Как мы можем сделать это, не зная как выглядит $J(\beta_0, \beta_1)$?



Градиентный спуск с линейной регрессией

- Вычислить градиент, $\nabla J(\beta_0, \beta_1)$, который указывает в направлении самой большой увеличение!

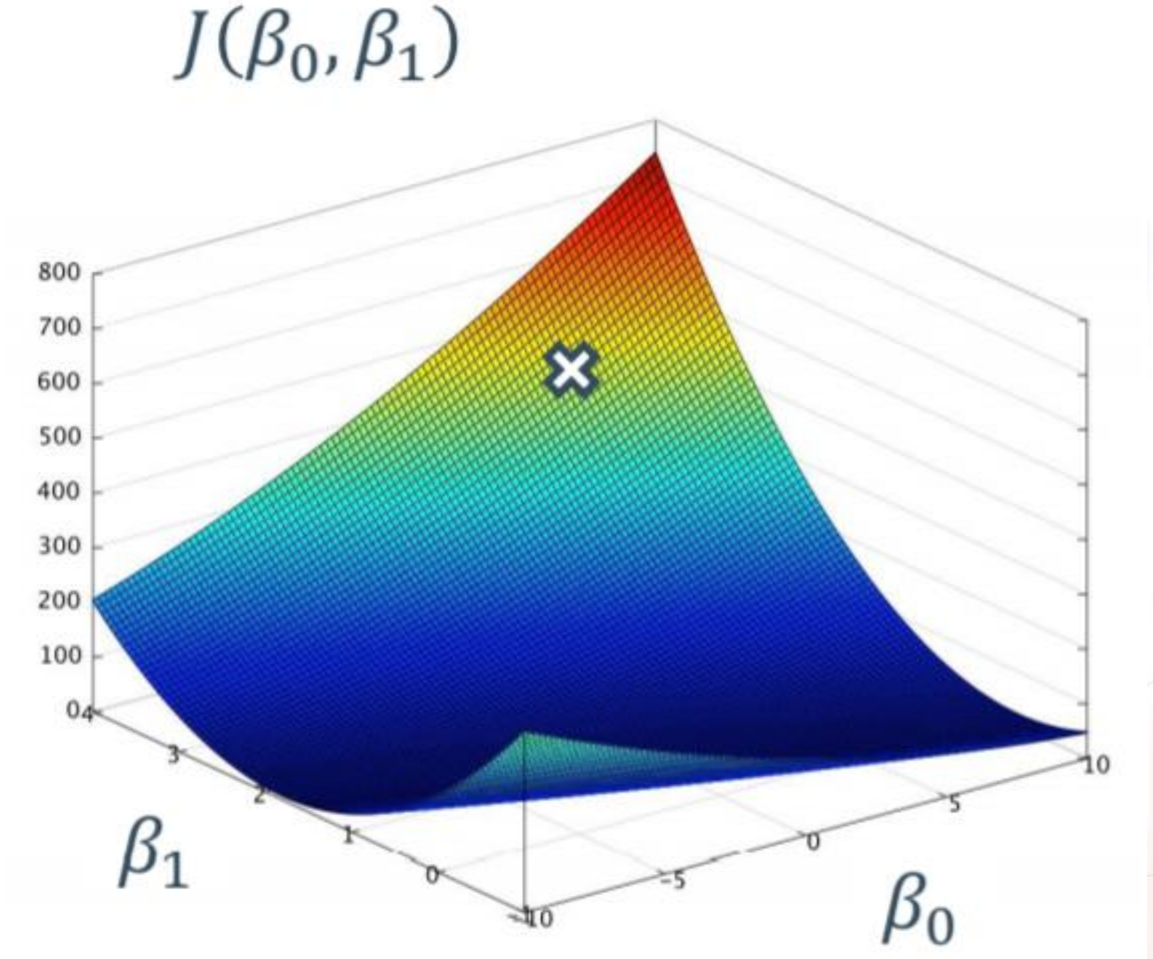
- -$\nabla J(\beta_0, \beta_1)$ (отрицательный градиент) указывает на самое большое снижение в этой точке!



Градиентный спуск с линейной регрессией

- Градиент - это вектор, координаты которого состоят из частных производных параметров

$$\nabla J(\beta_0, \dots, \beta_n) = < \frac{\partial J}{\partial \beta_0}, \dots, \frac{\partial J}{\partial \beta_n} >$$

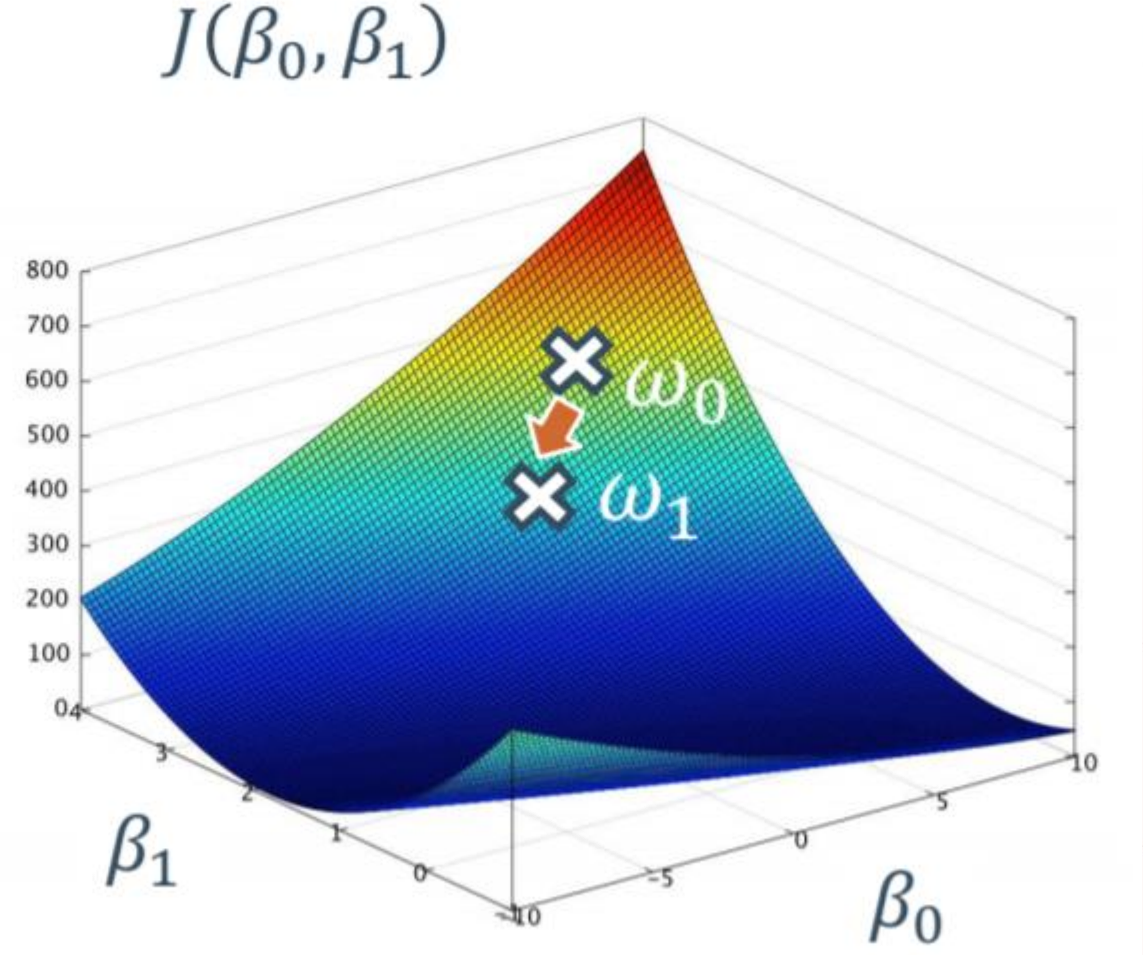


Градиентный спуск с линейной регрессией

- Затем используйте градиент ∇ функции потерь для расчета следующей точки w_1 от текущего w_0:

$$w_1 = w_0 - \lambda\nabla \frac{1}{2n}\sum_{\mathrm{i=1}}^{n}((\beta_0 + \beta_1 x^{(i)}) - y^{(i)})^2$$

- Скорость обучения α - это настраиваемый параметр, который определяет размер шага

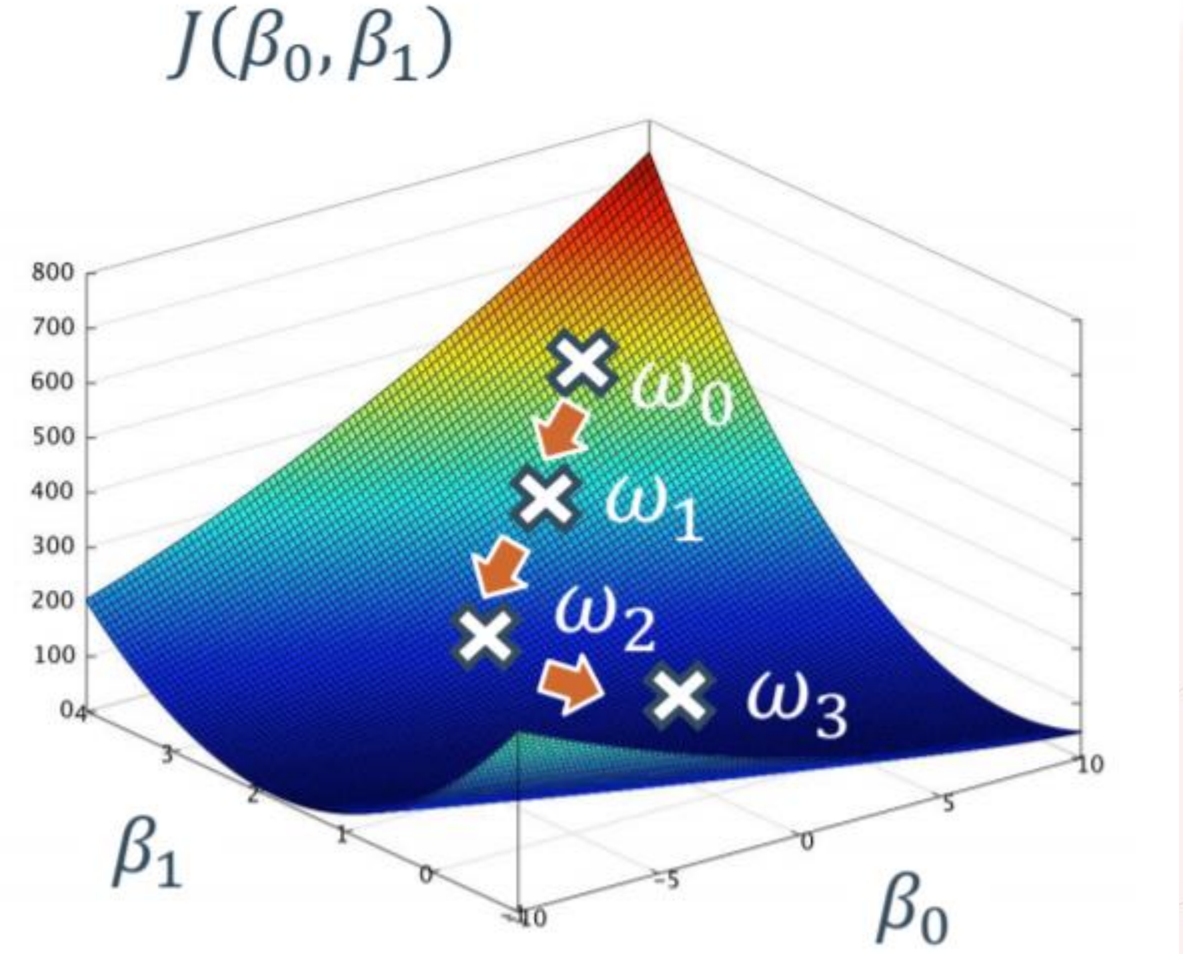


Градиентный спуск с линейной регрессией

- Каждая точка может быть итеративно рассчитана из предыдущей

$$w_2 = w_1 - \lambda\nabla \frac{1}{2n}\sum_{i=1}^{n}((\beta_0 + \beta_1 x^{(i)}) - y^{(i)})^2\,,$$

$$w_3 = w_2 - \lambda\nabla \frac{1}{2n}\sum_{i=1}^{n}((\beta_0 + \beta_1 x^{(i)}) - y^{(i)})^2$$

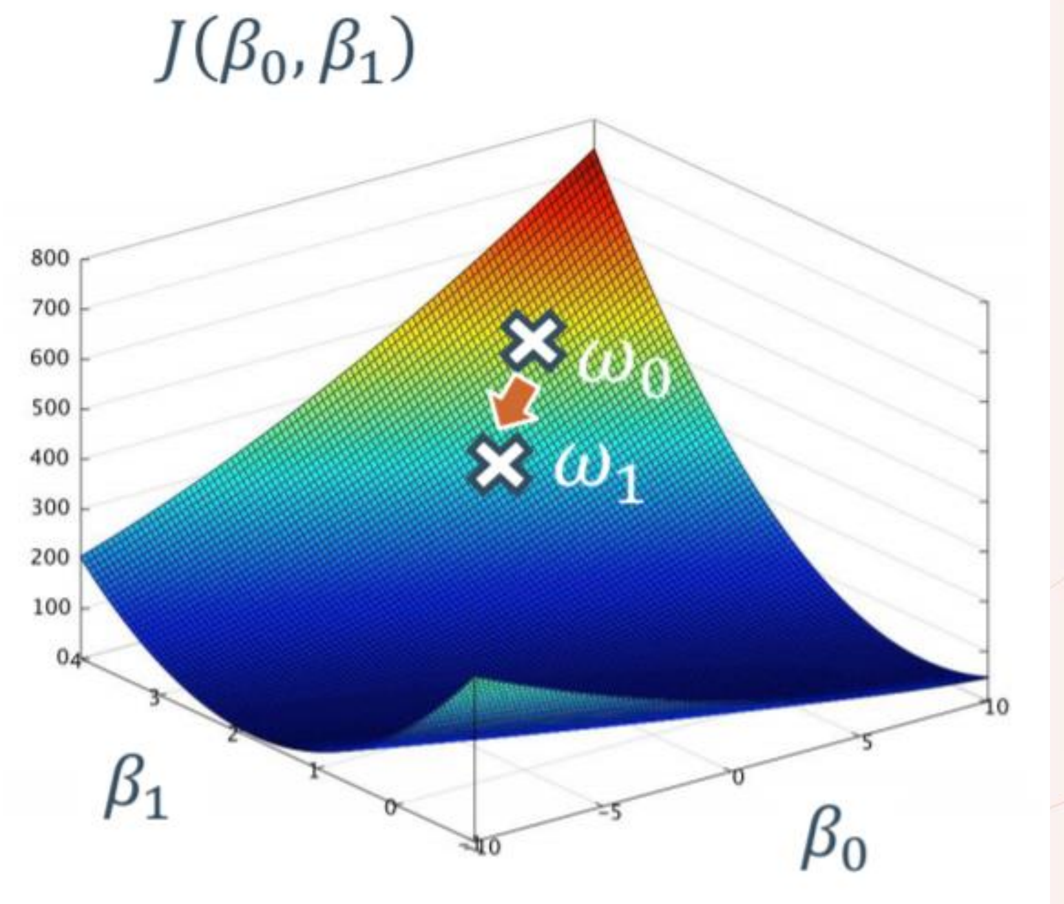


Стохастический градиентный спуск

- Используйте одну точку данных для определения функции градиента и стоимости вместо всех данных

$$w_2 = w_1 - \lambda\nabla\frac{1}{2n}\cancel{\sum_{i=1}^{n}}((\beta_0 + \beta_1 x^{(i)}) - y^{(i)})^2\,,$$

$$w_3 = w_2 - \lambda\nabla\frac{1}{2}((\beta_0 + \beta_1 x^{(i)}) - y^{(i)})^2$$



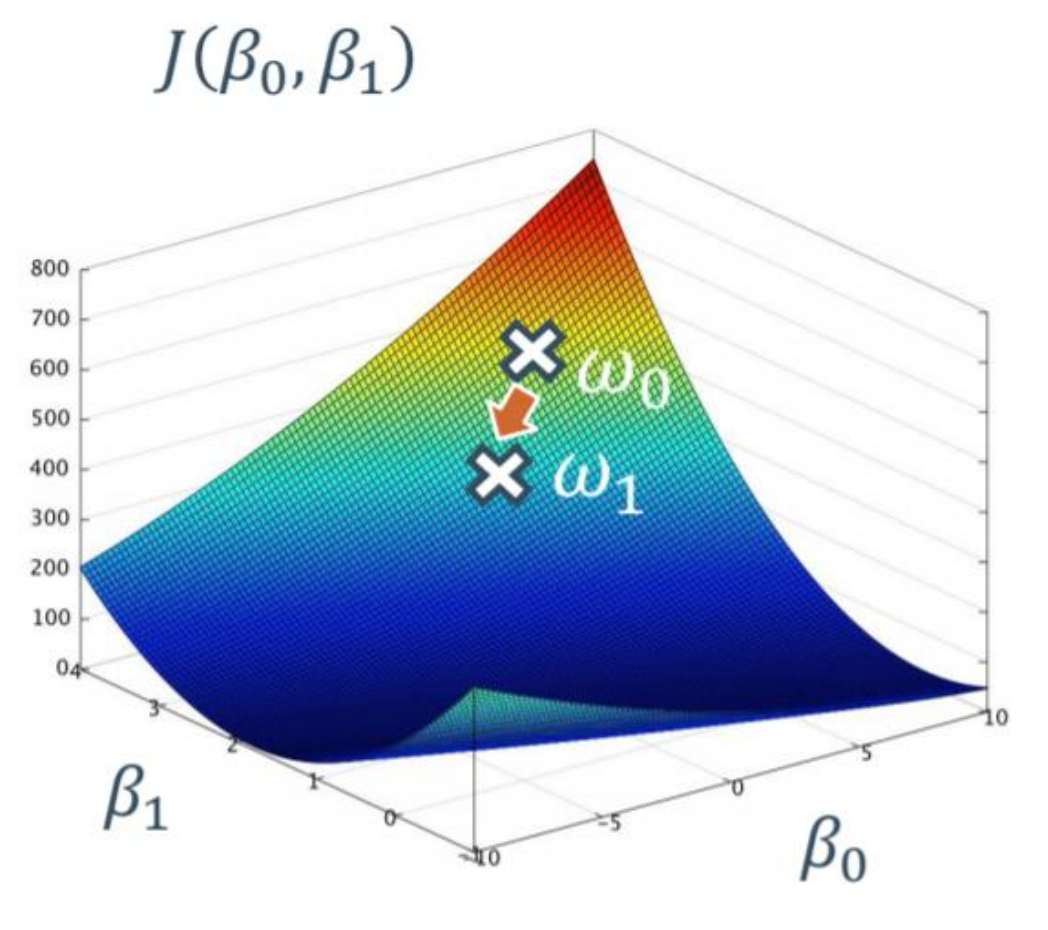
Стохастический градиентный спуск

- Используйте одну точку данных для определения функции градиента и стоимости вместо всех данных

$$w_2 = w_1 - \lambda\nabla\frac{1}{2n}\cancel{\sum_{i=1}^{n}}((\beta_0 + \beta_1 x^{(i)}) - y^{(i)})^2\,,$$

$$\downarrow$$

$$\boxed{w_3 = w_2 - \lambda\nabla\frac{1}{2}((\beta_0 + \beta_1 x^{(i)}) - y^{(i)})^2}$$



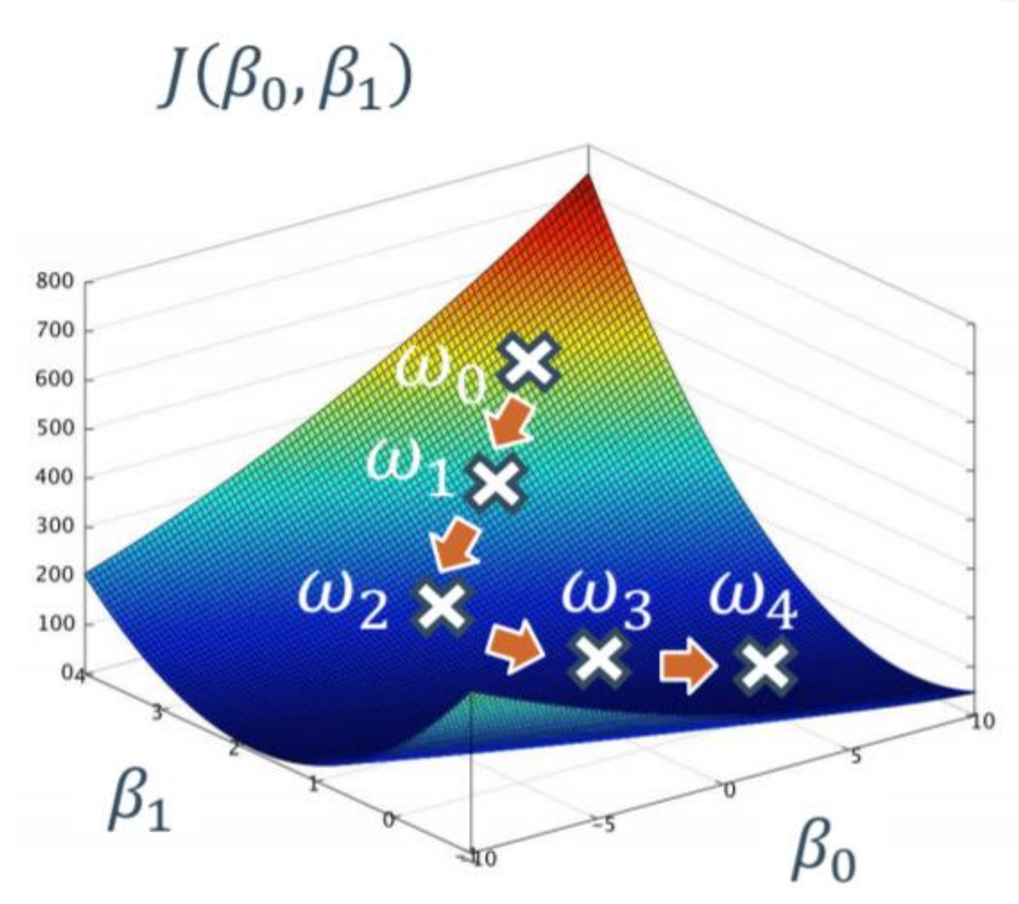
Стохастический градиентный спуск

- Используйте одну точку данных для определения функции градиента и стоимости вместо всех данных

$$w_1 = w_0 - \lambda \nabla \frac{1}{2}\left(\left(\beta_0 + \beta_1 x^{(0)}\right) - y^{(0)}\right)^2,$$

…

$$w_4 = w_3 - \lambda \nabla \frac{1}{2}\left(\left(\beta_0 + \beta_1 x^{(3)}\right) - y^{(3)}\right)^2$$



- Путь к минимуму менее прямой из-за шума в одной точке данных - «стохастичный»

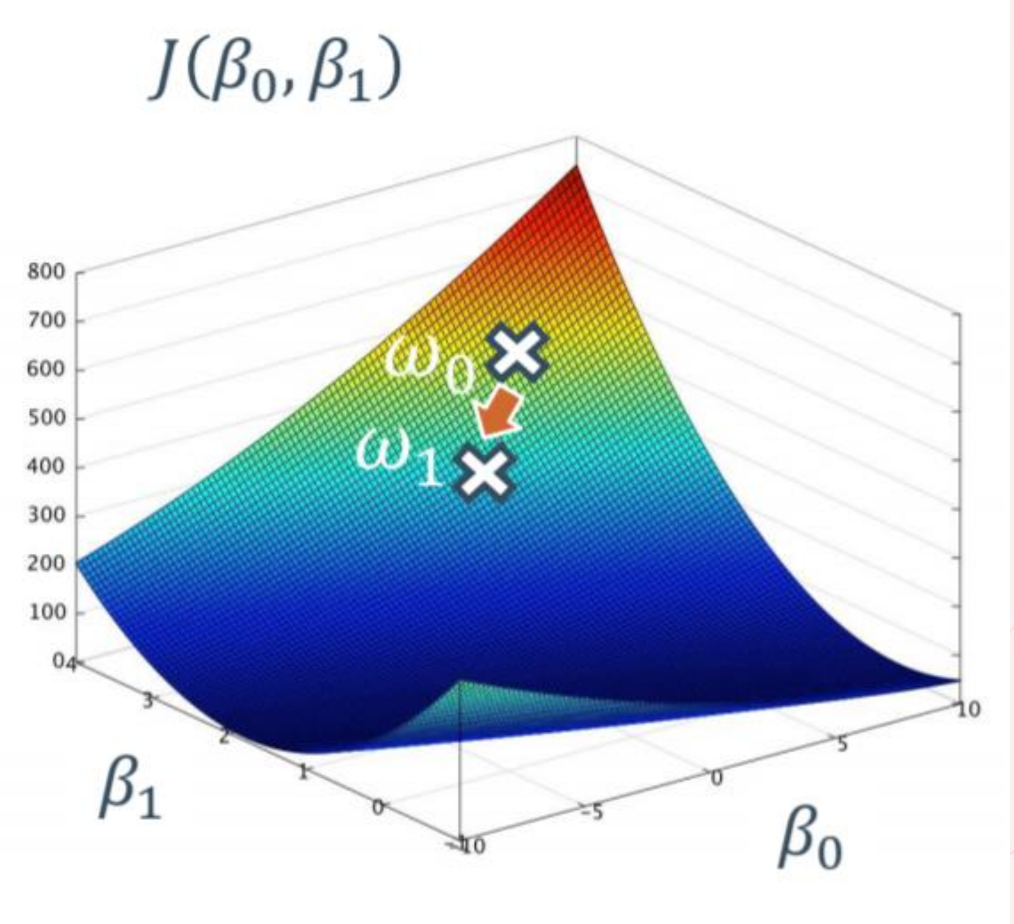
Мини пакетный градиентный спуск

- Выполните обновление для каждого m обучающих примеров

$$w_1 = w_0 - \lambda \nabla \frac{1}{2m} \sum_{\mathrm{i}=1}^{m} ((\beta_0 + \beta_1 x^{(i)}) - y^{(i)})^2$$

Лучшее из обоих миров:

- Снижение объема вычислений относительно классического градиентного спуска
- Менее шумный, чем стохастический градиентный спуск
- Мини-пакетная реализация обычно используемая для *нейронных сетей*
- Размеры партии варьируются от 50–256 точек
- Компромисс между размером партии и скоростью обучения α
- Индивидуальный график обучения: постепенно снижайте скорость обучения в течение определенной эпохи



Градиентный спуск

- Функция потерь

$$J(\beta_0, \beta_1) = \frac{1}{2n}\sum_{\mathrm{i}=1}^{n}((\beta_0 + \beta_1 x^{(i)}) - y^{(i)})^2\,,$$

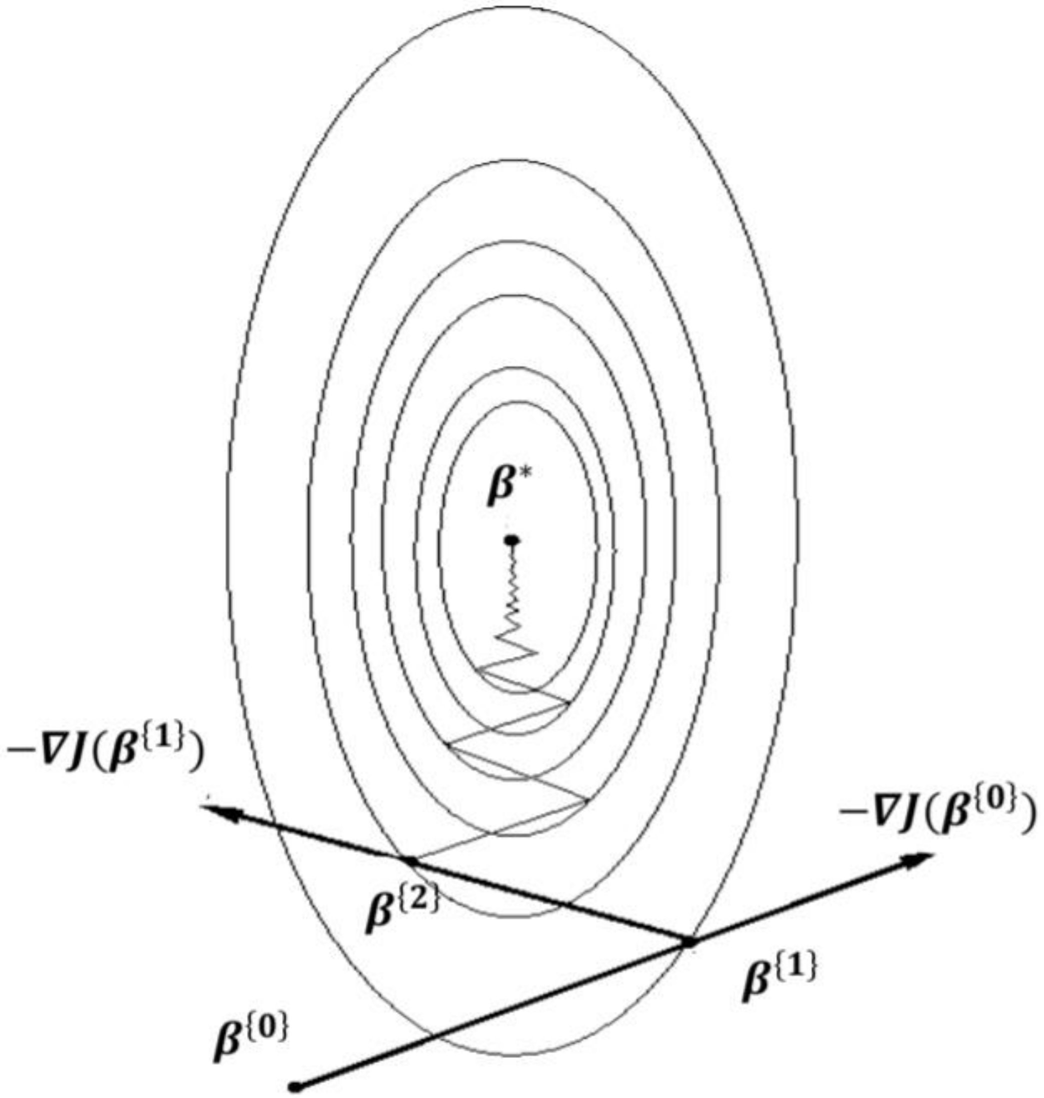
- Метод градиентного спуска

$$\beta_j^{\{k+1\}} = \beta_j^{\{k\}} - \lambda\frac{\partial}{\partial\beta_j}J(\beta_0, \beta_1)$$

- Результат

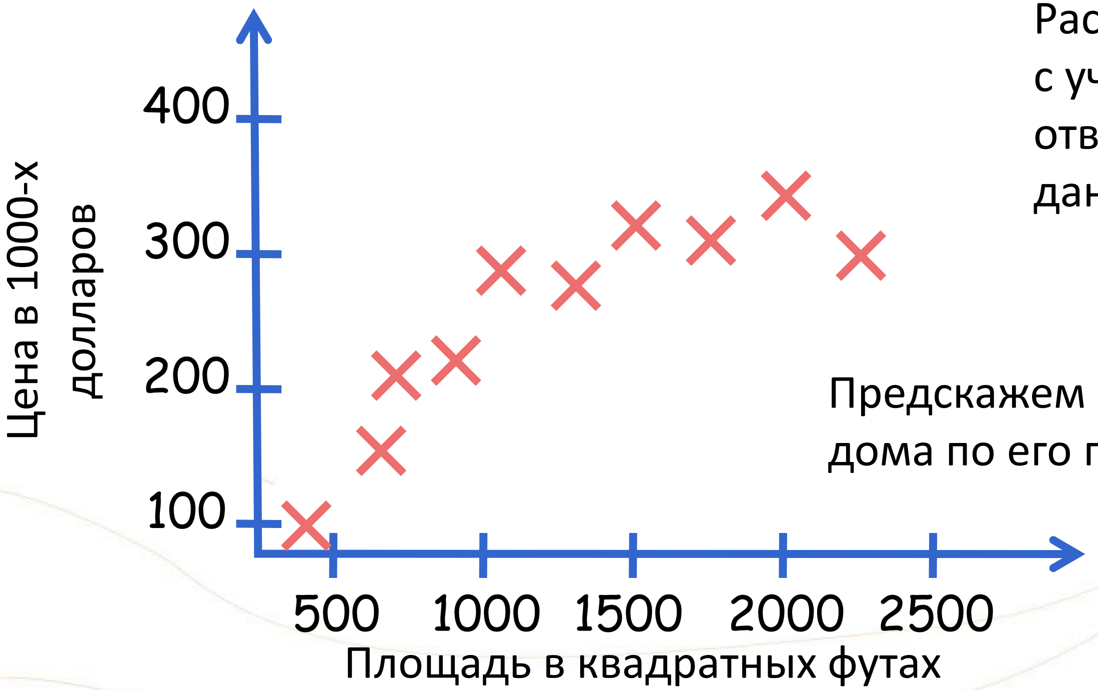
$$\beta_0^{\{k+1\}} = \beta_0^{\{k\}} - \lambda\frac{1}{n}\sum_{\mathrm{i}=1}^{n}((\beta_0 + \beta_1 x^{(i)}) - y^{(i)}))$$

$$\beta_1^{\{k+1\}} = \beta_1^{\{k\}} - \lambda\frac{1}{n}\sum_{\mathrm{i}=1}^{n}((\beta_0 + \beta_1 x^{(i)}) - y^{(i)}))x^{(i)}$$



Линейная регрессия с одной переменной

Регрессия (предсказание непрерывной выходной величины, например, цены на недвижимость)



Рассматриваем обучение с учителем (даны правильные ответы для каждого примера в данных)

Предскажем стоимость дома по его площади

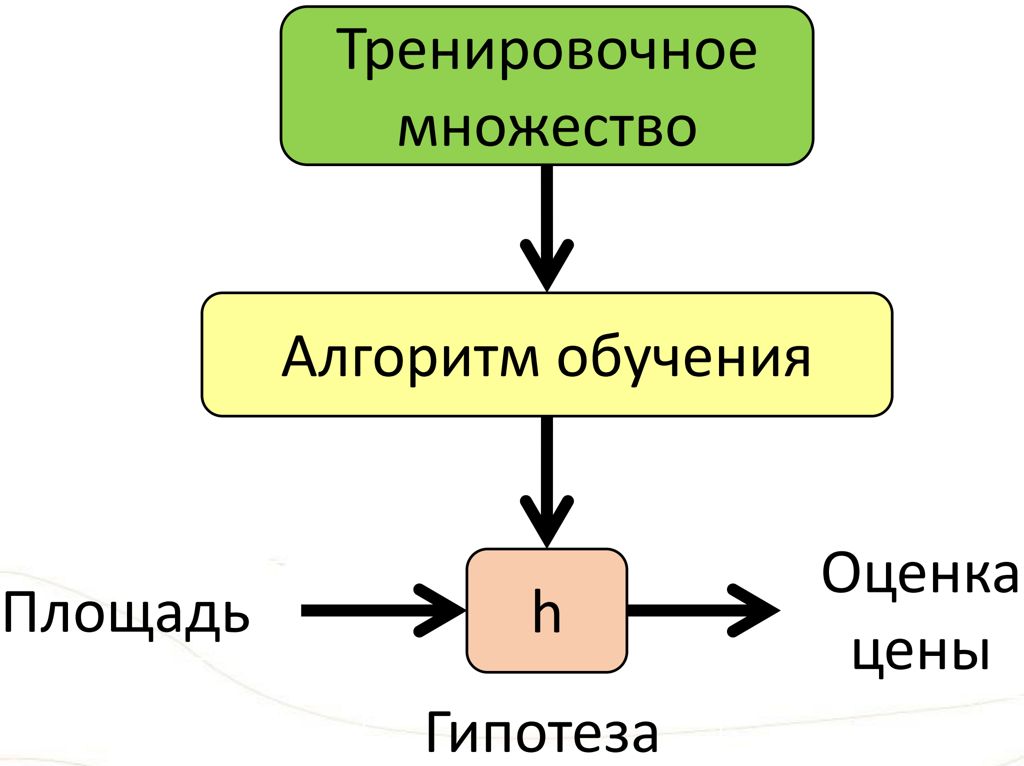
Линейная регрессия с одной переменной

Тренировочное множество данных (скажем, всего m)

Площадь (фут^2) – x	**Цена в 1000-х ($) – y**
2104	460
1416	232
1534	315
852	178
…	…

Обозначения: **n** = число тренировочных примеров
x = «входная» переменная / свойство
y = «выходная» переменная / «метка»
($\mathbf{x}^{(i)}$, $\mathbf{y}^{(i)}$) = i-й тренировочный пример

Линейная регрессия с одной переменной

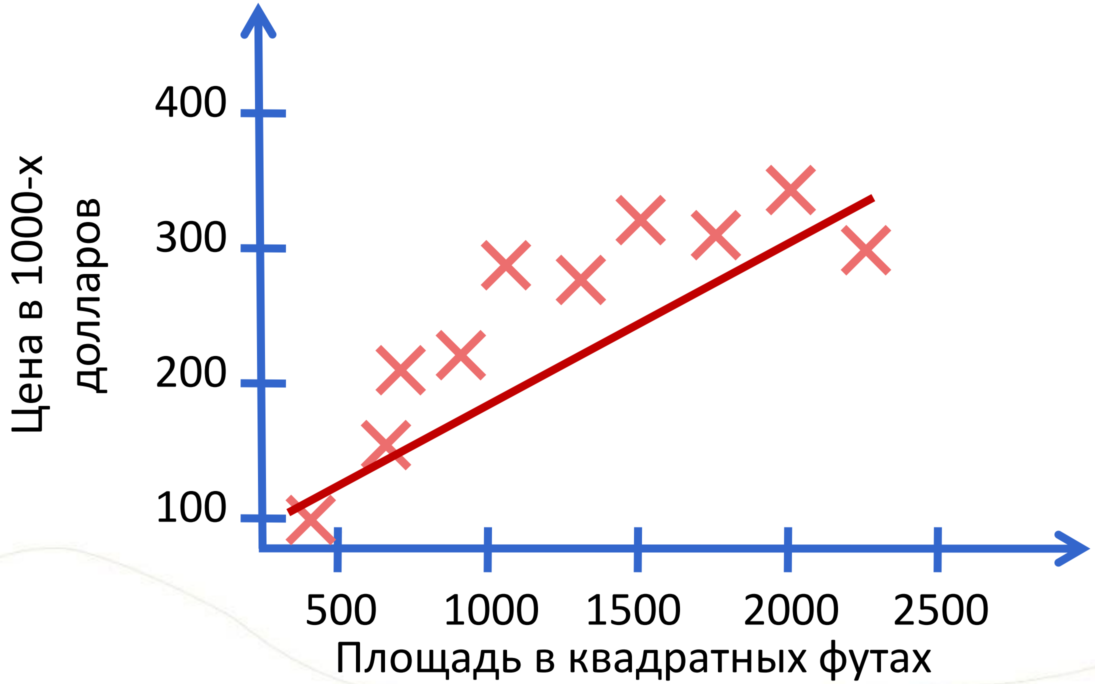


Гипотеза h выглядит так:
$h(x) = \beta_0 + \beta_1 x$

Как оценить β_0 и β_1 ?

Линейная регрессия с одной переменной

✓ Последовательность действий

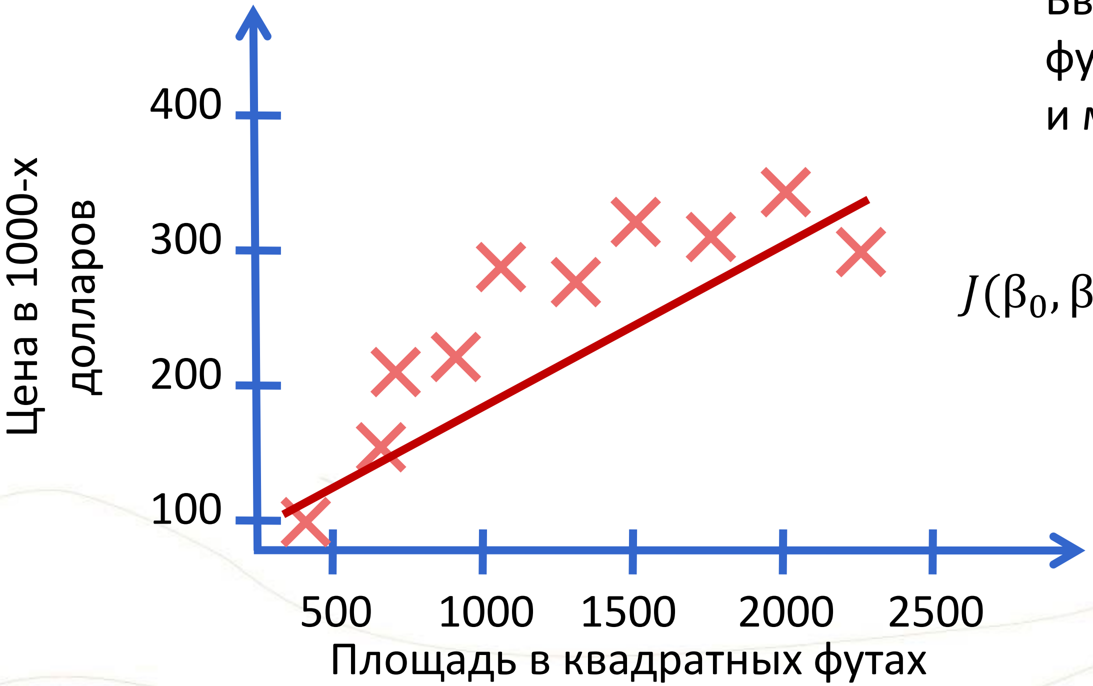


Гипотеза h выглядит так:
h(x) = $\beta_0 + \beta_1$x

Как оценить β_0 и β_1 ?

Стоимостная функция (Cost Function)

✓ Идея! Выбрать β_0 и β_1 так, чтобы $h_\beta(x)$ являлась близкой к значениям y для всех тренировочных примеров



Введем стоимостную функцию $J(\beta_0, \beta_1)$ и минимизируем ее:

$$J(\beta_0, \beta_1) = \frac{1}{2n}\sum_{i=1}^{n}(h(x^{(i)}) - y^{(i)})^2,$$

$$\min_{\beta_0,\, \beta_1} J(\beta_0, \beta_1)$$

Как минимизировать стоимостную функцию для линейной регрессии с одной переменной

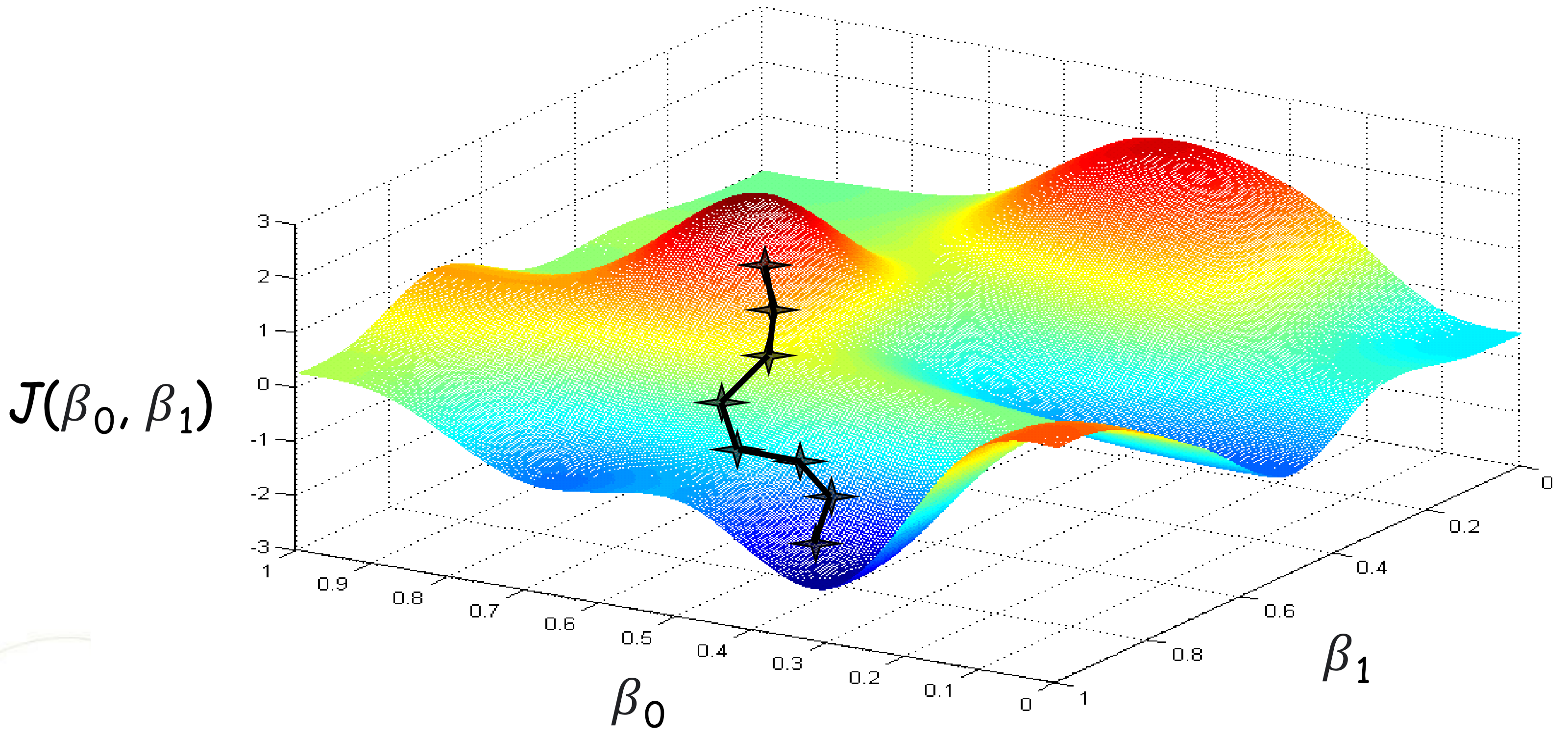
- ✓ Использование методов численной оптимизации, например, метода градиентного спуска
  - ✓ Более конкретно будем рассматривать групповой градиентный спуск («Batch» Gradient Descent). Групповой означает, что на каждом этапе градиентного спуска используются все тренировочные примеры
  - ✓ Подход может быть использован и для других методов машинного обучения, например, нейронных сетей

Метод градиентного спуска

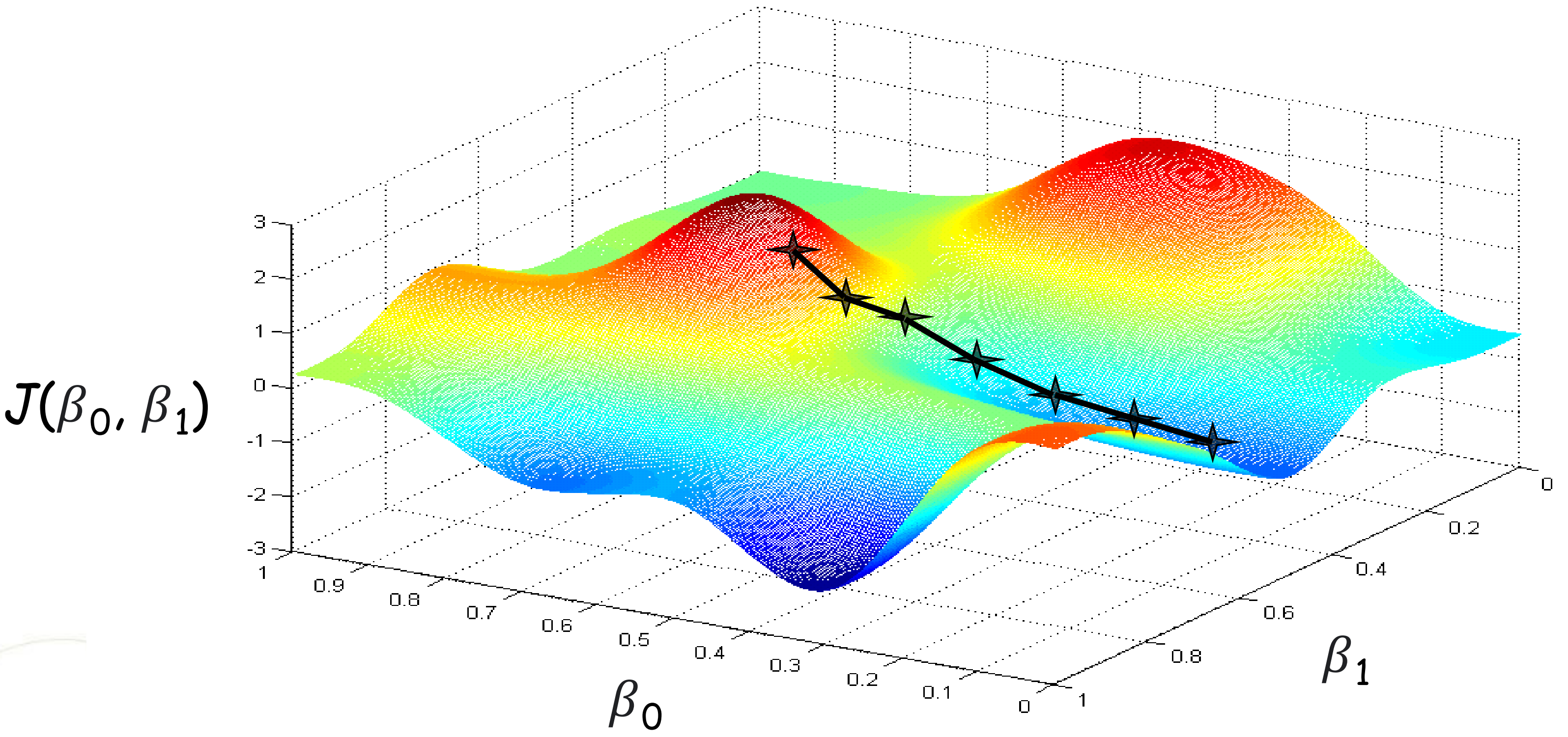
- Постановка задачи
  - Имеется некоторая стоимостная функция $J(\beta_0, \beta_1)$
  - Необходимо найти такие значения β_0, β_1, чтобы функция $J(\beta_0, \beta_1)$ стала минимальной

- Решение задачи
  - Стартуем из некоторых значений β_0, β_1, например, равных величине ноль
  - Продолжаем изменение значений β_0, β_1 до тех пор, пока не достигнем минимума. Минимум достижим не всегда!

Пример работы градиентного спуска



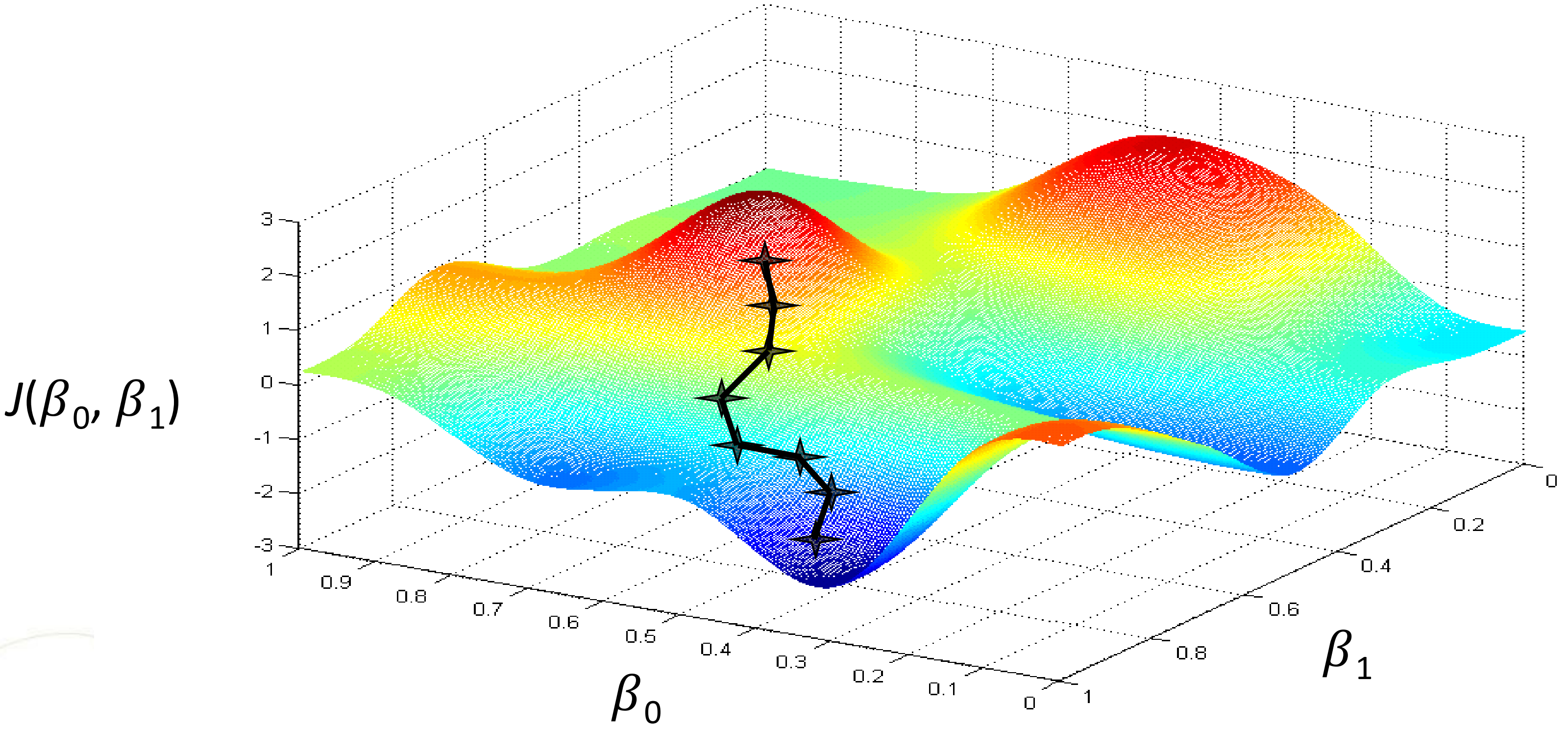
Пример работы градиентного спуска



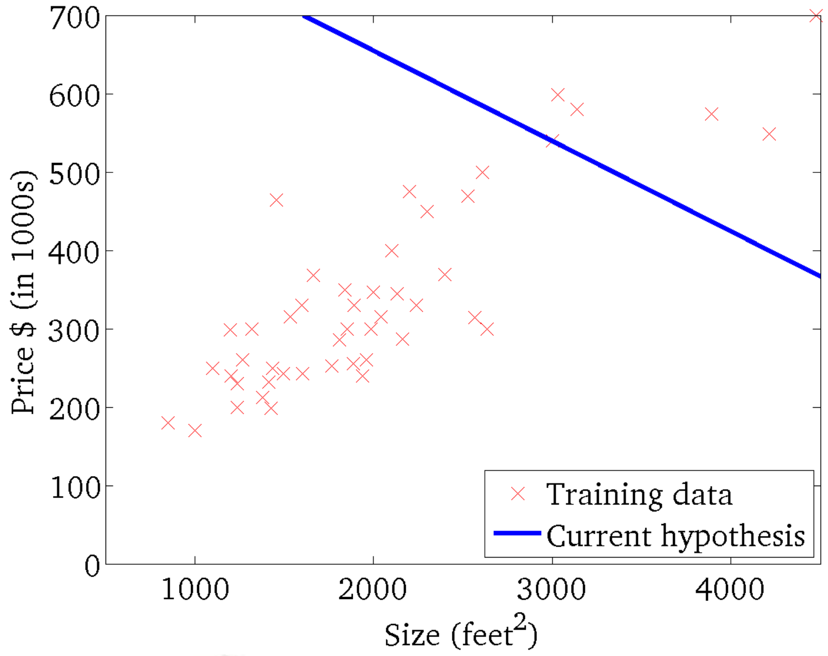
Метод градиентного спуска. Реализация

- ✓ Скорость сходимости алгоритма регулируется параметром α
  - ✓ Если α маленькое, то градиентный спуск может быть медленным
  - ✓ Если α большое, то градиентный спуск может проскочить минимум. Алгоритм может не сходиться или даже расходиться
- ✓ Градиентный спуск может сходиться к локальному минимуму, даже если α является фиксированным
  - ✓ При приближении к локальному минимуму градиентный спуск будет автоматически выполнять более малые шаги. Поэтому нет необходимости уменьшать α через некоторое время

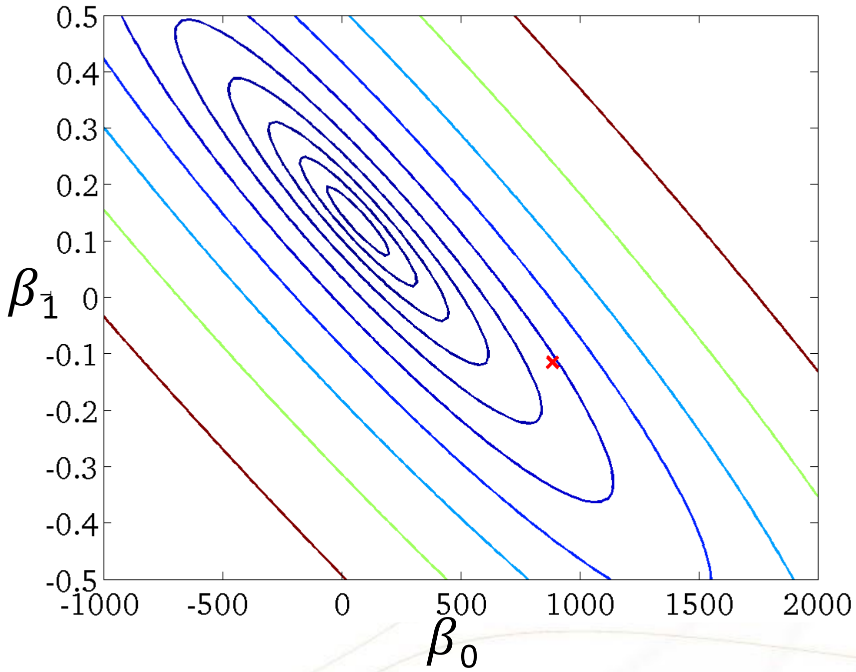
Пример формы стоимостной функции



Пример работы градиентного спуска для линейной регрессии

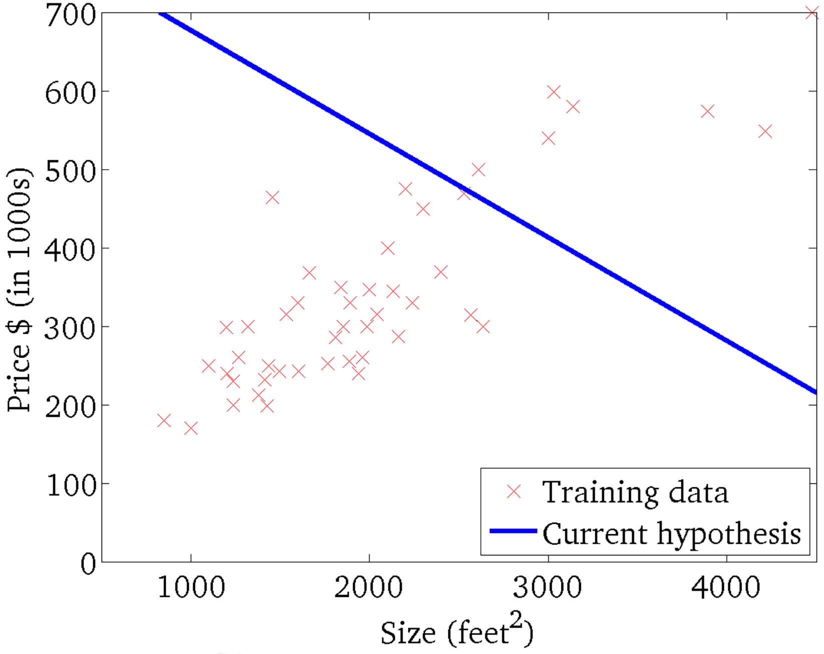


h(x) – это функция x для фиксированных β_0 и β_1

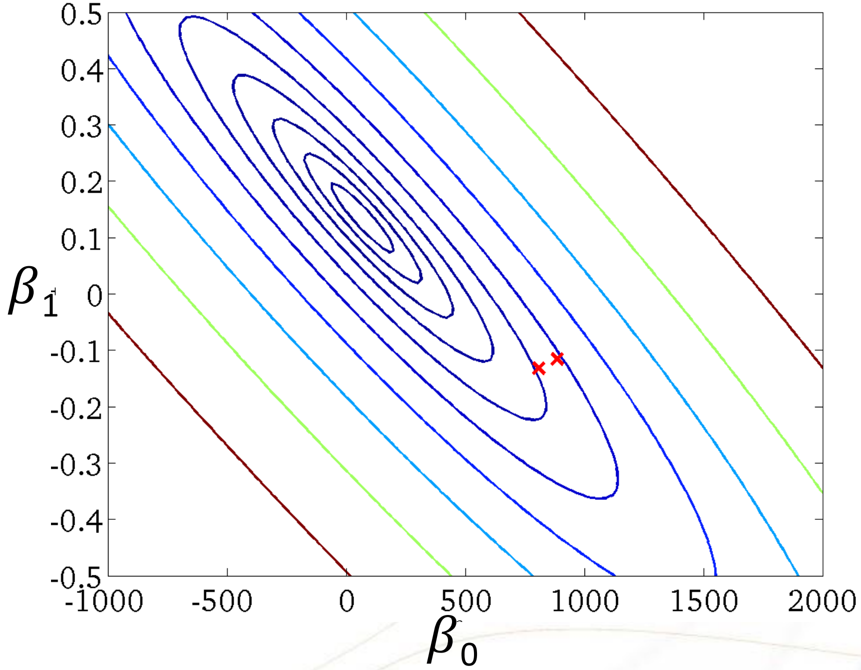


J(β_0, β_1) – это функция параметров β_0 и β_1

Пример работы градиентного спуска для линейной регрессии

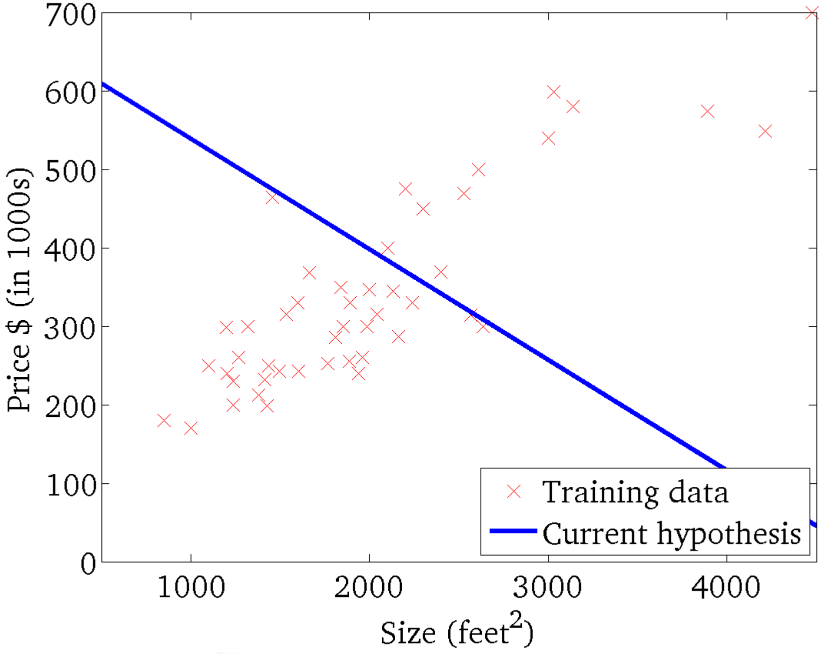


h(x) – это функция x для фиксированных β_0 и β_1

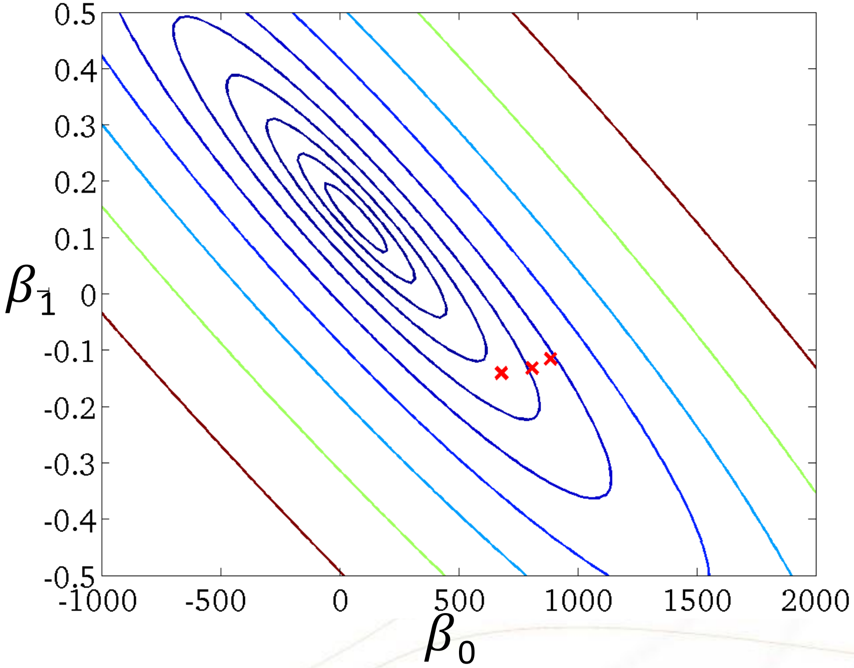


$J(\beta_0, \beta_1)$ – это функция параметров β_0 и β_1

Пример работы градиентного спуска для линейной регрессии

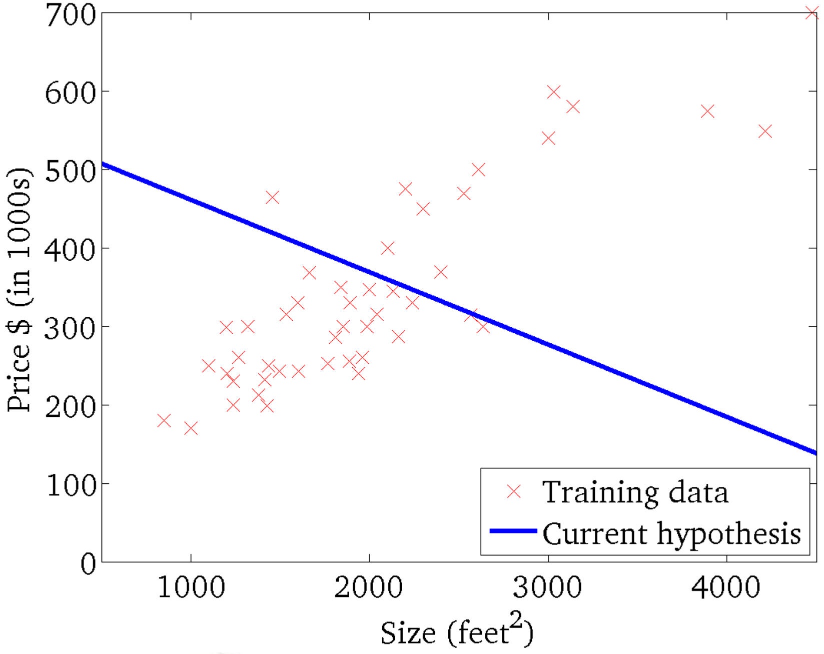


h(x) – это функция x для фиксированных β_0 и β_1

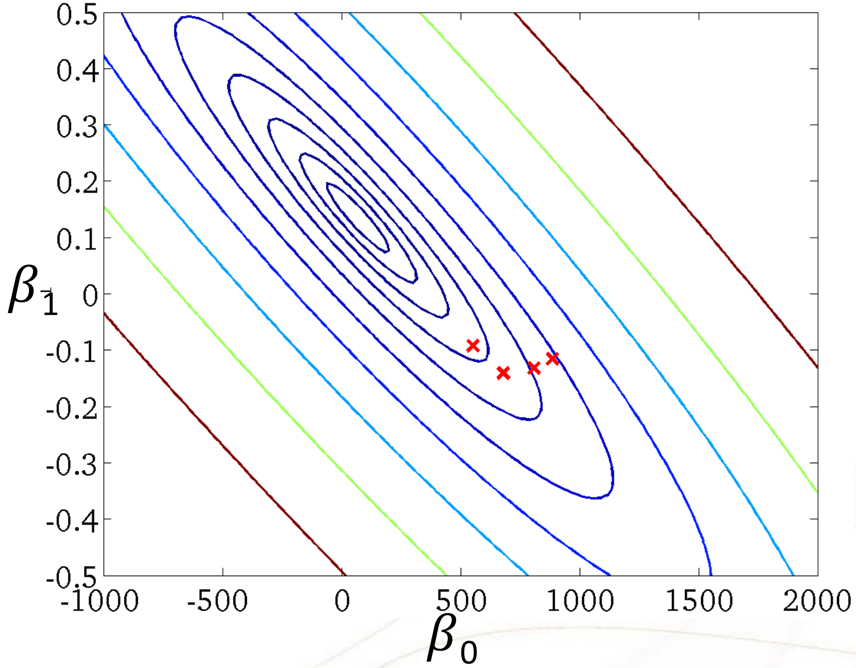


J(β_0, β_1) – это функция параметров β_0 и β_1

Пример работы градиентного спуска для линейной регрессии

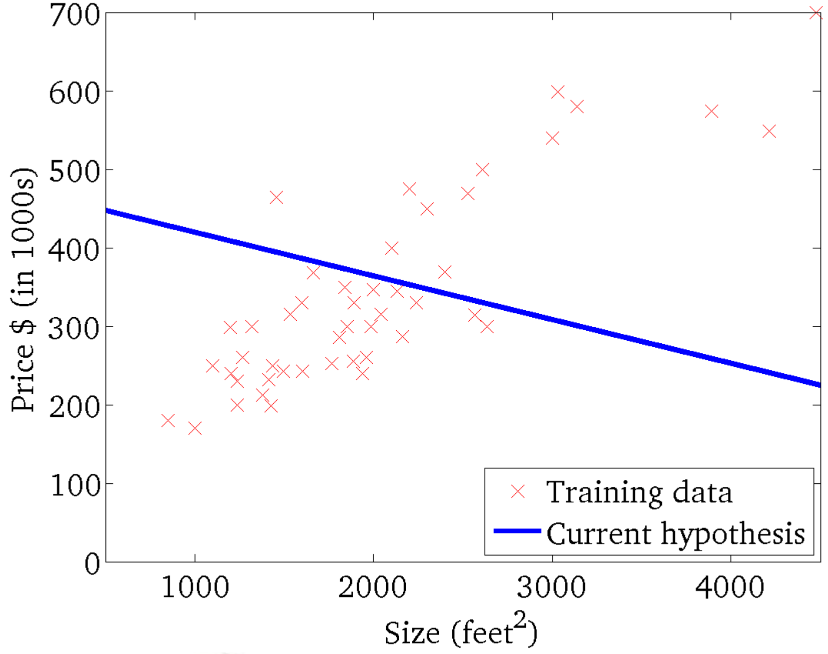


h(x) – это функция x для фиксированных β_0 и β_1

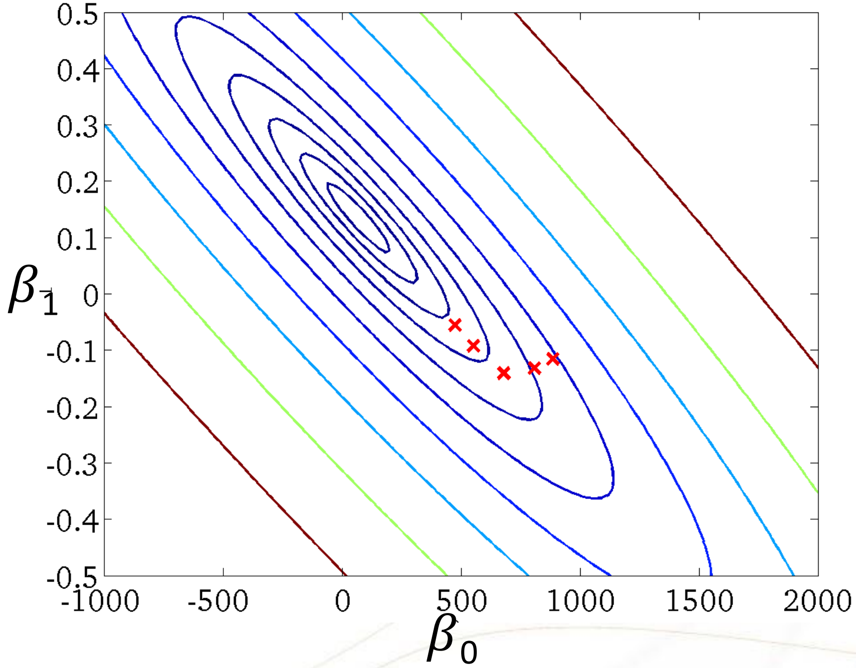


J(β_0, β_1) – это функция параметров β_0 и β_1

Пример работы градиентного спуска для линейной регрессии

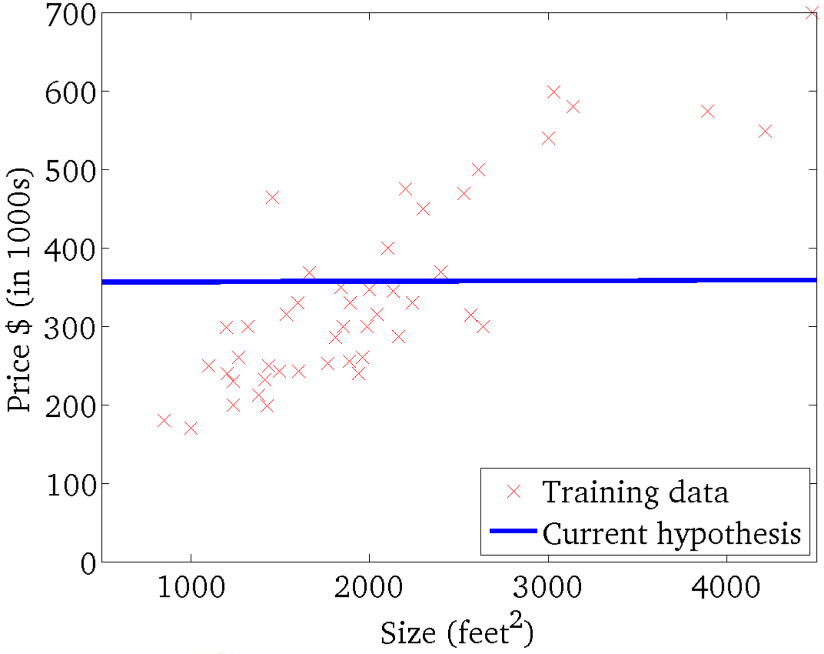


h(x) – это функция x для фиксированных β_0 и β_1

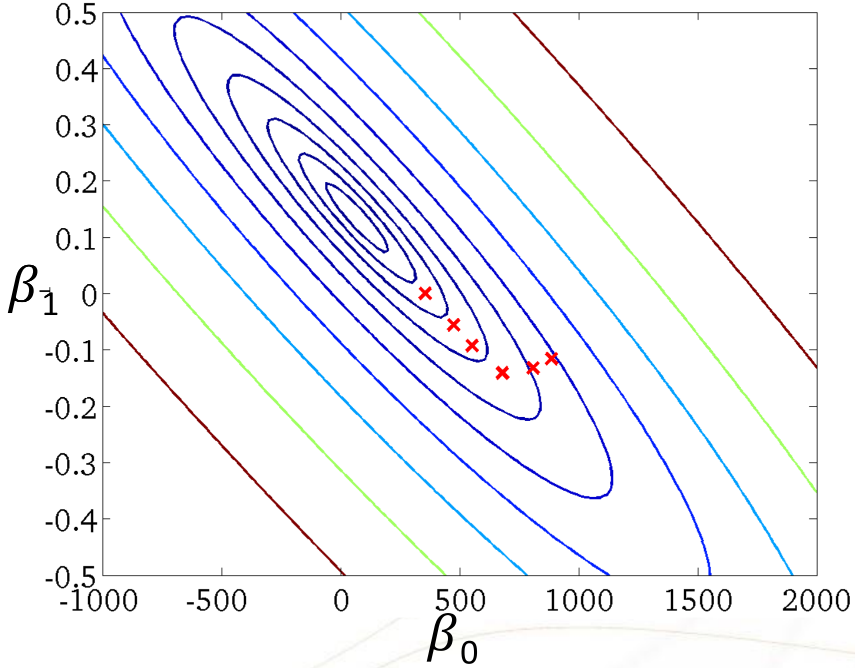


J(β_0, β_1) – это функция параметров β_0 и β_1

Пример работы градиентного спуска для линейной регрессии

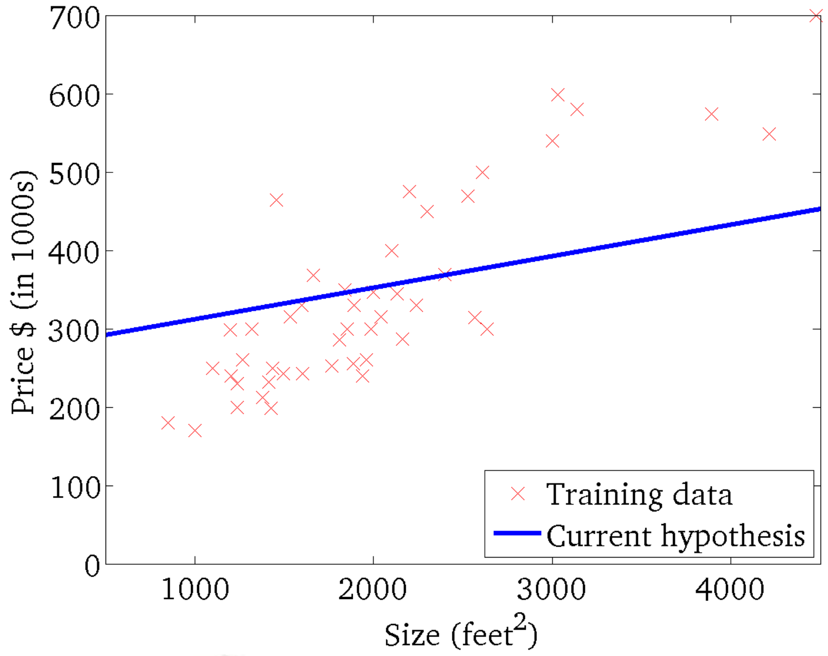


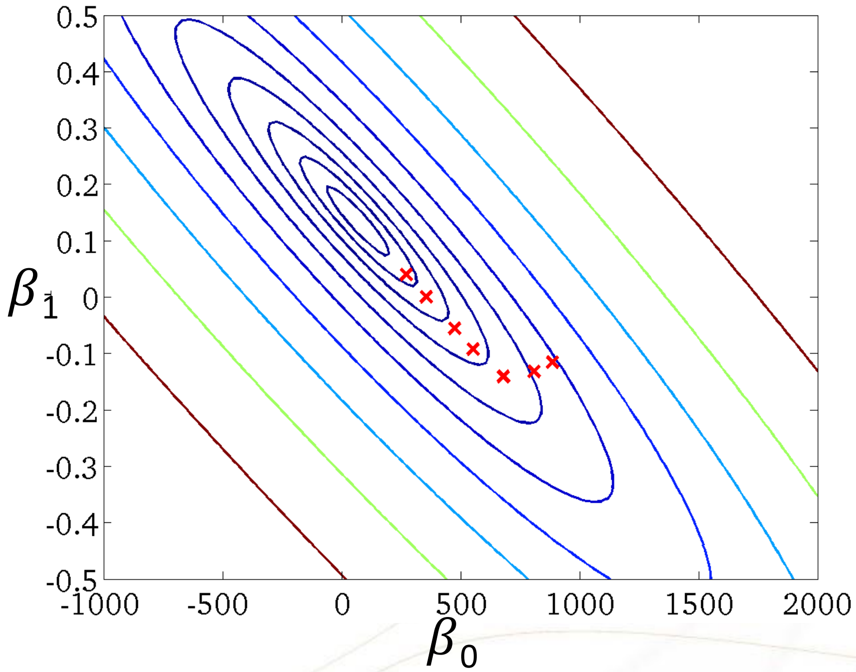
$h_Q(x)$ – это функция x для фиксированных β_0 и β_1



$J(\beta_0, \beta_1)$ – это функция параметров β_0 и β_1

Пример работы градиентного спуска для линейной регрессии

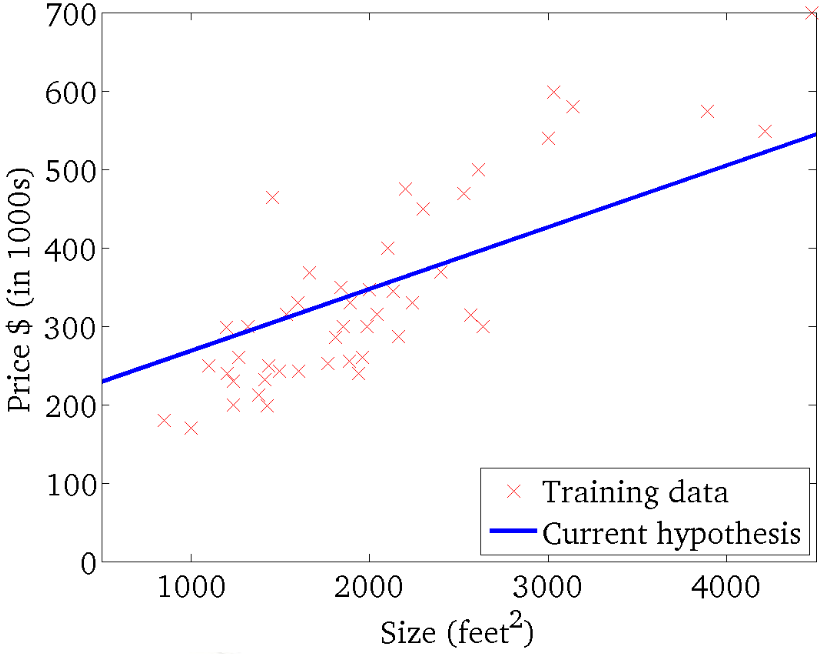




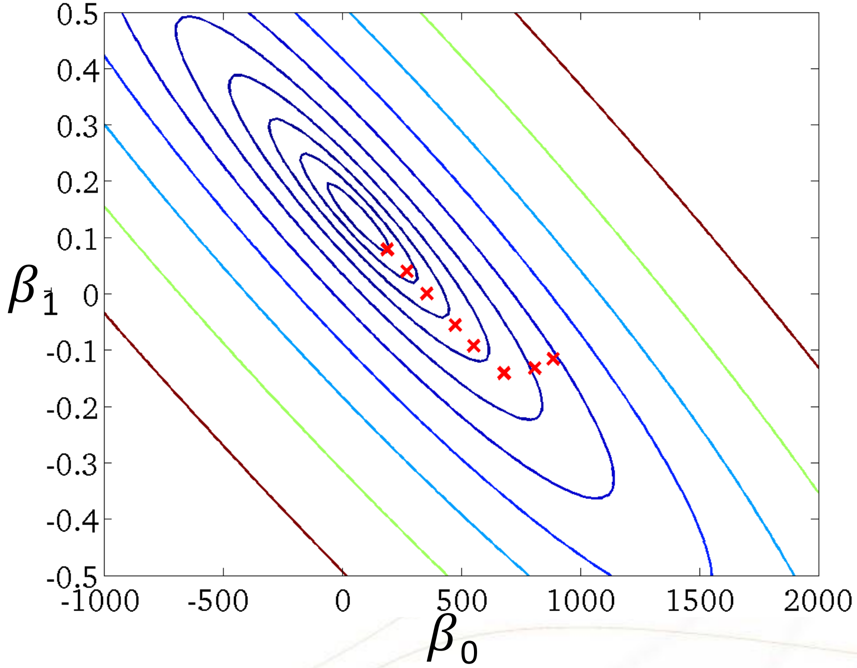
h(x) – это функция x для фиксированных β_0 и β_1

J(β_0, β_1) – это функция параметров β_0 и β_1

Пример работы градиентного спуска для линейной регрессии

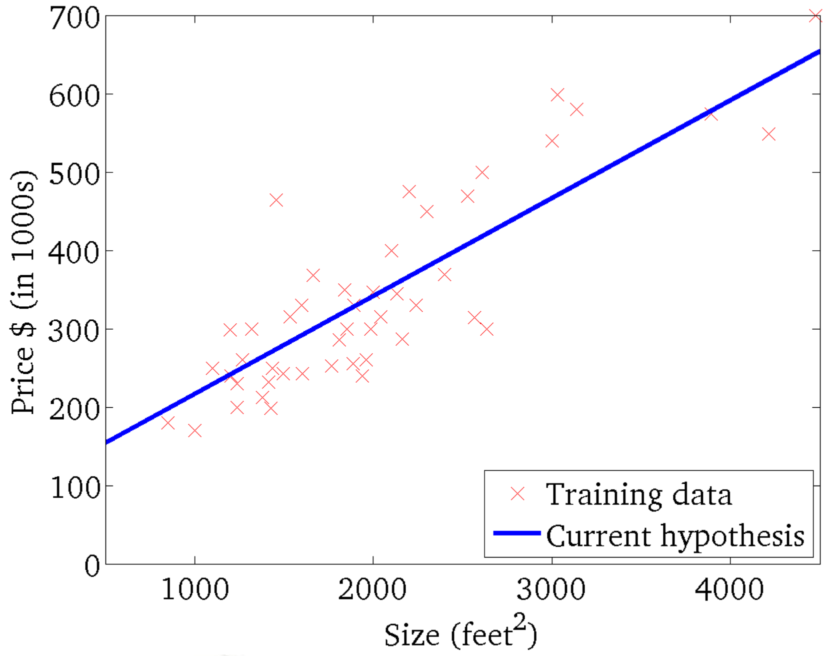


h(x) – это функция x для фиксированных β_0 и β_1

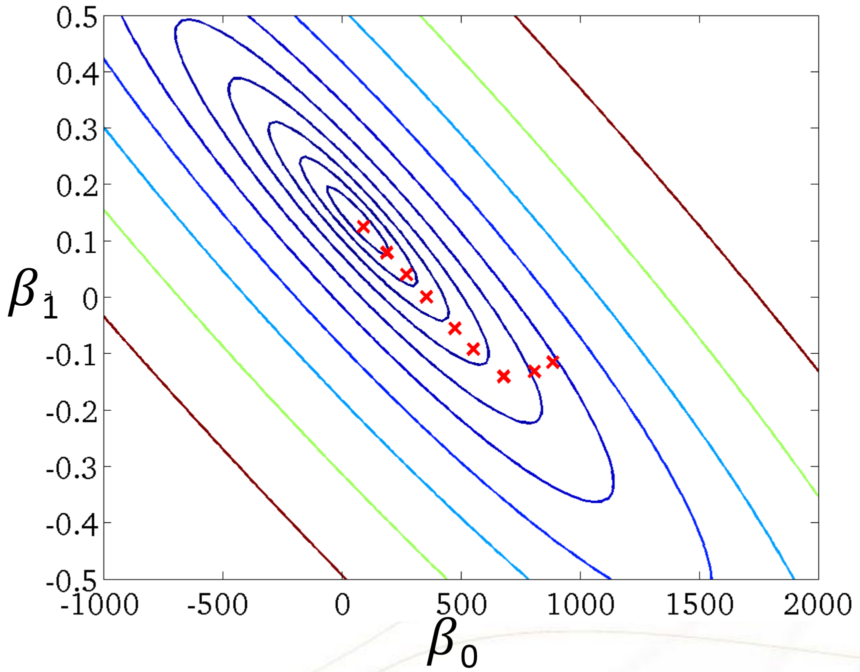


J(β_0, β_1) – это функция параметров β_0 и β_1

Пример работы градиентного спуска для линейной регрессии



h(x) – это функция x для фиксированных β_0 и β_1



J(β_0, β_1) – это функция параметров β_0 и β_1