

# Машинное обучение

Лекция 4

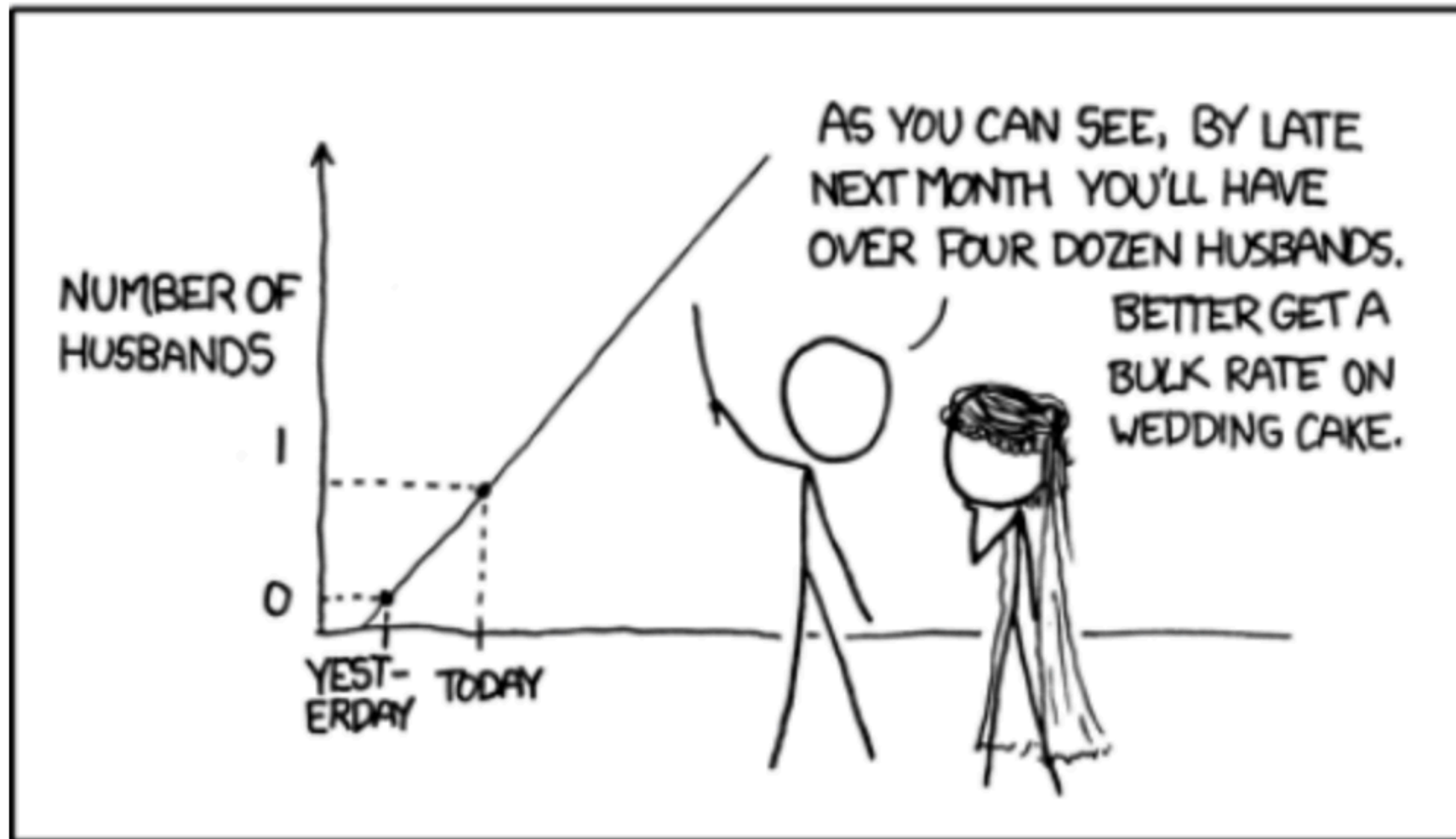
Линейные модели

*Власов Кирилл Вячеславович*



# Линейные модели

MY HOBBY: EXTRAPOLATING



# Линейные модели

Линейная модель - взвешенная сумма признаков и член смещения (*bias term*), который также называют свободным членом (intercept term)

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n$$

$\hat{y}$  Предсказываемое значение

$n$  Количество признаков

$x_1, x_2, \dots, x_n$  Значения признаков

$\theta_0, \theta_1, \theta_2, \dots, \theta_n$  Веса признаков и свободный член

Векторная форма:

$$\hat{y} = h_{\theta}(x) = \theta^T \cdot x$$

$\theta$  Вектор весов и свободный член

$x$  Вектор значений признаков примера, где  $x_0 = 0$

# Линейные модели

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n$$

*Средняя квадратичная ошибка (Mean Squared Error, MSE):*

$$MSE = \frac{1}{l} \sum_{i=1}^l (f(x_i) - y_i)^2$$

$$MSE = \frac{1}{l} \sum_{i=1}^l (\hat{y}^{(i)} - y_i)^2 \rightarrow \frac{1}{l} \sum_{i=1}^l (\theta^T \cdot x^{(i)} - y_i)$$

# Метод наименьших квадратов

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n$$

*Средняя квадратичная ошибка (Mean Squared Error, MSE):*

$$MSE = \frac{1}{l} \sum_{i=1}^l (f(x_i) - y_i)^2$$

$$MSE = \frac{1}{l} \sum_{i=1}^l (\hat{y}^{(i)} - y_i)^2 \rightarrow \frac{1}{l} \sum_{i=1}^l (\theta^T \cdot x^{(i)} - y_i)^2 \rightarrow \min$$

*Аналитический способ поиска оптимальных весов (нормальное уравнение):*

$$\hat{\theta} = (X^T \cdot X)^{-1} \cdot X^T \cdot y$$

$X$  Матрица объектов признаков

$y$  Вектор целевой переменной

$\hat{\theta}$  Оптимальный вектор весов, который сводит к минимуму MSE

Подробный и понятный вывод: Открытый курс машинного обучения: Тема 4

# Метод наименьших квадратов

## Теорема Гаусса — Маркова

оценки **метода наименьших квадратов** оптимальны в классе линейных несмещённых оценок

*Условия для парной регрессии:*

1. модель данных правильно специфицирована. Нет лишних переменных, или учтены все важные  $Y = \theta_0 + \theta \cdot X + \epsilon$
2. все  $X$  детерминированы и не все равны между собой. Иными словами, переменные не должны быть постоянными.
3. Ошибки не носят систематического характера, то есть  $E(\epsilon_i) = 0 \forall i$
4. Дисперсия ошибок одинакова (гомоскедастичность) и равна некоторой  $\sigma^2 = const$
5. Ошибки некоррелированы, то есть  $cov(\epsilon_i, \epsilon_j) = 0 \forall i, j$

*Условия для Множественной регрессии:*

1. модель данных правильно специфицирована. Нет лишних переменных, или учтены все важные
2.  $rang(X) = m$
3.  $E(\epsilon_i) = 0 \forall i$
4.  $cov(\epsilon_i, \epsilon_j) = 0 \forall i, j$

**Реализация в python**

# Проблемы нормального уравнения

$$\hat{\theta} = (X^T \cdot X)^{-1} \cdot X^T \cdot y$$



# Проблемы нормального уравнения

$$\hat{\theta} = (X^T \cdot X)^{-1} \cdot X^T \cdot y$$

$(X^T \cdot X)$  – матрица размером  $n \times n$ , где  $n$  – количество признаков

Вычислительная сложность для обратной матрицы:  $O(n^3)$

# Градиентный спуск

$$MSE = \frac{1}{l} \sum_{i=1}^l (\hat{y}^{(i)} - y_i)^2 \rightarrow \frac{1}{l} \sum_{i=1}^l (\theta^T \cdot x^{(i)} - y_i)^2 \rightarrow \min$$

# Градиентный спуск

$$MSE = \frac{1}{l} \sum_{i=1}^l (\hat{y}^{(i)} - y_i)^2 \rightarrow \frac{1}{l} \sum_{i=1}^l (\theta^T \cdot x^{(i)} - y_i)^2 \rightarrow \min$$

*Градиент - Вектор указывающий направление наибольшего возрастания функции, компоненты которого равны частным производным по всем её аргументам.*

$$\nabla \varphi = \left( \frac{\partial \varphi}{\partial x_1}, \frac{\partial \varphi}{\partial x_2}, \dots, \frac{\partial \varphi}{\partial x_n}, \right)$$

# Градиентный спуск

$$MSE = \frac{1}{l} \sum_{i=1}^l (\hat{y}^{(i)} - y_i)^2 \rightarrow \frac{1}{l} \sum_{i=1}^l (\theta^T \cdot x^{(i)} - y_i)^2 \rightarrow \min$$

Градиент - Вектор указывающий направление наибольшего возрастания функции, компоненты которого равны *частным производным* по всем её аргументам.

$$\nabla \varphi = \left( \frac{\partial \varphi}{\partial x_1}, \frac{\partial \varphi}{\partial x_2}, \dots, \frac{\partial \varphi}{\partial x_n} \right)$$

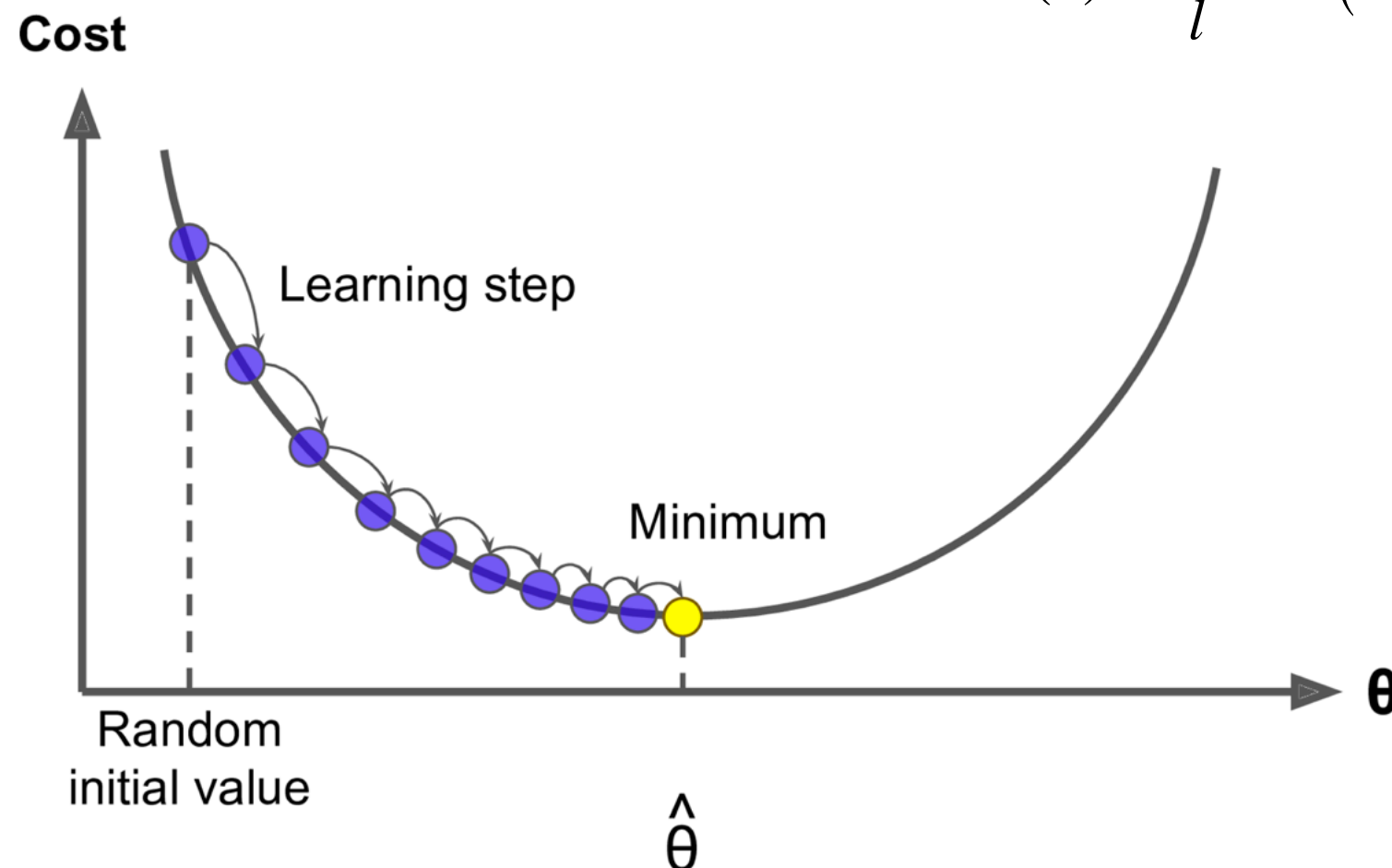
$$\frac{\partial MSE(\theta)}{\partial \theta_j} = \frac{2}{l} \sum_{i=1}^l (\theta^T \cdot x^{(i)} - y^{(i)}) \cdot x_j^{(i)} \quad \nabla MSE(\theta) = \begin{pmatrix} \frac{\partial MSE(\theta)}{\partial \theta_0} \\ \frac{\partial MSE(\theta)}{\partial \theta_1} \\ \dots \\ \frac{\partial MSE(\theta)}{\partial \theta_n} \end{pmatrix} = \frac{2}{l} X^T \cdot (X \cdot \theta - y)$$

# Градиентный спуск

$$MSE = \frac{1}{l} \sum_{i=1}^l (\hat{y}^{(i)} - y_i)^2 \rightarrow \frac{1}{l} \sum_{i=1}^l (\theta^T \cdot x^{(i)} - y_i)^2 \rightarrow \min$$

Градиент - Вектор указывающий направление наибольшего возрастания функции, компоненты которого равны *частным производным* по всем её аргументам.

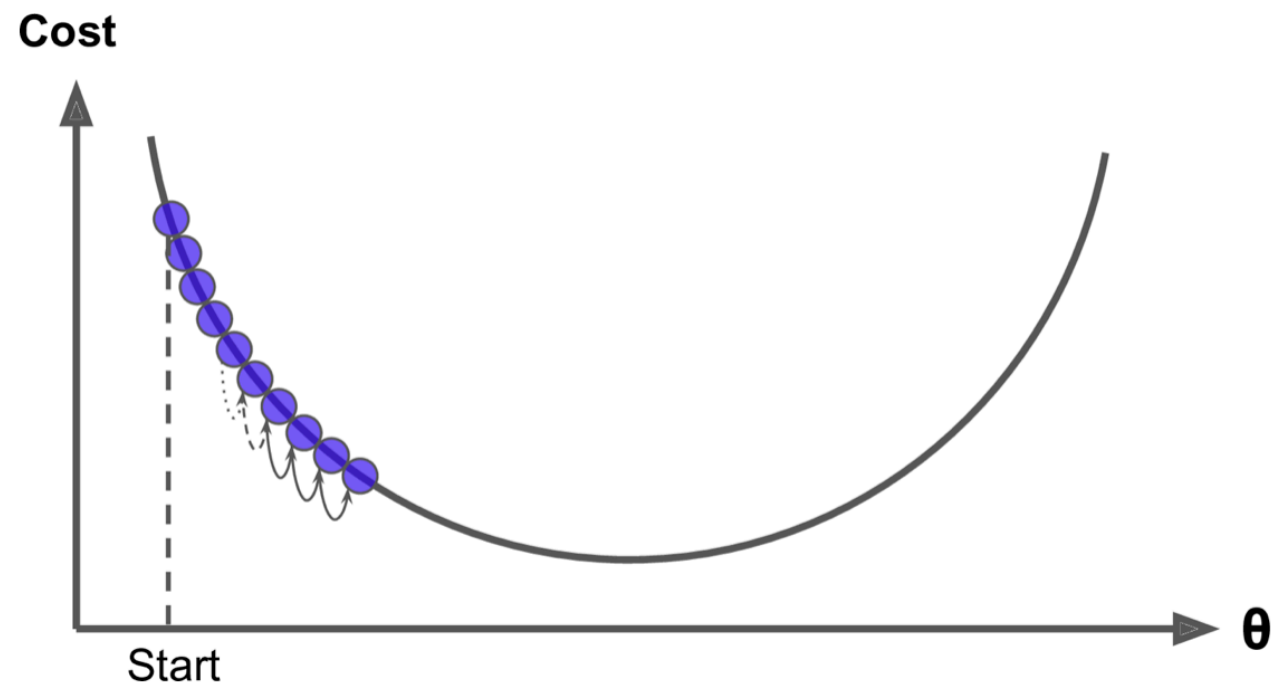
$$\nabla MSE(\theta) = \frac{2}{l} X^T \cdot (X \cdot \theta - y)$$



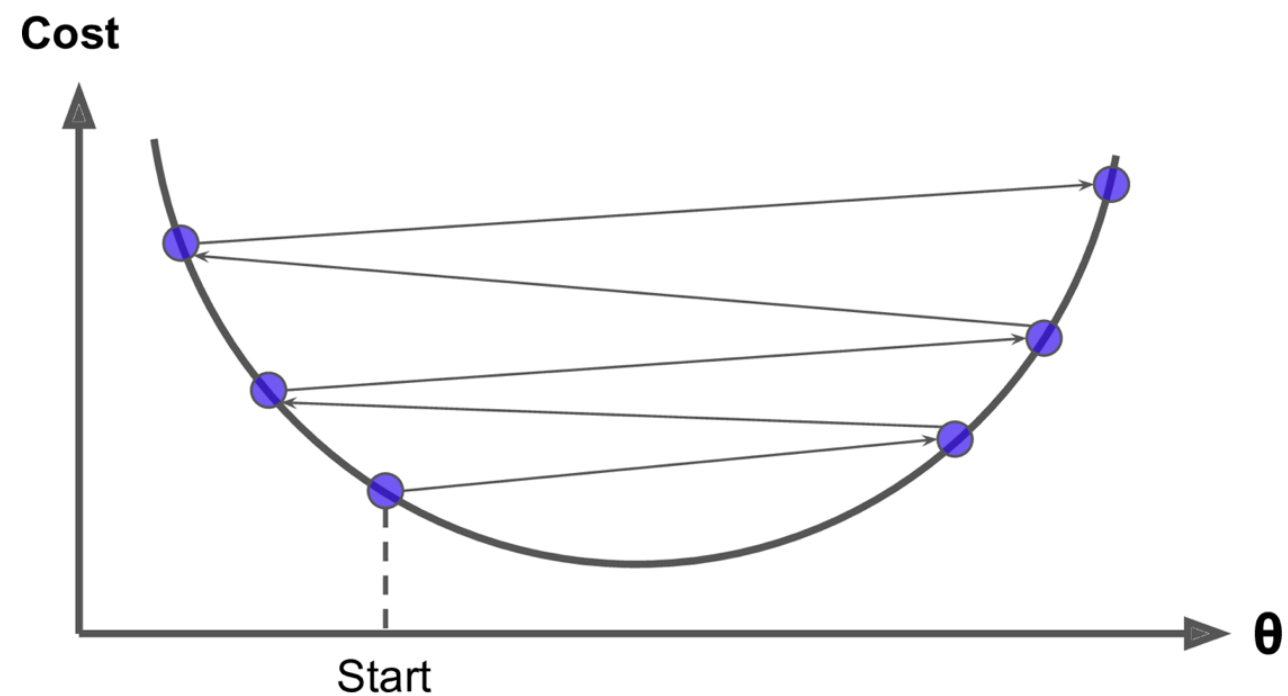
1. Случайно задаем веса
2. Считаем градиент в точке
3. Изменяем веса путем вычитания градиента
4. Повторяем п.2

\* Мы можем контролировать скорость обучения (learning rate) умножая градиент на шаг обучения

# Выбор шага обучения градиентного спуска



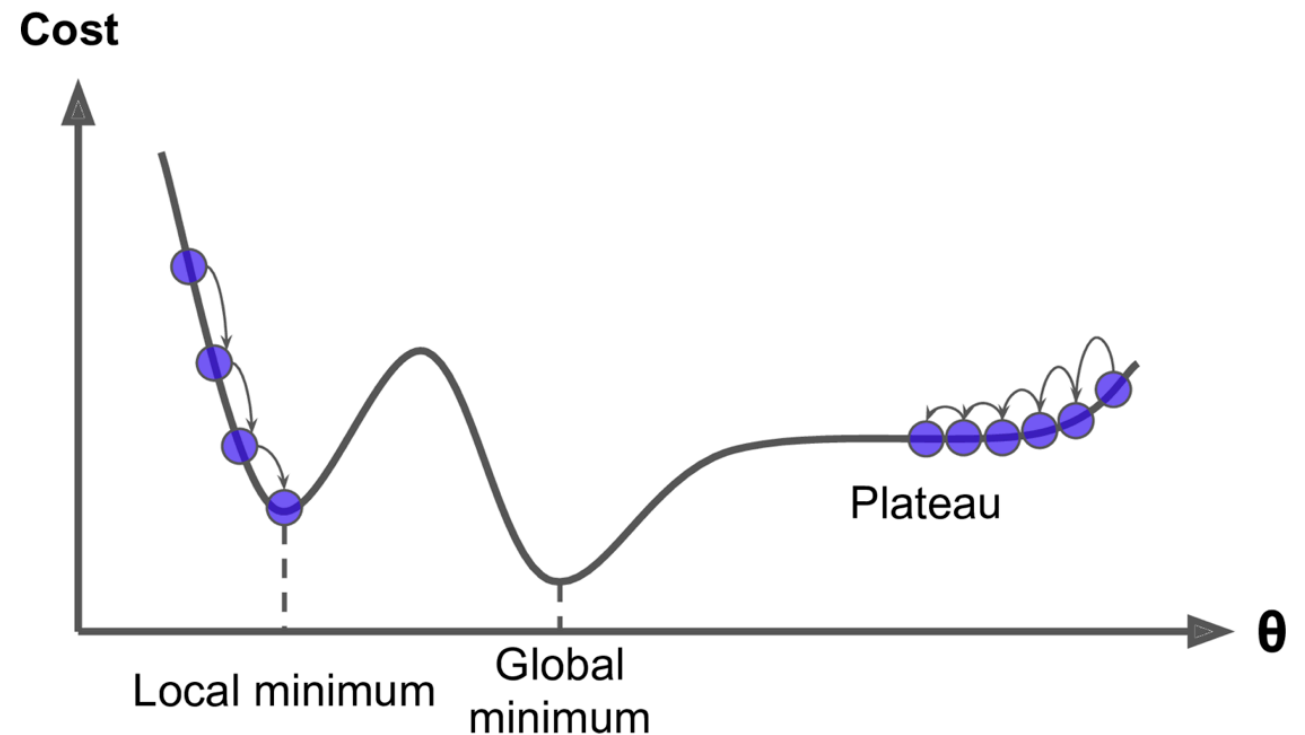
**Слишком маленький шаг обучения**  
рисуем не дойти до минимума



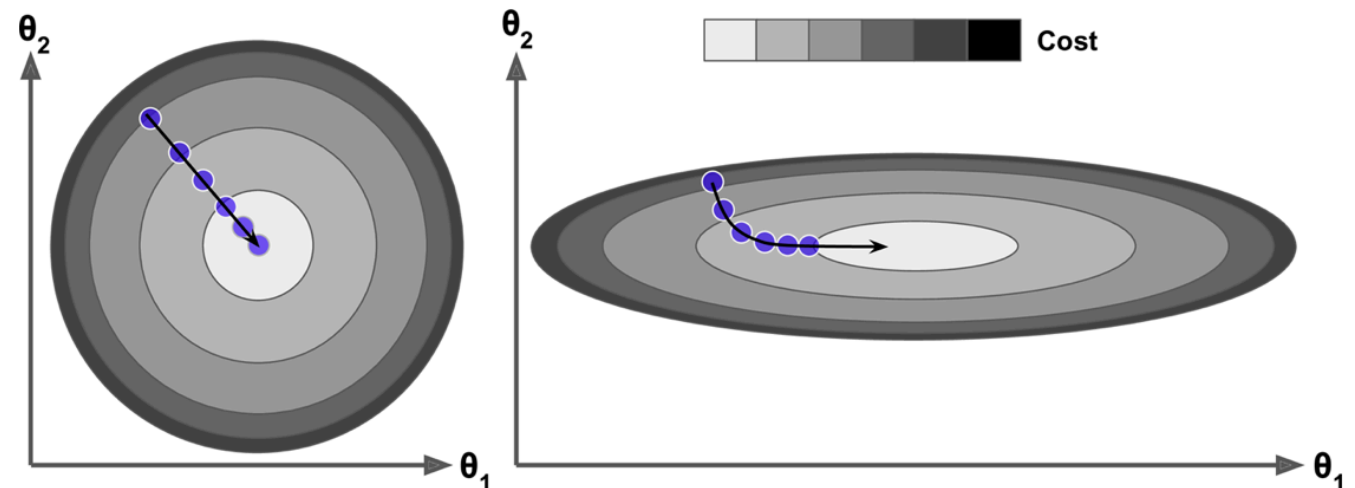
**Слишком большой шаг обучения**  
рисуем проскочить минимум

# Другие проблемы градиентного спуска

Не все функции потерь  
одинаково полезны  
имеют форму чаши  
(параболоид)



Важно масштабировать  
данные



# Реализация в python



# Стохастический градиентный спуск

$$MSE = \frac{1}{l} \sum_{i=1}^l (\hat{y}^{(i)} - y_i)^2 \rightarrow \frac{1}{l} \sum_{i=1}^l (\theta^T \cdot x^{(i)} - y_i)^2 \rightarrow \min$$


$$\frac{\partial MSE(\theta)}{\partial \theta_j} = \frac{2}{l} \sum_{i=1}^l (\theta^T \cdot x^{(i)} - y^{(i)}) \cdot x_j^{(i)}$$

# Стохастический градиентный спуск

$$MSE = \frac{1}{l} \sum_{i=1}^l (\hat{y}^{(i)} - y_i)^2 \rightarrow \frac{1}{l} \sum_{i=1}^l (\theta^T \cdot x^{(i)} - y_i)^2 \rightarrow \min$$

$$\frac{\partial MSE(\theta)}{\partial \theta_j} = \frac{2}{l} \sum_{i=1}^l (\theta^T \cdot x^{(i)} - y^{(i)}) \cdot x_j^{(i)}$$

Пакетный градиентный спуск



$$\frac{\partial MSE(\theta)}{\partial \theta_j} = (\theta^T \cdot x^{(i)} - y^{(i)}) \cdot x_j^{(i)}$$

$$\nabla MSE(\theta) = x_i^T \cdot (x_i \cdot \theta - y)$$

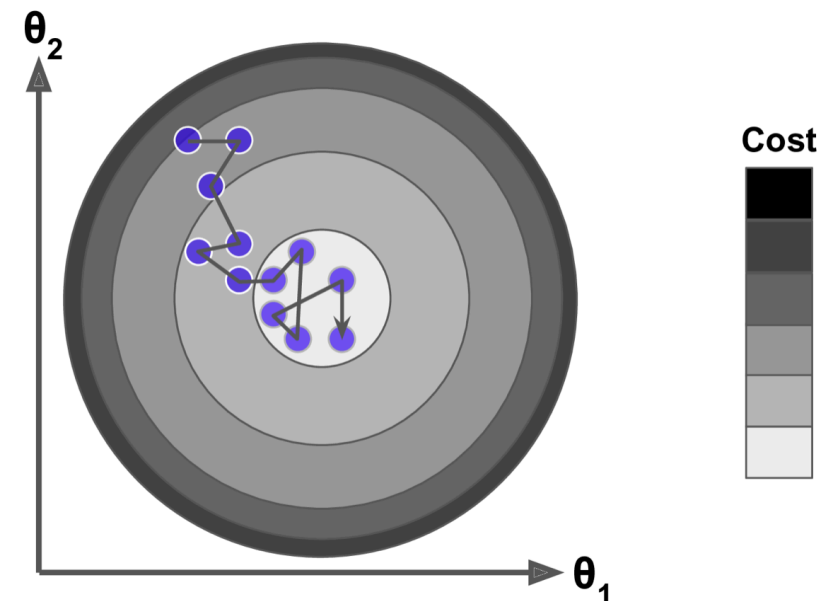
# Стохастический градиентный спуск

$$MSE = \frac{1}{l} \sum_{i=1}^l (\hat{y}^{(i)} - y_i)^2 \rightarrow \frac{1}{l} \sum_{i=1}^l (\theta^T \cdot x^{(i)} - y_i)^2 \rightarrow \min$$

$$\frac{\partial MSE(\theta)}{\partial \theta_j} = \frac{2}{l} \sum_{i=1}^l (\theta^T \cdot x^{(i)} - y^{(i)}) \cdot x_j^{(i)}$$

$$\frac{\partial MSE(\theta)}{\partial \theta_j} = (\theta^T \cdot x^{(i)} - y^{(i)}) \cdot x^{(i)}$$

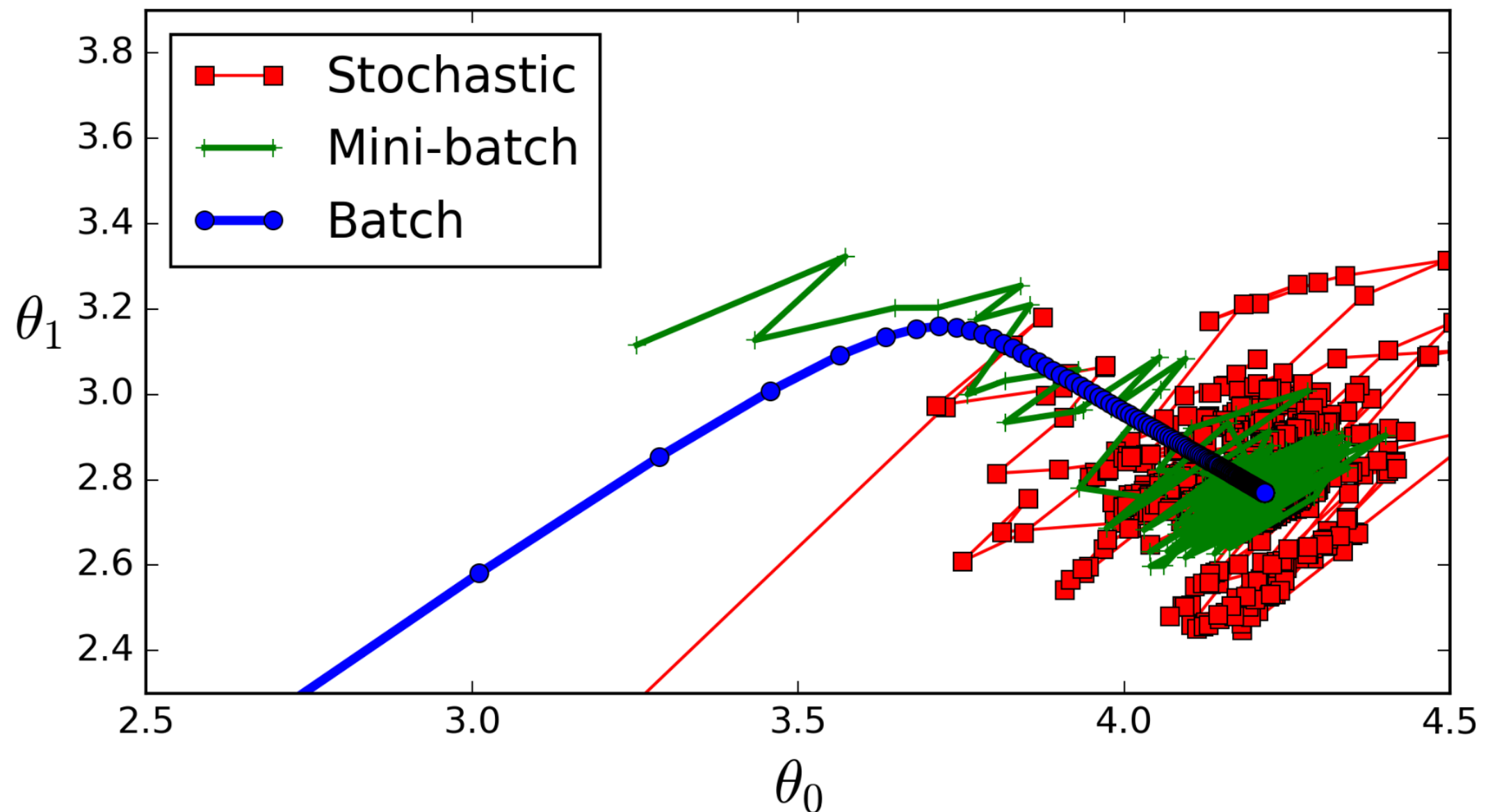
$$\nabla MSE(\theta) = x_i^T \cdot (x_i \cdot \theta - y)$$



Случайно выбираем объект, и двигаемся к минимуму.  
Каждый объект выборки может прогоняться несколько раз или быть не выбран вообще

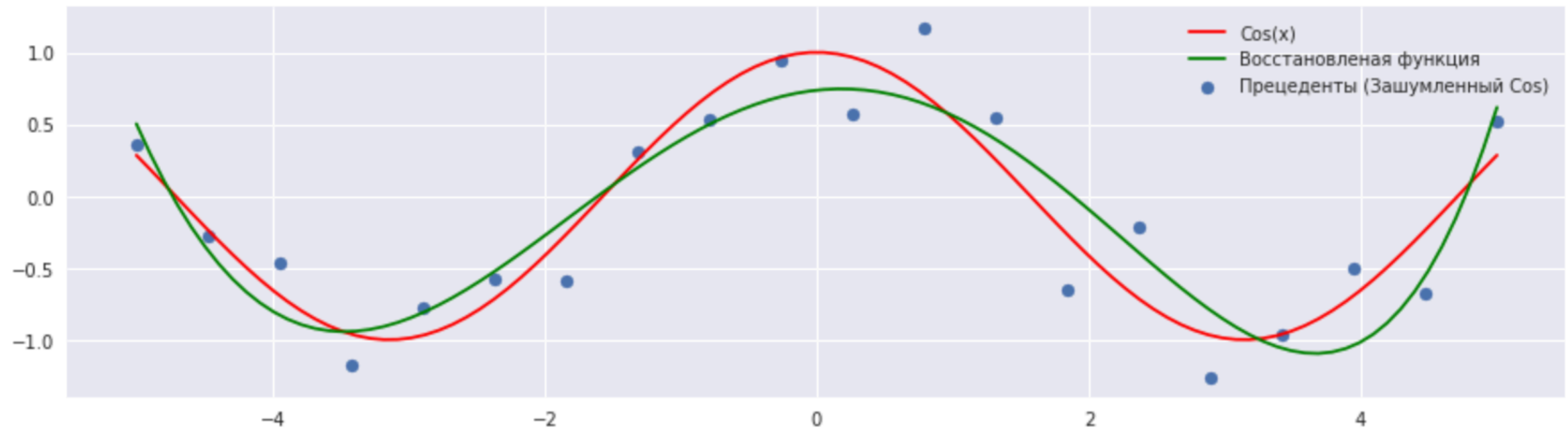
# Выбор метода градиентного спуска

Можно скрестить два подхода: Стохастический и пакетный и выбирать случайные подборки например из 100 объектов, тогда получится: метод называемый Mini-batch

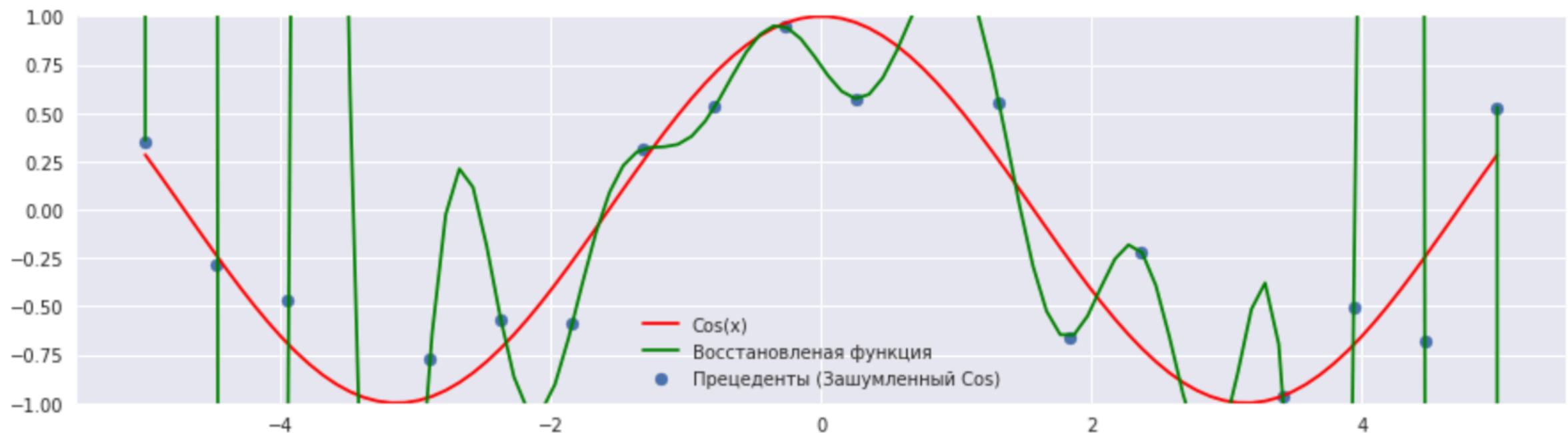


# Напоминание из лекции 2

Восстановим зависимость с помощью полинома 5-ого порядка



Восстановим зависимость с помощью полинома 11-ого порядка



# Полиномиальные функции

$$\hat{y} = \theta_0 + \theta_1 x + \theta_2 x^2 + \dots + \theta_n x^n$$



`sklearn.preprocessing.PolynomialFeatures`

*(degree=2, interaction\_only=False, include\_bias=True)*

**Используя описанные выше методы, с помощью линейной регрессии мы можем обучать нелинейные модели**

При увеличении степени полинома выше третьей модель начинает *интерполировать* данные, вместо *экстраполяции*.

Подробнее: [Базовые принципы машинного обучения на примере линейной регрессии](#)

# Регуляризация линейных моделей

1. Одно из условий Гауса-Маркова:  $\text{rang}(X) = m$

Если оно не выполняется, то решение МНК  $\hat{\theta} = (X^T \cdot X)^{-1} \cdot X^T \cdot y$  не существует

так как Матрица  $X^T \cdot X$  сингулярна (вырождена)

Значит нам нужно сделать так, чтобы сделать матрицу вырожденной (регулярной)

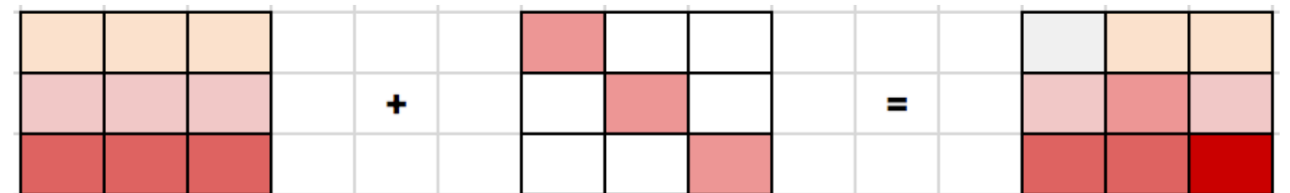
2. Мультиколлениарность

Собственные значения будут стремиться к 0, а при обращении матрицы к  $\infty$

# Регуляризация линейных моделей

L2 - регуляризация, гребневая регрессия, ridge

$$MSE(\theta) + \alpha \frac{1}{2} \sum_{i=1}^n \theta_i^2 \rightarrow \min$$

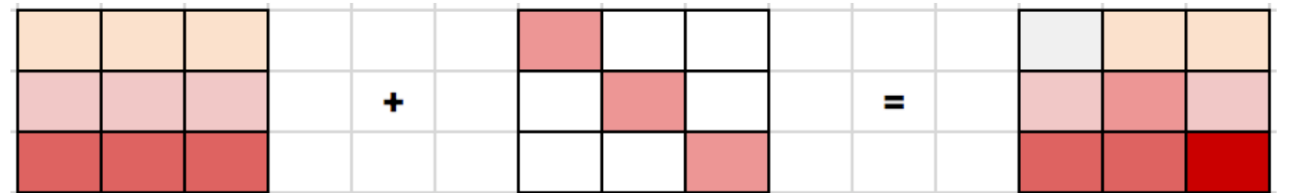




# Регуляризация линейных моделей

L2 - регуляризация, гребневая регрессия, ridge

$$MSE(\theta) + \alpha \frac{1}{2} \sum_{i=1}^n \theta_i^2 \rightarrow \min$$



L1 - регуляризация, Lasso (Least absolute shrinkage and selection operator)

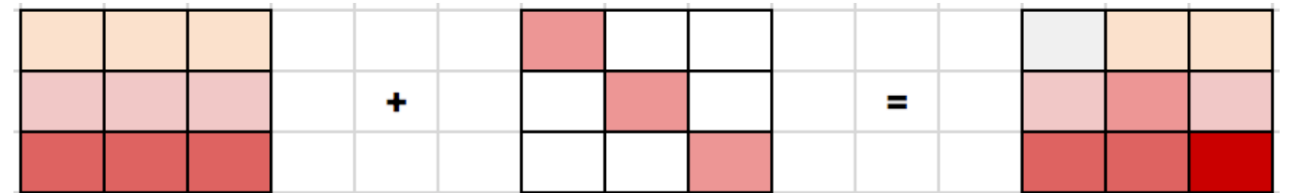
$$MSE(\theta) + \alpha \sum_{i=1}^n |\theta_i| \rightarrow \min$$

*введение ограничений на норму вектора коэффициентов модели  
приводит к обращению в 0 некоторых коэффициентов модели.*

# Регуляризация линейных моделей

L2 - регуляризация, гребневая регрессия, ridge

$$MSE(\theta) + \alpha \frac{1}{2} \sum_{i=1}^n \theta_i^2 \rightarrow \min$$



L1 - регуляризация, Lasso (Least absolute shrinkage and selection operator)

$$MSE(\theta) + \alpha \sum_{i=1}^n |\theta_i| \rightarrow \min$$

*введение ограничений на норму вектора коэффициентов модели приводит к обращению в 0 некоторых коэффициентов модели.*

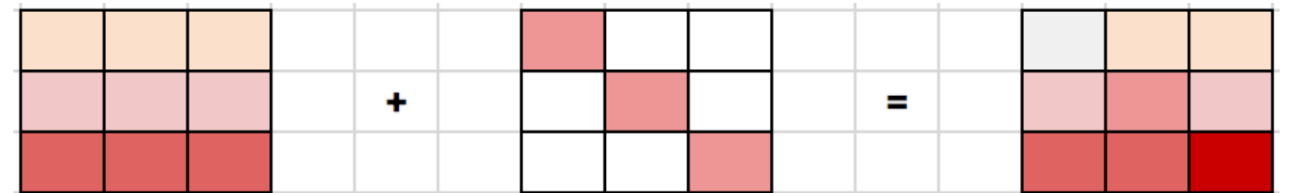
ElasticNet

$$MSE(\theta) + \alpha r \sum_{i=1}^n |\theta_i| + \alpha \frac{1-r}{2} \sum_{i=1}^n \theta_i^2 \rightarrow \min$$

# Регуляризация линейных моделей

L2 - регуляризация, гребневая регрессия, ridge

$$MSE(\theta) + \alpha \frac{1}{2} \sum_{i=1}^n \theta_i^2 \rightarrow \min$$



L1 - регуляризация, Lasso (Least absolute shrinkage and selection operator)

$$MSE(\theta) + \alpha \sum_{i=1}^n |\theta_i| \rightarrow \min$$

*введение ограничений на норму вектора коэффициентов модели приводит к обращению в 0 некоторых коэффициентов модели.*

ElasticNet

$$MSE(\theta) + \alpha r \sum_{i=1}^n |\theta_i| + \alpha \frac{1-r}{2} \sum_{i=1}^n \theta_i^2 \rightarrow \min$$