Лекция 3: модели регрессии

Евгений Борисов

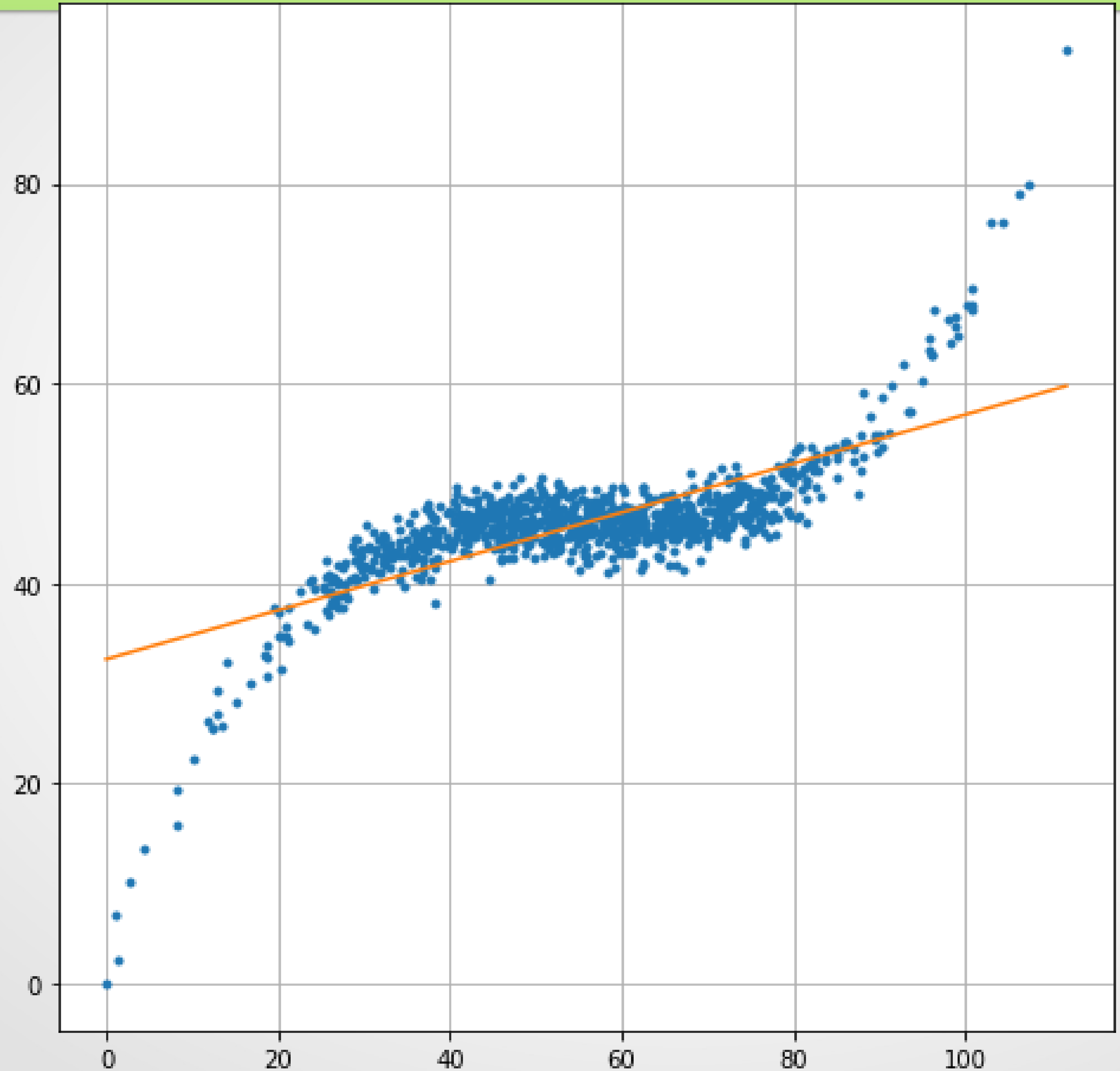
четверг, 20 сентября 2018 г.

Регрессия: определение

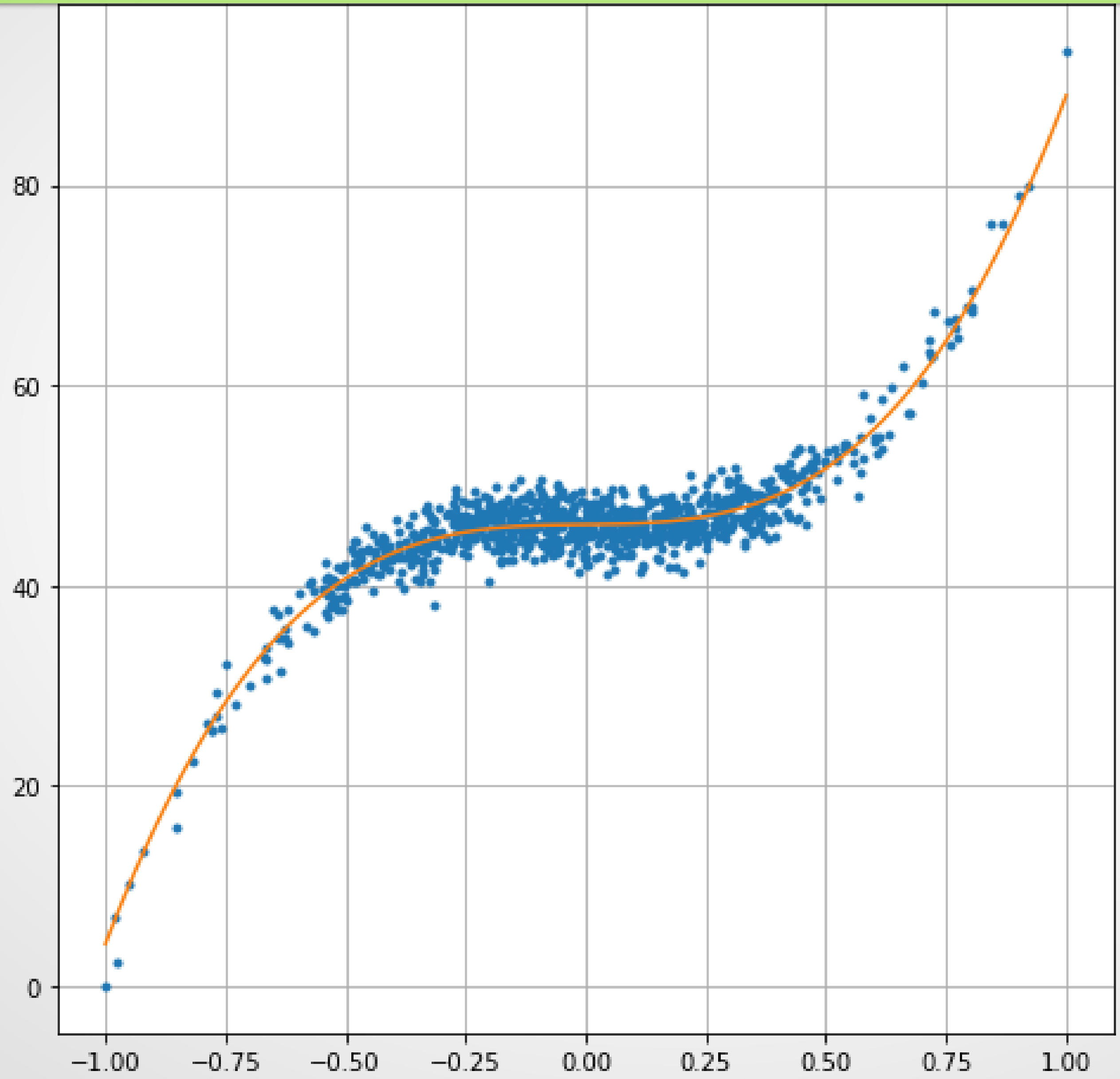
зависимость математического ожидания случайной величины ζ от одной или нескольких других случайных величин ξ (свободных переменных).

$$\zeta = f(\xi_1, \xi_2, \ldots \xi_n)$$

Регрессия: линейная



Регрессия: нелинейная



Регрессия: пример использования

Оценка недвижимости по статистике продаж

```
цена = оценка(
        район,
        площадь,
        этаж,
        лифт,
        ремонт,
      )
```

Регрессия: постановка задачи 1

$$\begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \dots & x_n^{(1)} & y^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \dots & x_n^{(2)} & y^{(2)} \\ \vdots & \vdots & & \vdots & \vdots \\ x_1^{(m)} & x_2^{(m)} & \dots & x_n^{(m)} & y^{(m)} \end{bmatrix}$$

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \ldots + \theta_n x_n$$

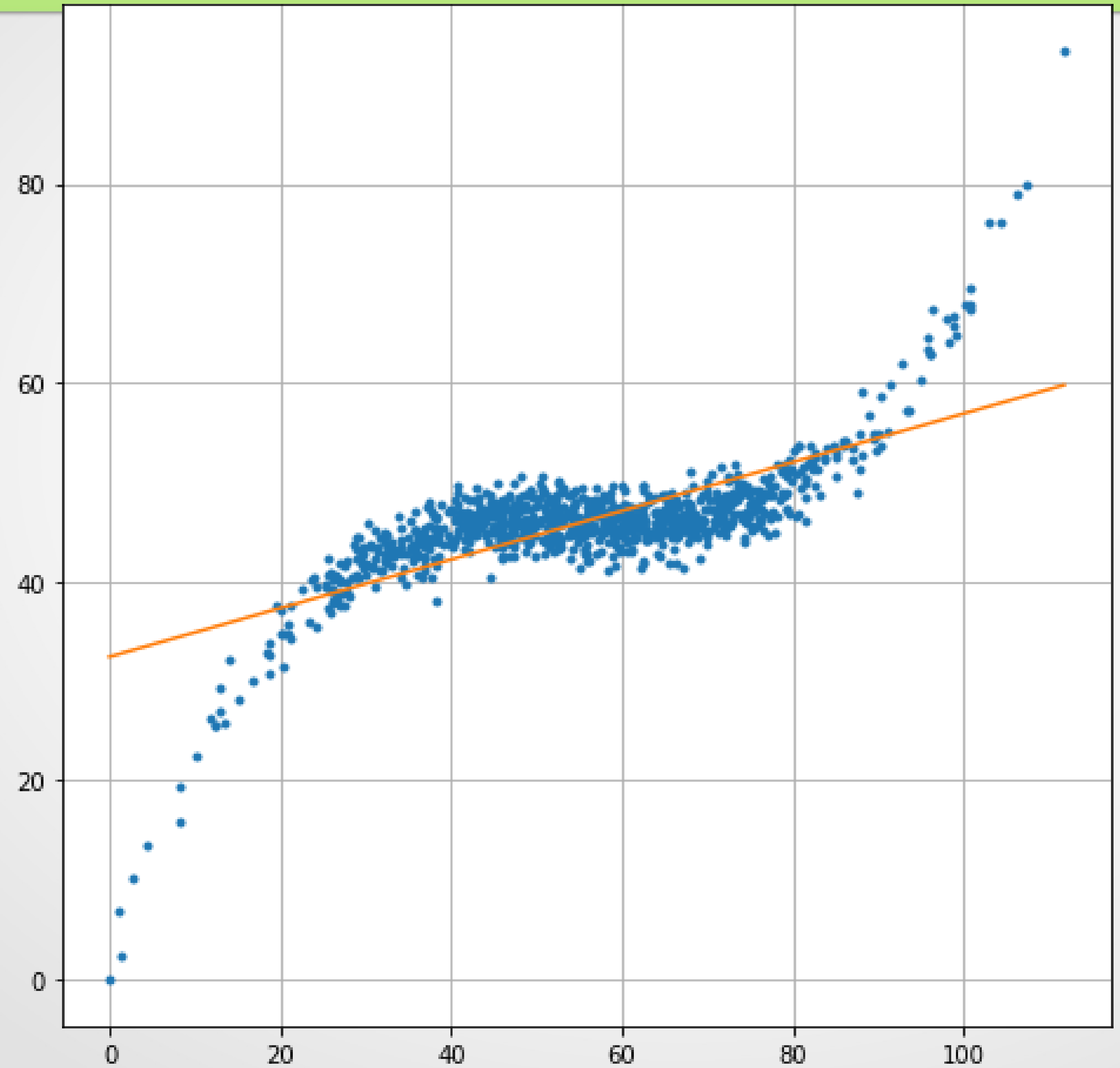
$$y = h(x)$$

Регрессия: постановка задачи 2

$$x = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} \quad ; \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix}$$

$$h_\theta(x) = \theta^T x$$

Регрессия: линейная



Регрессия: решение аналитическое 1

метод наименьших квадратов:

минимизация суммы квадратов отклонений
функции от искомых переменных

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix} ; \quad X = \begin{bmatrix} 1 & x_1^{(1)} & \dots & x_n^{(1)} \\ 1 & x_1^{(2)} & \dots & x_n^{(2)} \\ & \vdots & & \\ 1 & x_1^{(m)} & \dots & x_n^{(m)} \end{bmatrix} ; \quad y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}$$

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

Регрессия: решение аналитическое 2

ограничения метода:

не работает для больших наборов данных

$\mathbf{X}^T\mathbf{X}$ может быть необратимая (вырожденная)
- строки **X** линейно зависимы
- признаков больше чем примеров

Регрессия: решение численное

er(θ) = h(x) - y ф-ция ошибки

J(er(θ)) ф-ция потери (loss)

средняя квадратичная ошибка (MSQE)

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

$$\min_\theta J(\theta)$$ задача оптимизации

Регрессия: метод градиентного спуска 1

задача оптимизации

$$\mathbf{min_\theta(J)}$$

градиент функции $\nabla J(\theta)$ в точке θ

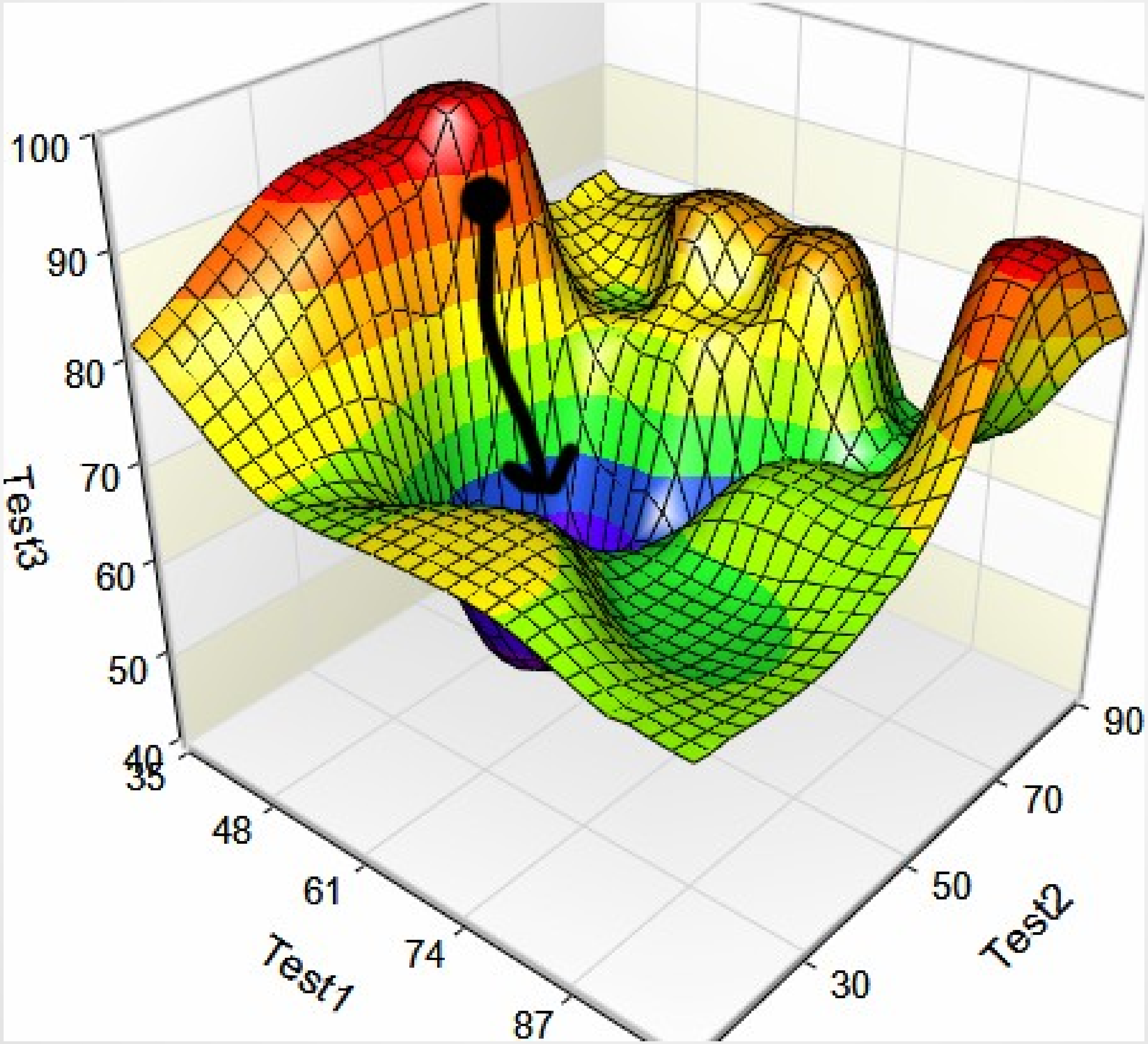
$$\nabla J(\theta) = [\ \partial J/\partial\theta_1 \ \ldots\ ,\ \partial J/\partial\theta_n]$$

∇J направление наискорейшего возрастания ф-ции **J**

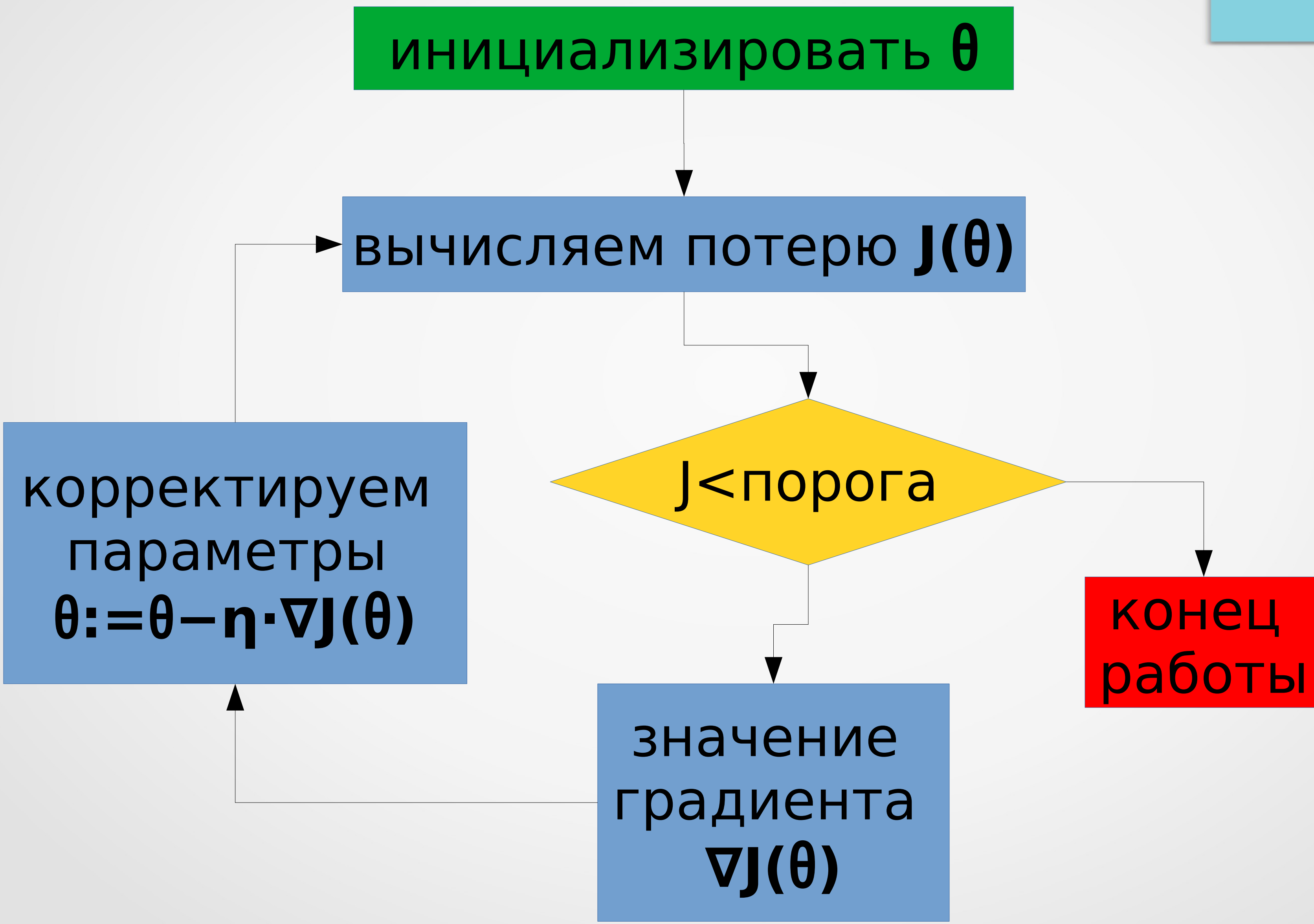
двигаем параметры в противоположную сторону

$$\theta := \theta - \nabla J(\theta)$$

Регрессия: метод градиентного спуска 2



Регрессия: метод градиентного спуска 3



Регрессия: метод градиентного спуска 3

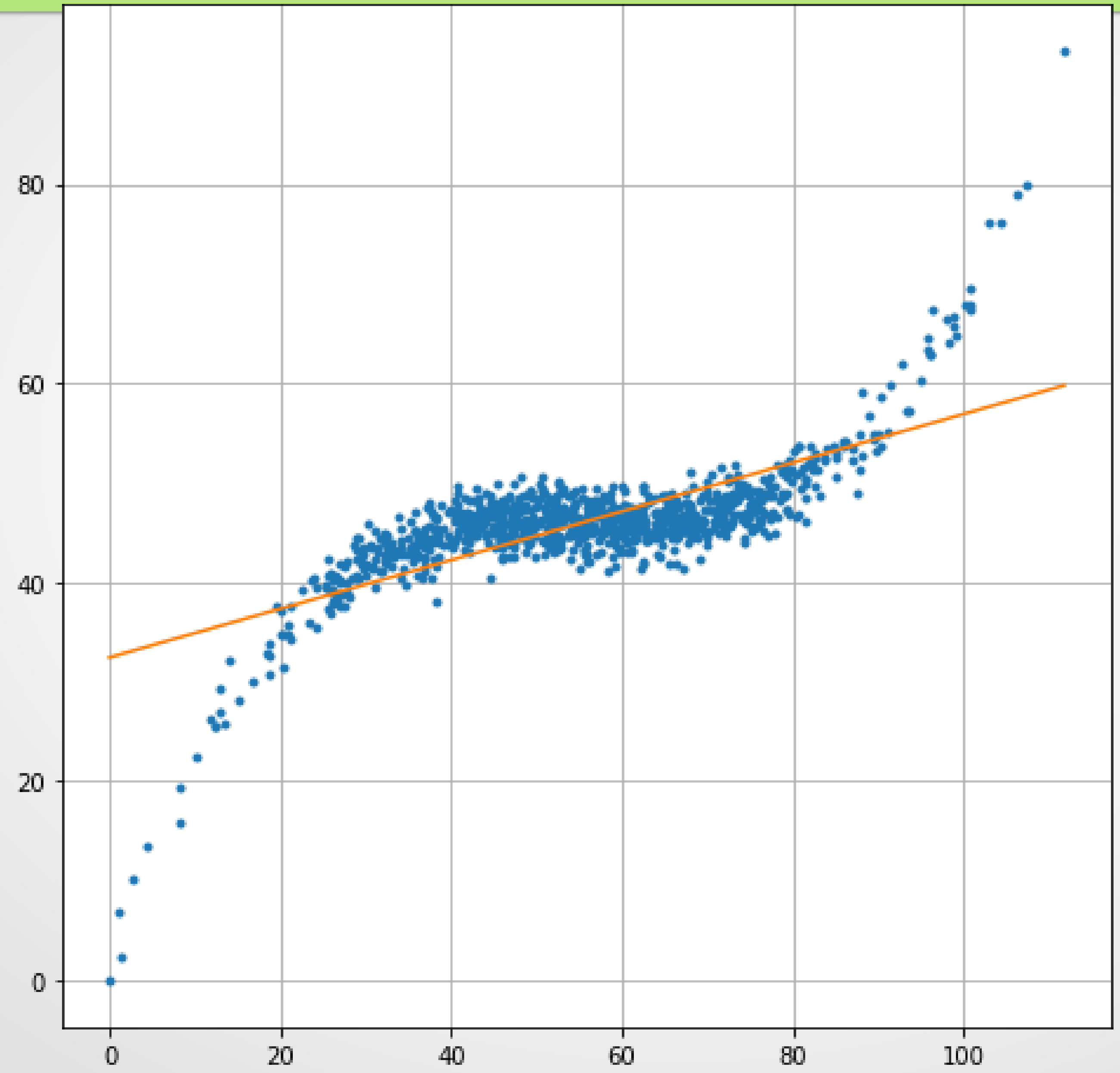
J - ф-ция потери (MSQE)

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

градиент ∇J и изменение весов θ

$$\theta_j := \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta) = \theta_j - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)x_j^{(i)}$$

Регрессия: линейная



Регрессия: нелинейная 1

преобразование исходных данных

увеличиваем размерность пространства

строим полином степени **k**
на признаках **x**, размера **n**

пример для одномерного пространства (n=1) :

строим полином k=1

линейная: $h_{lin}(\theta, x) = \theta_0 + \theta_1 x$

строим полином k=3

нелинейная: $h_{nlin}(\theta, x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$

Регрессия: нелинейная 2

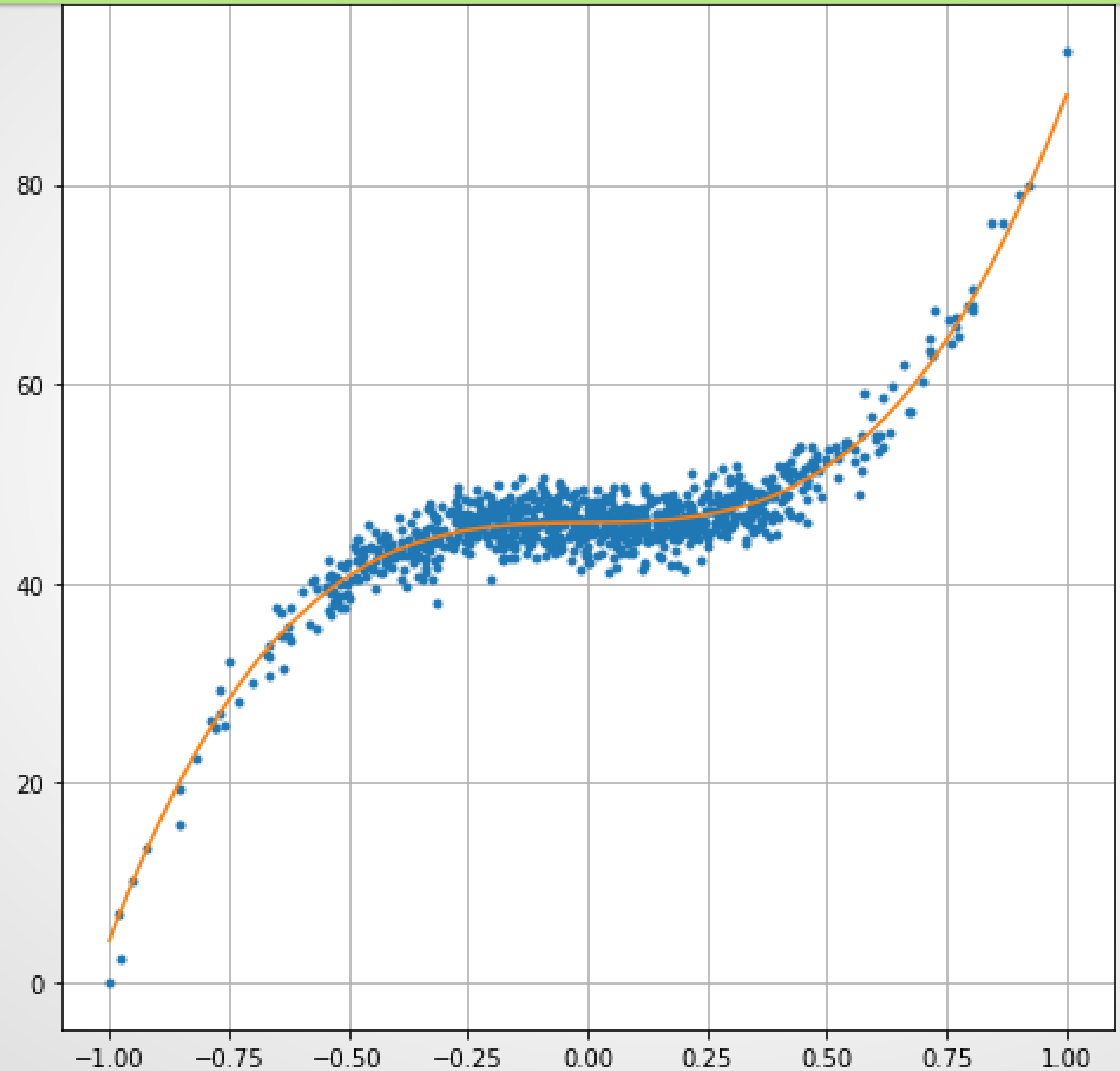
m примеров, размера **n**, **n+1** параметр θ

исходные данные **X** = $\begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \dots & x_n^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \dots & x_n^{(2)} \\ \vdots & \vdots & & \vdots \\ x_1^{(m)} & x_2^{(m)} & \dots & x_n^{(m)} \end{bmatrix}$

строим полином степени **k** на переменных **x**,

комбинируем столбцы матрицы **X**
число параметров **θ** увеличивается

Регрессия: нелинейная 3



Регрессия: почти последний слайд...

Вопросы ?

Регрессия: литература

Борисов Е.С. Модели математической регрессии

http://mechanoid.kiev.ua/ml-regression.html

Регрессия: практика

Numpy

matplotlib

sklearn.datasets

sklearn.preprocessing.**PolynomialFeatures**

sklearn.preprocessing.**MinMaxScaler**

векторизация вычислений!

for i in r : X[i] *Y[i]

X*Y X.dot(Y)