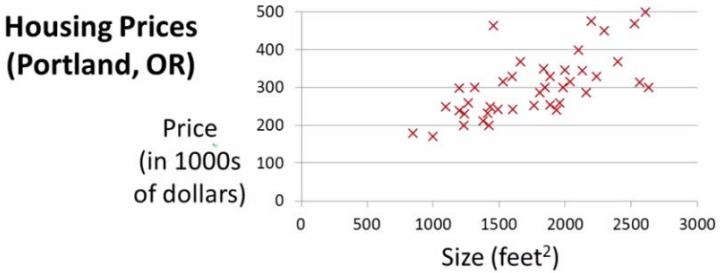
Постановка задачи



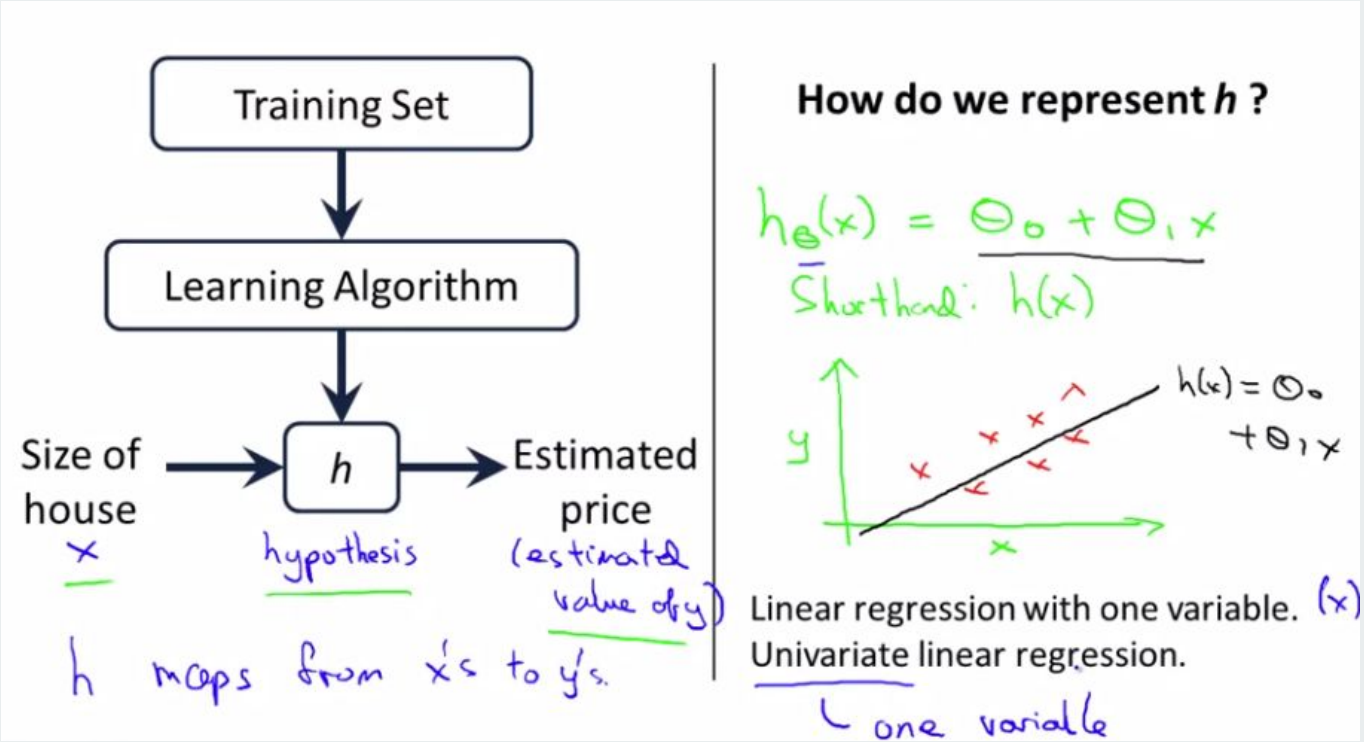
Training set of housing prices (Portland, OR)

Size in feet² (x)	Price ($) in 1000's (y)
2104	460
1416	232
1534	315
852	178
…	…

Notation:

- **m** = Number of training examples
- **x**'s = "input" variable / features
- **y**'s = "output" variable / "target" variable

Гипотеза/модель



Функция потерь (Loss Function, Cost Function, Error Function; J)

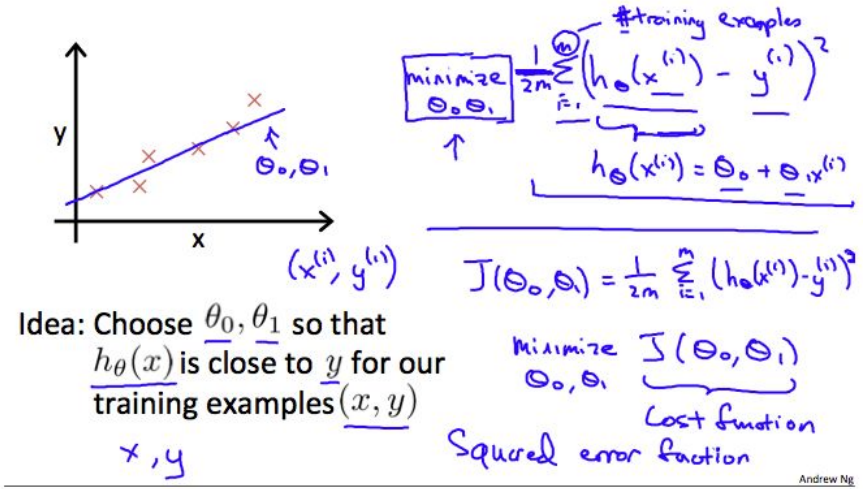
Cost Function

We can measure the accuracy of our hypothesis function by using a **cost function**. This takes an average difference (actually a fancier version of an average) of all the results of the hypothesis with inputs from x's and the actual output y's.

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (\hat{y}_i - y_i)^2 = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)^2$$

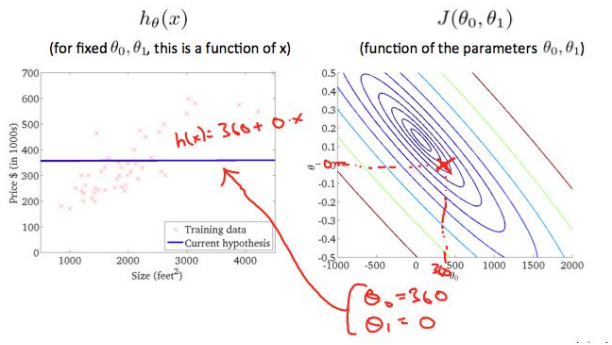
To break it apart, it is $\frac{1}{2}\ \bar{x}$ where $\bar{x}$ is the mean of the squares of $h_\theta(x_i) - y_i$, or the difference between the predicted value and the actual value.

This function is otherwise called the "Squared error function", or "Mean squared error". The mean is halved $\left(\frac{1}{2}\right)$ as a convenience for the computation of the gradient descent, as the derivative term of the square function will cancel out the $\frac{1}{2}$ term. The following image summarizes what the cost function does:

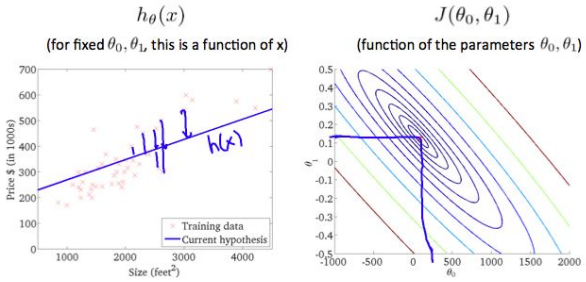


Пример 1

Taking any color and going along the 'circle', one would expect to get the same value of the cost function. For example, the three green points found on the green line above have the same value for $J(\theta_0, \theta_1)$ and as a result, they are found along the same line. The circled x displays the value of the cost function for the graph on the left when θ_0 = 800 and θ_1 = -0.15. Taking another h(x) and plotting its contour plot, one gets the following graphs:



When θ_0 = 360 and θ_1 = 0, the value of $J(\theta_0, \theta_1)$ in the contour plot gets closer to the center thus reducing the cost function error. Now giving our hypothesis function a slightly positive slope results in a better fit of the data.



The graph above minimizes the cost function as much as possible and consequently, the result of θ_1 and θ_0 tend to be around 0.12 and 250 respectively. Plotting those values on our graph to the right seems to put our point in the center of the inner most 'circle'.

MSE, RMSE, MLE (x, vt)

MSE - Mean squared error
MLE - Maximum likelihood estimation

1.1. Linear Models

The following are a set of methods intended for regression in which the target value is expected to be a linear combination of the features. In mathematical notation, if $\hat{y}$ is the predicted value.

$$\hat{y}(w, x) = w_0 + w_1 x_1 + \ldots + w_p x_p$$

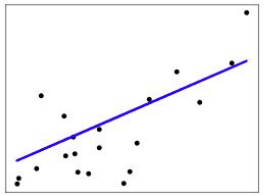
Across the module, we designate the vector $w = (w_1, \ldots, w_p)$ as `coef_` and w_0 as `intercept_`.

To perform classification with generalized linear models, see Logistic regression.

1.1.1. Ordinary Least Squares

`LinearRegression` fits a linear model with coefficients $w = (w_1, \ldots, w_p)$ to minimize the residual sum of squares between the observed targets in the dataset, and the targets predicted by the linear approximation. Mathematically it solves a problem of the form:

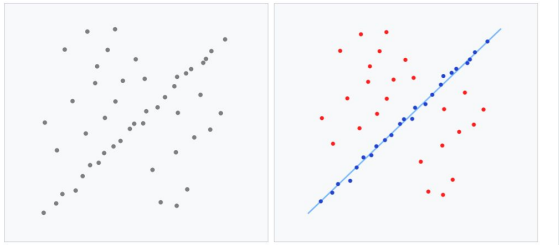
$$\min_w ||Xw - y||_2^2$$



`LinearRegression` will take in its `fit` method arrays X, y and will store the coefficients w of the linear model in its `coef_` member:

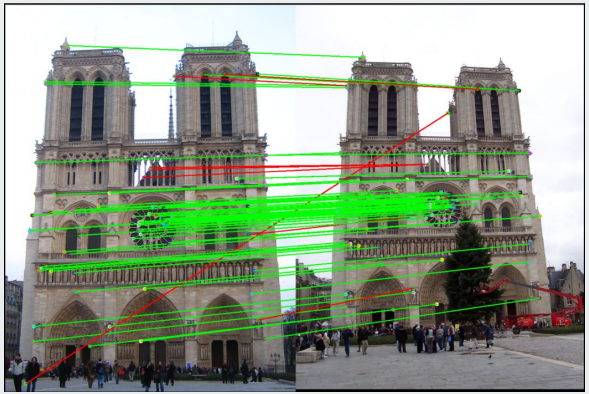
```
>>> from sklearn import linear_model
>>> reg = linear_model.LinearRegression()
>>> reg.fit([[0, 0], [1, 1], [2, 2]], [0, 1, 2])
LinearRegression()
>>> reg.coef_
array([0.5, 0.5])
```

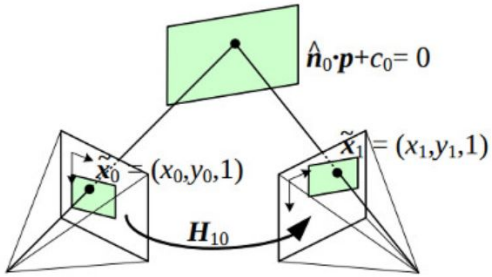
Пример 2 (LM - levenberg–marquardt)

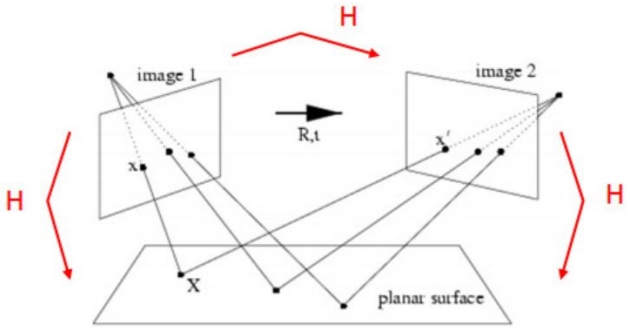


Набор данных, в который надо вписать прямую. выбросы присутствуют в большом количестве.

Предложенная алгоритмом RANSAC прямая. выбросы не влияют на результат.

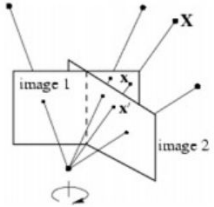


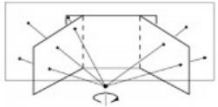




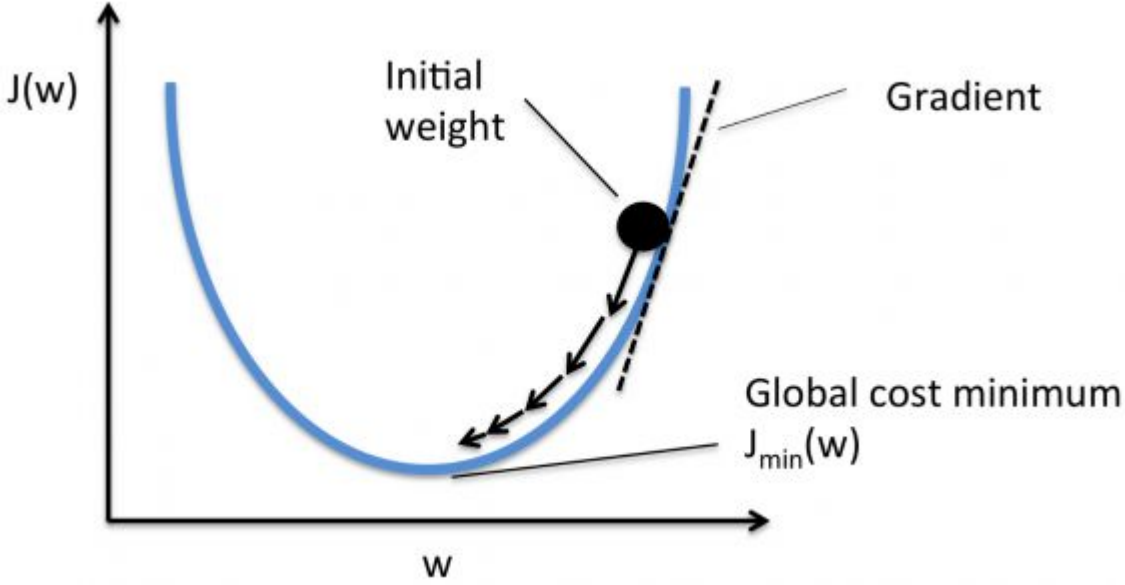
- a rotating camera around its axis of projection, equivalent to consider that the points are on a plane at infinity (image taken from 2)

Rotating camera, arbitrary world





Градиентный спуск

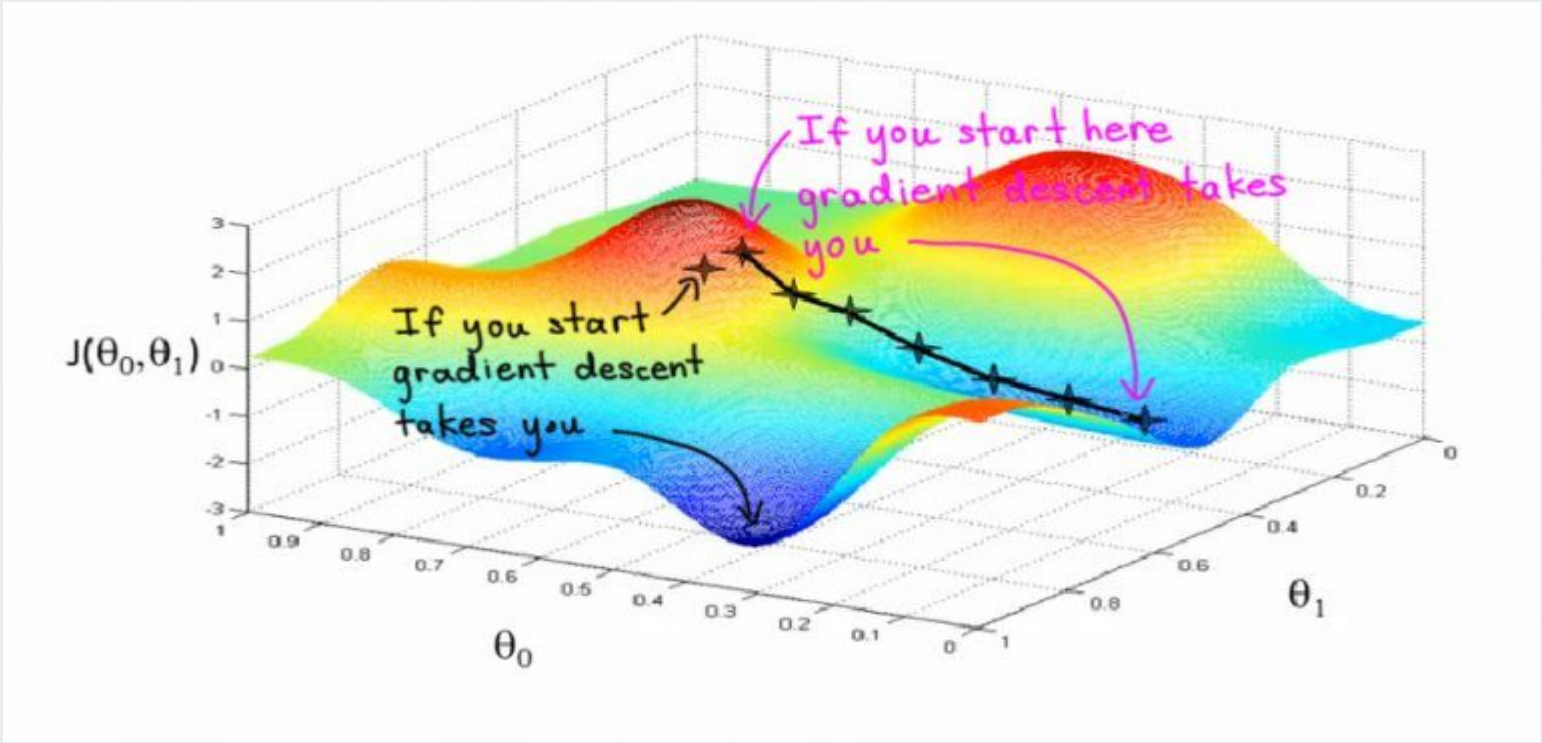


$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

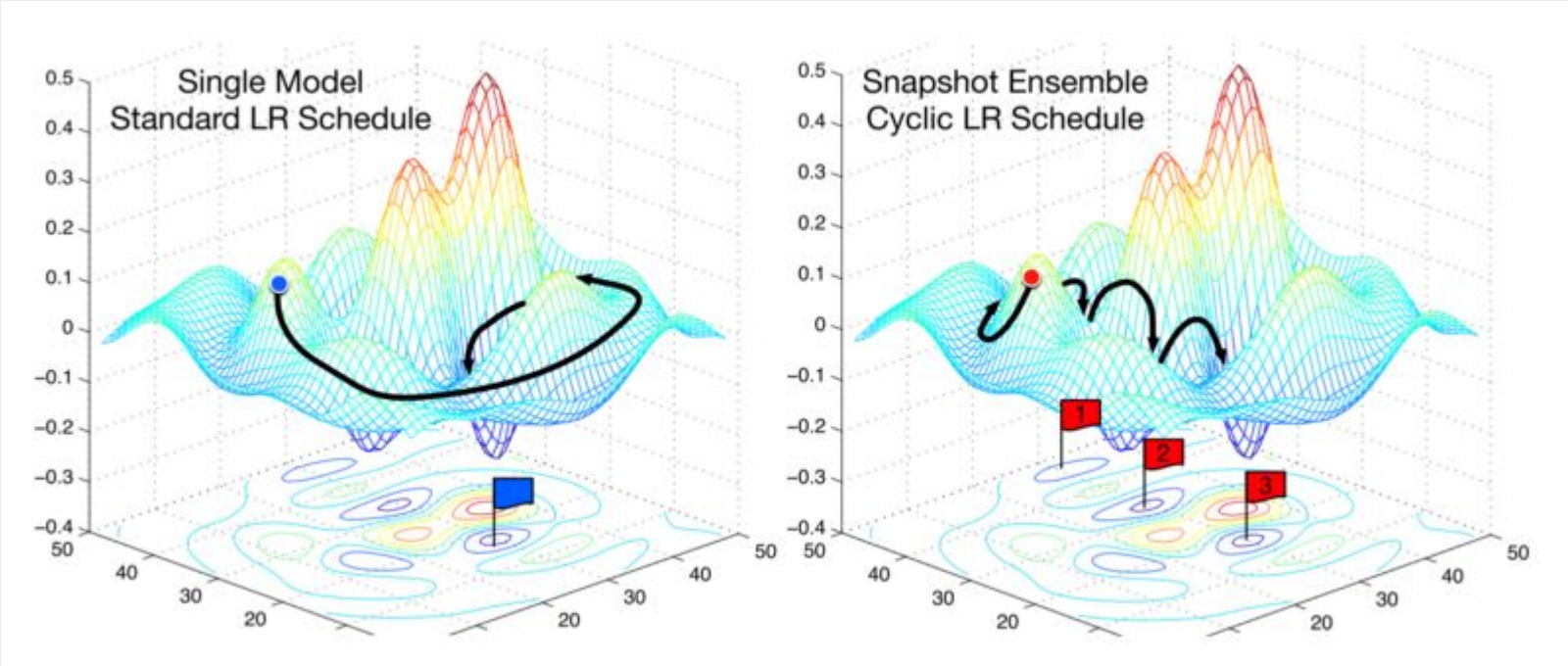
$$\theta_0 := \theta_0 - \frac{1}{m} \alpha \sum_{i=1}^{m} (h_o x_i - y_i)$$

$$\theta_1 := \theta_1 - \frac{1}{m} \alpha \sum_{i=1}^{m} (h_o x_i - y_i)$$

Local vs global min



Local vs global min



Пример 3 (MVG)

Refs:

- https://www.coursera.org/learn/machine-learning/lecture/db3jS/model-representation
- https://scikit-learn.org/stable/modules/linear_model.html
- https://id-lab.ru/posts/developers/funkcii/
- https://blog.paperspace.com/intro-to-optimization-in-deep-learning-gradient-descent/
- https://docs.opencv.org/4.x/d9/dab/tutorial_homography.html
- https://github.com/openMVG/openMVG
- https://habr.com/ru/post/419757/
- https://medium.com/data-breach/introduction-to-feature-detection-and-matching-65e27179885d