Linear Regression

What is Linear Regression and how it works ?

Data science and Machine learning Online Course

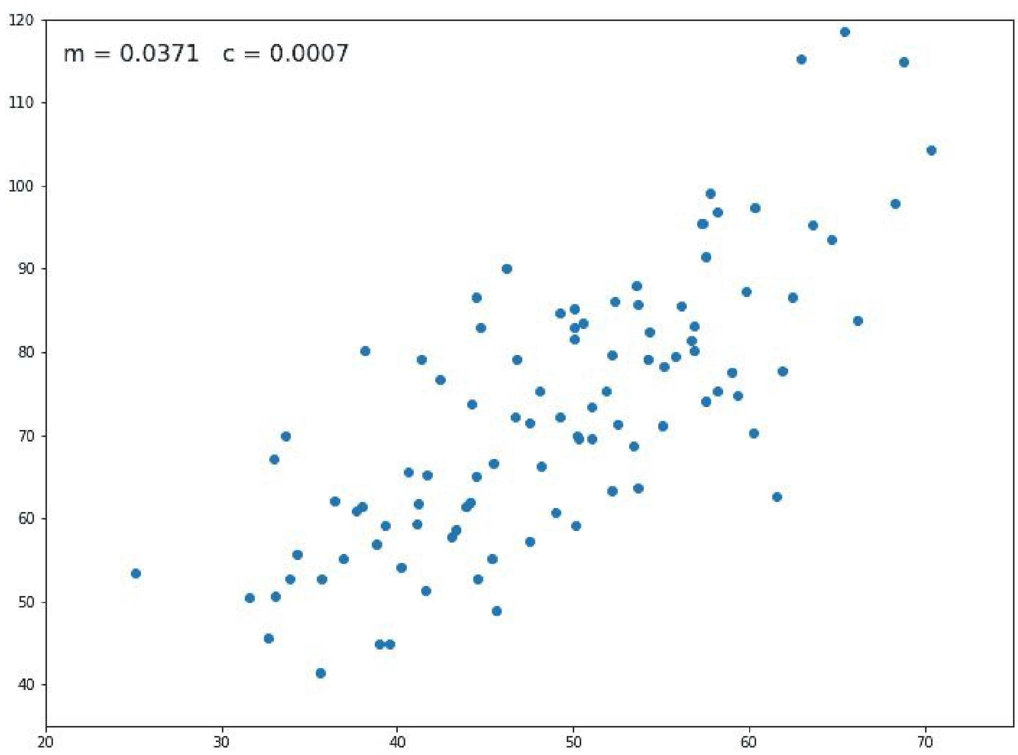
What is Linear Regression ?

Linear Regression is a method used to define a relationship between a dependent variable (Y) and independent variable (X).

Where y is the dependent variable, m is the scale factor or coefficient, b being the bias coefficient and X being the independent variable.

$$y = mx + b$$

The goal is to draw the line of best fit between X and Y which estimates the relationship between X and Y.



Model Representation:

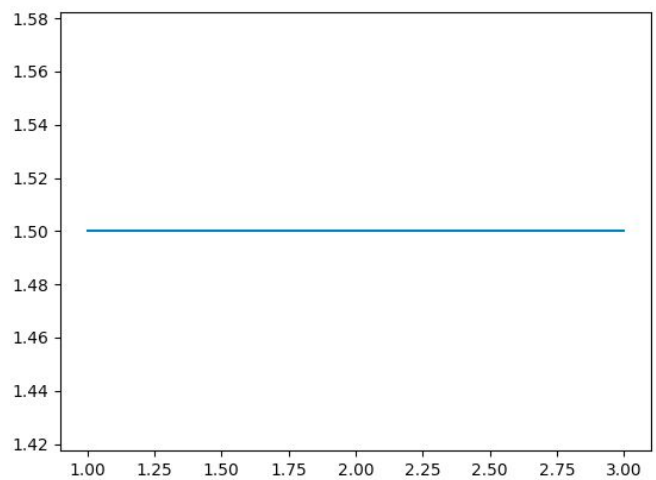
First, the goal of most machine learning algorithms is to construct a model.
a hypothesis that can be used to estimate Y based on X. The model maps inputs to outputs.

$$h_\theta(x) = \theta_0 + \theta_1 x$$

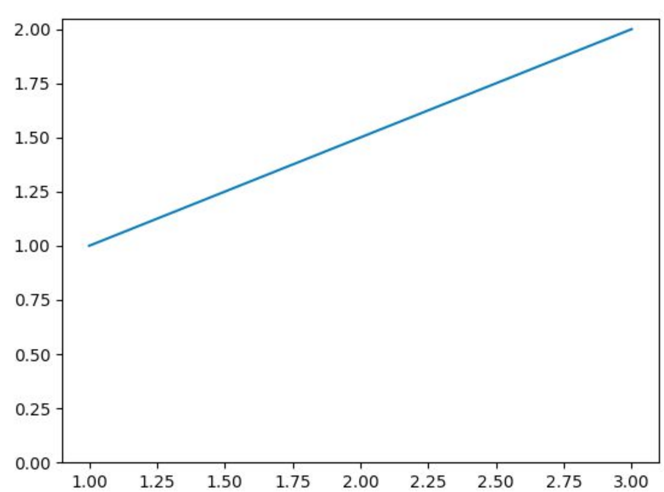
The **THETA** values are the parameters.

So let's plot some lines with our model:

$$\theta_0 = 1.5$$
$$\theta_1 = 0$$

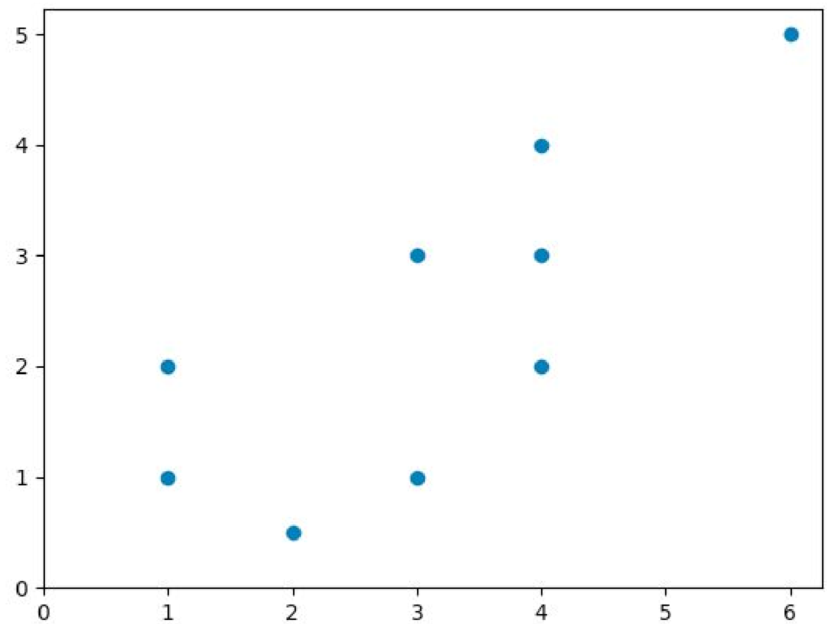


$$\theta_0 = 1$$
$$\theta_1 = 0.5$$



- The goal of creating a model is to choose parameters, or theta values, so that h(x) is close to y for the training data, x and y.

Imagine we have these data and we plot them on figure.



We'll try to find the best line fit using linear regression.

What is Cost Function ?

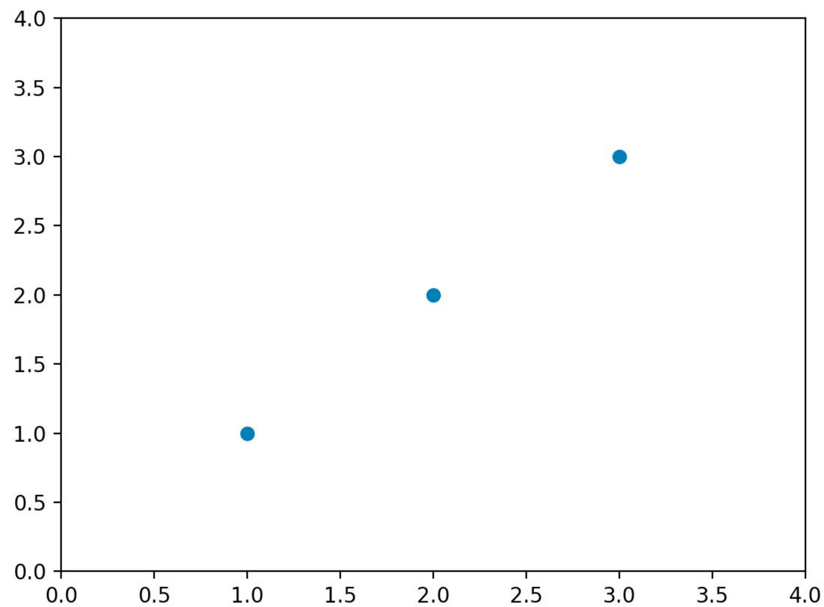
We need a function that will minimize the parameters over our dataset. One common function that is often used is mean squared error **(MSE)**, which measure the difference between the estimator (**the dataset**) and the estimated value (**the prediction**)

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2.$$

We can change it a little to find better equation and more understandable.

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

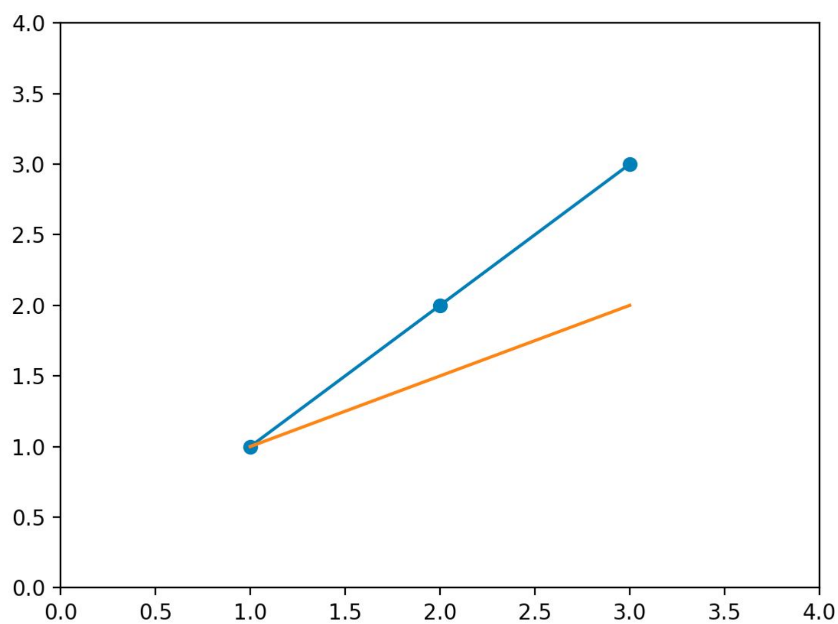
Let's apply our MSE to simple data.

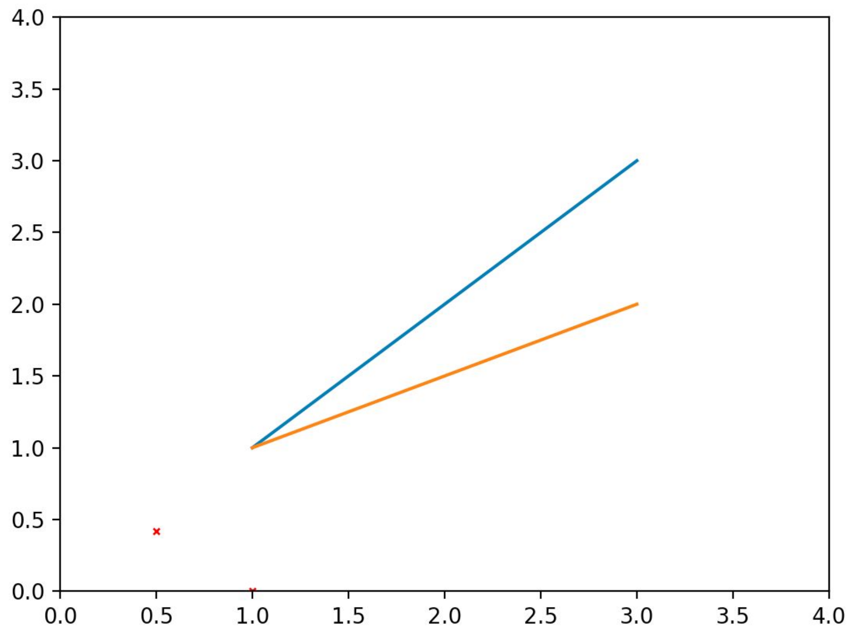


For now we will calculate some theta values, and plot the cost function by hand. Since this function passes through (0, 0), we are only looking at a **single value of theta**.

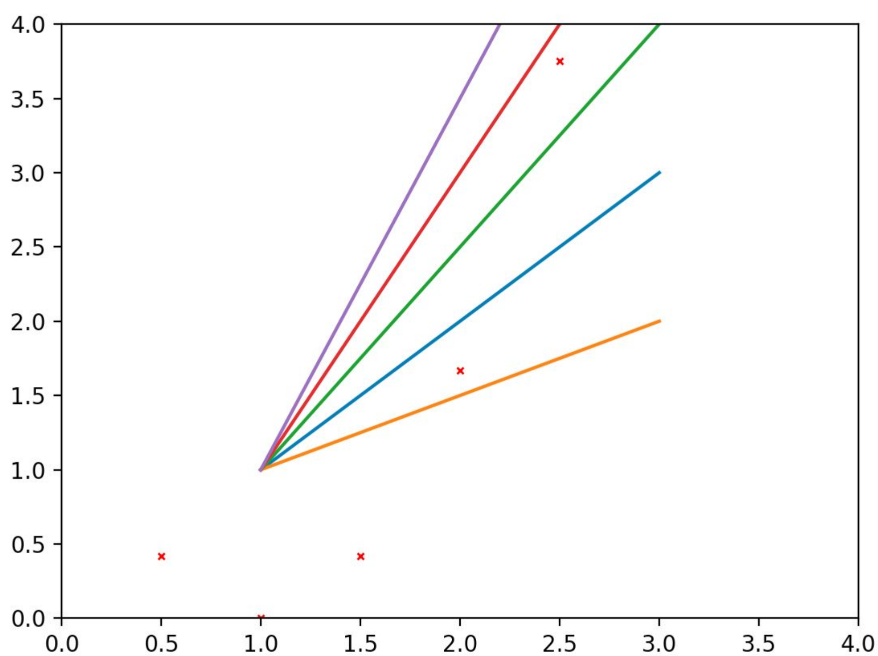
The MSE function gives us a value of 0.58

J(1) = 0
J(0.5) = 0.58

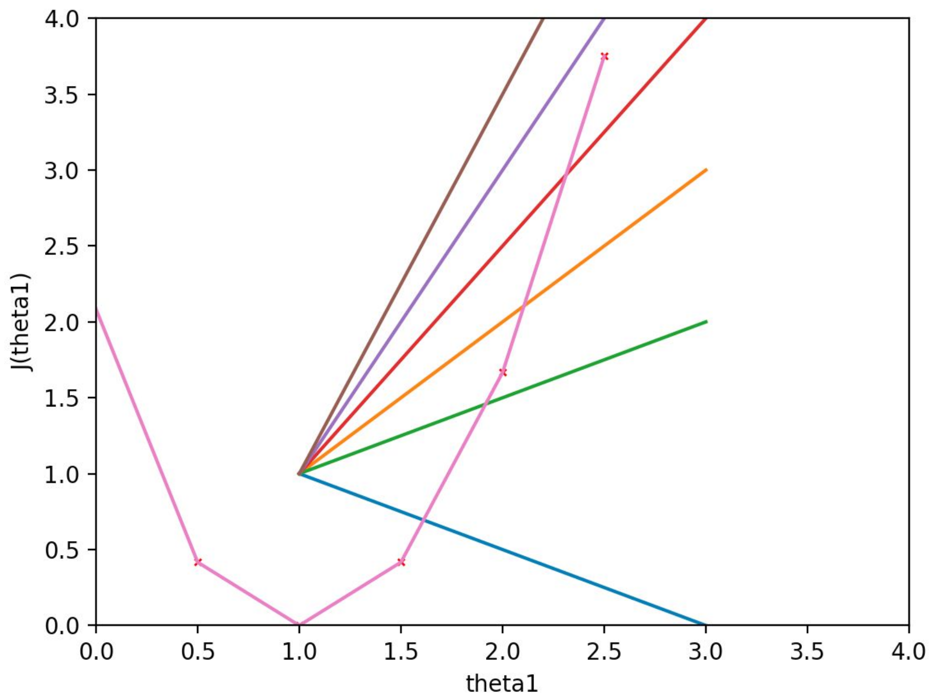




We’ll go ahead and calculate some more values of J(ϴ)



And if we join the dots together nicely.



We can see that the cost function is at a minimum when theta = 1

What is Gradient Descent ?

We minimized J(Θ) by trial and error above — just trying lots of values and visually inspecting the resulting graph.

Queue *gradient descent.* Gradient Descent is a general function for minimizing a function, in this case the Mean Squared Error cost function.

- Gradient Descent basically just does what we were doing by hand — change the theta values, or parameters, bit by bit, until we *hopefully* arrived at a minimum.

We start by initializing theta0 and theta1 to any two values, say **0** for both, and go from there.

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \qquad (\text{for } j = 0 \text{ and } j = 1)$$

where α, alpha, is the learning rate, or how quickly we want to move towards the minimum. If α is too large, however, we can overshoot.