


## Mean Squared Error

$$mse = \frac{1}{n} \sum_{i=1}^n (y_i - y_{predicted})^2$$

## Mean Squared Error

$$mse = \frac{1}{n} \sum_{i=1}^n (y_i - (mx_i + b))^2$$

 Cost Function

**Gradient descent** is an algorithm that finds best fit line for given training data set

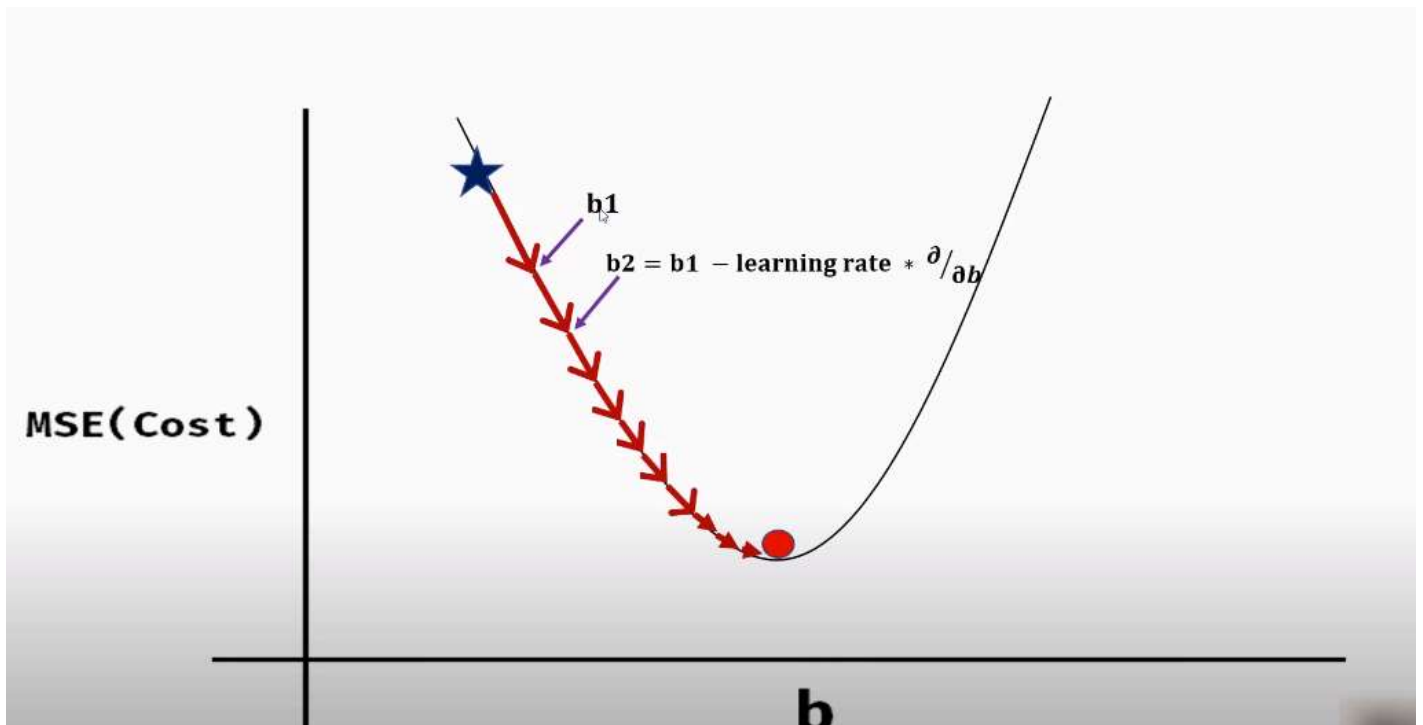
$$mse = \frac{1}{n} \sum_{i=1}^n (y_i - (mx_i + b))^2$$

$$\frac{\partial}{\partial m} = \frac{2}{n} \sum_{i=1}^n -x_i (y_i - (mx_i + b))$$

$$\frac{\partial}{\partial b} = \frac{2}{n} \sum_{i=1}^n -(y_i - (mx_i + b))$$

$$m = m - \text{learning rate} * \frac{\partial}{\partial m}$$

$$b = b - \text{learning rate} * \frac{\partial}{\partial b}$$



```
import numpy as np
def gradient_descent(x,y):
    m_curr=b_curr=0
    iterations=1000
    n=len(x)
    learning_rate=0.001
    for i in range(iterations):
        y_predicted=m_curr*x+b_curr
        md=-(2/n)*sum(x*(y)-y_predicted)
        bd=-(2/n)*sum(y-y_predicted)
        m_curr=m_curr-learning_rate*md
        b_curr=b_curr-learning_rate*bd
```

```
x=np.array([1,2,3,4,5])
y=np.array([5,7,9,11,13])
```

```
gradient_descent(x,y)
```

```
⇒ m 0.062,b 0.018000000000000002,iteration 0
m 0.123592000000000001,b 0.035592000000000006,iteration 1
m 0.184779264,b 0.052779264000000006,iteration 2
m 0.245565029888,b 0.06956502988800001,iteration 3
m 0.305952509648896,b 0.085952509648896,iteration 4
m 0.3659448895717048,b 0.10194488957170483,iteration 5
m 0.42554533045513115,b 0.11754533045513119,iteration 6
m 0.48475696781149014,b 0.13275696781149013,iteration 7
m 0.5435829120689982,b 0.1475829120689982,iteration 8
m 0.6020262487724463,b 0.16202624877244623,iteration 9
m 0.6600900387822667,b 0.17609003878226667,iteration 10
m 0.7177773184720085,b 0.18977731847200854,iteration 11
m 0.7750910999242324,b 0.20309109992423247,iteration 12
m 0.8320343711248386,b 0.21603437112483861,iteration 13
m 0.8886100961558399,b 0.2286100961558399,iteration 14
m 0.9448212153865931,b 0.24082121538659318,iteration 15
m 1.0006706456635004,b 0.25267064566350045,iteration 16
m 1.0561612804981924,b 0.26416128049819243,iteration 17
m 1.1112959902542068,b 0.27529599025420687,iteration 18
m 1.1660776223321732,b 0.2860776223321732,iteration 19
m 1.2205090013535158,b 0.2965090013535158,iteration 20
m 1.2745929293426876,b 0.30659292934268767,iteration 21
m 1.3283321859079462,b 0.31633218590794615,iteration 22
m 1.3817295284206828,b 0.3257295284206826,iteration 23
m 1.4347876921933174,b 0.3347876921933171,iteration 24
m 1.4875093906557708,b 0.3435093906557706,iteration 25
m 1.5398973155305247,b 0.3518973155305244,iteration 26
m 1.5919541370062804,b 0.3599541370062802,iteration 27
m 1.64368250391023,b 0.36768250391022994,iteration 28
m 1.6950850438789482,b 0.3750850438789481,iteration 29
m 1.7461643635279167,b 0.3821643635279165,iteration 30
m 1.7969230486196934,b 0.3889230486196932,iteration 31
m 1.8473636642307358,b 0.39536366423073566,iteration 32
m 1.89748875491689,b 0.4014887549168898,iteration 33
```

```
m 1.9473008448775548,b 0.4073008448775547,iteration 34
m 1.9968024381185343,b 0.41280243811853423,iteration 35
m 2.045996018613586,b 0.417996018613586,iteration 36
m 2.0948840504646773,b 0.4228840504646773,iteration 37
m 2.1434689780609597,b 0.4274689780609599,iteration 38
m 2.191753226236472,b 0.4317532262364722,iteration 39
m 2.2397392004265804,b 0.43573920042658043,iteration 40
m 2.287429286823168,b 0.4394292868231678,iteration 41
m 2.3348258525285823,b 0.4428258525285824,iteration 42
m 2.3819312457083535,b 0.44593124570835374,iteration 43
m 2.428747795742687,b 0.4487477957426869,iteration 44
m 2.4752778133767452,b 0.45127781337674544,iteration 45
```