ALGORITHMS: → one input feature & One output feature

SIMPLE LINEAR REGRESSION: (It is a Supervised ML)

As we know the Supervised ML classified into two types of problems

1) Regression → o/p feature is Continous in nature

2) Classification → o/p feature is Binary / Multi-class categories. ~~classification~~

Let's assume a simple dataset with two features (one dependent & one independent)

Weight (Independent feature)	height (Dependent feature)
74	170
80	180
90	190
75	175.5
⋮	⋮

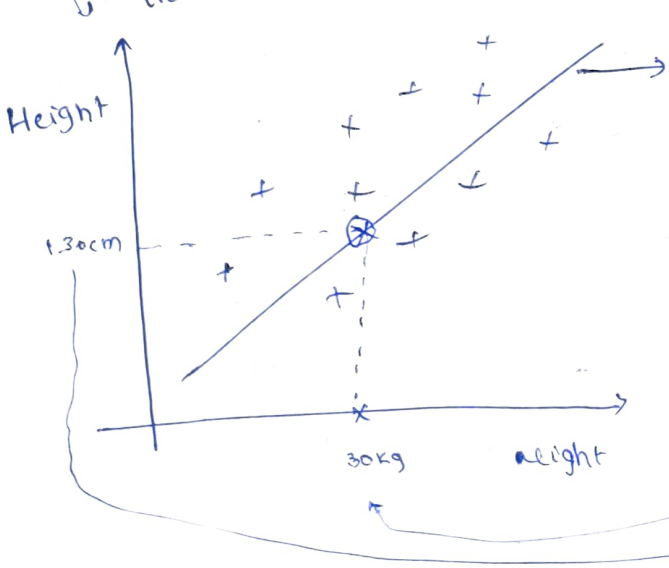
If our client want's to predict height based on weight.

⇒ Here our model needs to find height based on weight so, Weight is Independent feature (input feature), height is dependent feature (o/p or target feature)

1) Initially we train our model with the data & we try to predict the height using weights using test data. (Test data only have weights data, based on weight's our model will predict height).

GEOMETRICAL INTUITION BEHIND SIMPLE LINEAR REGRESSION: (It will tell you how our model will Predict new height values based on weight data)

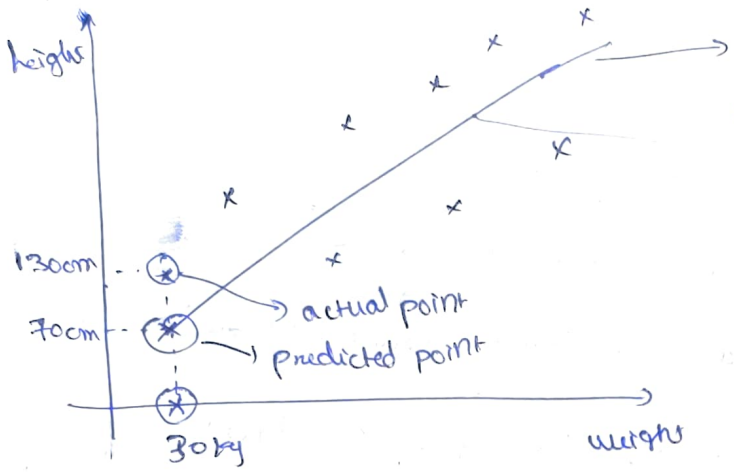
Plotting the data ↓



Linear reg^n find
Our Model will ~~create~~ best fit line based on training data

Then based on best fit line we will find the height values for new weight values like

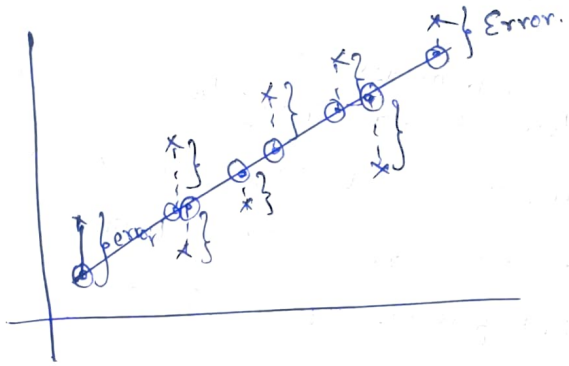
Our Model predicts height = 130cm for input weight of 30kg



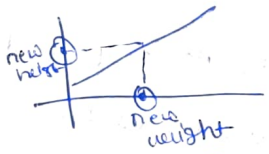
→ after traing the ML model, it creats a Best fit line

→ points on best fit line are classified as predicted points

Best fit line is created in such a way that the distance (sum of error) b/w actual data point and predicted data point is minimum.

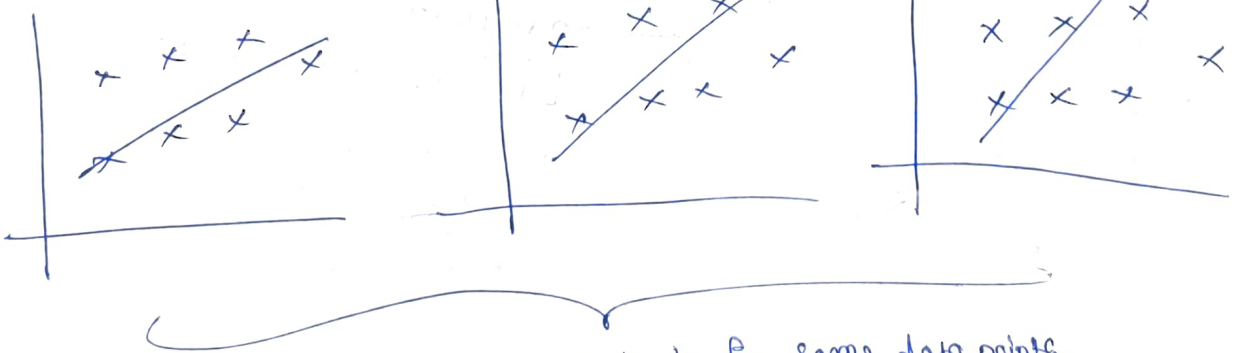


purpose of best fit line → Used to predict height values based on new weight values



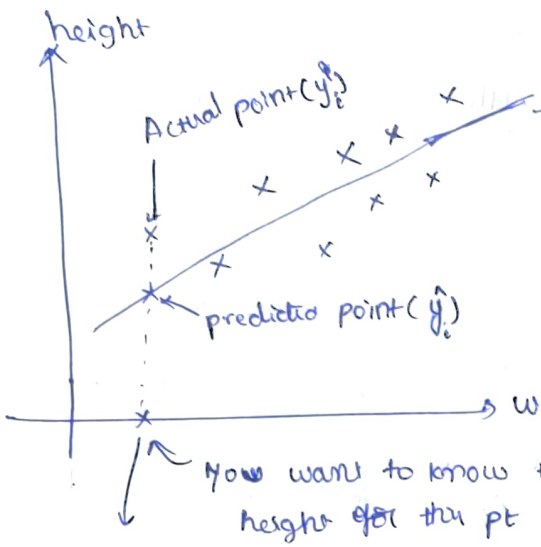
How exactly the best fit line will be created ? → will see later.

To be More precise



3 Best fit line's for same data points,

whatever best fit line has less sum of error b/w actual & predicted value it will be considered.



The equation to represent straight line is

$y = mx + c$ or $y = wx + b$

slope — data points — intercept

or

$y = \beta_0 + \beta_1 x$

All are same

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

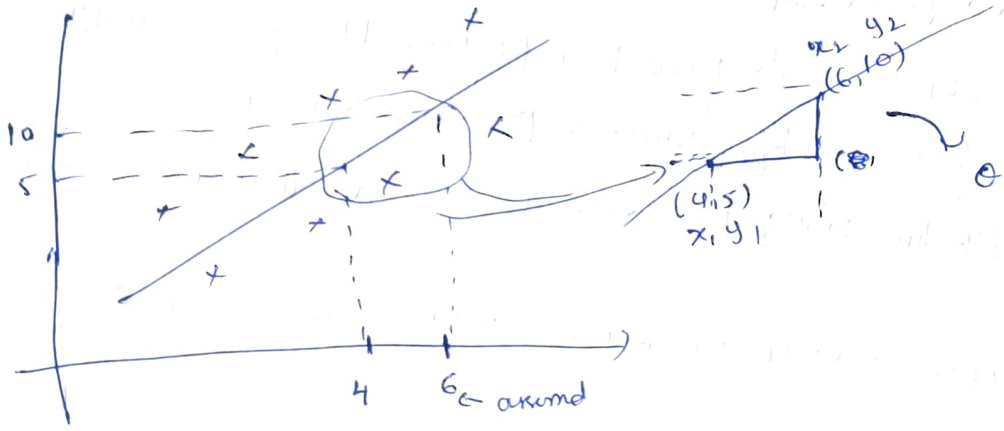
Researches are using this equation.

θ_0 => intercept

θ_1 => slope or co-efficient

Error => $(y_i - \hat{y}_i)$ — actual, predicted

Slope or co-efficient :- with the movement on x-axis how ~~feature changing~~ much movement we have in y-axis?

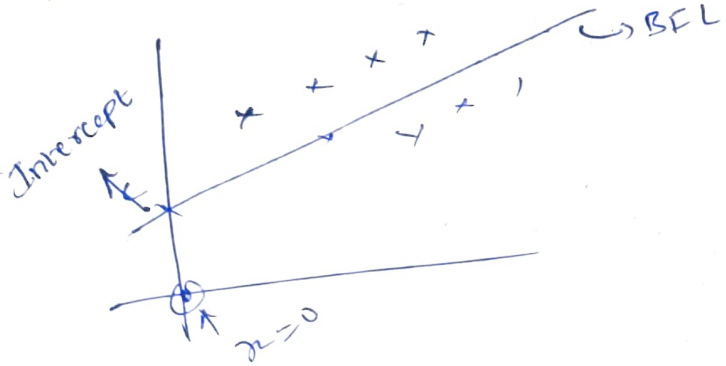


$\theta_1 = \frac{y_2 - y_1}{x_2 - x_1}$

$= \frac{10 - 5}{6 - 4}$

$= \frac{5}{2} = 2.5$

Intercept :- if our x-axis value = 0, then the value of y is represented as intercept

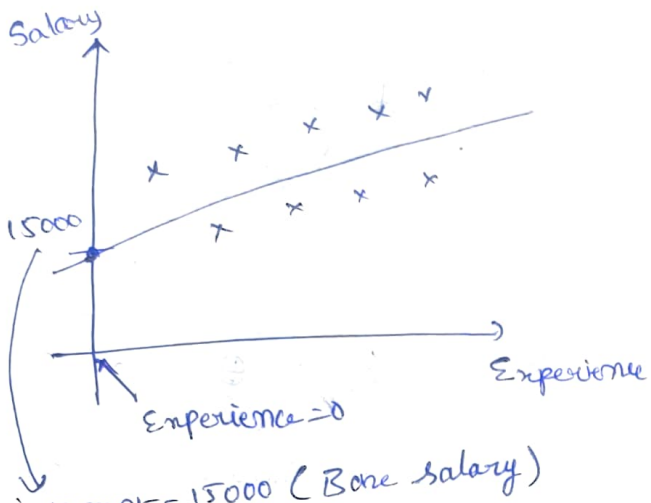


$y = mx + c$

$y = m(0) + c$

$y = c$

Small ex: When we plot the graph experience vs salary. eventhough we joined as a fresher we have some basic salary right



intercept = 15000 (Base salary)

* ~~$h = \theta_0 +$~~ $h_\theta(x) = \theta_0 + \theta_1 x$ ← data points

So, Based on θ_0 & θ_1 we will fit & create the best fit line.

Initially, we initiate random values for θ_0 & θ_1 & we will create a best fit line & based on best fit line we will calculate the ~~error~~ error b/w actual & predicted point. If the error is high then again we change the value of θ_0 & θ_1 to create new best fit line. This process will repeat ~~till~~ till we get less error b/w actual & predicted data point.

Whatever best fit line giving less error, we consider that line as our best fit line.

(THE WHOLE PROCESS WILL BE EXPLAINED IN DETAIL LATER)

(This process is called optimization)

So, In order to derive the optimization we need to derive the

cost function:

$$J(\theta_0, \theta_1) = \frac{1}{n} \sum_{i=1}^{n} (y_i - h_\theta(x))^2$$ ⇒ Mean Squared error

n ↓ no. of data points.

y_i ↓ actual, $h_\theta(x)$ ↓ predicted

Σ $h_\theta(x) - y_i$ ✓

Eqn tells you that, Our Aim is to find the best fit line in such a way that our mean squared error should be minimized.

Sum of Squares of Actual - predicted values.

n = no. of data points

y_i = Actual value

$h_\theta(x)$ = predicted value.

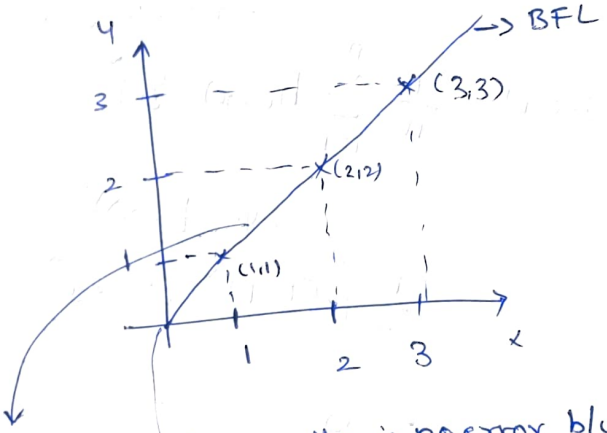
Final Aim : → To get Best fit line Minimize $J(\theta_0, \theta_1) = \frac{1}{n} \sum_{i=1}^{n} (y_i - h_\theta(x))^2$

$$\text{Minimize } J(w,b) = \frac{1}{n} \sum_{i=1}^{m} (y_i - \hat{y}_i)^2$$

Optimization : (How do we Minimize the cost function)

Dataset :

x → independent	y → dependent
1	1
2	2
3	3



As we can observe ther is no error b/w actual & $\hat{y}_i$ beccause thre the BFL is passed exactly through all points

$h_\theta(x) = \theta_0 + \theta_1 x$ ⇄ $y = mx + c$

Here our intercept is zero. i.e $\theta_0 = 0$ ⇄ $c = 0$

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

If we assume $\theta_1 = 1$

at $x = 1 \Rightarrow h_\theta(x) = 0 + 1(1)$

$h_\theta(x) = 1$

at $x = 2 \Rightarrow h_\theta(x) = 0 + 1(2)$

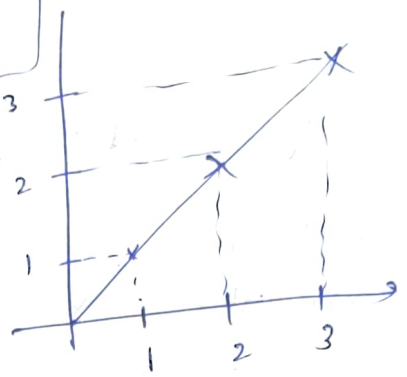
$h_\theta(x) = 2$

at $x = 3 \Rightarrow h_\theta(x) = 0 + 1(3)$

$= 3$

x	y	$h_\theta(x)$
1	1	1
2	2	2
3	3	3

$\Rightarrow$ If we plot



After creating BFL, we need to find cost function.

cost fn: ~~$\theta_0 = 0$~~ $\rightarrow$ Initial. $\boxed{\theta_0 = 0} \Rightarrow \boxed{h_\theta(x) = \theta_1 x_i}$

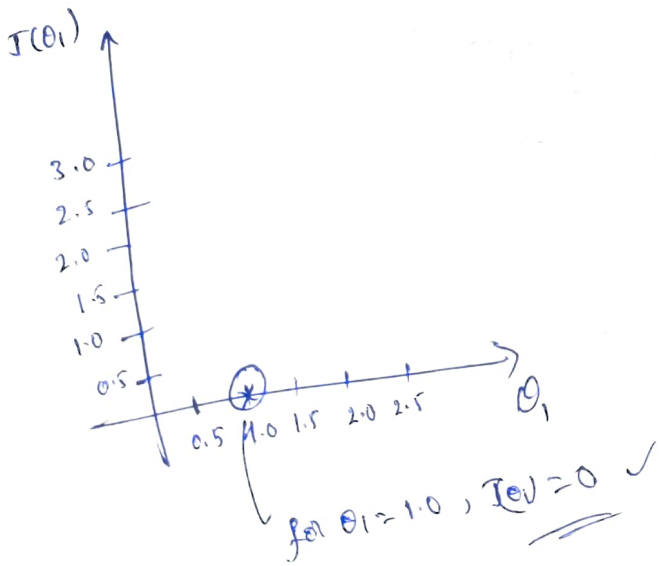
$$J(\theta_1) = \frac{1}{n} \sum_{i=1}^{n} \underbrace{(h_\theta(x) - y_i)^2}_{\text{error b/w actual \& predicted}}$$

$$(y_i - h_\theta(x))^2$$

$$= \frac{1}{3} \left([1-1]^2 + [2-2]^2 + [3-3]^2 \right)$$

$$= 0$$

Creating another graph that plotting $J(\theta_1)$ vs θ_1



* Now assume $\theta_1 = 0.5$

at $x=1$

$h_\theta(x) = 0 + 0.5(1)$

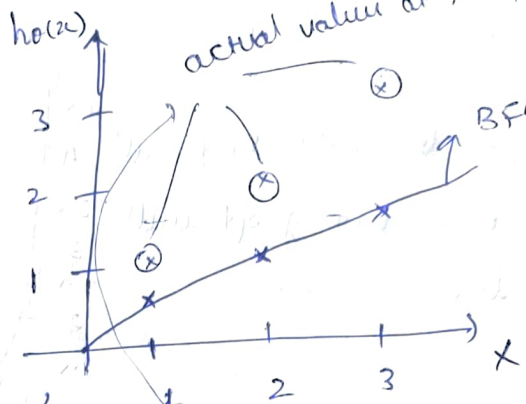
$= 0.5$

at $x=2$ $h_\theta(x) = 0 + 0.5(2)$

$= 1$

at $x=3$ $h_\theta(x) = 0 + 0.5(3)$

$= 1.5$



Now the BFL is changed

x	y	$h_\theta(x)$ ← predicted val
1	1	0.5
2	2	1
3	3	1.5

↓ actual ↓ predicted.

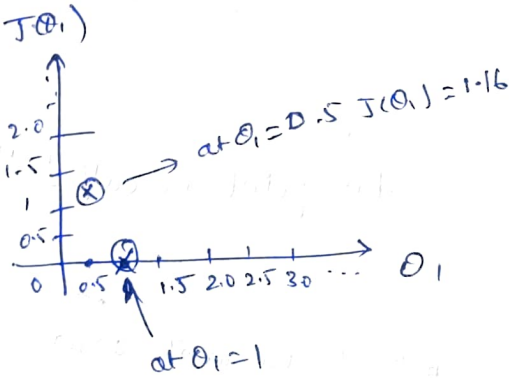
Now cost fn :

$$\Rightarrow \frac{1}{n} \sum_{i=1}^{n} (y_i - h_\theta(x))^2$$

$$= \frac{1}{3} \left((1-0.5)^2 + (2-1)^2 + (3-1.5)^2 \right)$$

$$= \frac{1}{3} (0.25 + 1 + 2.25)$$

$J(\theta_1) = 1.16$ when $\theta_1 = 0.5$



~~cost fn~~ :

Now assume $\theta_1 = 0$

at $x=1$ $h_\theta(x) = 0 + 0(1)$

$= 0$

at $x=2$ $h_\theta(x) = 0 + 0(2)$

$= 0$

at $x=3$ $h_\theta(x) = 0$

=

x	y	$h_\theta(x)$
1	1	0
2	2	0
3	3	0

cost fn :

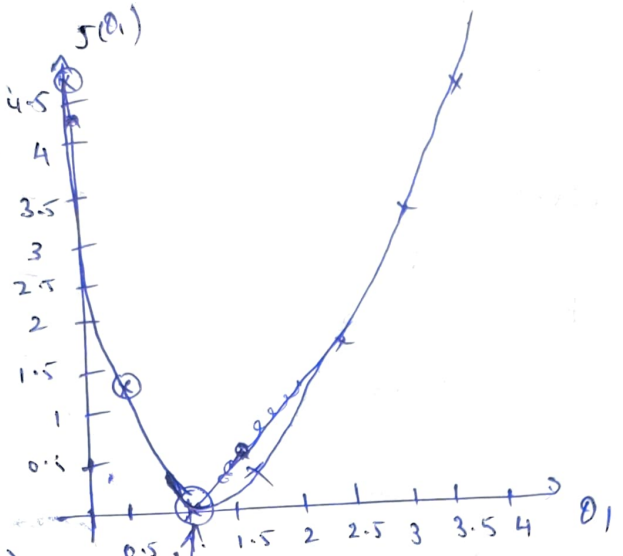
$$= \frac{1}{3} \left((1-0)^2 + (2-0)^2 + (3-0)^2 \right) = \frac{1}{3} (1+4+9)$$

$$= 14/3 \Rightarrow 4.66$$

=

at $\theta_1 = 0$, $J(\theta_1) = 4.66$ ..

if we plot $J(\theta_1)$ vs θ_1 for all values of θ_1 then graph will looks like. →



We will get a parabola like this. (U-shape)

GRADIENT DESCENT CURVE

At this point only our cost fn is zero i.e error is zero

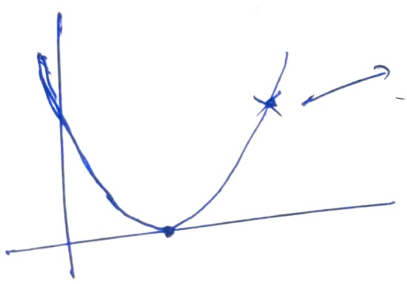
So, This is point where we got our best fit line for the training data, as error is less b/w actual & predict points.

This point is called "Global minima" (cost fn is less = error less)

so,

~~Initially~~ we won't assign the multiple θ_1 values but we will change (either increment or decrement the θ_1 values based on our requirement)

i.e



→ for ex; if you are here then to find the global minima then we need to ~~de~~increment the θ_1 value right so like that we need to develop our algorithm.

↓

it is CONVERGENCE ALGORITHM

Convergence algorithm: → Moving towards global minima

In the convergence algorithm, we optimize the change of θ_j

Algorithm:

Repeat until convergence

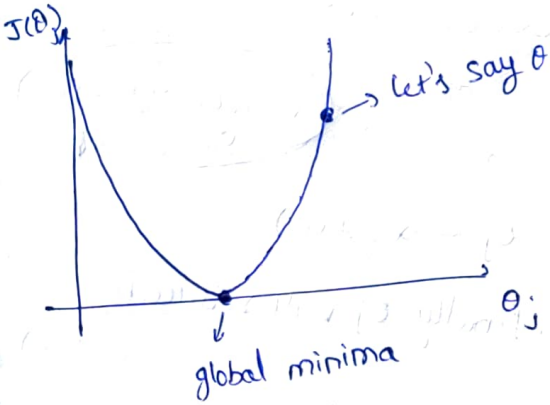
{ $j = 0, 1$

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} (J(\theta_j))$$

↑ Learning rate

}

Let's consider



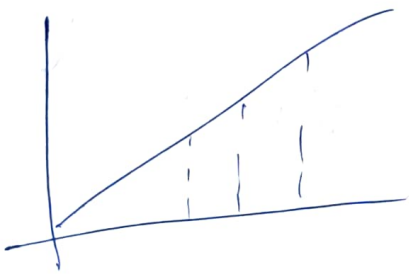
→ Let's say θ_1 is this point. } → So Now we need to move towards global minima.

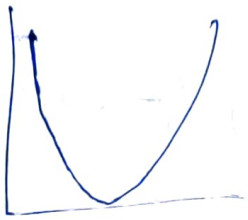
So, How we can move θ_j value towards global minima.

(By seeing graph we can say that we need to reduce the value of θ_j to reach global minima but for algorithm we need to find ~~derivative~~ slope at that particular point)

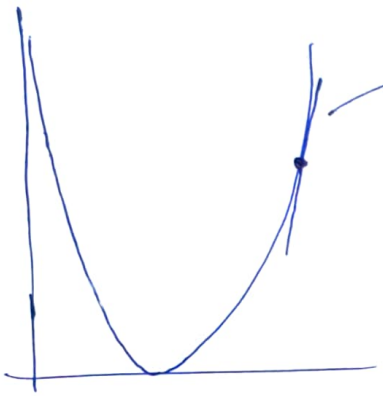
As the curve is parabola, we need to find ~~derivative~~ slope to decide whether we need to ↑ is or ↓ it's the θ_j value

Ex: for straight line slope is same at every point but for parabola for the slope will changes at each point.





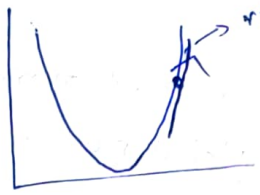
To find the slope value of parabola curve, we need to find derivative at that particular point



if you want to find derivative at particular point then we need to draw a tangent then we need to find whether it is +ve or -ve slope

$$\text{derivate} = \frac{\partial}{\partial \theta_j} J(\theta_j)$$

as a Notation, if our tangent right side of is showing up, then it has +ve slope



right side of tangent showing upwards. i,e it has +ve slope

α always +ve say $\alpha = 0.01$

So, $\theta_j = \theta_j - \alpha(+ve)$ (new, old)

So, finally θ_j will reduce by $\alpha(+ve)$

$\theta_{j\,new} << \theta_{j\,old}$ ✓

if the tangent right side is downwards then we get -ve slope

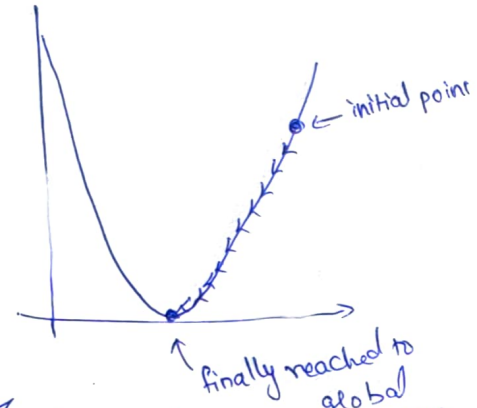
So, $\theta_j = \theta_j - \alpha(-ve)$ ⇒ final θ_j value will increase.

Finally,

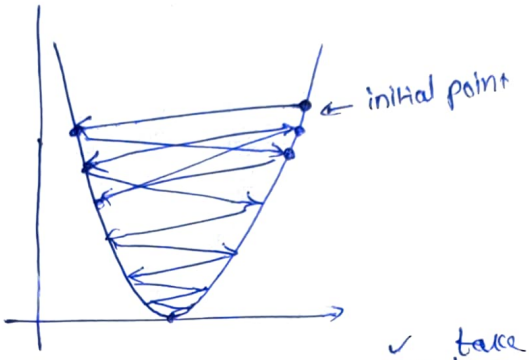
After reaching global minima, Slope = 0 because of horizontal line

Then $\theta_j = \theta_j - \alpha(0) \Rightarrow \theta_j$ (only θ_j will be there)

α → It will be very small, It is used to converge the θ_j value quickly, if it is large value then it will take so much of time to converge.



if α is less



if α is large then it take ~~will keep~~ ✓ So much of time to converge

Sometimes it won't reach global minima if α is very large

NOTE:

if α is less, converge ~~more~~ time is less

if α is very large, convergence takes very large amount of time.

Once we reach the global minima, then we consider the θ_1, θ_0 values at Global minim & we create our best fit line.