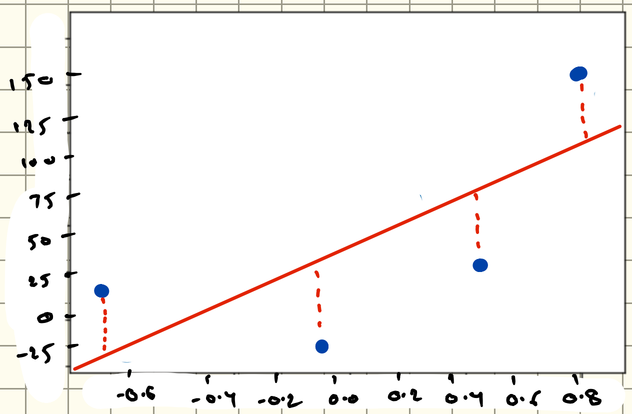
Gradient Descent

GD is an optimization algorithm. We give it any function, no matter what it is, as long as it is differentiable, and GD finds the minimum of that function.

⇒ Do we use GD only in LR ? Not at all. → it is used in many algorithm like Logistic R, and in deep learning, it is used everywhere, we can't even study deep L without GD.

Intuition



cgpa	lpa
-	-
-	-
-	-
-	-

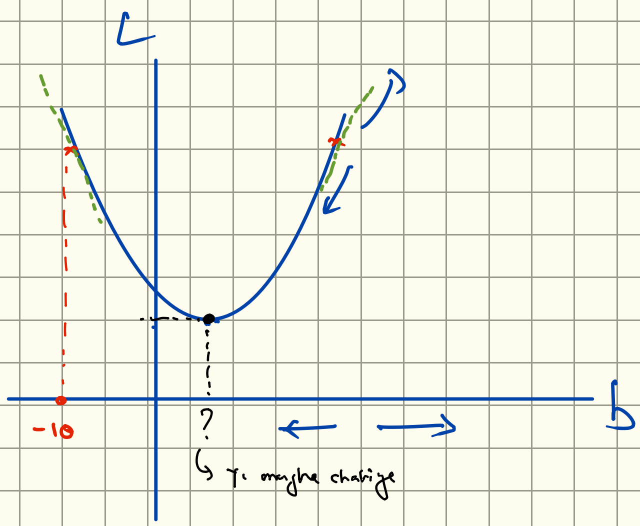
$$\boxed{\hat{y}_1 = mx_1 + b}$$

$$\boxed{m = 78.35}$$

Loss ⇒ $L = \sum_{i=1}^{4} (y_i - \hat{y}_i)^2$

$$\boxed{L = \sum_{i=1}^{4} (y_i - mx_i - b)^2}$$

$$\boxed{L = \sum_{i=1}^{4} (y_i - 78.35x_i - b)^2}$$



$\boxed{L = b^2}$ → hame aise b ki value chahiye jisse L minimum ho

hame aise b ki value chahiye jisse L minimum ho

⇒ In this graph, we have to find the value of b for which L is minimum. Now, how can we find it.?

↳ if we had used OLS, we could have directly taken the derivative, set it to 0, and solved the equation – just like we did in the previous class

but we can't do that here because in higher dimensions this become difficult.

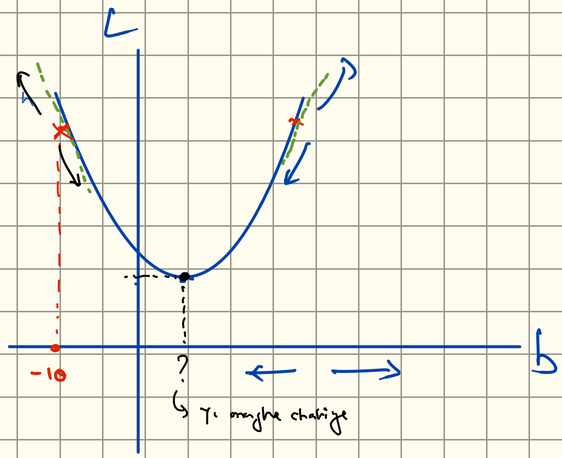
That's where Gradient Descent (GD) comes in.

Step 1 :- Select a random b_{min}

Step 2 :-

slope = -ve → Agar slope -ve h toh hamko aage jaana h
b → increment
slope = +ve ⇒ Agar slope +ve h toh hamko Piche jana h
b → decrement

Let say we take $b = -10$



In the graph, the red × shows this point, and we need to move it to the black point •, which is the minimum.

We can easily see from the graph where the minimum is, but the algorithm doesn't know in which direction to move.

There are two possible direction - either increase b (move right) or decrease b (move left).

So how will the algorithm know whether increasing or decreasing b will take it toward the minimum?

From the graph, we can tell that if we increase b, we move toward the minimum, and if we decrease b, we move away from it.

So, at any point, we need to decide whether to move forward or backword —
and that is the main trick of the gradient Descent algorithm.

Now how will we find the direction?

↳ We will calculate the slope of the equation at the point, from the direction of the slope, we can know whether to move forward or backword.

→ if the slope is negative (-ve), we should move forward (increase b).

→ if the slope is positive (+ve), we should move backword (decrease b)

slope = -ve → Agar slope -ve h toh hamko aage jaana h
b → increment

slope = +ve ⇒ Agar slope +ve h toh hamko Piche jana h
b → decrement

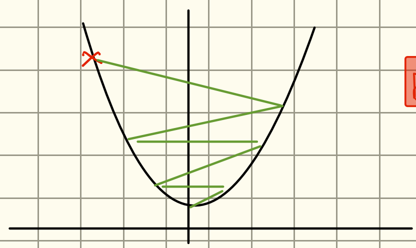
$b_{new} = b_{old} - \eta \, slope$ ⇒ gradient descent

(η → step size)

$\boxed{\eta = 0.01}$ ↳ learning rate

Note

we noticed that when we substracted the slope of bold, the red point × started moving in a zigzag way. To stop this, we multiply the slope by a learning rate (η). usually, the value of the learning rate is 0.01, but we can increase or decrease it.



$\boxed{b_{new} = b_{old} - \eta\, slope}$ ⇒ gradient descent

$\eta\, slope$ → step_size

$\boxed{\eta = 0.01}$ → learning rate

⇒ **when to stop**

① diff b_{old} with b_{new}

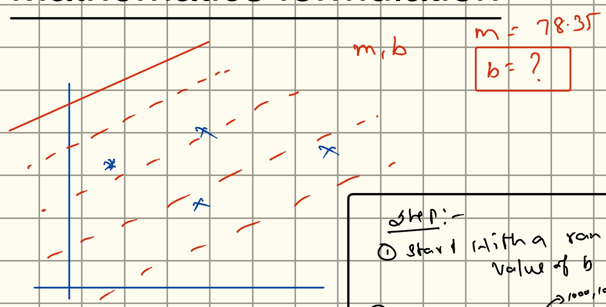
$b_{old} - b_{new} \Rightarrow 0.0001$

② iteration = $\boxed{1000}$ → ham loop chalate hai ki itne me sol^n mil hi jayega.

↓

$\boxed{\text{we call this epochs}}$

*Mathematics formulation



⇒ we already know the value of m = 78.35. now we will find the slope in form of b.

step:-

① start with a random value of b

② for i in epochs: (1000, 100)

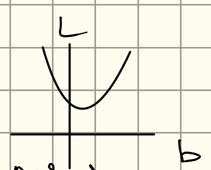
$b_{new} = b_{old} - \eta \times slope$

$\eta = 0.01$

$$L = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

loss

→ slope



$$\frac{dL}{db} = \frac{d}{db}\left(\sum_{i=1}^{n} (y_i - \hat{y}_i)^2\right)$$

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

$$slope = -2\sum_{i=1}^{n} (y_i - mx_i - \underbrace{b}_{old})$$

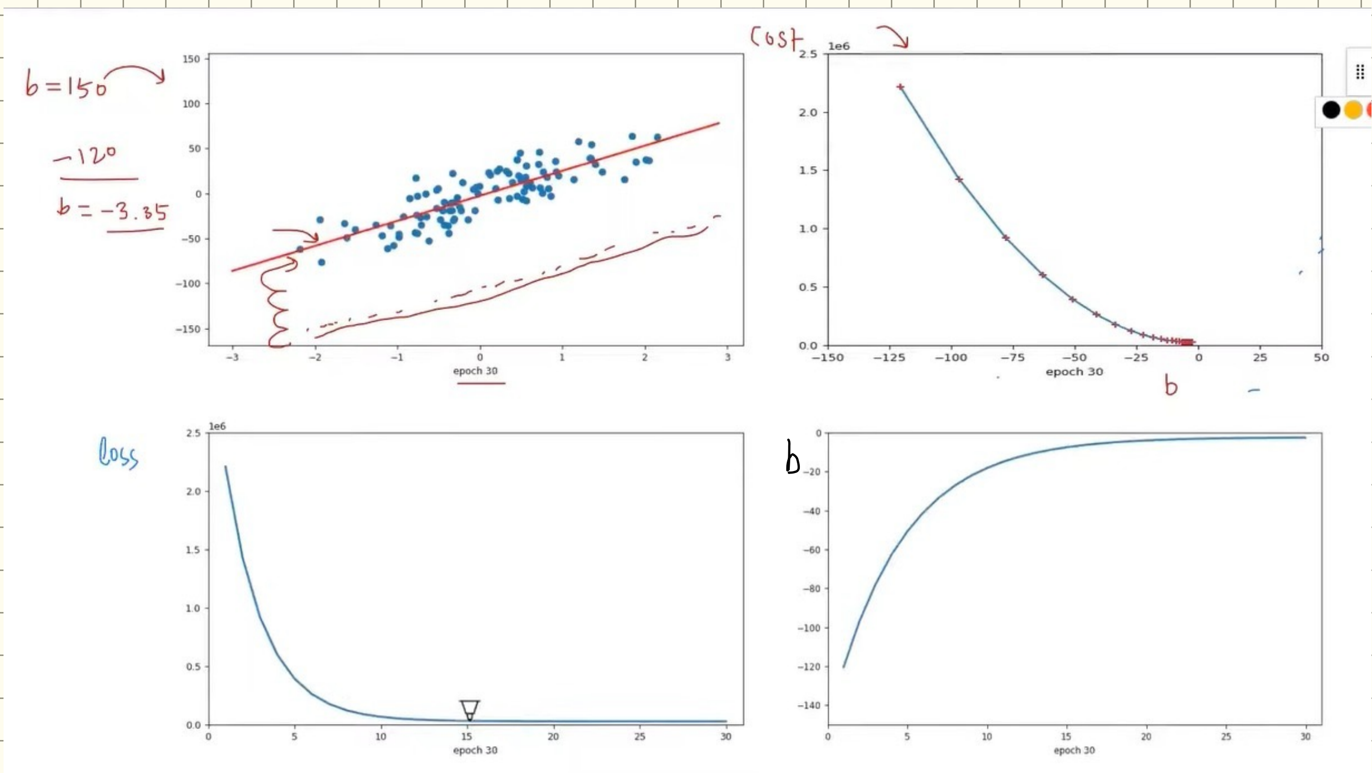
1st

$m = 78.35$

$b = 0$

$$\boxed{\text{Slope}(b) = -2 \sum_{i=1}^{n} (y_i - 78.35x_i - 0)}$$

$$\boxed{b_{new} = b_{old} - \eta \, \text{slope}_{(b=old)}}$$

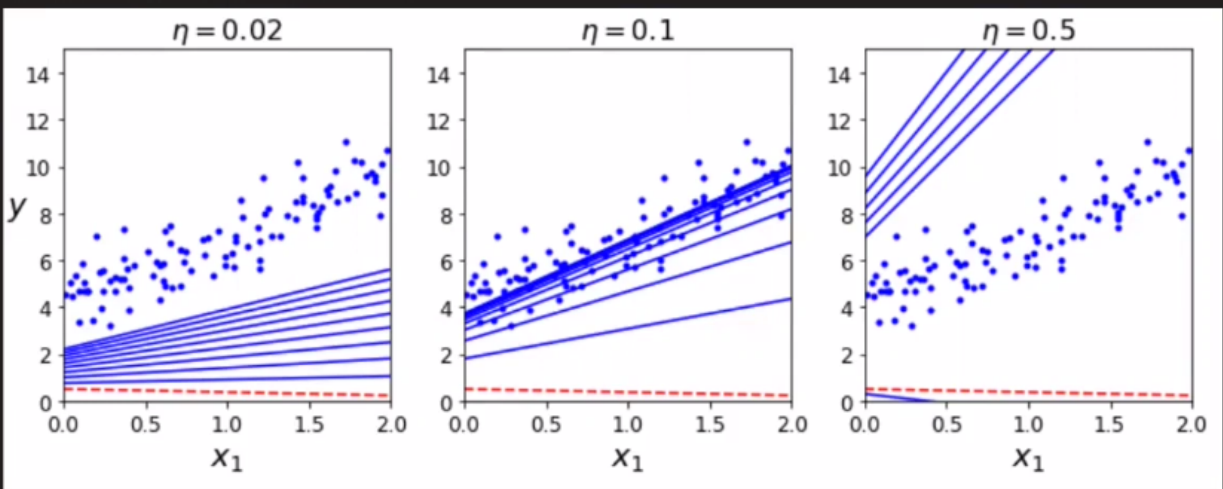


Few Discussion

[redacted]

1. Effect of Learning rate
2. The universality of Gradient Descent

epochs = 10



Adding m into the mix

Step

① initilize random values for m and b

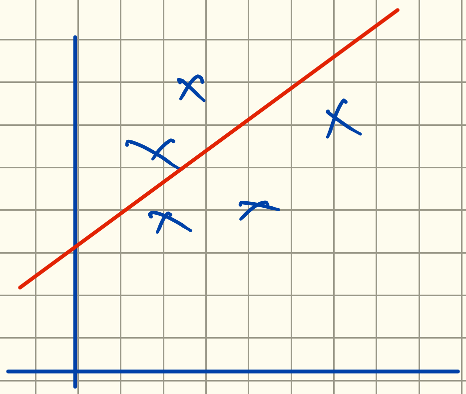
$m = 1$ & $b = 0$

② epochs = 100, $\eta = 0.01$

for i in epochs

$b = b - \eta$ slope

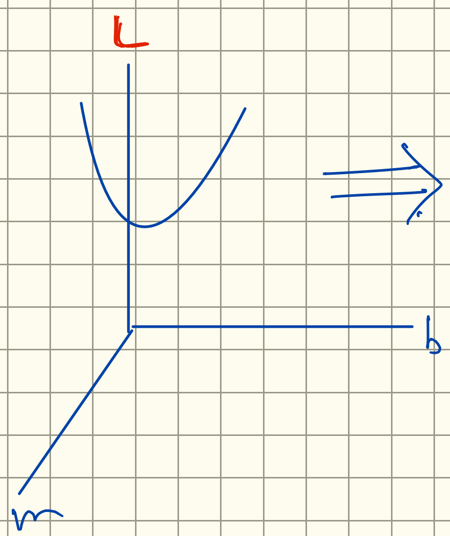
$m = m - \eta$ slope

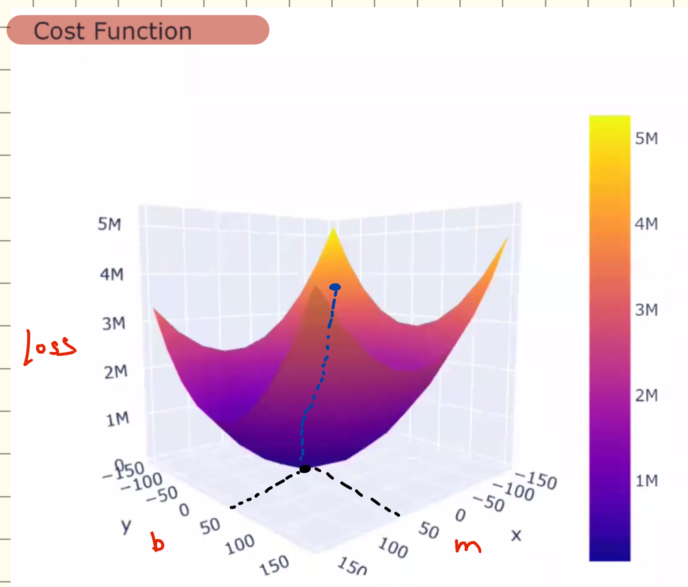


$$L = \sum (y_i - \hat{y}_i)^2$$

$$L = \sum (y_i - m x_i - b)^2$$

$L(m,b)$





$$L(m,b) = \sum (y_i - m x_i - b)^2$$

$$\boxed{b\text{-slope} = \frac{\partial L}{\partial b}}$$

$$\boxed{m\text{-slope} = \frac{\partial L}{\partial m}}$$

derivative w.r.t b

$$\frac{\partial L}{\partial b} = 2\sum (y_i - m x_i - \underset{-1}{b})$$

$$\boxed{\frac{\partial L}{\partial b} = -2\sum (y_i - m x_i - b)}$$

= slope-b at b=0

derivative w.r.t m

$$\frac{\partial L}{\partial m} = 2\sum (y_i - \underset{-x_i}{m x_i} - b)$$

$$\boxed{\frac{\partial L}{\partial m} = -2\sum (y_i - m x_i - b)\, x_i}$$

slope-m at m=1