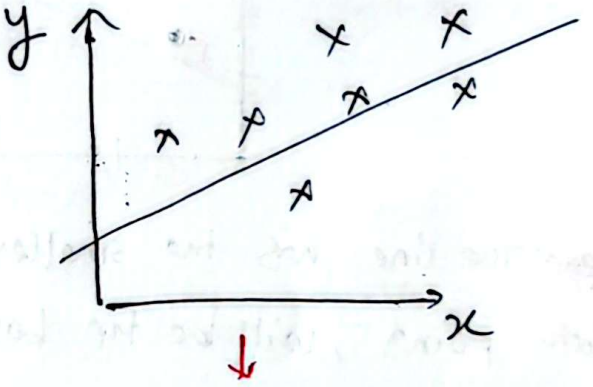
# Gradient Descent

## Linear Regression

Linear Regression is a statistical method that estimates the relationship between a dependent variable and one or more independent variables by fitting a linear equation equation to the observed data. ~~It is used for both~~

This process involves finding a 'best fit' line through the data points to minimize the error between the predicted and actual ~~data~~ values.


non linear relationship. We have used decision Tree (Regression Tree)

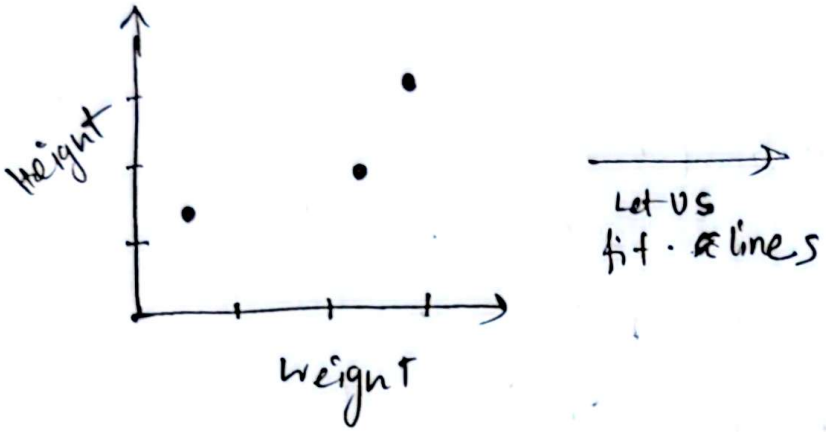



Linear relationship between dependent (y) and independent (x) variables. We can fit a linear (best fit) line.

For example, we have the following Dataset →

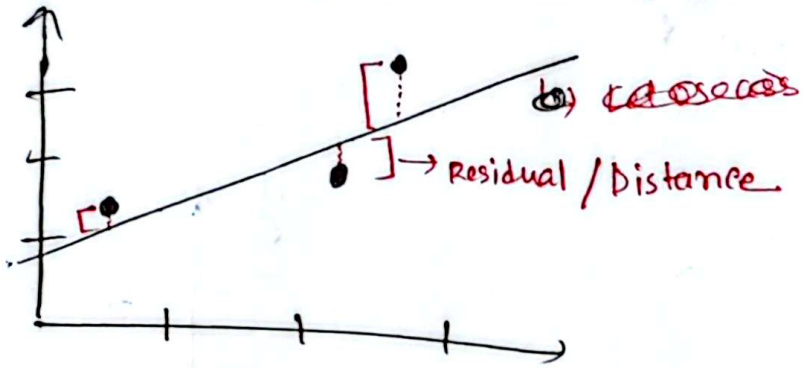
| Weight (x) | Height (y) |
|---|---|
| 0.5 - $[x^{(1)}]$ | 1.4 $[y^{(1)}]$ |
| 2.3 $[x^{(2)}]$ | 1.9 $[y^{(2)}]$ |
| 2.9 $[x^{(3)}]$ | 3.2 $[y^{(3)}]$ |

Let us plot the points →




Among these lines which one will be best fitted.?

The best fitted line will be the one, that has the smallest error(distance) from the actual points. Let us fit a line →




~~From each~~ The line has the smallest Residual or distance from the data points, will be the best fitted line. To calculate Residual or distance, we can use many error / Loss function, naming SSR, MSE, MAE. We will use SSR for the lecture.

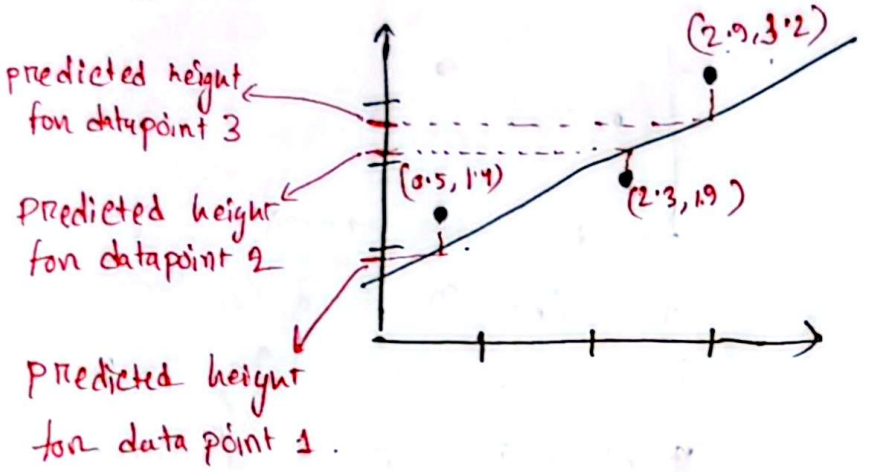
We know the equation of straight line is

$$y = mx + c$$

According to our data set,

Predicted Height = Slope x weight + intercept

This straight line equation gives us predicted target value from the actual feature values.




We calculate SSR by subtracting the predicted y (height) from the actual y (height). As we can get negative value we square the subtraction value and sum all the errors.

$$\therefore SSR = \sum_{i=1}^{i=n} \left(y_{actual}^{(i)} - y_{predicted}^{(i)}\right)^2, \quad n = \text{number of datapoints.}$$

For this dataset we are using, SSR will be,

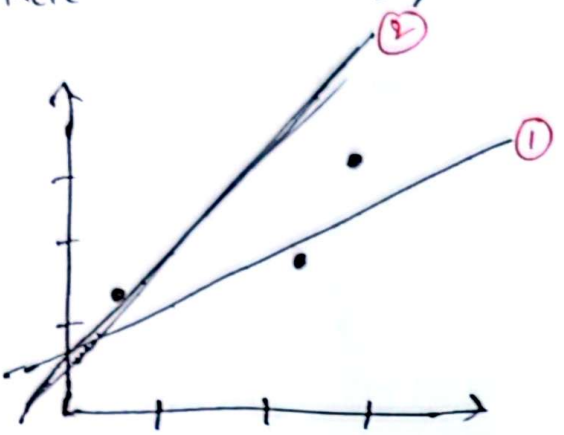
$$SSR = \left(1.4 - y_{pred}^{(1)}\right)^2 + \left(1.9 - y_{pred}^{(2)}\right)^2 + \left(3.2 - y_{pred}^{(3)}\right)^2$$

Here $y_{pred} = mx + c$.

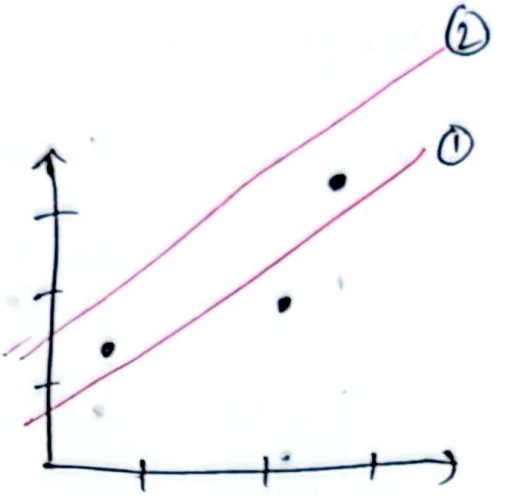
So, we need to find $y_{pred}$ values for all datapoints. For that we need the values of $m$, $x$, and $c$. Here $x$ can be easily found from the dataset. But $m$ and $c$ is unknown.

So we need find the $m$ and $c$.

Another intution is,




The line ① and line ② has intercepted the y axis at the same point. Only their slope is (m) is diffenent.




the line ① and ② has the same slope. Only their intercept point is diffenent

So, to find the best fit line, we will actually need to find the slope (m) and intercept (c) for which SSR will be minimum.

However the equation changes according to the number of features we have.

If we have 2 features, straight line equation →

$$y_{pred} = m_1 x_1 + m_2 x_2 + c$$ [need to find $m_1$, $m_2$ and $c$].

can be written as, $z = ax + by + c$ [need to find $a$, $b$, $c$]

($z$: predicted target; $x$: feature 1; $b$: feature 2)

Can be written as, $y = w_1 x_1 + w_2 x_2 + w_3$ [need to find $w_1, w_2, w_3$]

For 3 features, straight line equation,

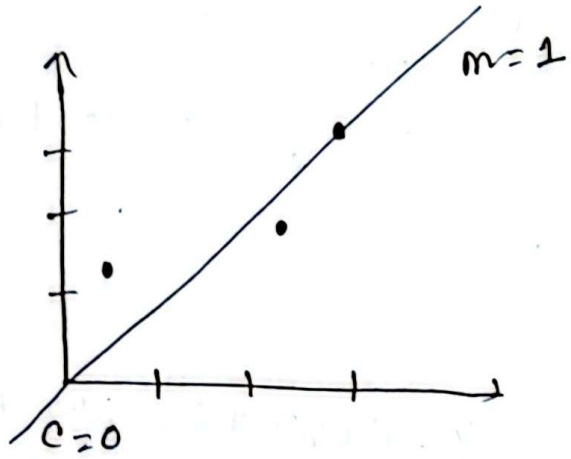
$$y = w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4$$ [need to find $w_1$ $w_2$ $w_3$ $w_4$]

To find the m and c of the best fit line, we will initialize m and c randomly at first and the optimize it. we will use Gradient Descent to optimize and find the best value for m and c.

## Gradient Descent:

Gradient Descent is is an iterative optimization algorithm used in machine learning to find the minimum of an function (here for m and c), typically a loss function.

Gradient Descent used in many other ML Algorithms like Logistic regression, PCA, neural network etc..




Here we will initialize c=0 and m=1 and then optimize these parameters for it's best fit line.

So Gradient descent works by repeatedly taking steps in the direction of the steepest descent to adjust parameter and minimize the difference between actual y and predicted y..

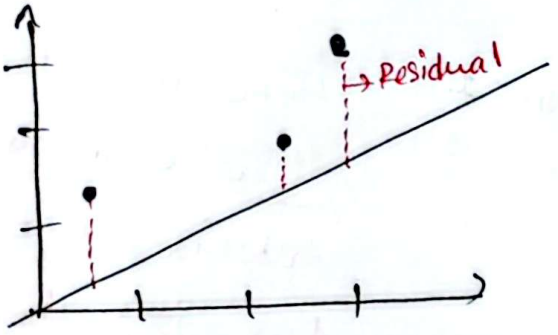
## Finding Intercept (c) using Gradient Descent:

Let us at first find the intercept (c) of a linear best fit line using Gradient Descent. For the previous data set, we will set $m = 0.64$. However, in the actual process we need to find both m and c parallely. For simplicity we will now skip the m.

### Step-1:

Pick a random value for the parameter, c.

Let's initialize c, $c = 0$. [we can pick any number]




$y_{pred} = mx + c$

$\Rightarrow y_{pred} = 0.64x + 0$

To evaluate the initial line ($y = 0.64x + 0$), we need to find SSR. Here SSR is used as Loss Function.

We have seen, for this dataset,

$$SSR = (1.4 - y_{pred}^{(1)})^2 + (1.9 - y_{pred}^{(2)})^2 + (3.2 - y_{pred}^{(3)})^2$$

$$= [1.4 - (0.64x^{(1)} + 0)]^2 + [1.9 - (0.64x^{(2)} + 0)]^2 + [3.2 - (0.64x^{(3)} + 0)]^2$$

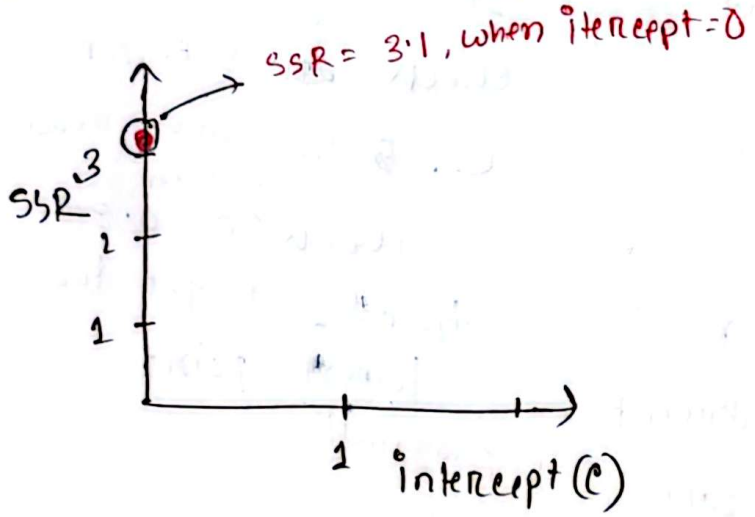
$$= [1.4 - (0.64 \times 0.5 + 0)]^2 + [1.9 - (0.64 \times 2.3 + 0)]^2 + [3.2 - (0.64 \times 2.9 + 0)]^2$$

$$= (1.4 - 0.32)^2 + (1.9 - 1.5)^2 + (3.2 - 1.9)^2$$

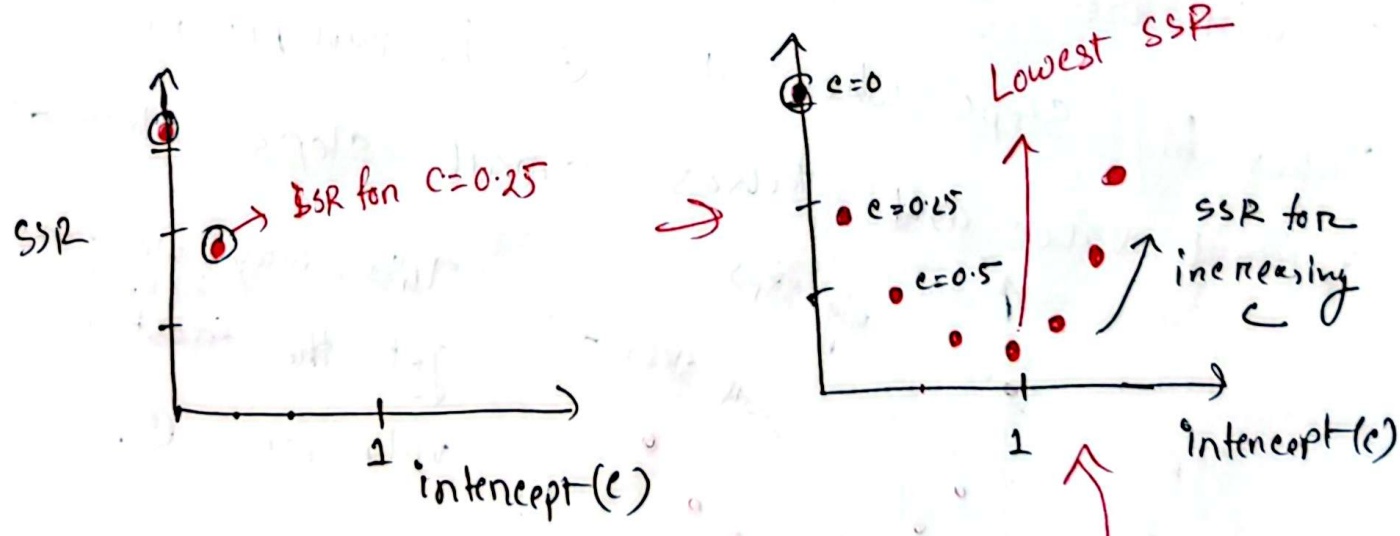
$$= (1.1)^2 + (0.4)^2 + (1.3)^2$$

$$= 3.1$$

we can plot the SSR value in the graph for c=0 like below →

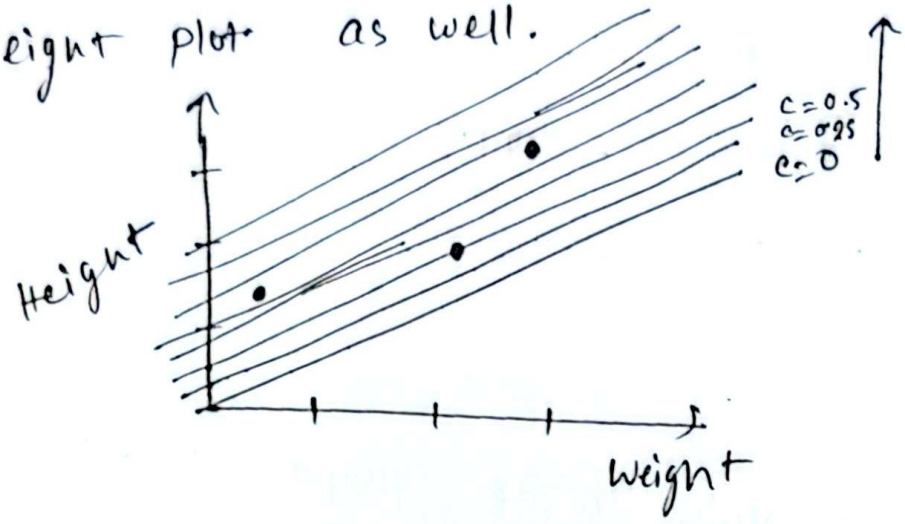



Now if we increase intercept to c = 0.25, we will get different SSR (probably less than 3.1). If we plot this →

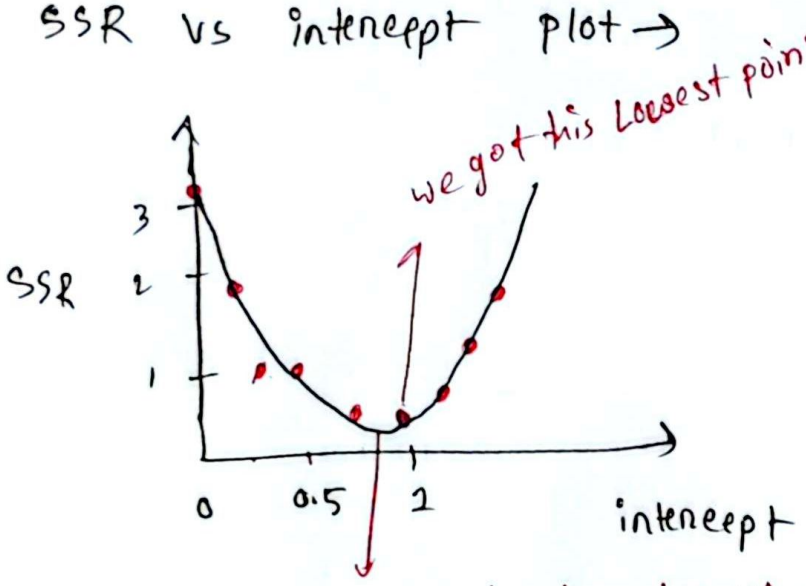



if we keep increasing c, we will get this plot ←

here. We can visualize the c in our Height vs Weight plot as well.

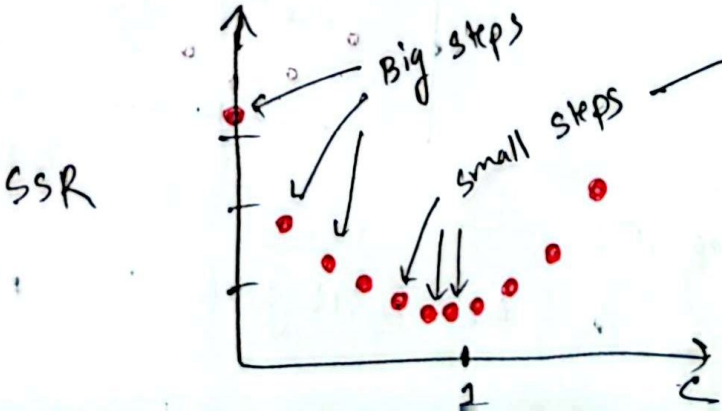



SSR vs inteneept plot →

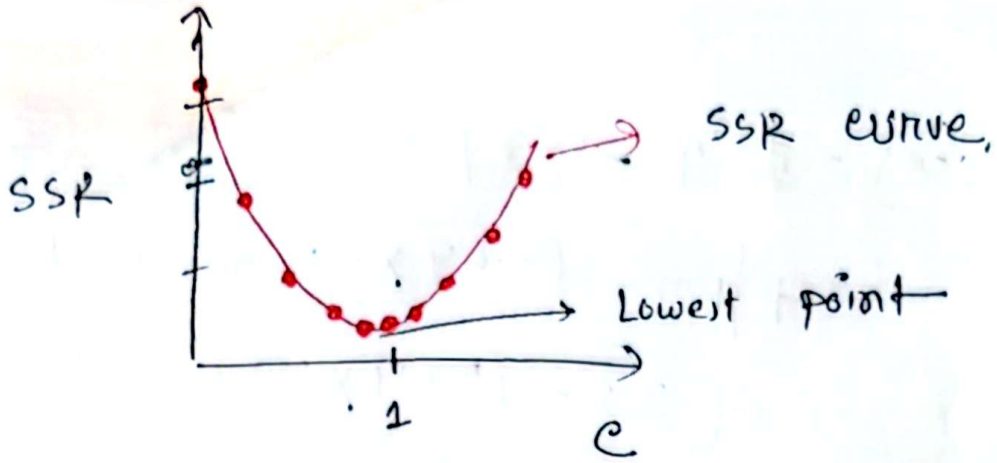



It is very tiresome to check SSR for C. We may need to check between c = 0.90 to c = 1 to get the lowest point.

As this manually plug-in method is not optimal way, we will use gradient Descent. Gradient Descent takes big steps when the c is far from Optimal value and takes small steps when close.




This way we can get the best/optimal value for C.




we need to go to the lowest point of SSR curve.

From Calculus we know →


~~The~~ ~~Previously we have~~

By taking derivative of this curves we get the slope. ~~and~~ Here we want to reach the slope = 0 point as it will give us minimum SSR.

Previously we saw the equation of SSR / Loss function

$$\text{Loss function (SSR)} = \left(y^{(1)}_{actual} - y^{(1)}_{pred}\right)^2 + \left(y^{(2)}_{actual} - y^{(2)}_{pred}\right)^2 + \left(y^{(3)}_{actual} - y^{(3)}_{pred}\right)^2$$

If we plug $y_{pred.} = mx + c$ where $m = 0.64$

$x = x^{(1)}, x^{(2)}, x^{(3)}$

we get,

$$\text{Loss function (SSR)} = [1.4 - (0.64 \times 0.5 + c)]^2 + [1.9 - (0.64 \times 2.3 + c)]^2 + [3.2 - (0.64 \times 2.9 + c)]^2$$

So, by taking derivative of this function we can descent to the lowest point of the curve where the SSR is lowest. As we are finding c, we will take derivative in terms of c.

Step-1: Taking derivative of the Loss function in terms of parameters, here c.

$$\frac{d(\text{Loss Function})}{dc} = \frac{d}{dc}[1.4 - (0.64 \times 0.5 + c)]^2 + \frac{d}{dc}[1.9 - (0.64 \times 2.3 + c)]^2 + \frac{d}{dc}[3.2 - (0.64 \times 2.9 + c)]^2$$

$$= 2 \times [1.4 - (0.64 \times 0.5 + c)] \times (-1) + 2 \times [1.9 - (0.64 \times 2.3 + c)] \times (-1) + 2 \times [3.2 - (0.64 \times 2.9 + c)] \times (-1)$$

$$= -2[1.4 - (0.64 \times 0.5 + c)] - 2[1.9 - (0.64 \times 2.3 + c)] - 2[3.2 - (0.64 \times 2.9 + c)]$$

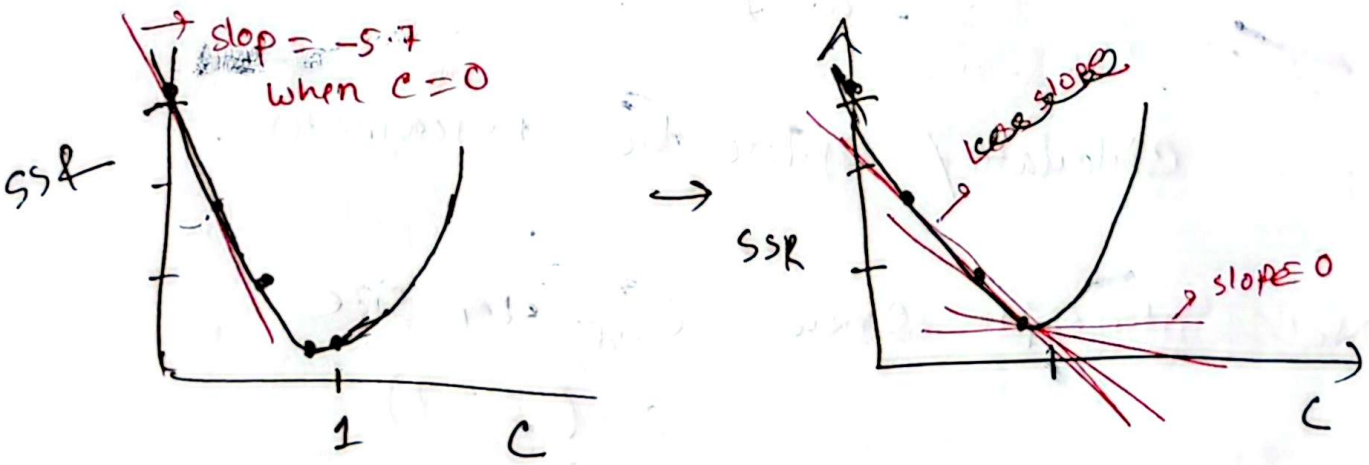
Instead of ~~finding~~ solving $\frac{d}{dc}(\text{Loss function}) = 0$ to find the minimum SSR, Gradient descent finds minimum value by taking steps ($c=0$, $c=0.25$...) from initial value, until it reaches the best value. This makes Gradient Descent useful when it's not possible to solve $\frac{d}{dc}(\text{Loss function}) = 0$.

Step-3: Plug the parameter values (here c) in the derivative.

So, for $c = 0$.

$$\frac{d}{dc}(\text{Loss function}) = -2[1.4 - (0.64 \times 0.5 + 0)] - 2[1.9 - (0.64 \times 2.3 + 0)] - 2[3.2 - (0.64 \times 5.9 + 0)]$$

$$= -5.7$$




The closer we get the optimal value of c, the closer the slope of the curve will get to 0.

When we close to optimal value we should take small baby step ( $c = 0.90 \rightarrow c = 0.93$ ) when the slop step = 0.03 is far from zero, we should take big step. The size of the step is related to slope, which tells us if we should take baby step or big step.

~~In Gradient~~

Step-4: Calculate step size of parameter (here c)

Using,

**Step size = Slope x learning rate ($\alpha$)**

Here step size = $\frac{d}{dc} \times$ learning rate ($\alpha$)

$= -5.7 \times 0.1$

$= -0.57$ [when $c = 0$]

Step-5: Calculate / Update the parameter.

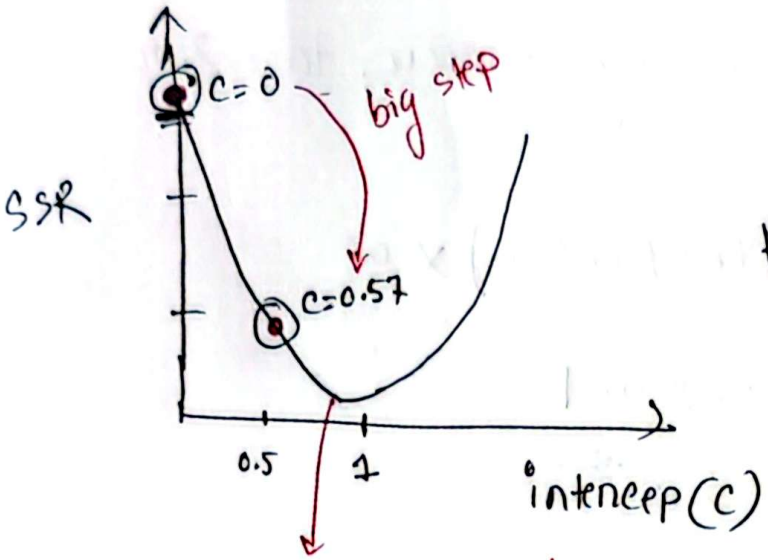
New intercept $C_{new} = C_{old}$ step size

$= 0 - (-0.57)$

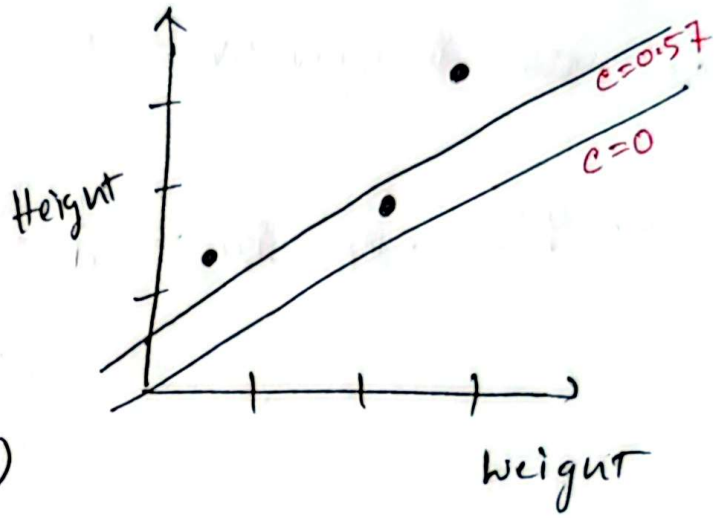
$= 0.57$

**New parameter = old parameter - Step size**

So, new intercept is $c = 0.57$.




Optimal value for c is here. So we need to take more steps.




To take more steps we need to continue our process (Step 3 to step 5) Again and again; We are done with our First Iteration.

2nd Iteration:

Step 3: plug parameter (c here) in the derivative.

now, $c = 0.57$.

$$\therefore \frac{d}{dc}(\text{Loss function}) = -2[1.4 - (0.64 \times 0.5 + \underset{c}{0.57})] - 2[1.9 - (0.64 \times 2.3 + \underset{c}{0.57})] - 2[3.2 - (0.64 \times 2.9 + \underset{c}{0.57})]$$

$$= -2.3$$

$\frac{d}{dc} SSR(c=0) = -5.7$ and $\frac{d}{dc} SSR(c=0.57) = -2.3$

so, derivative is moving closer to zero.

**Step-4:** Step size $= \frac{d}{dc}(\text{Loss function}) \times \alpha$

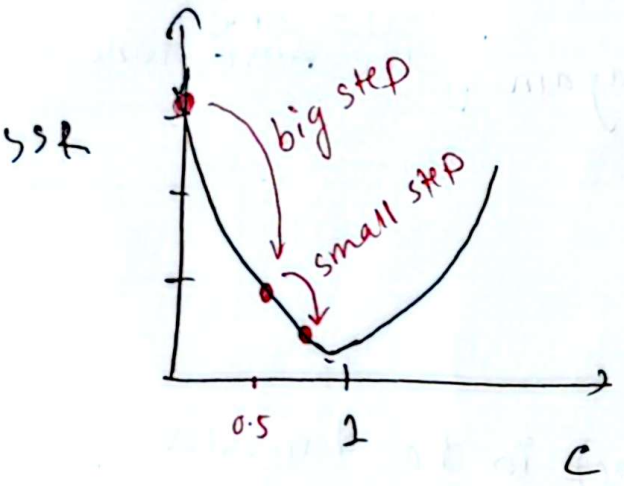
$= -2.3 \times 0.1$

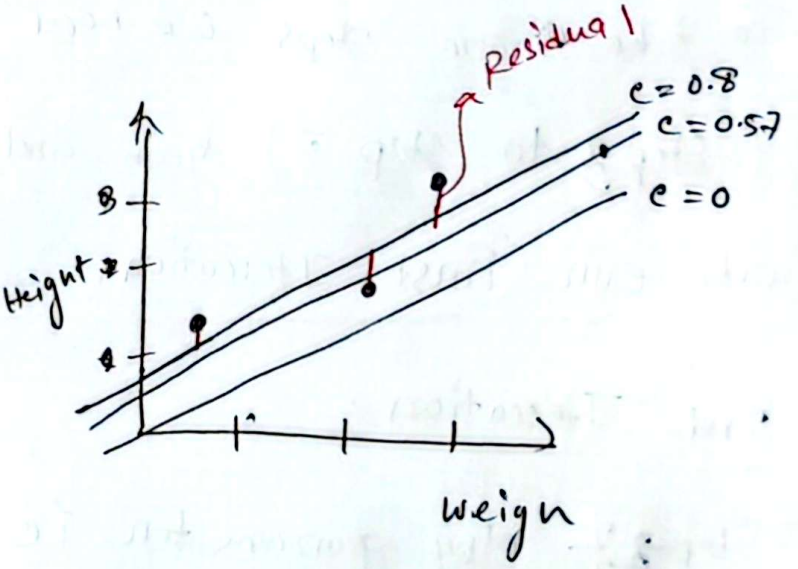
$= -2.3$

**Step-5:** $c_{new} = c_{old} - \text{step size}$

$= 0.57 - (-2.3)$

$= 0.8$




Residuals are getting smaller.

If we continue our iteration - the next c value will be →

$c_4 = 0.89$

$c_5 = 0.92$

$c_6 = 0.94$

$c_7 = 0.95$

Here each steps are getting smaller and smaller and we are getting close to optimal c.

Now we need to know when to stop this iteration. We can do this in two ways →

(1) Stop when step size is close to zero. In practice when step size is 0.001 or smaller we stop the iteration.

(2) We can also limit on the number of iteration. In practice if number of iteration 1000 or greater, we stop.

We have done the process of Gradient Descent for finding c. But we need to find m (slope) as well. Now we will find the best fit line $y = mx + c$, aka, we will find m and c both at the same time.

## Gradient Descent to find m and c:

Step-1: Taking Derivative of the Loss function in terms of parameters (m and c both here)

we will find m as well.

Loss function (SSR) $= [1.4 - (m \times 0.5 + c)]^2 + [1.9 - (m \times 2.3 + c)]^2 + [3.2 - (m \times 2.9 + c)]^2$

$$\frac{d}{dc}(\text{Loss function}) = -2\left[1.4 - (m \times 0.5 + c)\right] - 2\left[1.9 - (m \times 2.3 + c)\right] - 2\left[3.2 - (m \times 2.9 + c)\right]$$

Similarly,

$$\frac{d}{dm}(\text{Loss function}) = \frac{d}{dm}\left[1.4 - (m \times 0.5 + c)\right]^2 + \frac{d}{dm}\left[1.9 - (m \times 2.3 + c)\right]^2 + \frac{d}{dm}\left[3.2 - (m \times 2.9 + c)\right]^2$$

↓ derivative of loss function in terms of m

$$= 2\left[1.4 - (m \times 0.5 + c)\right] \times (0.5) + 2\left[1.9 - (m \times 2.3 + c)\right] \times (-2.3) + 2\left[3.2 - (m \times 2.9 + c)\right] \times (-2.9)$$

$$\therefore \frac{d}{dm}(\text{Loss function}) = -2 \times 0.5\left[1.4 - (m \times 0.5 + c)\right] - 2 \times 2.3\left[1.9 - (m \times 2.3 + c)\right] - 2 \times 2.9\left[3.2 - (m \times 2.9 + c)\right]$$

When we have two or more derivative of a function, they are called **Gradient.** we will use this Gradient to descent to the lowest point of the Loss function (SSR curve). This is why the Algorithm is call **Gradient Descent.**

Step-2: pick a random value for the parameters (here m and c).

Let $m = 1$ and $c = 0$.

Step-3: plug the parameter values in their derivatives.

$$\therefore \frac{d}{dc}(\text{Loss function}) = -2[1.4 - (1 \times 0.5 + 0)] - 2[1.9 - (1 \times 2.3 + 0)] - 2[3.2 - (1 \times 2.9 + 0)]$$

$$= \boxed{-1.6}$$

$$\therefore \frac{d}{dm}(\text{Loss function}) = -2 \times 0.5[1.4 - (1 \times 0.5 + 0)] - 2 \times 2.3[1.9 - (1 \times 2.3 + 0)] - 2 \times 2.9[3.2 - (1 \times 2.9 + 0)]$$

$$= \boxed{-0.8}$$

Step-4: calculate step size of parameters (m and c)

$$\therefore \text{Step size}(c) = \frac{d}{dc} \times \text{learning rate}$$

$$= -1.6 \times 0.01$$

$$= -0.016$$

→ Previous learning rate will not work this time. Gradient descent is sensitive to learning rates.

$$\therefore \text{Step size}(m) = \frac{d}{dm} \times \text{learning rate}$$

$$= -0.8 \times 0.01$$

$$= -0.008$$

Step-5: calculate the parameters (c and m)

new intercept $c_{new} = c_{old} - \text{step size}$

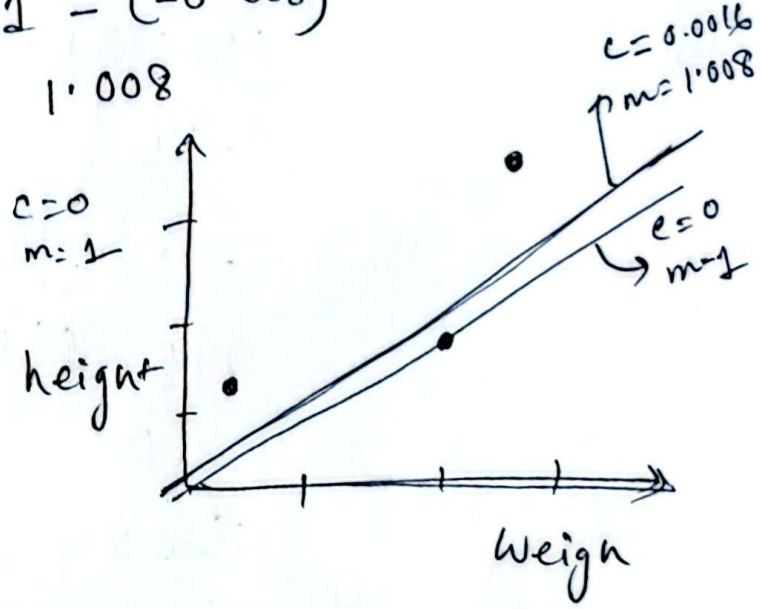
$= 0 - (-0.016)$

$= 0.016$

new slope $m_{new} = m_{old} - \text{step size}$

$= 1 - (-0.008)$

$= 1.008$




Now we will ~~continue~~ repeat this process until we reach our termination condition.

For this data set, we will find, $c = 0.95$ and $m = 0.64$ as the best fitted line.

- If we have three features →

| x | y | z (actual) |
|---|---|---|
| 1 $x^{(1)}$ | 2 $y^{(1)}$ | 13 $z^{(1)}$ |
| 2 $x^{(2)}$ | 3 $y^{(2)}$ | 18 $z^{(2)}$ |
| 3 $x^{(3)}$ | 4 $y^{(3)}$ | 23 $z^{(3)}$ |

↓ feature 1 ↓ feature 2 ↓ target

The equation for best fit line will be →

$$z_{pred} = ax + by + c$$

or $z = m_1 x + m_2 y + c$

So, to find the best fit line we need to find a, b, c or $m_1, m_2$ ; c using gradient Descent.

Step-1 : Derivative of the loss function in term of parameters (a, b, c).

If we use SSR as loss function,

$$\text{Loss function (SSR)} = (z^{(1)}_{actual} - z^{(1)}_{pred})^2 + (z^{(2)}_{actual} - z^{(2)}_{pred})^2 + (z^{(3)}_{actual} - z^{(3)}_{pred})^2$$

$$= [13 - (a x^{(1)} + b y^{(1)} + c)]^2 + [18 - (a x^{(2)} + b y^{(2)} + c)]^2 + [23 - (a x^{(3)} + b y^{(3)} + c)]^2$$

$$= [13 - (a + 2b + c)]^2 + [18 - (2a + 3b + c)]^2 + [23 - (3a + 4b + c)]^2$$

Then we have to find $\frac{d}{da}$(Loss function), $\frac{d}{db}$(Loss function) and $\frac{d}{dc}$(Loss function)

a, b, c are parameters which we need to fine-tune.

Step-2&3: plug in ~~initial~~ value of parameters into the derivatives.

We can set $a = 1$, $b = 1$ and $c = 1$ and plug these values into $\frac{d}{da}$(Loss function), $\frac{d}{db}$(Loss function), $\frac{d}{dc}$(Loss function)

Step 4-5: update the parameter (a, b, c)

Step size = derivative × learning rate.

$\therefore a_{new} = a_{old} - \text{Step size}$

$$a_{new} = a_{old} - \frac{d}{da}(\text{Loss function}) \times \alpha$$

Similarly

$$b_{new} = b_{old} - \frac{d}{db}(\text{Loss function}) \times \alpha$$

$$c_{new} = c_{old} - \frac{d}{dc}(\text{Loss function}) \times \alpha$$