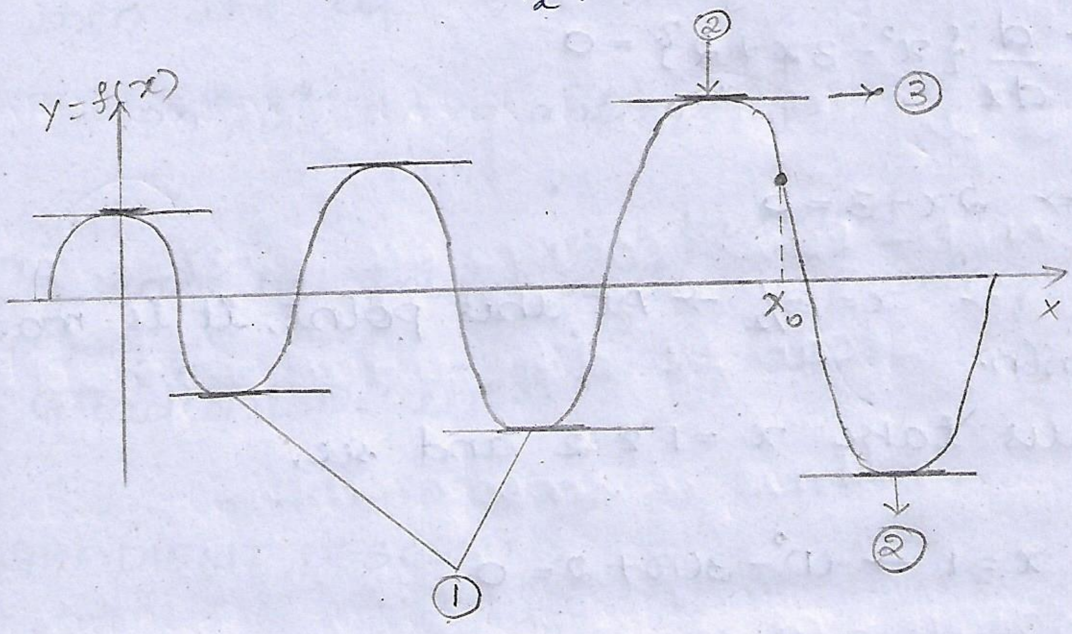
→ Let us consider an $eq^n$ and minimise it.

↳ $eq^n \Rightarrow y = x^2 - 3x + 2$.

$\Rightarrow$ min → $x^* = \underset{x}{\arg\min}\{x^2 - 3x + 2\}$




① → LOCAL MAXIMA

② → GLOBAL MAXIMA.

③ → TANGENT

→ The slopes of the tangents at max (or) min is equal to "zero".

$\Rightarrow \left[\frac{d}{dx}\{f(x)\}\right]_{x_0}$ – It gives the slope of the function.

$\Rightarrow x^* = \underset{x}{\arg\min}\{x^2 - 3x + 2\}$

As we know that, at min (or) max slope is equal to zero.

slope $- \frac{dy}{dx} = 0 \rightarrow$ If this holds true.

$\Rightarrow \frac{d}{dx}\{x^2 - 3x + 2\} = 0$

$\Rightarrow 2x - 3 = 0$

$\Rightarrow x = 3/2 \rightarrow$ At this point, it is max. (or) min.

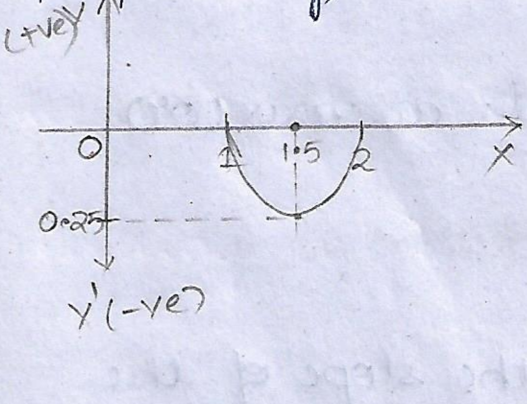
Let us take $x = 1$ & $2$ and see,

$\Rightarrow x = 1 \rightarrow (1)^2 - 3(1) + 2 = 0$

$\Rightarrow x = 2 \rightarrow (2)^2 - 3(2) + 2 = 0$

$\Rightarrow x = 1.5\ (3/2) \rightarrow (1.5)^2 - 3(1.5) + 2 = -0.25$

Graphically,




$\rightarrow$ The value lies in the -ve Y-axis.

FROM OPTIMISATION OF LINEAR REGRESSION

we know that,

$$\Rightarrow m^*, c^* = \underset{m,c}{\arg\min}\{\Sigma(y_i - \{mx_i + c\})^2\}$$

let us consider $[y_i - \{mx_i + c\}]$ as $q$.

then the eq$^n$ will be

$$\Rightarrow m^*, c^* = \underset{m,c}{\arg\min}\{\Sigma q^2\}$$

↳ '$q$' can be found very fast and easily by using Gradient Descent.

GRADIENT DESCENT :

It is an optimization algorithm used to minimize some function by iteratively moving in the direction of steepest descent as defined by the negative of the gradient.

→ In machine learning, we use

Gradient descent to update the parameters of our model.

→ This is a iterative algorithm.

→ In modern computers, it is very fast.

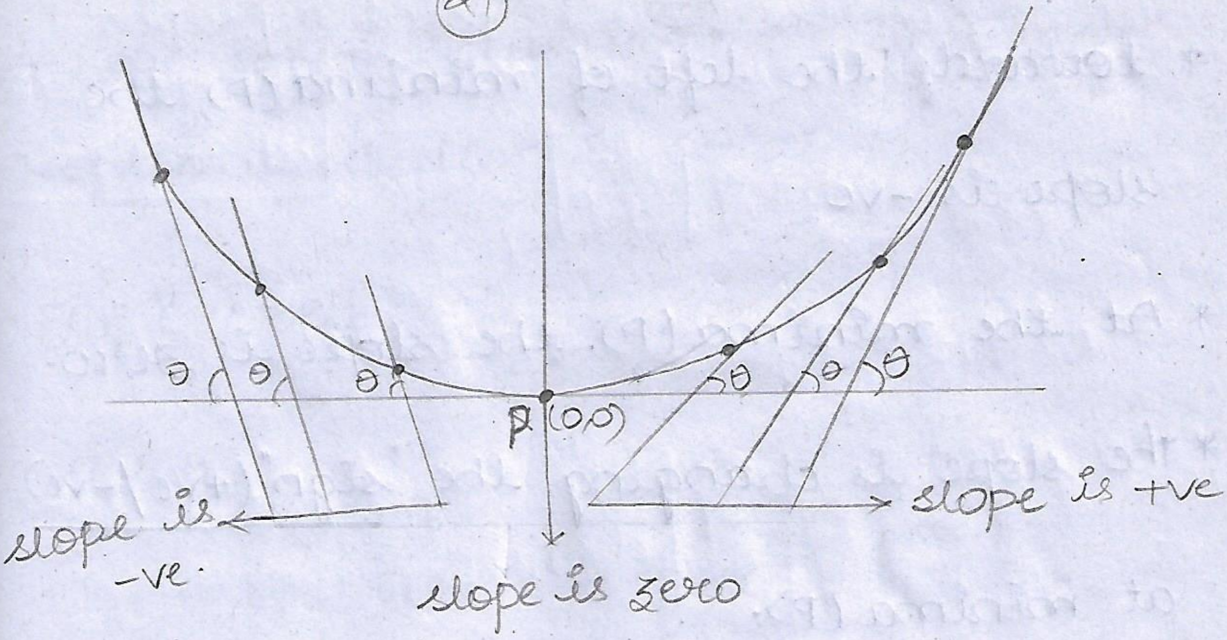
NOTE:

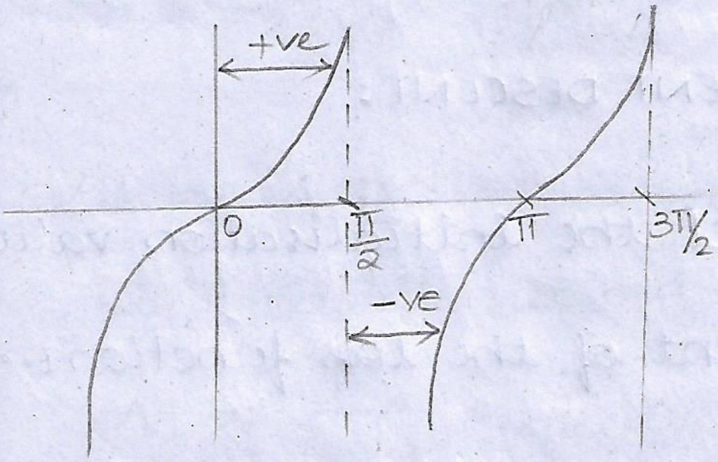
↳ If it is

*→ SQUARED - Gives continuous function and can be differentiated at any point, need to have a smooth curve.

*→ ABSOLUTE - Gives discontinuous function and can't be differentiable at any point.

Let us look in detail about this by Plotting a graph.




STEPS:

↳ We need to figure out the tangents

slope $\left(\frac{d}{dx}\right)$

where $\theta \to$ tangent

OBSERVATION FROM THE GRAPH:

↳ As we know to compute the slope,

* towards the right of minima (P), the slope is +ve

* towards the left of minima(P), the slope is -ve

* At the minima (P), the slope is zero.

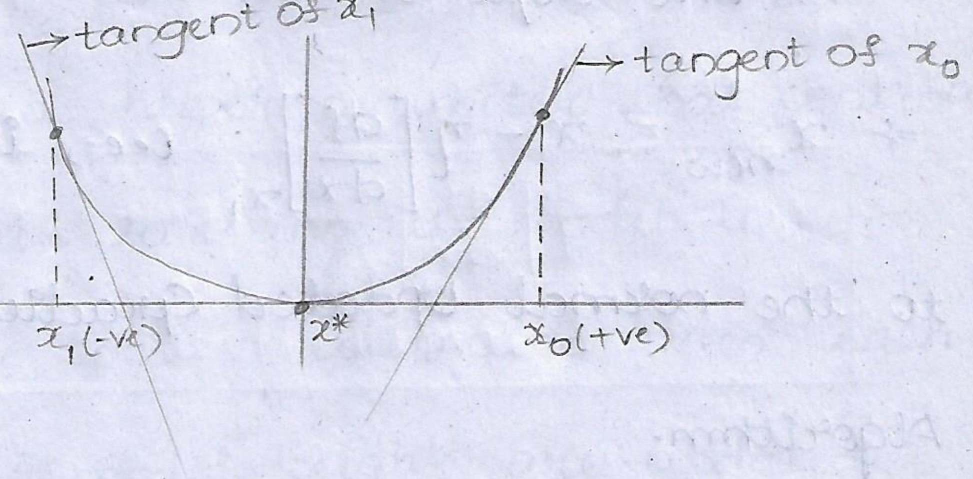
** The slope is changing the "sign" (+ve/-ve) at minima (P).

STEPS FOR GRADIENT DESCENT:

1. Randomly select the initialisation values.
2. Take the gradient of the loss function i.e., the derivative of the old function for each parameter in it.
3. Substitute the parameter values in the gradient.
4. Calculate step size by using the appropriate learning rate.
5. Calculate the new parameters.

6. Repeat the steps from $3^{rd}$ until an optimal solution is obtained.

FOR EXAMPLE:




→ As per step-1:

we randomly picked some values i.e., $x_0$, $x_1$ & $x^*$ respectively.

→ As per step-2:

we take the derivative i.e.,

$$\Rightarrow \boxed{x_{new} = x_0 - \eta \left[\frac{df}{dx}\right]_{x_0}}$$ where $\eta$ - magnitude (eta)

↳ "this is the UPDATED GRADIENT DESCENT ALGORITM".

↳ Make a step (move) in the direction opposite to the gradient, opposite direction of slope increases from the current point by alpha times the gradient at that point.

Similarly,

$$\Rightarrow x_{new} = x_1 + \eta\left[\frac{df}{dx}\right]_{x_1}$$

As the slope is -ve, then

$$\Rightarrow x_{new} = x_1 - \eta\left[\frac{df}{dx}\right]_{x_1}$$ i.e., it is back

to the normal Updated Gradient Descent Algorithm.

↳ ** $\eta$ is called as the LEARNING RATE.

LEARNING RATE:

It determines the stepsize at each iteration while moving towards a minimum of a loss function.

→ It is a tuning parameter in a optimization algorithm.

→ In setting a learning rate, there is a trade-off (a balance achieved between two desirable values) between the rate of convergence and overshooting.

→ This is a hyperparameter that controls how much to change the model in response to the estimated error each time the model weights are updated.

→ As per step-6:

we will repeat the steps from 3 to 5 until the convergence i.e., minima point (or) optimal value.

| |
|---|
| So, for this, we take for loop since it is needed to repeat multiple times. |

Now let us take $x_0$ and compute the Gradient Descent Algorithm.

STEP-1: Randomly pick a value of $x = x_0$

STEP-2: Compute $x_1$

$$\boxed{\Rightarrow x_1 = x_0 - \underbrace{\eta}_{\text{LEARNING RATE}} \left[\frac{df}{dx}\right]_{x_0} \rightarrow \text{UPDATE FUNCTION}}$$

STEP-3: From Optimisation of Linear Regression we know that

$$\Rightarrow m^*, c^* = \underset{m,c}{\arg\min} \sum \left(Y_{i\,pred} - Y_{act}\right)^2$$

$$\Rightarrow m^*, c^* = \underset{m,c}{\arg\min} \sum \left(Y_{i\,pred} - \{mx_i + c\}\right)^2$$

As $x_0$ is calculated, then

$$m_0^* \rightarrow m_1 \quad \& \quad c_0^* \rightarrow c_1$$

$$\Rightarrow m_1 = m_0 - \eta \left[\frac{\partial f}{\partial m}\right]_{m_0}$$

where $\frac{\partial f}{\partial m} \rightarrow$ partial differenciation since it is unknown.

$$\Rightarrow c_1 = c_0 - \eta\left[\frac{\partial f}{\partial c}\right]_{c_0}$$

STEP-4: Repeat step-2, until it reaches the optimal value i.e., convergence.

NOTE:

As the slope changes, the values get decreased.

Finally, we can say that

$$\Rightarrow \boxed{x_{new} - x_{old} \underbrace{\approx}_{\text{very close}} 0}$$

i.e., when the difference between the $x_{new}$ & $x_{old}$ tends to be very close to zero, then it is said to be converged.

$$\Rightarrow \frac{\partial f}{\partial m} = \frac{\partial \Sigma \left(y_i - \{m x_i + c\}\right)^2}{\partial m}$$

$$\Rightarrow \frac{\partial f}{\partial m} = \Sigma\, 2 \cdot \left(y_i - \{m x_i + c\}\right) \left\{ \cancel{\frac{\partial y}{\partial m}}^{\,0} - \frac{\partial (m x_i + c)}{\partial m} \right\}$$

$$\boxed{\Rightarrow \frac{\partial f}{\partial m} = \Sigma\, 2\left(y_i - \{m x_i + c\}\right)(-x_i)}$$

similarly,

$$\Rightarrow \frac{\partial f}{\partial c} = \frac{\partial \Sigma \left(y_i - \{m x_i + c\}\right)^2}{\partial c}$$

$$\Rightarrow \frac{\partial f}{\partial c} = \Sigma\, 2\left(y_i - \{m x_i + c\}\right) \left\{ \frac{\partial y}{\partial c} - \left[ \frac{\partial (m x_i)}{\partial c} + \frac{\partial}{\partial c}(c) \right] \right\}$$

$$\boxed{\Rightarrow \frac{\partial f}{\partial c} = -2\left(y_i - \{m x_i + c\}\right)(1)}$$