1) Why Stochastic Gradient Descent (SGD)?

→ In Gradient Descent:

$$W_1 = W_0 - \frac{1}{N}\alpha \sum_{i=1}^{N} \left(\frac{\partial L}{\partial W}\right)_{W_0}$$

$N \Rightarrow$ Total no. of Training Examples

$L \Rightarrow$ loss function Eg: (MSE) $L = \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{N}$

→ Since finding gradient for whole Dataset takes lot of time.

∴ Stochastic Gradient Descent

2) SGD:

$$W_1 = W_0 - \frac{\alpha}{n} \sum_{i=1}^{n} \left(\frac{\partial L}{\partial W}\right)_{W_0}$$

$\alpha \Rightarrow$ Learning rate
$L \Rightarrow$ loss function
$n \Rightarrow$ samples from "N"

Step 1) Initialize the weight & bias matrices with random values & weight matrix shape = (1, no. of dim)

```
W = np.random.randn(1, no.of.dim)
b = np.random.randn(1, 1)
```

Step 2) Pick randomly "n" no. of samples from total Dataset A of "N" data points

Step 3) Separate X-to & y-to.

Step 4) Differentiate the loss function (MSE)

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

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

Linear Regn

$\hat{y}_i = w^T x_i + b$

$$\frac{\partial L}{\partial w} = \sum_{i=1}^{n} (-2x_i)\left[y_i - w^T x_i - b\right]$$

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{n} (-2)\left[y_i - w^T x_i - b\right]$$

Here x_i, w are vectors

Eg: x_1 ⇒ for 1st training example datapoint (Boston Data) 13 columns

$x_1 = [x_{11}\ x_{12}\ x_{13}, \ldots - x_{1,13}]_{(1\times 13)}$

$w_0 = [w_{11}\ w_{12}\ w_{13}, \ldots - w_{1,13}]_{(1\times 13)}$

y_i ⇒ scalar value

Steps) Epoch 1

5.1) Initialize sum of $\frac{\partial L}{\partial w}$ & sum of $\frac{\partial L}{\partial b}$ to zero values initially

5.2) Iterate over "n" training examples with first $w = w_0$ & find sum of $\frac{\partial L}{\partial w}$ over "n" examples & also $\frac{\partial L}{\partial b}$ sum over "n" examples.

Eg:- if $n=3$

$$\text{sum}\left(\frac{\partial L}{\partial w}\right) = \left[(-2x_1)(y_1 - w_0^T x_1 - b) + (-2x_2)(y_2 - w_0^T x_2 - b) + (-2x_3)(y_3 - w_0^T x_3 - b)\right]$$

$$\text{sum}\left(\frac{\partial L}{\partial b}\right)$$

step 5.3) Find next "w" $W_{old} \leftarrow$ W previous step

$$W_{new} = W_{old} - \alpha \frac{[\text{sum of } \partial L/\partial w]}{n}$$

$$b_{new} = b_{old} - \alpha \left[\frac{\text{sum of } \partial L/\partial b}{n}\right]$$

$w_1 \Leftarrow$ $b_1 \Leftarrow$

step 5.4) Find $(y_{pred})_{train} \Rightarrow w^T x_i$

i

Eg: ~~if n=3~~

if $n=3$

& $i = 1, 2, 3$

$$(y_{pred})_1 = w_1^T x_1 + b_1$$

↳ append to list y_{pred}

~~$(y_{pred})_2$~~ =

step 6) Iterating over "n", now "i" takes 2 & 3;

$n=3$ ~~sum~~ $\left(\frac{\partial L}{\partial w}\right) = [(-2x_1)(w_1^T$ ~~x_1~~

6.1) $$\text{sum}\left(\frac{\partial L}{\partial w}\right) = (-2x_1)[y_1 - w_1^T x_1 - b_1] + (-2x_2)[y_2 - w_1^T x_2 - b_1] + (-2x_3)[y_3 - w_1^T x_3 - b_1]$$

ii) sum $(\partial L/\partial b)$

6.2)

$$w_2 = w_1 - \alpha \left[\frac{\text{sum of } \partial L/\partial w}{n}\right]$$

$$b_2 = b_1 - \alpha \left[\frac{\text{sum of } \partial L/\partial b}{n}\right]$$

step 6.3) Find $(y_{pred})_2 = W_2^T X_2 + b_2$

$\therefore i = 2$

2nd Training example. & W2

$\Rightarrow$ append $(y_{pred})_2$ to the list y_{pred}.

step 7) Repeat step 6 till $(y_{pred})_n = W^T X_n$

7.1) n times

So; we Find "W", "n" times

if $n = 3 \Rightarrow$ we find "W" 3 times.

step 7.2) Calculate MSE $\Rightarrow$ mse(y_pred, y_tr) $(w_0)_1 \; (w_1)_1 \; (w_n)_1$

step 8) epoch = 2 $\Rightarrow$ $W = W_{prev.epoch}$

$\Rightarrow$ use "W" calculated in previous Epoch.

$\Rightarrow$ Repeat steps 6, 7.

step 9) Find mse(y_pred, y_tr) $(w_0)_2 \; (w_1)_2 \; (w_2)_2 \; \cdots \; (w_n)_2$ ← separately calculated

with $(w_0)_2$ $(w_1)_2$

epoch number

0th example

step 10) returns "Final" W & b, y_pred.

step 11) Predict on Test dataset

Use W, b of (Train data calculated)

Test Data has 5 examples:

$y_1 = W^T \cdot X_1 + b$

$y_2 = W^T \cdot X_2 + b$

$y_3 = W^T \cdot X_3 + b$

$y_5 = W^T X_5 + b$

"W" is the final/last "W" of last Epoch