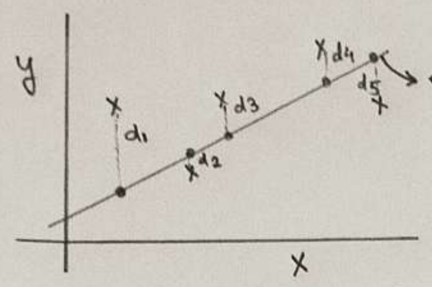
Jeet Banerjee

Mathematical Formulation
(Batch Gradient Descent + Simple Linear Regression)

Simple Linear Regression → single i/p (x)
single o/p (y)



→ equation of simple Linear Regression line:

$$\hat{y} = mx + b$$

m → slope, b → intercept

Total loss in prediction $= L = d_1^2 + d_2^2 + d_3^2 + \cdots + d_n^2$

$$\Rightarrow L = (y_1 - \hat{y}_1)^2 + (y_2 - \hat{y}_2)^2 + (y_3 - \hat{y}_3)^2 + \cdots + (y_n - \hat{y}_n)^2$$

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

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

$$\Rightarrow \boxed{L(m,b) = \sum_{i=1}^{n} (y_i - mx_i - b)^2} \quad -(1)$$

In gradient descent algorithm, we need to iteratively update (m,b) till its convergence. And, for that converged value of (m,b); the prediction loss would be minimum.

The contemporary expressions to iteratively update (m,b) in gradient descent:

$$\boxed{\begin{aligned} m_{updated} &= m_{initial} - \eta * slope(L)_m \\ b_{updated} &= b_{initial} - \eta * slope(L)_b \end{aligned}} \quad -(2) \Rightarrow$$

- selection of 'η' should ensure optimal change in (m,b)
- for -ve slope(L), in each iteration, (m,b) are increasing, and vice-versa

↳ where η = learning rate

slope$(L)_m$ = slope of $L(m,b)$ w.r.t m

slope$(L)_b$ = slope of $L(m,b)$ w.r.t b

Now, $slope(L)_m = \frac{\partial L}{\partial m} = \sum_{i=1}^{n} 2(y_i - mx_i - b)(-x_i)$

$$\Rightarrow \boxed{slope(L)_m = -2\sum_{i=1}^{n} (y_i - mx_i - b)x_i} \quad -(3)$$

Jeet Banerjee

Mathematical Formulation
(Batch Gradient Descent + Simple Linear Regression)

Similarly, $\text{slope}(L)_b = \frac{\partial L}{\partial b} = \sum_{i=1}^{n} 2(y_i - mx_i - b)(-1)$

$$\Rightarrow \boxed{\text{Slope}(L)_b = -2\sum_{i=1}^{n}(y_i - mx_i - b)} \quad -(4)$$

By substituting eq(3) and eq.(4) in eq.(2), we can iteratively update the values of (m,b) till its convergence, and the selection of no. of iterations (epochs) is depending on the required steps for the converged (m,b) with a selected learning rate (η).

Point to Note:

Say, at 1st iteration, to upate 'm' or 'b', we need to partially derivative loss function (L) 'n' no. of times, i.e., no. of rows present in the data.

Hence, for Batch Gradient Descent with Simple Linear Regression:

if no. of iterations = N

no. of rows in the data = n

the number of times, we need to partially derivative the loss function is $2n * N$, which is computationally intensive for large data size.