Previous one for 2D

Types of Gradient Descent

for 3D ← Batch GD | Stochastic GD | minibatch GD

i) Batch Gradient Descent:

→ uses the Entire datasets to complete the gradient before updating the parameters

advantage

→ very accurate & stable updates

→ Guaranteed to move smoothly toward minimum

disadvantage

→ very slow for large dataset

→ Requires a lot of memory

Mathematical Formulation:-

→ Consider 3 cols dataset ⇒

x_1 cgpa	x_2 iq	y lpa
8.1	93	3.2
7.5	95	3.5

$$\boxed{\underset{(lpa)}{y} = \beta_0 + \beta_1 \underset{(cgpa)}{x_1} + \beta_2 \underset{(iq)}{x_2}}$$

i) Random ~~variables~~ values ⇒ $\beta_0 = 0, \beta_1, \beta_2 = 1$

ii) epoch = 100, lr = 0.1

$$\begin{cases} \beta_0 = \beta_0 - \eta \text{ slope} \\ \beta_1 = \beta_1 - \eta \text{ slope} \\ \beta_2 = \beta_2 - \eta \text{ slope} \end{cases}$$

→ so here Loss function is depending on $\{\beta_0, \beta_1, \beta_2\}$ these three

$$\left(\frac{\partial L}{\partial \beta_0}\right), \left(\frac{\partial L}{\partial \beta_1}\right), \left(\frac{\partial L}{\partial \beta_2}\right)$$

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

$$= \frac{1}{2}\left[(y_1-\hat{y}_1)^2 + (y_2-\hat{y}_2)^2\right]$$

$$\frac{\partial L}{\partial \beta_0} = \frac{1}{2}\left[(y_1 - \underbrace{\beta_0 - \beta_1 x_{11} - \beta_2 x_{12}}_{\hat{y}_i})^2 + (y_2 - \beta_0 - \beta_1 x_{21} - \beta_2 x_{22})^2\right]$$

$$= \frac{1}{2}\left[2(y_1 - \hat{y})(-1) + 2(y_2 - \hat{y})(-1)\right]$$

$$= \frac{-2}{2}\left[(y_1 - \hat{y}_2) + (y_2 - \hat{y}_2)^2\right]$$

→ for n

$$\frac{\partial L}{\partial \beta_0} = \frac{-2}{n}\left[(y_1 - \hat{y}_2) + (y_2 - \hat{y}_2) + \dots (y_n - \hat{y}_n)\right] \Rightarrow \boxed{\frac{\partial L}{\partial \beta_0} = \frac{-2}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)}$$

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

$$L = \frac{1}{2} \left[(y_1 - \hat{y}_1)^2 + (y_2 - \hat{y}_2)^2 \right]$$

$$L = \frac{1}{2} \left[\Big(y_1 - \underbrace{\beta_0 + \beta_1 x_{11} + \beta_2 x_{12}}_{\hat{y}_1} \Big)^2 + \Big(y_2 - \underbrace{\beta_0 - \beta_1 x_{21} - \beta_2 x_{22}}_{\hat{y}_2}$$

$$\frac{\partial L}{\partial \beta_1} = \frac{1}{2} \left[2(y_1 - \hat{y}_1)(-x_{11}) + 2(y_2 - \hat{y}_2)(-x_{21}) \right]$$

$$\frac{\partial L}{\partial \beta_1} = \frac{-2}{n} \left[(y_1 - \hat{y}_1)(+x_{11}) + (y_2 - \hat{y}_2)(x_{21}) + \dots\dots + (y_n - \hat{y}_n)(x \right.$$

→ for n

$$\boxed{\frac{\partial L}{\partial \beta_1} = \frac{-2}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)(x_{i1})}$$

So here x_{i1} representing columns

$$\begin{bmatrix} x_{11} \\ x_{21} \\ x_{31} \\ \vdots \\ x_{n1} \end{bmatrix}$$

→ x_{i1} ⟹ 1st column of data

→ β_1 ⟹ weigh or values of 1col

$$\boxed{\frac{\partial L}{\partial \beta_2} = \frac{-2}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)(x_{i2})}$$

x_{i2} → 2nd column of dataset

→ so for m cols $\{\beta_0 - \beta_m\}$

$$\boxed{\frac{\partial L}{\partial \beta_m} = \frac{-2}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)(x_{im})}$$

x_{im} → m^{th} column of dataset