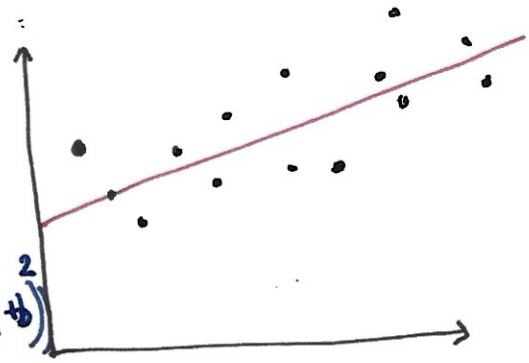
Linear Regression from scratch (using Python)

Estimation $\Rightarrow \hat{y} = wx + b$

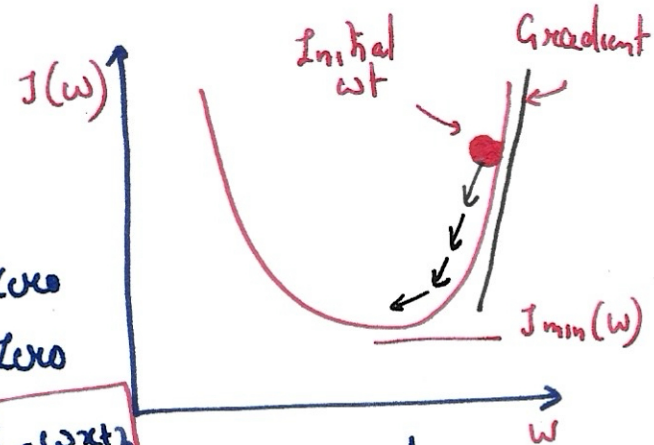


Mean Square error

$$MSE = J(w,b) = \frac{1}{N}\sum_{i=1}^{n}(y_i - (wx_i+b))^2$$

$$J'(m,b) = \begin{bmatrix} df/dw \\ df/db \end{bmatrix} = \begin{bmatrix} \frac{1}{N}\sum -2x_i\,(y_i - (wx_i+b)) \\ \frac{1}{N}\sum -2(y_i - (wx_i+b)) \end{bmatrix}$$

Gradient Descent



During training

- → Initialize weight as zero
- → Initialize bias as zero
- → Predict result using $\hat{y} = wx + b$
- → Calculate error
- → Use gradient descent to figure out new weight & bias values
- → Repeat n times
- → Test using $\hat{y} = wx + b$

$w = w - \alpha . dw$

$b = b - \alpha . db$

updating Parameters

$$dw = \frac{dJ}{dw} = \frac{1}{N}\sum_{i=1}^{n} -2x_i(y_i - (wx_i + b)) = \frac{1}{N}\sum_{i=1}^{n} -2x_i(y_i - \hat{y})$$

$\hat{y}$

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

Similarly for

$$db = \frac{dJ}{db} = \frac{1}{N}\sum_{i=1}^{n} 2(\hat{y} - y_i)$$

Doing it efficiently i.e. all calculations for all datapoint in one time

$\hat{y} = wx + b \longleftrightarrow$ Vectorized $= [wX + b]$

$X = [x_1\ x_2 \dots x_n]$

$$X^T = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

Vectorized $= [wx_1 + b \quad wx_2 + b \dots wx_n + b]$