09/10/22

Linear Regression Algorithm :

$h_\theta(x) = \theta_0 + \theta_1 x$

$h_\theta(x) : \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \cdots \quad \theta_{n \times n}$

Convergence Algorithm : repeat until convergence

{

$\theta_j := \theta_j - \alpha \left[\frac{\partial}{\partial \theta_j} \{ J(\theta_j) \} \right]$ → derivative, slope

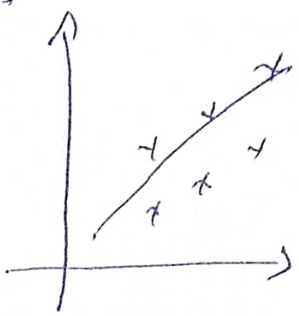
}

here we are doing the summation

Cost function → MSE

$$J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$$

predicted point, truth point



[loss function Vs cost function]

every observation.

loss function :- $\left(h_\theta(x^{(i)}) - y^{(i)} \right)^2 = \left(\hat{y}_{(i)} - y_{(i)} \right)^2$

$\hat{y}_{(i)}$ → Predicted value; $y_{(i)}$ → Actual value

Convergence : Repeat until convergence

$$\{ \quad \theta_j = \theta_j - \alpha \left[\frac{\partial}{\partial \theta_j} J(\theta_j) \right] \quad \text{learning curve}$$

$$\frac{\partial}{\partial x}(x^2) = 2x$$

$$\frac{\partial}{\partial x}(x)^n = n x^{n-1} \quad (1)$$

$$\frac{\partial}{\partial x}(x+1)^2 = 2(x+1) \times (1+0) = 2(x+1)$$

$$\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \frac{\partial}{\partial \theta_0} \left[\frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2 \right]$$

If $j=0$: $h_\theta(x) = \theta_0 + \theta_1 x$

$$= \frac{\partial}{\partial \theta_0} \left[\frac{1}{2m} \sum_{i=1}^{m} \left((\theta_0 + \theta_1 x)^{(i)} - y^{(i)} \right)^2 \right]$$

$$= \frac{\partial}{\partial x} \frac{1}{2m} \left[(1+x) - y^{(i)} \right]^2 \quad \rightarrow \text{let us consider: } (1+x) - \theta_0 + \theta_1 x.$$

$$= \frac{2}{2m} \left[(1+x) - y^{(i)} \right] * 1$$

$$= \frac{1}{m} \sum_{i=1}^{m} \left[\theta_0 + \theta_1 x - y^{(i)} \right] * 1 \qquad \frac{\partial}{\partial \theta_0}(\theta_0 + \theta_1 x_1) \Rightarrow 1.$$

If $j=1$:

$$\frac{\partial}{\partial \theta_1} \left[\frac{1}{2m} \sum_{i=1}^{m} \left(\theta_0 + \theta_1 x^{(i)} - y^{(i)} \right)^2 \right]$$

$$= \frac{2}{2m} \left((\theta_0 + \theta_1 x)^{(i)} - y^{(i)} \right) * [x] \quad \rightarrow \text{learning curve} \rightarrow \text{speed of convergence}$$

Repeat until convergence

$$\{ \quad \theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{n} \left(h_\theta(x^{(i)}) - y^{(i)} \right)$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{n} \left(h_\theta(x^{(i)}) - y^{(i)} \right) x_1^{(i)}$$

$$\}$$

Cost functions :-

1. MSE 2. MAE 3. RMSE (Root Mean Squared Error) 4. Hubel loss.

MSE = Mean Squared Error

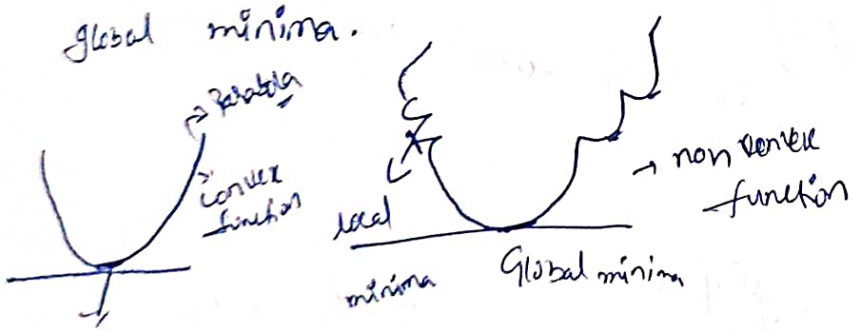
$$MSE = \sum_{i=1}^{n} \frac{(y-\hat{y})^2}{n}$$

$\hat{y} = \theta_0 + \theta_1 x$ → predicted value

quadratic equation → $a^2 + b^2 - cab$

Advantages

1. This equation is differentiable
. This also has one global minima.



. There is no local minima.

Exp	Salary (lakhs) INR → dependent
–	–
–	–
–	–
–	–

$(y-\hat{y})^2$ → squared → Error → Penalized

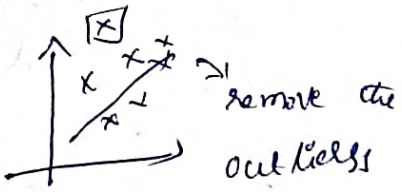
$(\text{Salary} - \text{Predicted Salary})^2 \times (\text{lakh})^2$

∴

MSE ⇒ ↓↓↓

Disadvantages

. This equation is not robust to outliers



. Penalizing the error ⇓ increasing

2. Mean Absolute Error (MAE)

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y-\hat{y}|$$

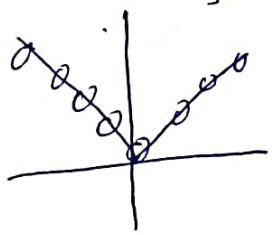
Advantage

1. Robust to outliers, here we are not squaring the value
2. It will also be in the same unit.

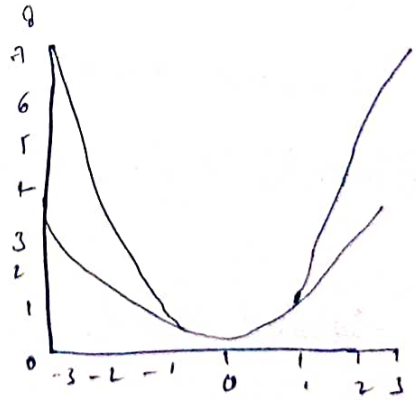
Disadvantage.

- Convergence usually takes more time - optimization is a complex task

Sub gradient concept



- Time consuming.
- There may be a local minima.



Huber loss

Combination of MSE & MAE

If there is outlier use MAE

otherwise MSE

$$L_\delta(a) = \frac{1}{2}a^2 \quad \text{for } |a| \le \delta$$
$$\delta.(|a| - \frac{1}{2}\delta) \quad \text{otherwise}$$

$$L_\delta(y, f(x)) = \frac{1}{2}(y-f(x))^2 \quad \text{for } |y-f(x)| \le \delta.$$
$$\delta.(|y-f(x)| - \frac{1}{2}\delta).$$

Advantage :-

- Used in robust regression.
- Outliers are handled properly.
- Local minima situation is handled here

Disadvantages :-

a. In order to maximize model accuracy, the hyperparameter δ will also need to be optimized which increases the training requirements.

b. It is complex.

RMSE

Unit-

Outliers

Differentiable

$$\sqrt{MSE} = \sqrt{\frac{\sum_{i=1}^{N} \|y_i - \hat{y}_i\|^2}{N}}$$

Advantages

Disadvantages :

- It is helpful to have a single number to judge a model's performance, whether it be during training, cross-validation, or monitoring after deployment.

Performance Metrics

1. R squared error
2. Adjusted R squared error

1. R Squared error

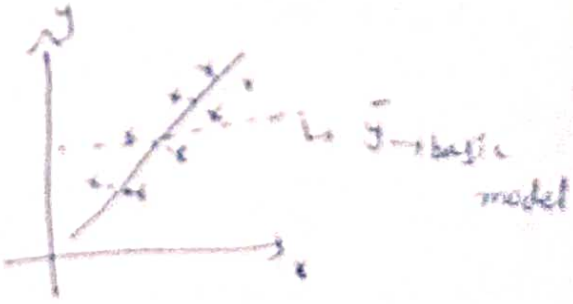
$$R\ Squared = 1 - \frac{SS_{Res}}{SS_{Total}}$$

SS_{Res} : Sum of Square Residuals

SS_{Tot} Sum of Square average

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

$\bar{y}$ = Avg of y



R Squared = $1 - \left\{ \frac{\text{Small number}}{\text{biggest num}} \right\}$ ⇒ Small number

≤ 1 .

R^2 = -ve .

i.e 0.85 ⇒ 85% accurate

0.75 ⇒ 75% ⇒ accurate

Performance of the model that you have created

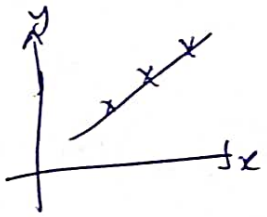
Can R Squared value be -ve?

⇓

Model is very bad.

$\bar{y}$ = Avg. of y

$R^2 = 1$



2. Adjusted R^2 squared :.

size of the house, city, location, No. of bedrooms, Gender, Salary, Price

R^2 65%.

R^2 75%.

R^2 88%. } - No.

R^2 90%.

Adjusted R^2 = 63% P = 1

" = 73% P = 2

Adjusted R^2 = 85% P = N

$$\text{Adjusted } R^2 = 1 - \frac{(1-R^2)(N-1)}{N-P-1}$$

N = No. of data Points

P = No. of Independent feature

Train :- very good accuracy [90%] [Low Bias]

Test :- Bad accuracy (50%) [High Variance]

↓

Overfitting

→ HyperParameter tuning.

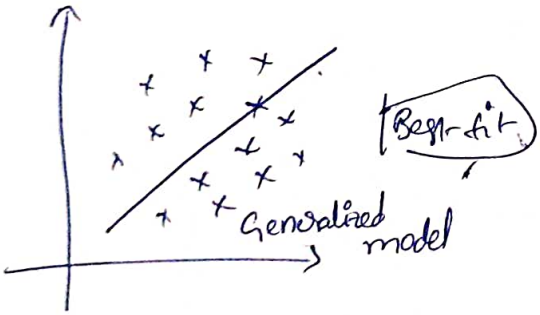
→ increase the data

Train = Model accuracy is low [High bias]

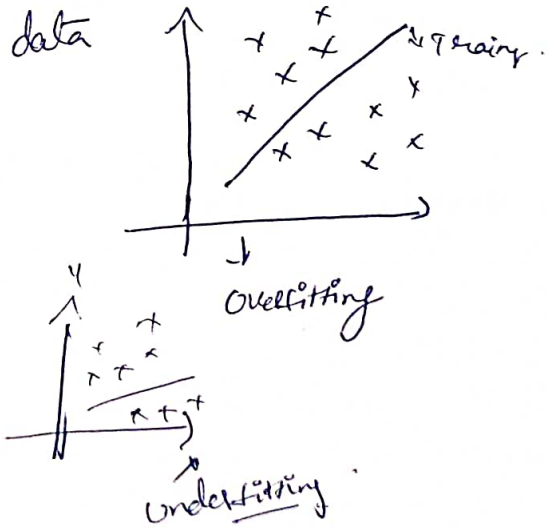
Test = Model ↓ low / high [low or high variance]

↓

model is Underfitting



Training data



- Linear Regression :-
  1. Calculation = $y = mx + c$
  2. loss [MAE, MSE, RMSE, Huber loss]
  3. Optimizing (m, c) → minimize the loss (gradient decent)