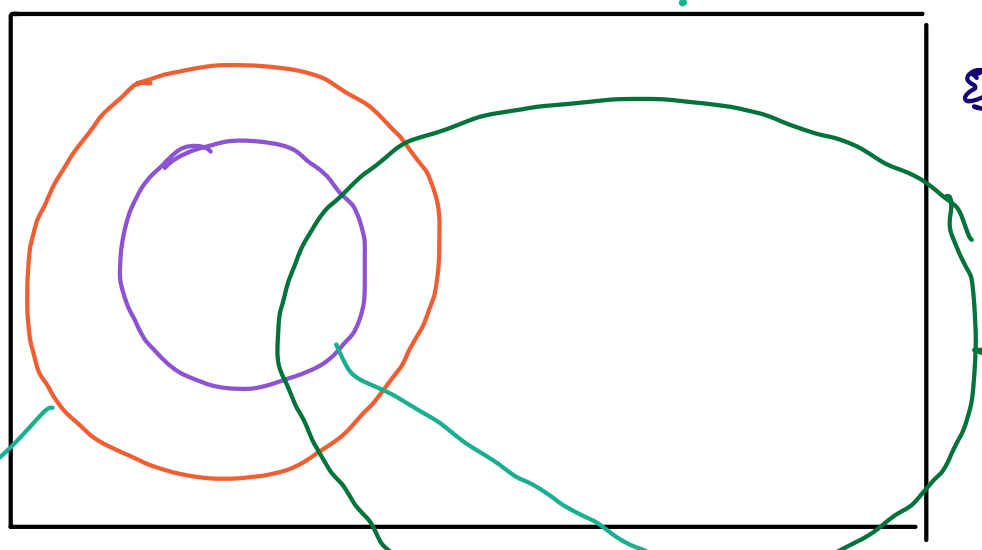
Universe of Data science

1. Artificial Intelligence
2. Machine Learning
3. Deep Learning
4. Data science



AI - To create an application which can perform its own tasks without any Human intervention

Eg: netflix Recommanda tor system

Data science
↓
overlaps everything

ML
↓
It provides stats tools to analyse, visualize, predict, forecast the data

Deep learning
↓
It was developed to mimic How Humans learn
↓
multi layer neural network

3 Types of Machine Learning:

AI
↓
- Supervised ML
- unsupervised ML
- Reinforcement Learning

Supervised ML Techniove → Regression / classification

→ Task is predict House price

Dataset

Size of House (Independent features)	# of Rooms (Independent features)	Price (Dependent (or) output Feature)
5000	5	450 K
6000	6	500 K
-	-	-
-	-	-
-	-	-

This value keeps on changing Based on Input (Independent Features)

(i) Dependent or o/p feature

(ii) if output is continuous

↓

Regression problem statement

(iii) if output is discrete / categorical

↓

classification problem statement

Output / Target feature is continuous value Here

so it is Regression problem statement

classification Example

No. of study Hours (I/p (or) Independent Feature)	No. of Play Hours	o/p (or) Dependent Feature
7	3	Pass
2	6	Fail

Pass / Fail

↓

categorical / Discrete

so ↓ classification

Binary classification (2 classes)

multi class classification (>2 classes)

(2) Unsupervised ML Technique :

Eg: Customer Segmentation

→ No specific output / dependent Feature

→ No Labelling data

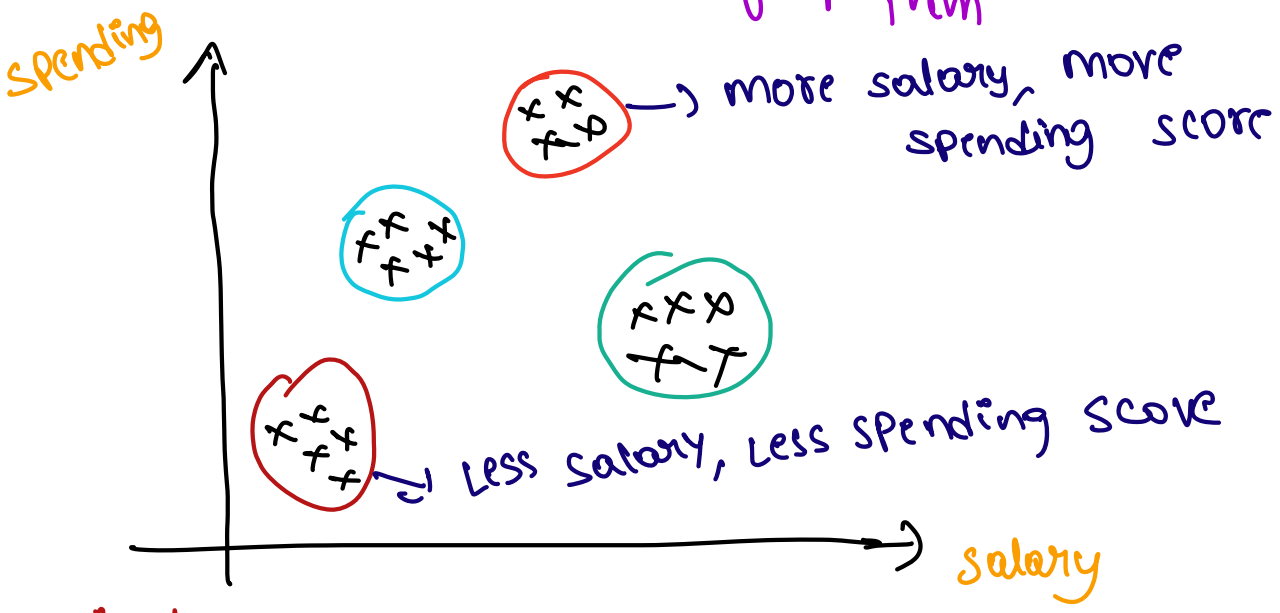
salary	spending score (1-10)
20000	9
45000	2
⋮	⋮

Ecommerce company

↓

Email with some discount coupons to specific spending score

→ Here we create clusters and grap them



Supervised ML Algorithms

1. Linear Regression
2. Ridge & Lasso Regression → Regression Algorithms
3. Elastic Net
4. Logistic Regression → classification Algorithm
5. Decision Tree → Both Regression / classification

6. Random Forest
7. Ada Boost
8. Xg Boost

} Both Regression / classification

Unsupervised ML:

1. K means
2. Hierarichal Mean
3. DB scan

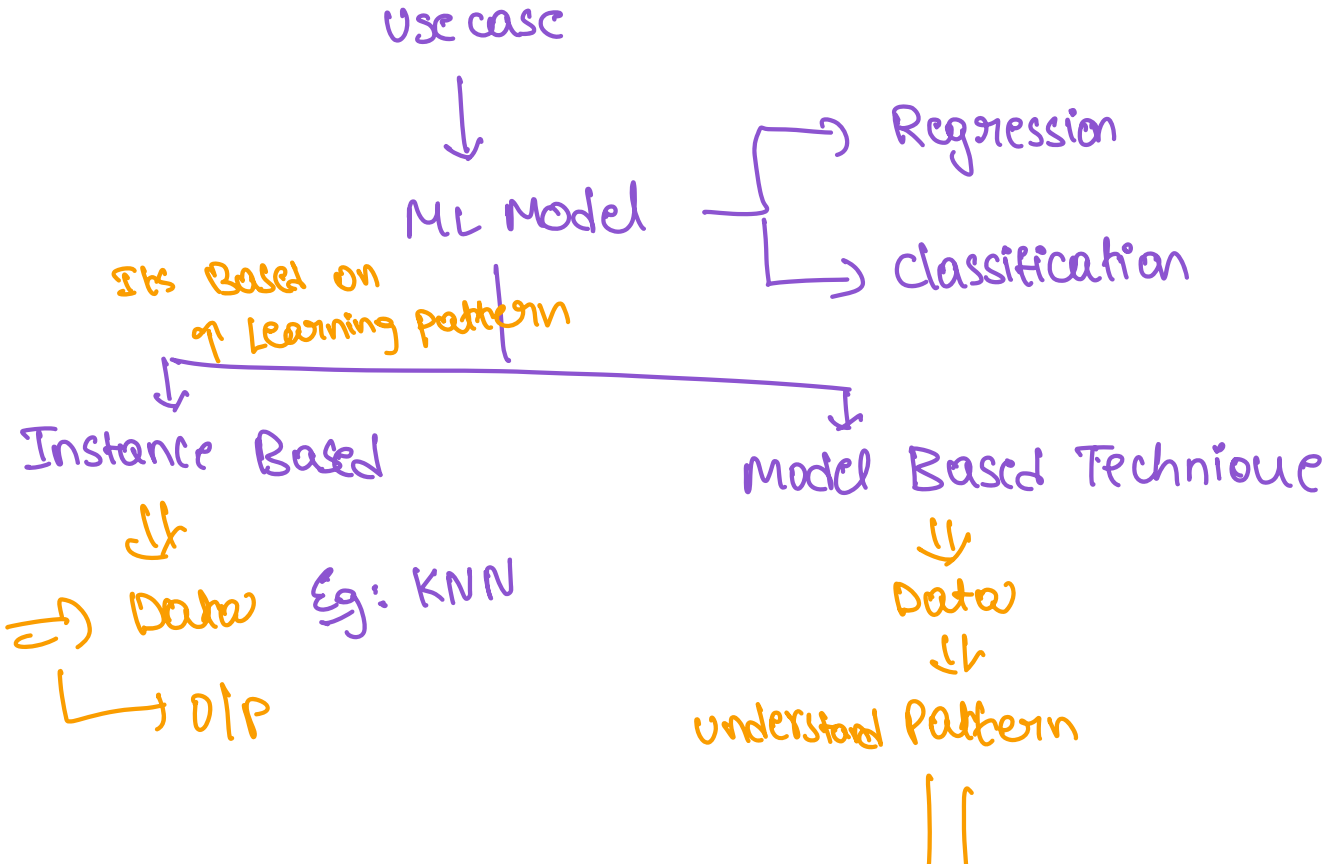
} clustering Algorithms

Reinforcement Learning :

→ It will learn itself by Recoarding | penality

Eg: Baby (OO) Kids learning

Instance Based Learning Vs Model Based learning



It will create Generalization method

where Model can predict correctly for new data point

Model Based Learning

Instance Based Learning

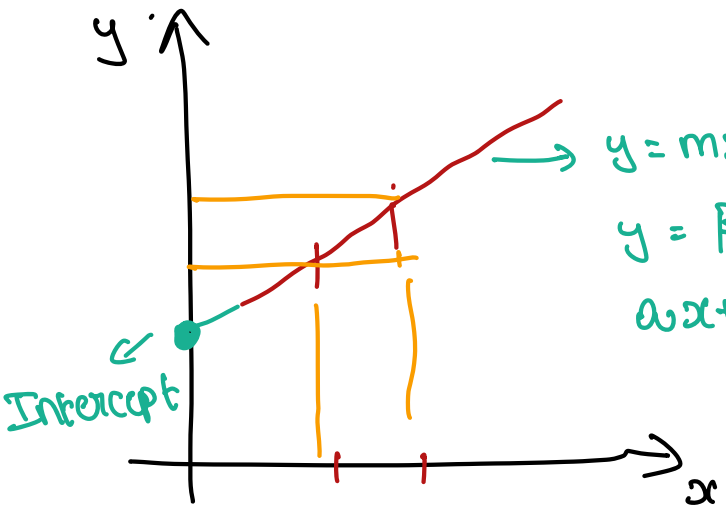
Usual/Conventional Machine Learning	Instance Based Learning
Prepare the data for model training	Prepare the data for model training. No difference here
Train model from training data to estimate model parameters i.e. discover patterns	Do not train model. Pattern discovery postponed until scoring query received
Store the model in suitable form (Pickle format)	There is no model to store
Generalize the rules in form of model, even before scoring instance is seen (Decision Boundary)	No generalization before scoring. Only generalize for each scoring instance individually as and when seen
Predict for unseen scoring instance using model	Predict for unseen scoring instance using training data directly
Can throw away input/training data after model training	Input/training data must be kept since each query uses part or full set of training observations
Requires a known model form	May not have explicit model form
Storing models generally requires less storage	Storing training data generally requires more storage
Scoring for new instance is generally fast	Storing for new instance may be slow

Learning → memorizing → Instance Based

Learning → Generalizing → Model Based

Linear Algebra :

Equation of Line, 3d Plane and Hyperplane (n-Dimension)



$y = mx + c$

$y = \beta_0 + \beta_1 x$

$ax + by + c = 0$

(1) $y = mx + c$ → Above line can be represented by this Equation

x, y → co-ordinates

m = Slope

↓

In the unit movement in the x-axis, what is the movement in the y-axis

c = Intercept

↓

if $x = 0$ in the Equation $y = mx + c \Rightarrow y = mx0 + c$ $\boxed{y = c}$

↓

where the straight line meets y-axis is called Intercept

(2) $ax + by + c = 0$

$by + c = -ax$

$by = -ax - c$

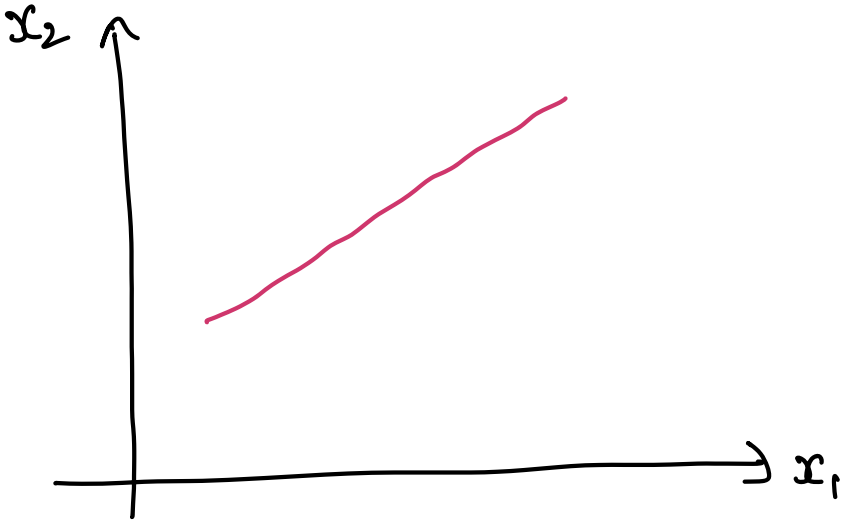
$$y = \left(\frac{-a}{b}\right)x - \frac{c}{b}$$

(m = $\frac{-a}{b}$, c = $\frac{c}{b}$)

↓

in the form of $\boxed{y = mx + c}$

For Eg we have more than one dimension :



Above line can be written as :

$$w_1x_1 + w_2x_2 + b = 0$$

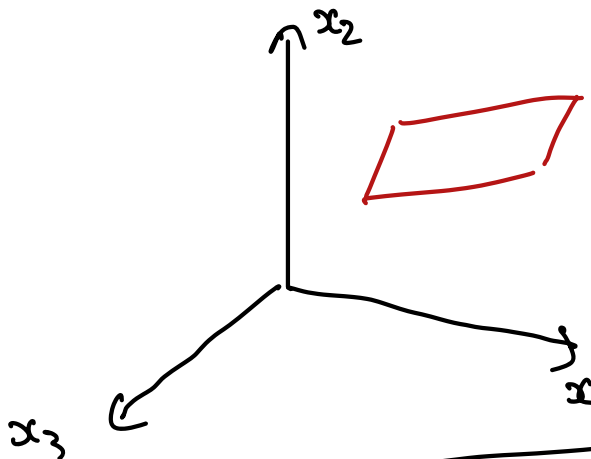
$w_1 = w$
$w_2 = b$
$b = c$

This can also be written as :

$$w^Tx + b = 0$$

This is nothing but a Equation of a Straight line

For Eg what if we have 3 dimensions :



Here we don't draw Straight line
↓
we draw plane

$$w_1x_1 + w_2x_2 + w_3x_3 + b = 0$$

→ Above Equation can be written as:

$$w^T x + b = 0$$

$$w = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} \quad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

→ If we have to do the dot product of w and x & we have to transpose one matrix

→ When we do matrix multiplication with 'w' and 'x', one of the matrix we have to Transpose it

Why we have to Transpose it?

(R1) In matrix multiplication, number of columns in the first matrix must equal the number of rows in the second matrix

(R2) Dot product Interpretation:

→ matrix multiplication involves taking the dot product of rows and columns of the matrices

→ Transposing one matrix ensures that the dot product is taken between corresponding elements of the matrices, facilitating correct multiplication

What About n-dimensional Plane:

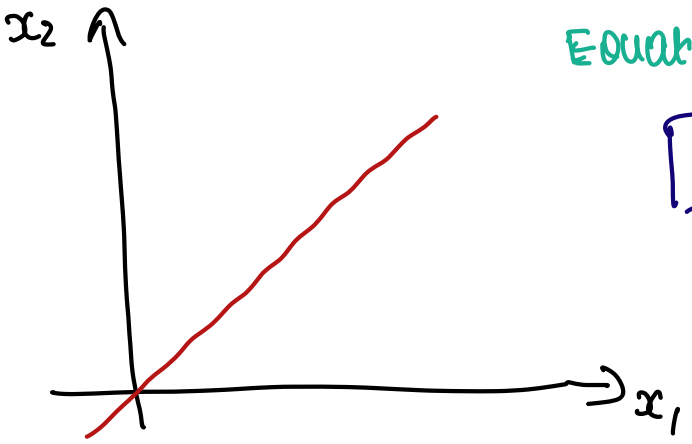
$$w_1 x_1 + w_2 x_2 + w_3 x_3 + \ldots + w_n x_n + b = 0$$

$$\Downarrow$$

$$w^T x + b = 0$$

For Eg:

What if the straight line passes through origin ?



Equation of a straight line:

$$w_1x_1 + w_2x_2 + b = 0$$

↓

If straight line passes through origin

↓

means Intercept, $b = 0$

↓

$$w_1x_1 + w_2x_2 = 0$$

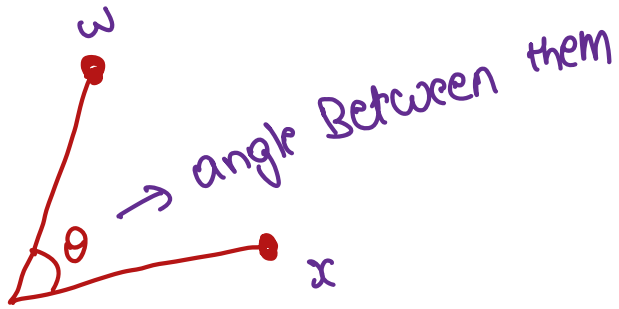
↓

$$w^Tx = 0$$

Equation of a straight line passing through origin

Equation of a plane : $\Pi_n = w^Tx = 0$

$$\begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_n \end{bmatrix} \quad \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix}$$



$$w \cdot x = w^Tx = \|w\| \|x\| \cos\theta = 0$$

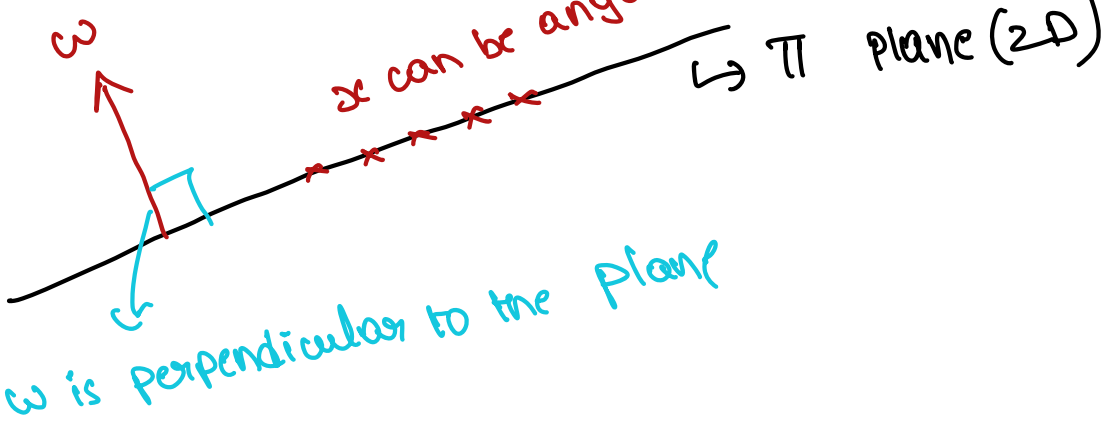
↓

magnitude of w * magnitude of x * Angle Between them

When $\cos\theta = 0$?

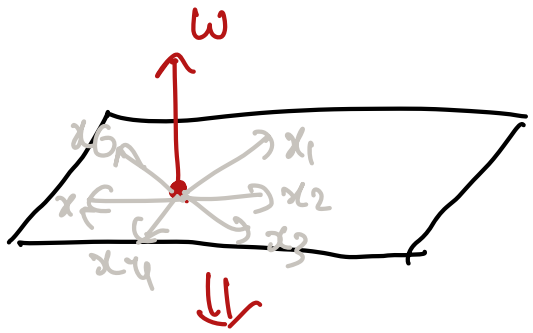
if $\theta = 90$, $\cos 90 = 0$

In 2D plane :



In 3D plane :

'w' is always perpendicular to the plane



$$w^T x = 0$$

Intercept, $b = 0$

Machine Learning Algorithms :

(1) Simple Linear Regression :

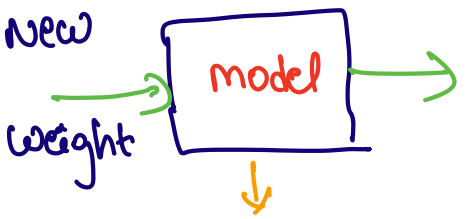
Supervised ML → Regression

For Eg :

Dataset

Weight (Input / Independent Feature)	Height (Output / Dependent Feature)
74	170 cm
80	180 cm
75	175.5cm
⋮	⋮

Simple Linear Regression



We should train this model with Dataset

↓

So that model can predict Height for New weight

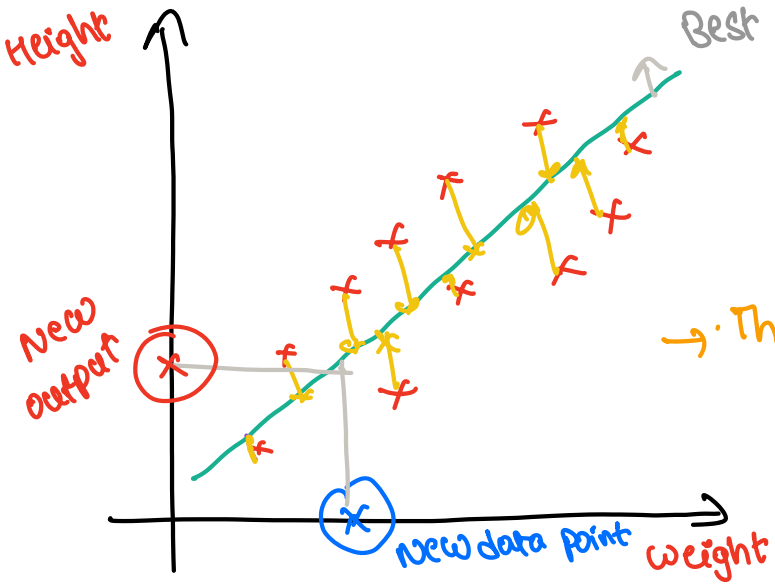
↓

which is not in Training set

Simple Linear Regression → one Input Feature

Multiple Linear Regression → more than one Input Feature

What are we trying to do in Geometric way :



x → Input and output data point (or) observations

→ The summation of all distances Should be minimum

↓

That will be the Best Fit line

What are we Trying to do in Linear Regression ?

↓

Trying to fit a Best line among all the data points

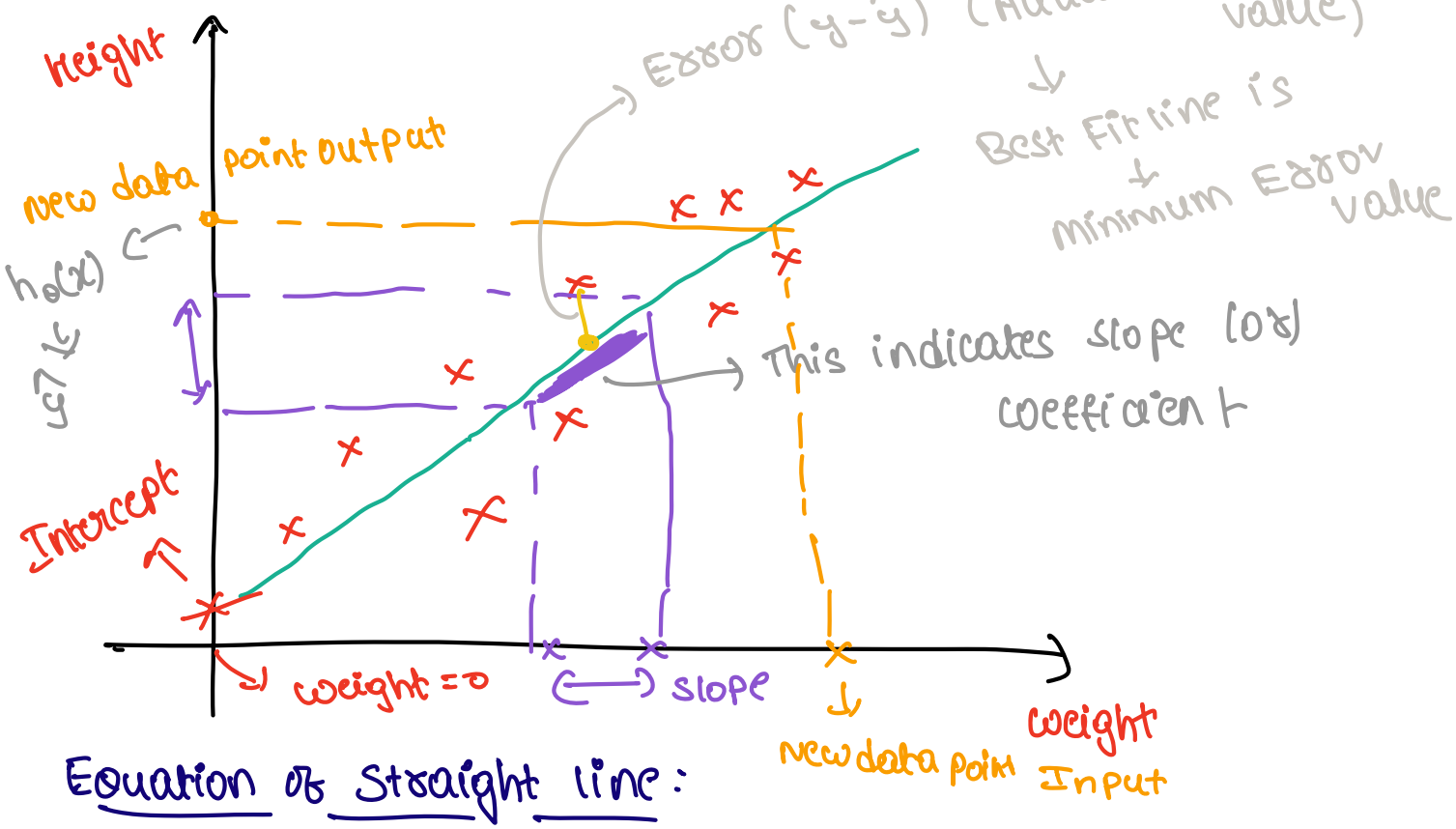
↓

This Best Fit line will help us predictions for the new weight

↓

All the distance Between Actual value and Predicted value is called Error

Mathematical Equation of Best Fit Line :



Equation of Straight line :

$y = mx + c$

$y = \beta_0 + \beta_1 x$

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

All are same

But MOSTly we will use this Equation

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

↓

x → independent (or) input Feature

↓

It is weight in our case

θ_0 → Intercept

if $x = 0$ → $h_\theta(x) = \theta_0$

↓

when $x = 0$, i.e, weight $= 0$

↓

where our best fit line meet y-axis

θ_1 = Slope (or) coefficient

↓

For a unit movement on x-axis, what is the movement w.r.to y-axis

For n-Features :

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \cdots + \theta_n$$

Predicted output → $h_\theta(x)$ → $\hat{y}$ (y hat)

How we are going to get Best Fit line ?

① Draw one simple straight

② Then we are going to change the angle and rotate it to find Best Fit line

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

θ_0 = Intercept

θ_1 = Slope (or) coefficient

→ By changing Intercept (θ_0) and Slope (θ_1) we are going to rotate the straight fit line to find the Best Fit line.

Cost Function :

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

$h_\theta(x)^{(i)}$ → Predicted point

$y^{(i)}$ → Actual (or) True point

$h_\theta(x) - y$ → Gives Error value

↓

The reason why we are squaring ?

↓

Because the cost Function is "Mean squared Error"

Final Aim, what we need to solve

→ minimize cost Function $J(\theta_0, \theta_1)$

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

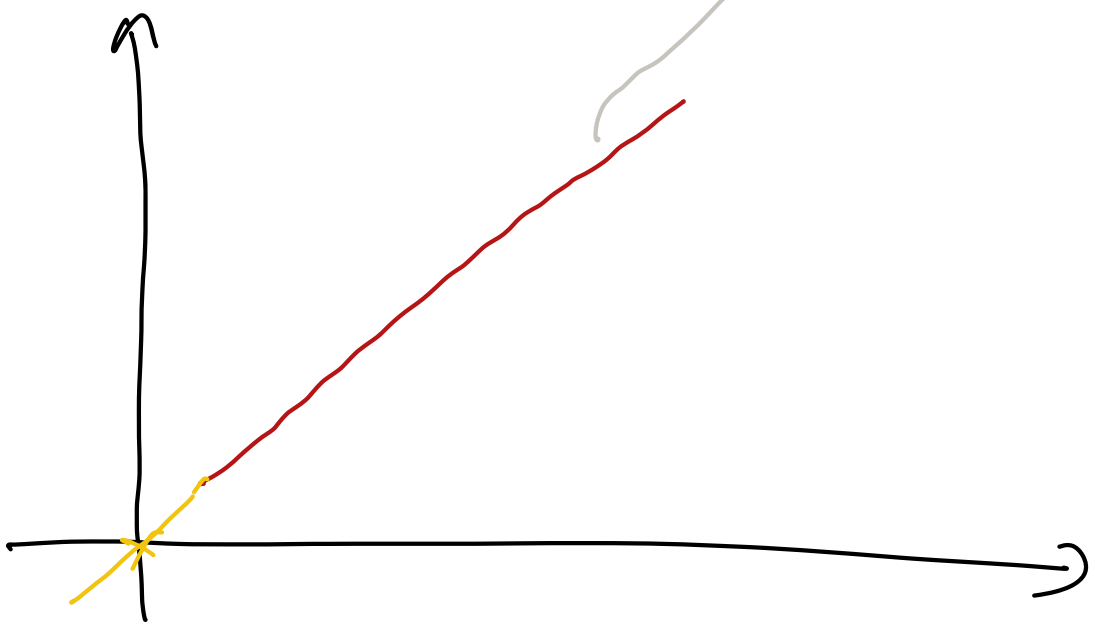
↓

This value should be minimal by changing (θ_0, θ_1)

How we are going to do this ?

Equation of straight line :

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



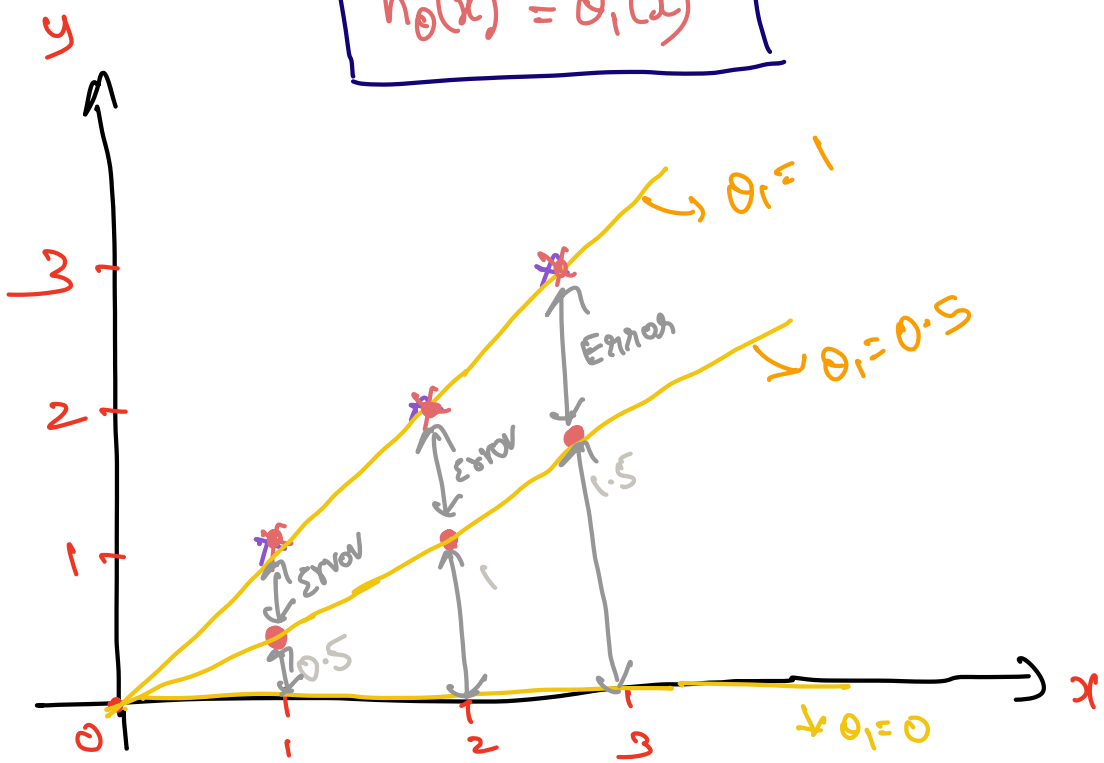
→ Assume our straight line passing through origin

$$\theta_0 = 0$$

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

Dataset

x	y
1	1
2	2
3	3



X → Actual points

● → Predicted points

line Equation : $h_\theta(x) = \theta_1 x$

Let $\theta_1 = 1$ {Slope} → we will keep on changing the slope

↓

$h_\theta(x) = 1$, when $x = 1$

$h_\theta(x) = 2$ $x = 2$

$h_\theta(x) = 3$ $x = 3$

→ $\theta_0 = 0$ (since line passing through origin, Intercept $\theta_0 = 0$)

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

↓

m = no. of datapoints, in our case $m = 3$

$$= \frac{1}{2\times 3}\left[(1-1)^2 + (2-2)^2 + (3-3)^2\right] = 0$$

↓

Cost Function is Zero

↓

If you see all predicted value & Actual values same in Above graph

Let $\theta_1 = 0.5$ $(h_\theta(x) = \theta_1 x)$

↓

$h_\theta(x) = 0.5$ if $x = 1$ → $0.5 \times 1 = 0.5$

$h_\theta(x) = 1$ $x = 2$ → $0.5 \times 2 = 1$

$h_\theta(x) = 1.5$ $x = 3$ → $0.5 \times 3 = 1.5$

Calculate cost Function $J(\theta_0, \theta_1)$

$\theta_0 = 0$ Assumption, Line passing through origin

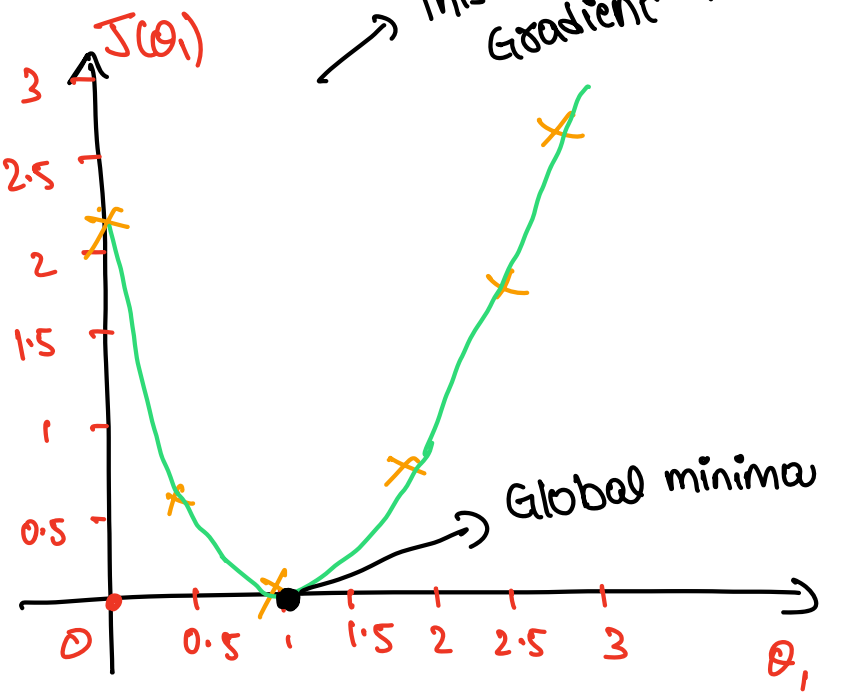
$$J(\theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left[h_\theta(x)^{(i)} - y^{(i)} \right]^2$$

$m = 3$

$$J(\theta_1) = \frac{1}{2 \times 3} \left[(0.5-1)^2 + (1-2)^2 + (1.5-3)^2 \right]$$

$J(\theta_1) = \approx 0.58$

when $\theta_1 = 0.5$



when $\theta_1 = 1$, $J(\theta_1) = 0$

when $\theta_1 = 0.5$, $J(\theta_1) = 0.58$

when $\theta_1 = 0$, $J(\theta_1) = 2.3$

→ when $\theta_1 = 0$

$$J(\theta_1) = \frac{1}{2 \times 3} \left[(0-1)^2 + (0-2)^2 + (0-3)^2 \right]$$

≈ 2.3 → Error quite High

If we keep on changing Above θ_1 values

↓

we will get Above curve

→ we have to reach Global minima point By changing (θ_0, θ_1)

ie, we are reducing value of cost Function

↓

→ we cannot keep on changing 'θ' values practically

↓

so, we will be needing convergence algorithm

↓

we have to select one 'θ_0' value and we have to find a mechanism to select suitable 'θ_1' values using convergence algorithm

Convergence Algorithm ;

→ optimize the changes of 'θ_1' value

↓

which is slope

→ Repeat until convergence (means reach Global minimum)

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_j)$$

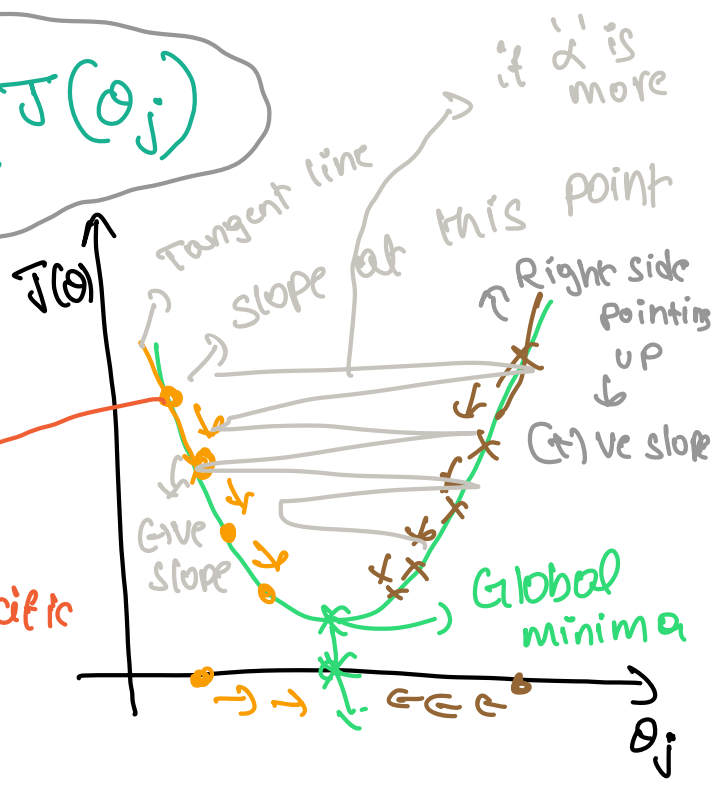
↓

change 'θ' value much more efficiently

derivative at this specific Point

↓

derivative means slope



How to find Derivative

→ Derivative is nothing but we are actually calculating slope

↓

for calculating Derivative, we have to draw a tangent line at that point

↓

we have to find whether it is positive slope (or) negative slope

↓

so we will know whether we can Increase (or) decrease 'θ_j' value

→ Just see the "right part" of Tangent line drawn at that point

↓

If it is facing downwards

↓

we will be calling it as negative slope

→ $\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_j)$ → From Above ↓ slope is (-ve)

$\theta_j = \theta_j - \alpha(-\text{ve value})$ $\quad (- \times - = +)$

$\theta_j = \theta_j + (+\text{ve})$ → so we are increasing θ_j value

→ How much θ_j value increase ?

↓

we don't know → This value will be decided by convergence algorithm by iterating

→ when slope is (+)ve

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_j)$$ → (+)ve slop

$$\theta_j = \theta_j - \alpha (+\text{ve slope value})$$

↓

θ_j will decrease

↓

move towards Global Minima

α : Learning rate ($\alpha = 0.001$) → start with small value

→ Learning rate controls the speed at which convergence should happen

$\alpha \rightarrow$ Small → model will take more time to converge

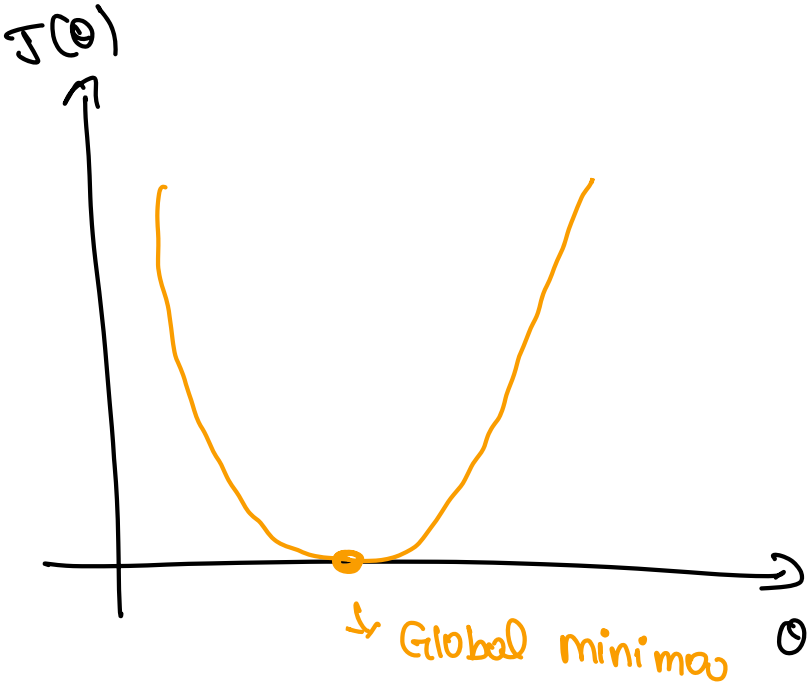
$\alpha \rightarrow$ Large → model will converge Fastly

↓

It may move very Fastly randomly and may not converge

Final Conclusion :

Gradient Descent :



Our Assumption :

→ our Best Fit line passes through origin

i.e, $\theta_0 = 0$

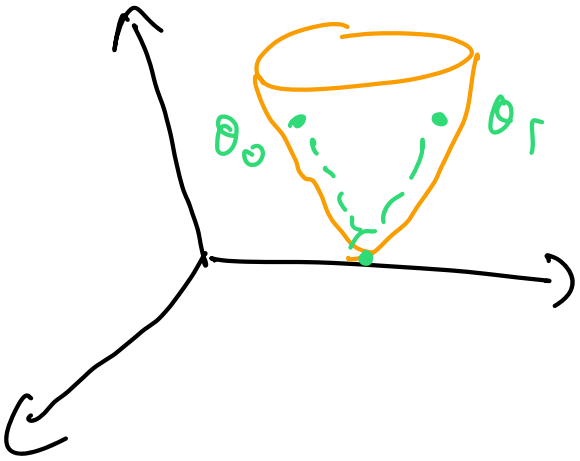
→ we only focussed on θ_1 = slope → changeable

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

↓

This is done to visualize Gradient descent in '2D' Diagram

For Eg: $\theta_0 \neq 0$ (i.e, our line not passing through origin)



Gradient Descent in '3D'

convergence Algorithm :

repeat until convergence → Including θ_0

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

→ $j = 0$ and 1

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

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

→ if $j = 0$

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

$\downarrow$

x^2

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

$\downarrow$

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

$$= \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x)^{(i)} - y^{(i)} \right) \times 1$$

$$h_\theta(x) = \theta_0 + \theta_1 x \rightarrow 0$$

θ_0 → 1

$\theta_1 x$ → constant

→ if $j = 1$

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

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

$\frac{\partial}{\partial \theta_1} [\theta_0 + \theta_1 x]$

↓ constant

↓ derivative will be x

Repeat until convergence

{

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

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

}

Multiple Linear Regression :

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

x = input Feature

θ_0 = Intercept

θ_1 = Slope

we won't change 'x'

↓

we only change

θ_0, θ_1

↓

Find Best Fit Line

what if we have multiple Input Features :

House pricing Dataset

3 Independent Features

output feature

No. of Rooms	Size of House	Location	Price
x_1	x_2	x_3	

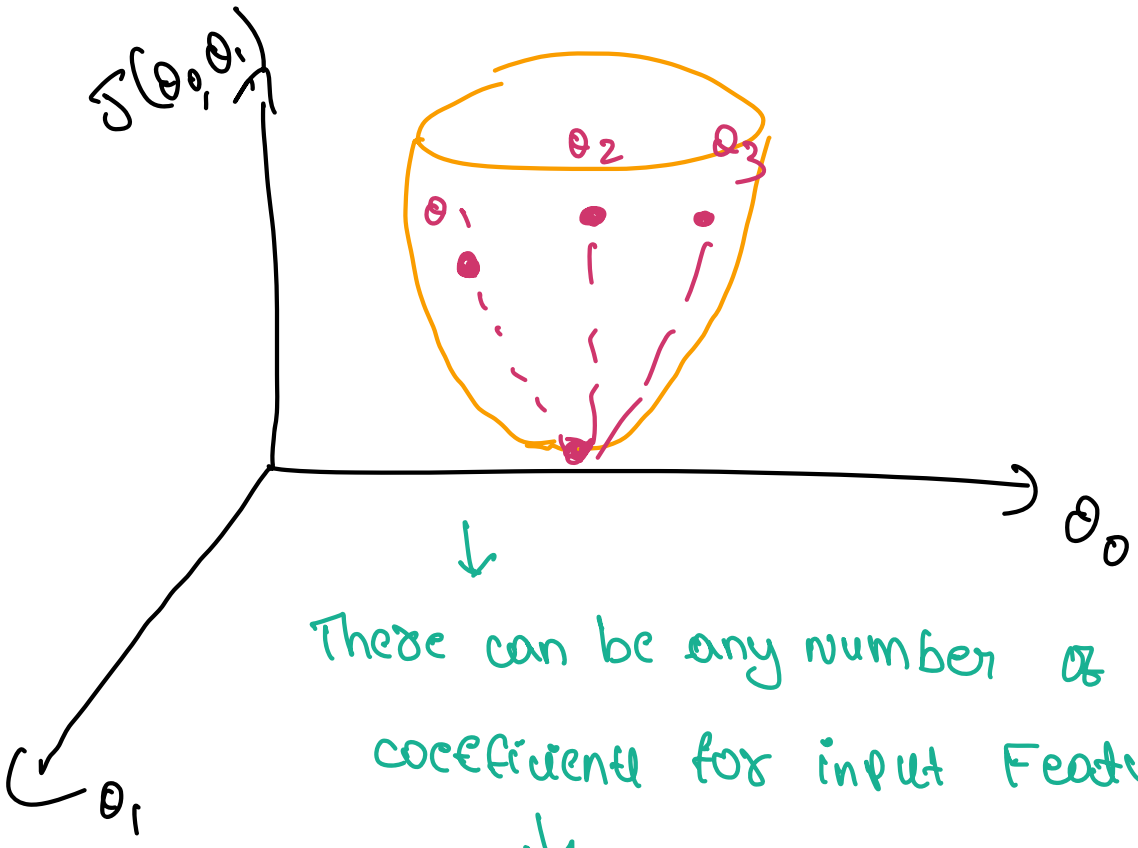
$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$$

↓

Equation for multiple Linear Regression

$\theta_1, \theta_2, \theta_3$ = coefficients (or) slope of x_1, x_2, x_3

θ_0 = Intercept will be only one



↓

These can be any number of 'θ' i.e, coefficients for input Feature

↓

Their only Aim is to reach Global minima

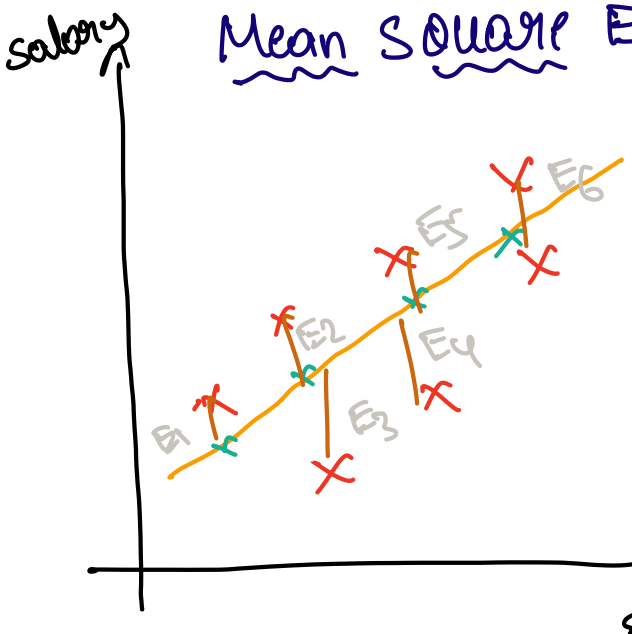
[Cost Function] → Performance Metrics :

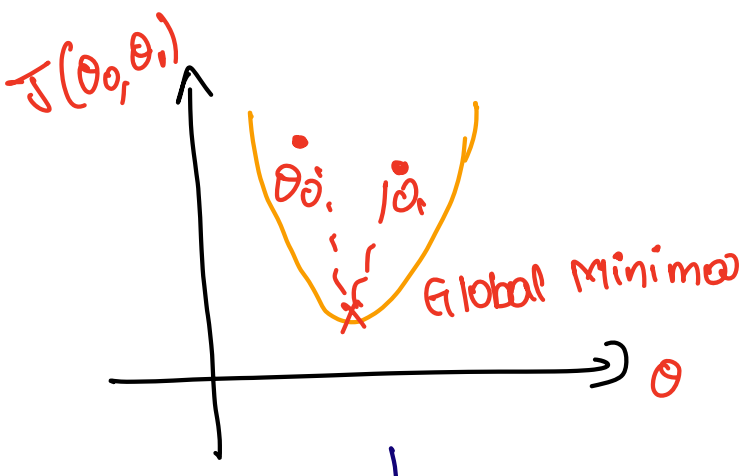
MSE → Mean Square Error

MAE → Mean Absolute Error

RMSE →

Mean Square Error





↓

This is for Gradient Descent

Data set

Input: Experience	Salary	o/p
⋮	⋮	

→ we need to reduce the cost Function

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

cost Function, MSE = $E1+E2+E3+E4+E5+E6$

↓

This value should be minimum

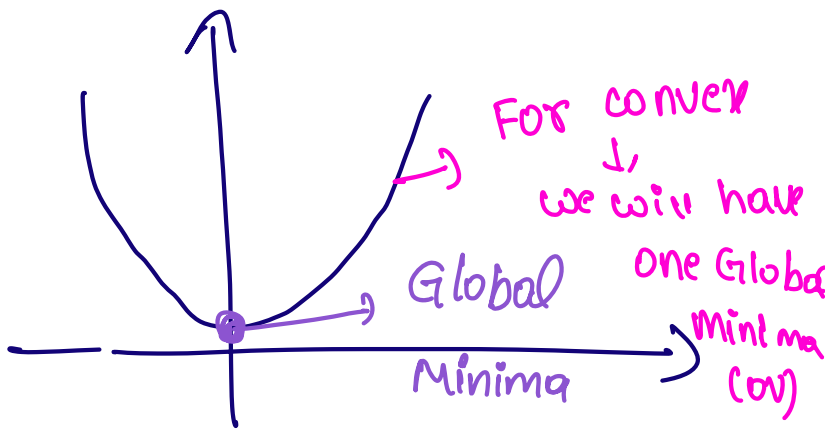
→ $(y-\hat{y})^2$ → It is Basically a Quadratic Equation

↓

$(a-b)^2 = a^2-2ab+b^2$

↓

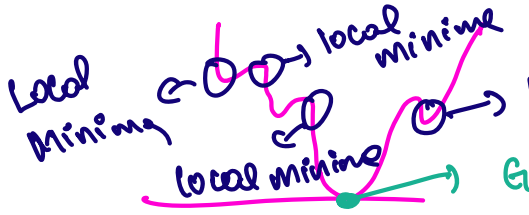
Graph for Quadratic Equation



This is very Similar to Gradient Descent ←

→ So By using (MSE) cost Function we can easily reach Global Minima

one Local Minima



→ For non-convex Graph like this we will have many local minimas

Advantages of MSE

1. It is Differentiable at every point
2. It has one local minima & global minima
3. Converge Faster

Disadvantages of MSE

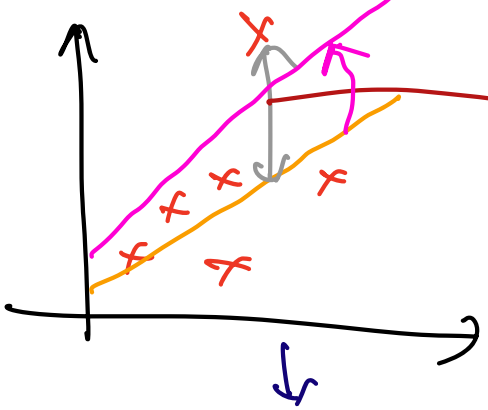
1. Not robust to outlier.
2. It is no longer in the same unit

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

→ Here we are penalizing the outlier

↓

Error will Increase



→ MSE will penalize this outlier

↓

our Best Fit line will move towards outlier (↗)

↓

This is the Impact of outlier

↓

MSE is Increasing

→ At local minima, the convergence will get stuck, Because slope at local minima will be "zero"

→ So if use any Equations other than Quadratic Equations may have many local minimas

→ So its Advisable to use Quadratic Equation

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

↓

If unit of y is lakhs

↓

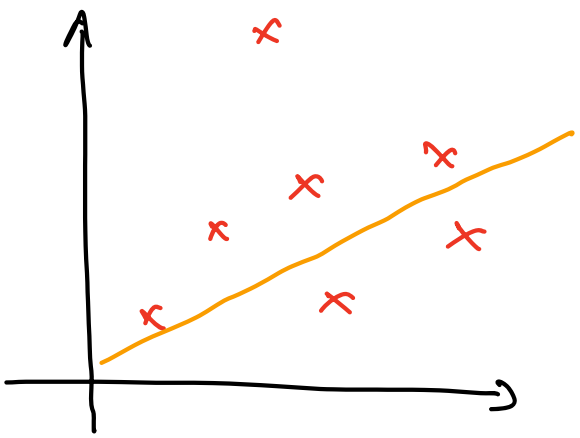
Doing $(y-\hat{y})^2$ → Becomes $(\text{lakhs})^2$

↓

which is changing the unit

Mean Absolute Error (MAE)

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



→ Here we are not squaring

↓

so it robust to outliers

→ It will be in the same unit

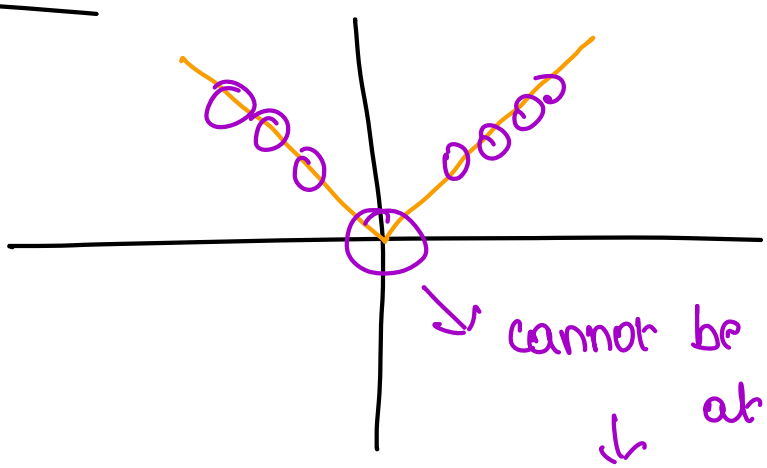
Advantages

1. Robust to outliers
2. It will be in the same unit

Disadvantages

1. Convergence usually take more time
2. optimization is a complex task
3. Time consuming

Graph of MAE :



↘ cannot be differentiable at 'zero'

↓

So we create sub gradient

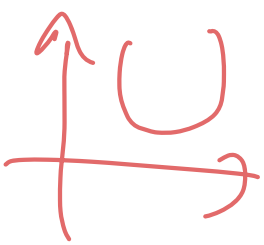
↓

Arrive at Global Minima

RMSE [Root mean Square Error]

$$RMSE = \sqrt{MSE}$$

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



Advantage

→ Same unit

→ Differentiable

Disadvantage

→ Not Robust to Outliers

Regression Performance Metrics

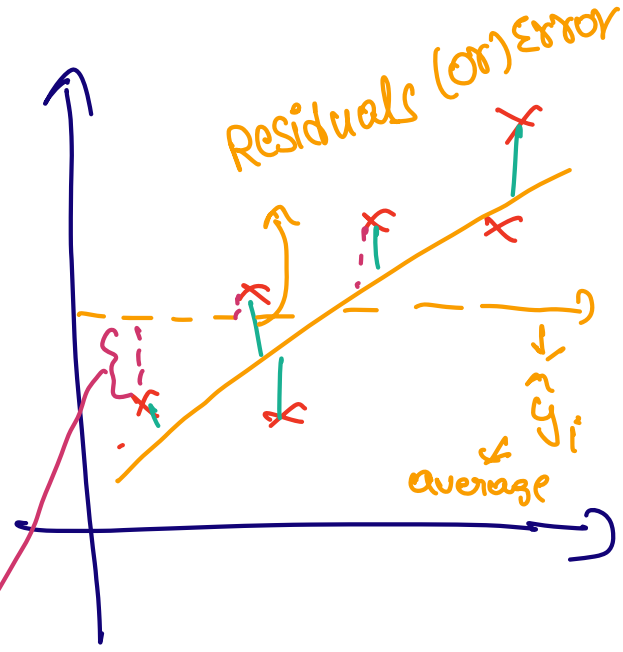
- MSE
- MAE
- RMSE
- R^2
- Adjusted R^2

R-Squared :

$$R^2 = 1 - \frac{SS_{RES}}{SS_{Total}}$$

$$= 1 - \frac{\text{Sum of squares of Residual}}{\text{Sum of squares of Total}}$$

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



$= 1 - \text{Smaller number}$
Bigger number

$= 1 - \text{smaller number}$

≈ 1

For Eg $R^2 = 0.70$, our model had 70% Accuracy

Adjusted R^2 :

Dataset

Size of House ↑ → Price ↑ ← Bedrooms, Location, Gender

↓

Both are having good positive correlation

$R^2 = 75\% \Rightarrow 0.75$

Bedrooms $\propto$ Price

$R^2 = 80\% \Rightarrow 0.80$

Location $\propto$ Price

$R^2 = 85\% \Rightarrow 0.85$

→ Gender doesn't have any correlation with price

R Squared = 87% ⇒ 0.87

→ AS features Increases, Due to the formula of R square value Increased

↓

Even though the Input feature is not at all correlated with output feature

↓

Rsquare value Increases

→ Adjusted R square will resolve this problem

↓

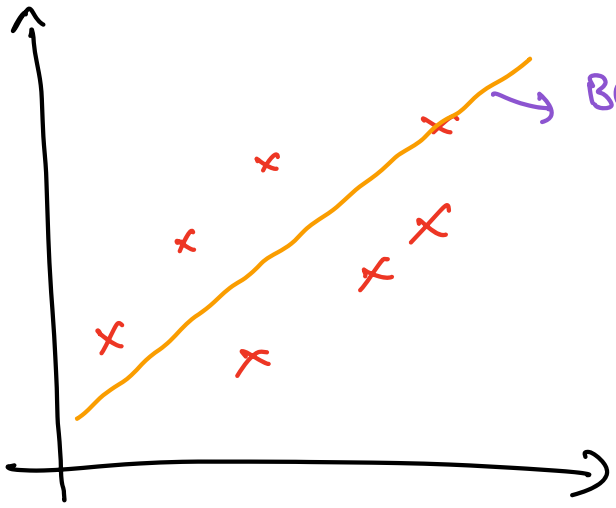
By penalizing the features which are not correlated with o/p feature

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

N = no. of data points

P = no. of Independent features

Linear Regression using OLS (ordinary Least square)



→ Best line was found using optimizer Algorithm Before

↓

Gradient Descent

↓↓

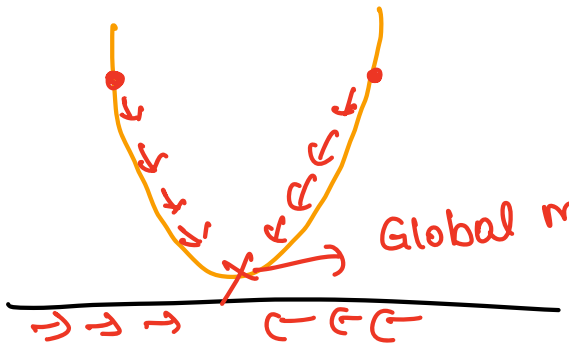
Loss function

↓↓

Every point we calculated Derivate

↓↓

Drive towards Global minima



OLS Technique ;

→ using formula and calculate β_0 & β_1

→ Aim of OLS is to reduce cost function error

ordinary Least square :

Error

$$S(\beta_0, \beta_1) = \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \beta_0 - \beta_1 x_i\right)^2$$

↓↓

Aim is to calculate β_0 & β_1

calculative derivatives w.r.to β_0

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

$$\frac{\partial S}{\partial \beta_0}(\beta_0, \beta_1) = \frac{2}{n} \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)(0 - 1 - 0)$$

$$= \frac{-2}{n} \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i) = 0 \quad \rightarrow (1)$$

$$\frac{\partial S}{\partial \beta_1}(\beta_0, \beta_1) = \frac{2}{n}\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)(0 - 0 - x_i)$$

$$= \frac{-2}{n}\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)(x_i) = 0 \rightarrow (2)$$

Eq $\rightarrow$ (1)

$$\frac{-2}{n}\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i) = 0$$

$$-\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i) = 0$$

$$-\sum_{i=1}^{n} y_i + n * \beta_0 + \beta_1 \sum_{i=1}^{n} x_i = 0$$

$$n\beta_0 + \beta_1 \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i$$

$$\beta_0 = \frac{\sum_{i=1}^{n} y_i}{n} - \beta_1 \frac{\sum_{i=1}^{n} x_i}{n}$$

$$\Downarrow$$

$$\beta_0 = \bar{y} - \beta_1 \bar{x}$$

Intercept can be calculated by using this formula

Eq $\rightarrow$ (2)

$$\frac{-2}{n}\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)(x_i) = 0$$

$$\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)(x_i) = 0$$

$$\sum_{i=1}^{n} x_i y_i - \beta_0 \sum_{i=1}^{n} (x_i) - \beta_1 \sum_{i=1}^{n} (x_i)^2 = 0$$

$$\sum_{i=1}^{n} \left(x_i y_i - \beta_0 (x_i) + \beta_1 x_i^2 \right) = 0$$

Replace $\beta_0 = \bar{y} - \beta_1 \bar{x}$

$$\sum_{i=1}^{n} \left(x_i y_i - (\bar{y} - \beta_1 \bar{x}) x_i - \beta_1 x_i^2 \right) = 0$$

$$\sum_{i=1}^{n} \left(x_i y_i - x_i \bar{y} + \beta_1 \bar{x} x_i - \beta_1 x_i^2 \right) = 0$$

$$\sum_{i=1}^{n} \left(y_i - \bar{y} + \beta_1 \bar{x} - \beta_1 x_i \right) = 0$$

$$\sum_{i=1}^{n} \left[\left((y_i - \bar{y}) + \beta_1 (\bar{x} - x_i) \right) \right] = 0$$

$$\sum_{i=1}^{n} (y_i - \bar{y}) + \sum_{i=1}^{n} \beta_1 (\bar{x} - x_i) = 0$$

$$\sum_{i=1}^{n} \beta_1 (\bar{x} - x_i) = - \sum_{i=1}^{n} (y_i - \bar{y})$$

$$\beta_1 = \frac{- \sum_{i=1}^{n} (y_i - \bar{y})}{\sum_{i=1}^{n} (\bar{x} - x_i)}$$

$$\beta_1 = \frac{\sum_{i=1}^{n} (y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})}$$

$$\beta = \frac{\sum_{i=1}^{n} (y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})}$$

→ which is our slope (or) Coefficient

OLS formulas

Intercept

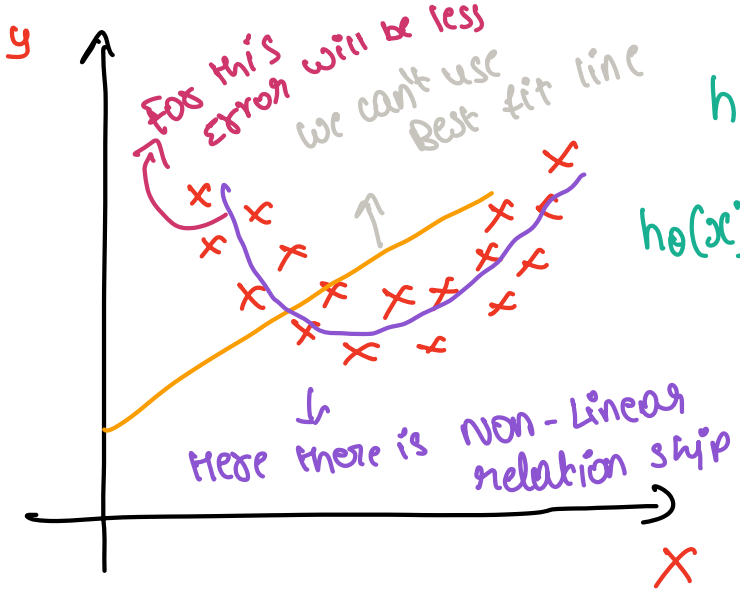
$$\beta_0 = \bar{y} - \beta_1 \bar{x}$$

coefficient

$$\beta = \frac{\sum_{i=1}^{n} (y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})}$$

x	y	$(y_i - \bar{y}) \div (x_i - \bar{x})$
⋮	⋮	⋮
$\bar{x}$	$\bar{y}$	

Polynomial Regression :



Only on Independent Feature

$h_\theta(x) = \beta_0 + \beta_1 x$ → Simple Linear Regression

$h_\theta(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$

↓

Multiple Linear Regression

↓

more than one Independent Feature

What if my dataset is not Linear ? (Like Above)

→ In Both Simple Linear Regression, Multiple Linear Regression we assume Data points are Linear

↓

In this case we will use Polynomial Regression

Polynomial Degrees

degree = 0

↓

Simple Polynomial Regression (one I/P, one O/P)

if degree = 0

↓

$h_\theta(x) = \beta_0 \times x^0$ ⇒ constant value

Polynomial degree = 1, $h_\theta(x) = \beta_0 \times x^0 + \beta_1 x^{(1)}$

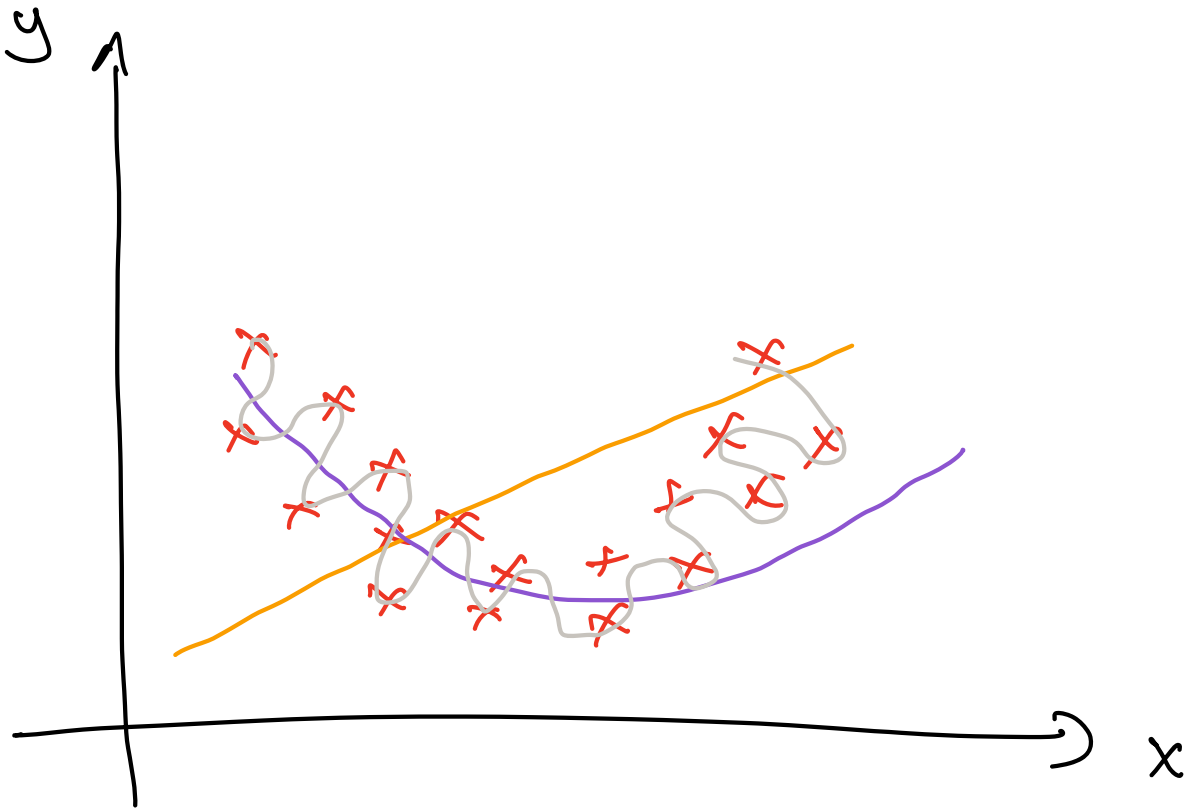
↓

Simple Linear Regression

Polynomial degree = 2, $h_\theta(x) = \beta_0 x^0 + \beta_1 x^{(1)} + \beta_2 x^{(2)}$

if degree = n,

$$h_\theta(x) = \beta_0 x^0 + \beta_1 x^{(1)} + \beta_2 x^{(2)} + \beta_3 x^{(3)} + \dots + \beta_n x^{(n)}$$



→ As polynomial degree increases, our line (or) curve will fit more data points

→ But if we are using much bigger polynomial degree, the line will try to fit every other data point and becomes overfit on data points

→ So we have to find the best degree for the curve to fit all data points without overfitting

For Eg if we have 2 Independent Features

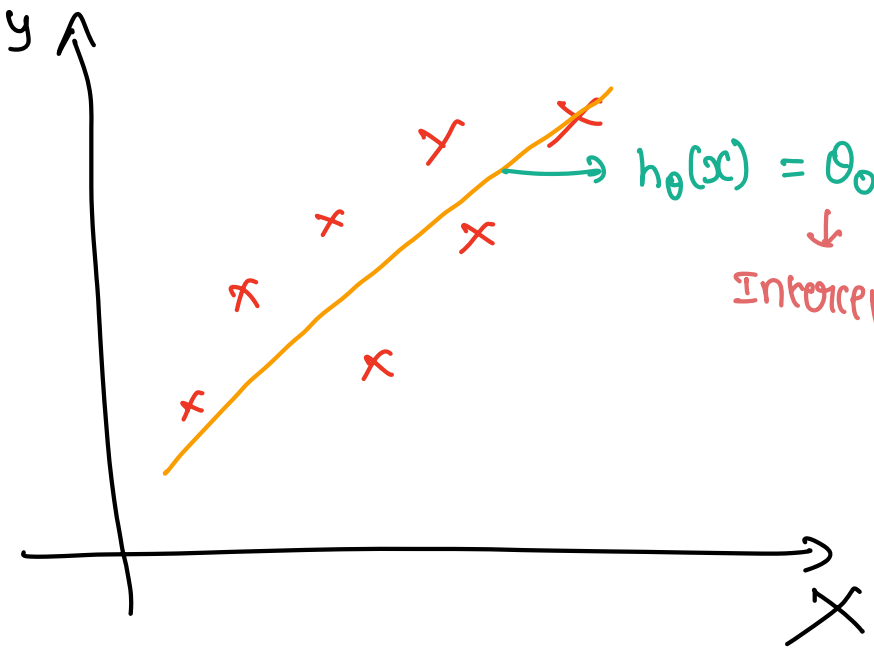
degree = 1, $h_\theta(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$

degree = 2, $h_\theta(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1^2 + \beta_4 x_2^2$

↓

Since degree is 2, we have to square Every input Feature

Summary of Linear Regression

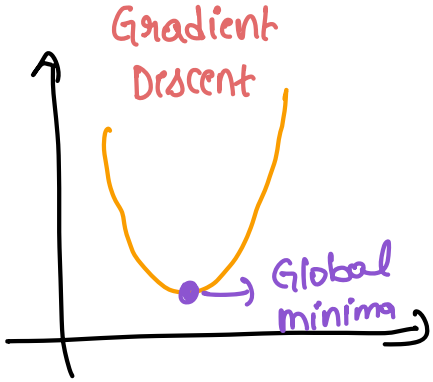


Single Independent Feature

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

θ_0 → Intercept

θ_1 → Slope



Cost function :

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

→ Mean Square Error

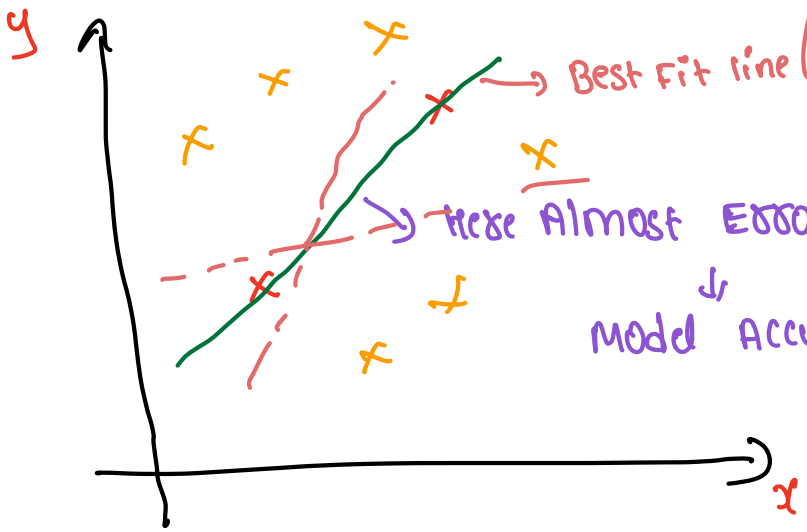
$h_\theta(x)^{(i)}$ → Predicted value

$y^{(i)}$ → Actual value

Ridge Regression :

↳ Assume single Independent Feature

↳ 2 data points



x → New Test Data

↓

Now the Error is Increasing on New Test Data

↓

Model is Over Fitting

↓

Train data → Accuracy ↑ → Low Bias

Test data → Accuracy ↓ → High Variance

↓

So, we use Ridge Regression for solving Over Fitting issue

Train error ↑ → Bias

Test Error → Variance

① Ridge Regression (or) L2 Regularization :

↳ Reduces Over fitting

$$\text{Cost Function} = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x)^{(i)} - y^{(i)} \right)^2$$

↓↓ this Becomes Zero during Over Fitting

0

→ So, we have to avoid the Cost Function shouldn't become zero

→ So we will add 2 parts to the cost Function as part of Ridge Regression

$$\text{Cost Function} = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2 + \lambda \sum_{i=1}^{n} (\text{slope})^2$$

↓ Hyper parameter

if $\lambda = 1$, $= (0) + (1)[(\theta_1)^2]$

↓

Here we are penalizing this '0' value By adding $(1)[(\theta_1)^2]$

↓

So the cost Function will not be '0' at any time

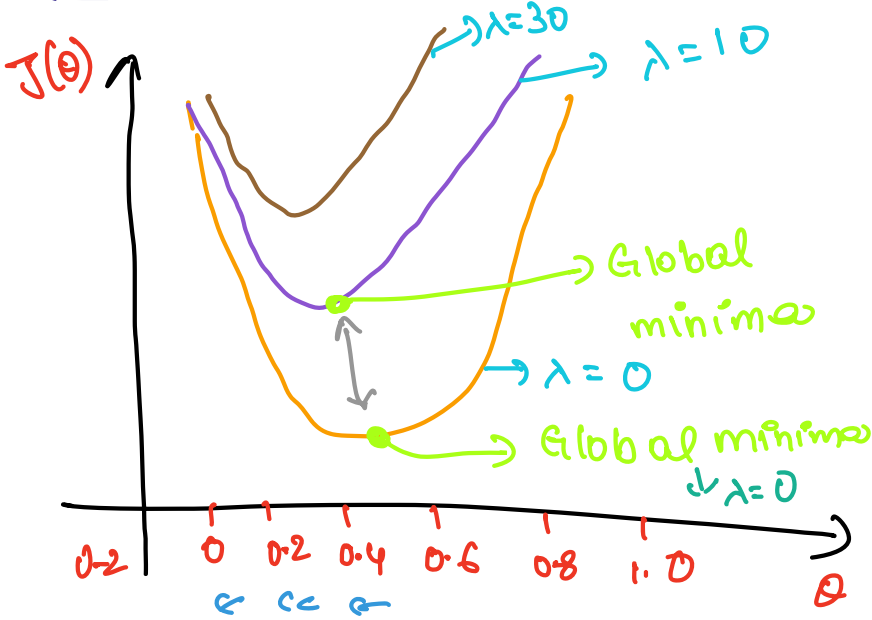
↓

This will avoid over Fitting

What is the Relationship B/w $\underline{\lambda}$ and $\underline{\text{slope}}$? (L2 Reg term)

↓

$\lambda \uparrow$ Slope ↓



$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$$

Intercept ← $= 0.34 + 0.52 x_1 + 0.48 x_2 + 0.24 x_3$ (Assume)

If we Apply Ridge Regression on this

↓

Above Coefficients will reduce

↓

But will Never Become Zero

$0.52 x_1$

↓

With the unit movement in x_1, what is the movement in y?

$x_1 \rightarrow 1$

$y \rightarrow 0.52$

↓

If x_1 is moving by '1'

↓

y will move by 0.52

For Eg :

In this case → The coefficient is '1'

$x \rightarrow 1$

$y \rightarrow 1$

→ If 'x' is moved by '1'

↓

'y' is moved by '1'

↓

meaning 'x' and 'y' Features are Highly correlated

After Applying Ridge Regression on Above :

$$h_\theta(x) = 0.34 + 0.40 x_1 + 0.38 x_2 + 0.14 x_3$$

These Coefficients are penalized (or) reduced

→ How Ridge Regression (or) L2 reducing overfitting :

↓

Reduces the impact of the coefficient, by reducing the coefficient of the feature that are not directly related, correlated with the output Feature

(2) Lasso Regression (L1 Regularization) → Feature selection

$$\text{Cost Function} = \frac{1}{am}\sum_{i=1}^{m}\left(h_\theta(x)^{(i)} - y^{(i)}\right) + \lambda \sum_{i=1}^{n} |\text{Slope}|$$

↓

Adding this part to cost function as part of Lasso Regression

Relationship B/w 'λ' and 'Slope' :

→ Here

$\lambda \uparrow$, $\theta \downarrow$

↓

But at some point of 'λ'

↓

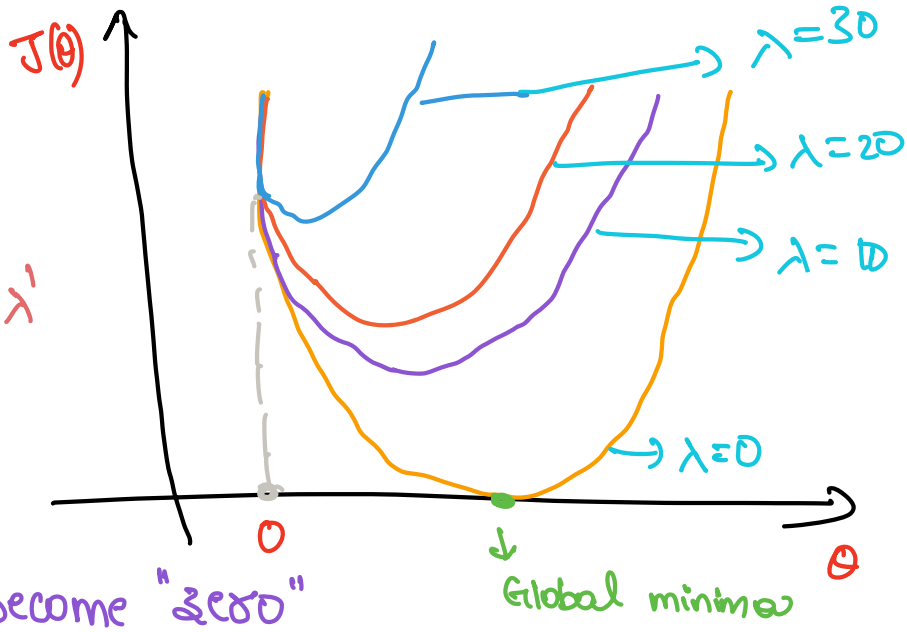
$\theta = 0$

↓

i.e, coefficient will become "zero"

↓

i.e, In short we are trying to remove that specific Feature



For Eg:

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_4 x_4$$

$$= 0.52 + 0.65 x_1 + 0.72 x_2 + 0.34 x_3 + 0.12 x_4$$

x_4 is the Feature that is not much correlated with O/P Feature

↓

very small coefficient 0.12

↓

After Apply Lasso Regression (or) L1 on this Cost Function

↓

Lasso will reduce it '0'

↓

'0.12' Becomes '0'

↓

x_4 Feature is Eliminated

$$h_\theta(x) = 0.52 + 0.5 x_1 + 0.60 x_2 + 0.14 x_3 + \cancel{0 \times x_4}$$

→ Which all Input Features that are not much correlated with output Feature will have small coefficient value

↓

On Applying Lasso Regression, it reduces those unimportant Features to 'zero'

↓

So we can use Lasso Regression for Feature selection

↓

If we have more Featuresse, we can use Lasso

③ Elastic Net Regression

→ It is the combination of Both Ridge & Lasso

- ① Reduce overfitting
- ② Feature selection

$$\text{Cost Function} = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x)^{(i)} - y^{(i)}\right)^2 + \lambda_1 \sum_{i=1}^{m}(\text{slope})^2 + \lambda_2 \sum_{i=1}^{n}|\text{slope}|$$

$\lambda_1 \sum_{i=1}^{m}(\text{slope})^2$ → L2 → Ridge Regression → Reduce overfitting

$\lambda_2 \sum_{i=1}^{n}|\text{slope}|$ → L1 → Lasso Regression → Feature Selection

→ For Hypermaker Tuning

↓

we will use Ridge Regression, Lasso Regression & Elastic Net Regression