INTRODUCTION

- **Supervised Learning:**
    - **Regression Problem:** When we have a continuos value in our output then it knows as regression problem.
    - **Classification Problem :** There are two types of classification problems 1) **Binary classification** and 2) **Multiclassification** problem

- **Unsupervised learning:**
    - **Clustering:** In Clustering, we don't have output feature or dependent variable, it's grouping algorithm ,**not classification problem**. We make clusters that describes info. For example, based on the data apple have customer who afford expensive products and cheap products. We can make two clusters, one who purchased expensive one consider as rich group of people and other one consider as middle class people. when they launch new products as per the price , it shows or give recommendation to the particular customers. Let's say apple launch i watch series 9 and i watch SE ,So based on the customer previous orders it suggest them either buy Series 9 or SE model.

1. Linear Regression

Equation of Linear regression is

h(x) = $\theta 0 + \theta 1 X$ where $\theta 1$ is slop or coefficient, and $\theta 0$ is intercept which means at what point this line meet at y axis.

- The main aim of LInear regression is to create best fit line so that the distance between predicted values and actual value would be minimal as much as we can. We use Cost function to find out distance between predicted and actual values,

  Cost Function:

$$\mathbf{J(\theta 0\ ,\ \theta 1) = \frac{1}{2m} \sum_{i=1}^{m} (h(Xi) - Yi)^2}$$

Where, h(Xi) is predicted values and Yi is actual values, and **1/m** is use to get average cost function , **½** will help us in derivative purpose.

So, this is know as **squared error function**

What we need to solve

We need to minimize the above cost function by adjusting the $\theta 0\ and\ \theta 1$.

Gradient Descent:

Gradient descent will play an important role to ensure that we get right slop value. In Gradient descent , we get global minima and that is the best fit line for linear regression where the distance between actual values and predicted values are minimal.

So, How do we find all points for gradient descent and get global minima? Well, for that we will use convergence algorithm

What convergence algorithm says that,

Repeat until convergence
{

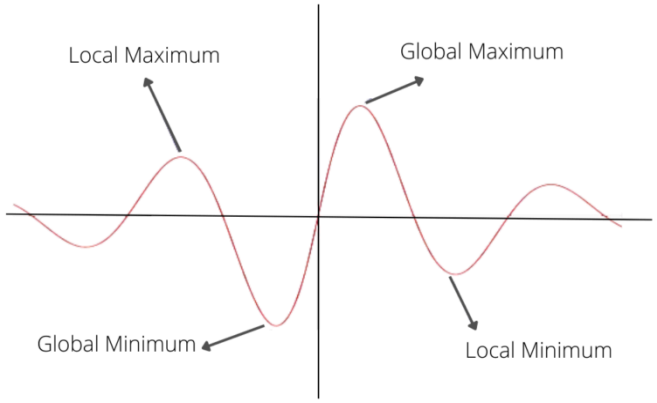
$$\theta j := \theta j - \alpha \frac{\delta}{\delta \theta_j} J(\theta 0, \theta 1)$$

}

Here, $\alpha \; is \; Learning \; Rate$, Alpha here play major role as if we take alpha value small like $\alpha = 0.01$, it takes small steps to move towards the optimal minima not even too small value , while if we take alpha value huge like $\alpha = 1$, the updation of $\theta 1$ value will jump and never reach the global minima.

Learning rate when we use small alpha value

Now, what if we have local minima in our Cost function



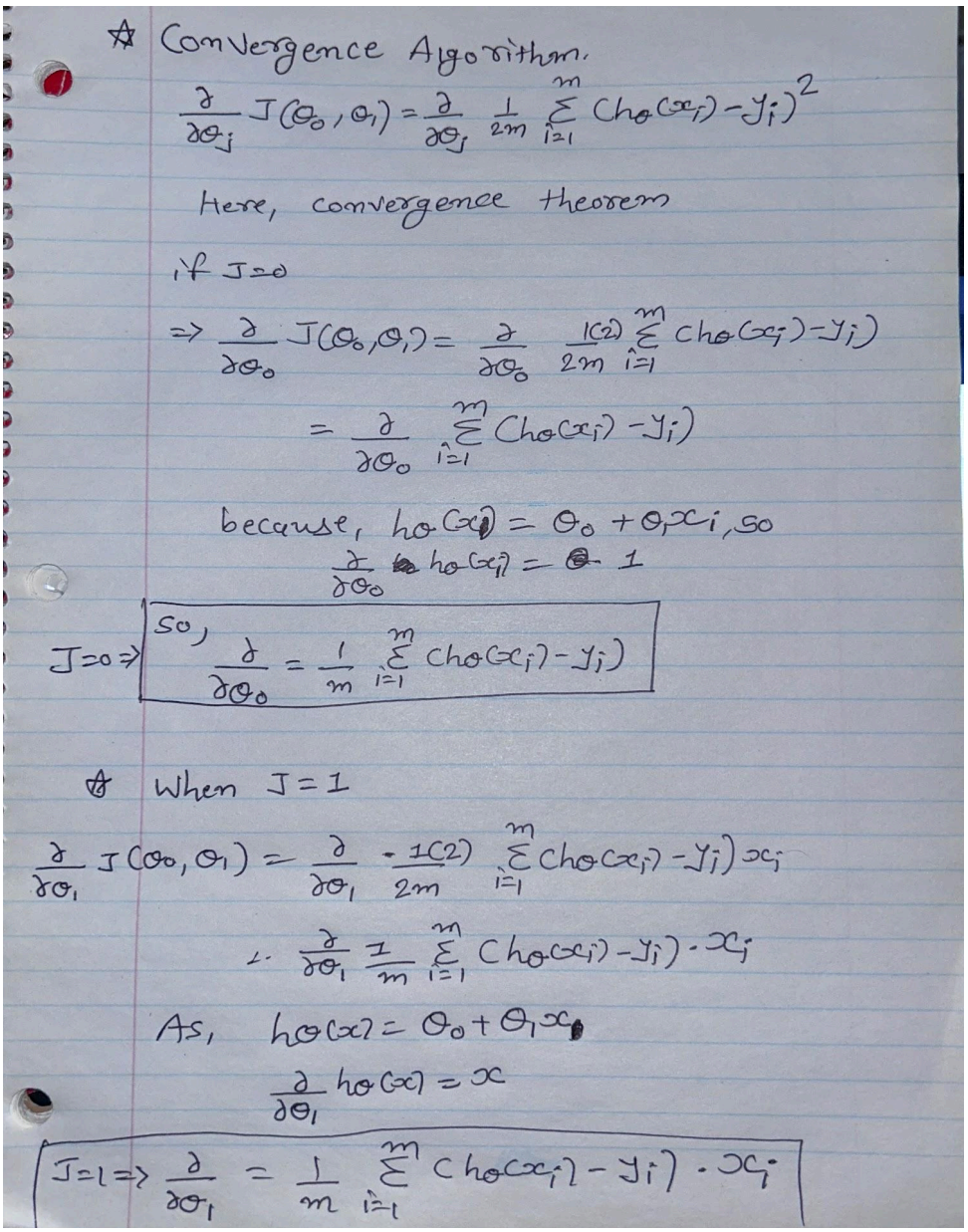
At local minima our Equation would be liked,

$\theta 1 := \theta 1 - \alpha(0)$, as slop($\frac{\delta}{\delta \theta_j} J(\theta 0, \theta 1)$) will be 0 at local minima, So,$\theta 1 := \theta 1.$

However, with the respect to the Gradient Descent and equation that we are using ,we don't get stuck in local minima.

Note:

What we do if we get local minima in Linear Regression, so Answer is that the Cost function we use ,will definitely not give us a local minima.



Convergence Algorithm

So,

Repeat until Convergence
{

$$\theta 0 := \theta 0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h(Xi) - Yi)$$

$$\theta 1 := \theta 1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h(Xi) - Yi) . (Xi)$$

}

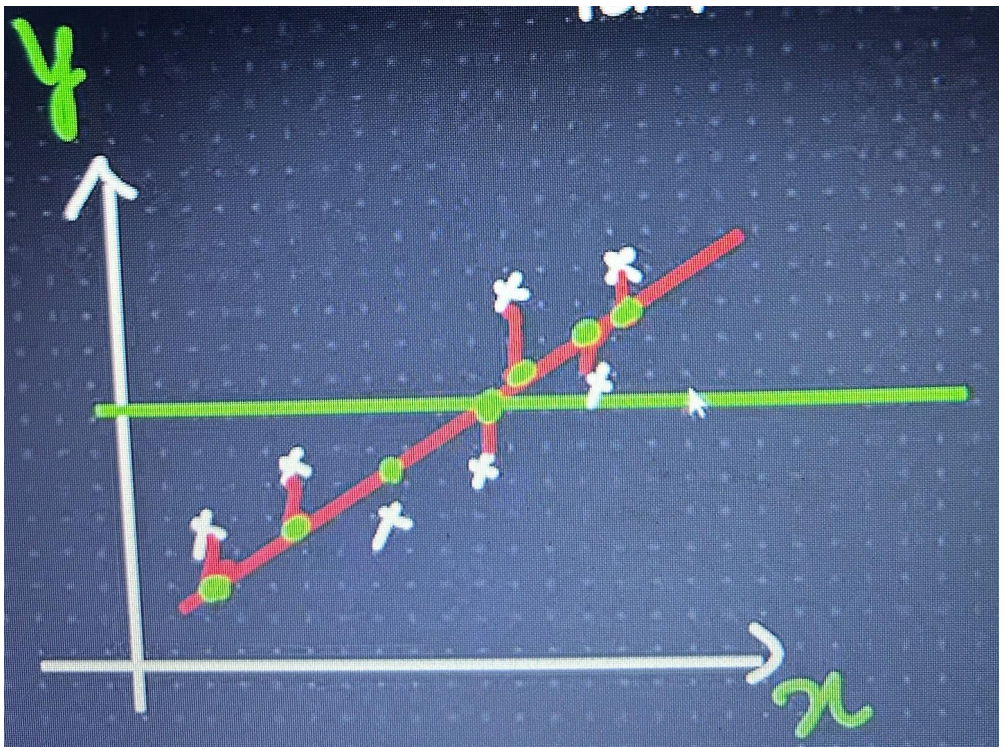
Performance Metrics:

We use two metrics to check the performance of model 1) **R^2** and 2) **Adjusted R^2**

R^2:

$$R^2 = 1 - \frac{SSresidual}{SStotal} = 1 - \frac{\Sigma(Yi - \bar{Yi})^2}{\Sigma(Yi - \bar{Y})^2}$$

Here, $\bar{Yi}$ is h(x) , and $\bar{Y}$ is mean of y



$\Sigma(Yi - \bar{Y})^2$ will have higher values than $\Sigma(Yi - \bar{Yi})^2$, So 1 - $\frac{Low\ value}{High\ value}$ after calculation of $\frac{Low\ value}{High\ value}$, we get smaller value then we minus it from 1 and eventually we get higher number. In conclusion , we can say that our model work perfectly and have good R^2.

Let's say we have features that are correlated than in that scenario our R^2 will increase. Even though we have features that are not correlated , R^2 will also go up. When we have features that highly correlated our R^2 will be increased tremendously and have less correlated features our R^2 will be small increased.

For example, let's say we have features like bedroom, location that are correlated with price so here R^2 increases, and when we add another feature like gender which is not correlated at all although R^2 will increase but we should not have been picked this model, instead we consider the first model where all features are correlated. So, in order to prevent this situation we use Adjusted R^2.

R^2 Adjusted:

R^2 Adjusted = $1 - \frac{(1-R^2)(N-1)}{N-P-1}$, where , N is total number of samples and P is Features or Predictors

As P value is getting increased, our R^2 adjusted will be decreased to adjust R^2.

Ridge and Lasso Regression:

Overfitting and Underfitting:

When our model works well on **TRAINING DATA (also knows as low bias)** and don't work well on **TEST DATA (High Variance)** that is called **Overfitting,** Whereas **Underfitting** means when not only on **Training data (HIgh bias)**, but also on **testing data (High variance)** our model doesn't work well.

Ridge Regression (L2 Regularization):

Whenever we get overfitting in Linear Regression, we use L2 regularization in order to prevent Overfitting.

Let's consider here, J(θ1) = 0

So, $J(\theta 0, \theta 1) = \frac{1}{2m} \sum_{i=1}^{m} (h\theta(x_i) - y_i)^2$

With the help of this equation if we get $J(\theta 0, \theta 1) = 0$, then we need to use L2 regularization, and the equation of L2 is

$$(h\theta(x_i) - y_i)^2 + \lambda(slop)^2$$

Here, lambda is a hyperparameter, we will find lambda through cross validation

Lasso Regression (L1 Regularization):

L1 Regularization is also used to prevent **Overfitting** , here the Equaton of L1 is

$$(h\theta(x_i) - y_i)^2 + \lambda|slop|$$

Apart from Overfitting , it is also useful for feature selection, let's take example here,

$$h\theta(x) = \bar{y} = \theta 0 + \theta 1x1 + \theta 2x2 + \theta 3x3 + \theta 4x4 + \ldots + \theta nxn$$

So, now we take a mode of all slops like this,

$$|\theta 0 + \theta 1 + \theta 2 + \theta 3 + \theta 4 + \ldots + \theta n|$$

As we go ahead whichever features do not play amazing role its slop value is small , here L1 will neglect that features.

In conclusion, we do two major things in Lasso regression 1) **To prevent Overfitting, and 2) Assist to select an important features and neglect the less important the less important features , While** we use Ridge regression **for only preventing Overfitting.**

Assumption of Linear Regression:

1) If our **features are normal or gaussian distribution**, Our model will be **trained well.** However, if our features does not follow normal or gaussian distribution, we will apply kind of mathematical equation in order to convert them into either normal or gaussian distribution.
2) **Standardization** is nothing but Scaling data with the help of **Z-score** in order to make **mean**(μ) **value 0** and **Standard deviation(σ) value 1**. If we want to **increase training time of model or optimize the model** it is inevitable to use standardization.
3) Linear Regression works well with respect to linearity.
4) We need to check **Multi-colinearity.**

   Let's say we have three features **X1,X2,X3** and output feature **Y,** now let's say **X1 and X2 are 95% correlated**, also these are **highly correlated with output feature Y**, then it's not necessary to use both features,so we can drop one of two features.

Hence, **Multicollinearity** is also solved with the help of **Variation inflation factor.**

2. Logistic Regression (Classification)

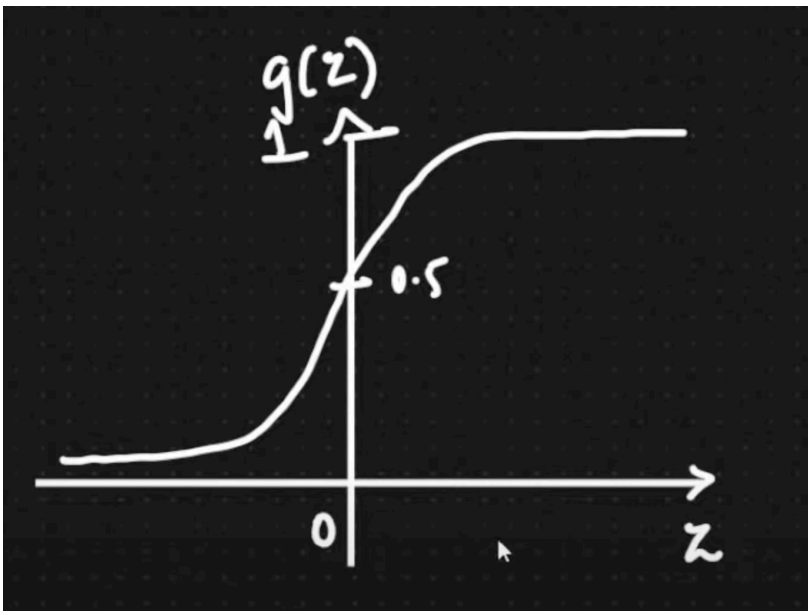
- This Regression is helpful when we have binary classification like yes or no , 0 or 1 , True or False.
- In order to solve this type of problem we can use **LInear Regression, however, we might face problems when we get outlinears.** Our maxima and minima values are 1 and 0 respectively, and sometimes **due to outliners** we may get value **grater than 1 or less than 0** in that case **we cann't use Linear Regression.**
- We apply mathematical formula on top of the linear regression to make a **Squash line at the beginning and in the end.**

$h\theta(x) = \theta 0 + \theta 1 X$, This is normal linear regression function in order to make logistic function we simply add **another mathematical function to make squash lline**, but first let's say $z = \theta 0 + \theta 1 X$

So, $h\theta(x) = g(z),$

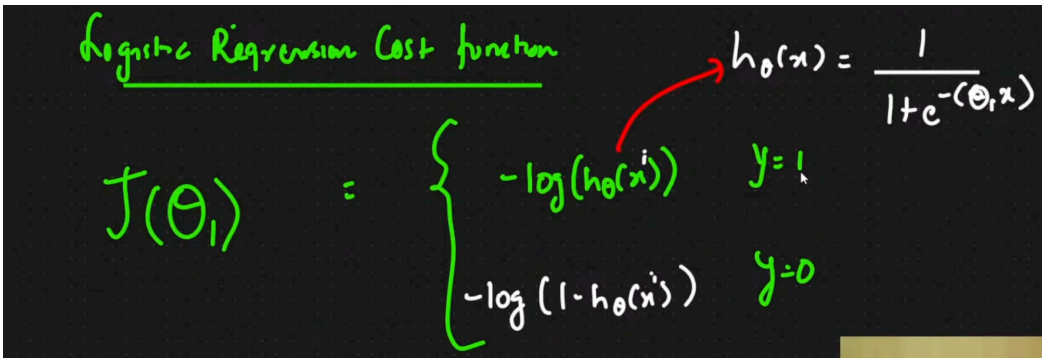
$$h\theta(x) = \frac{1}{1 + e^{-(\theta 0 + \theta 1 X)}}$$

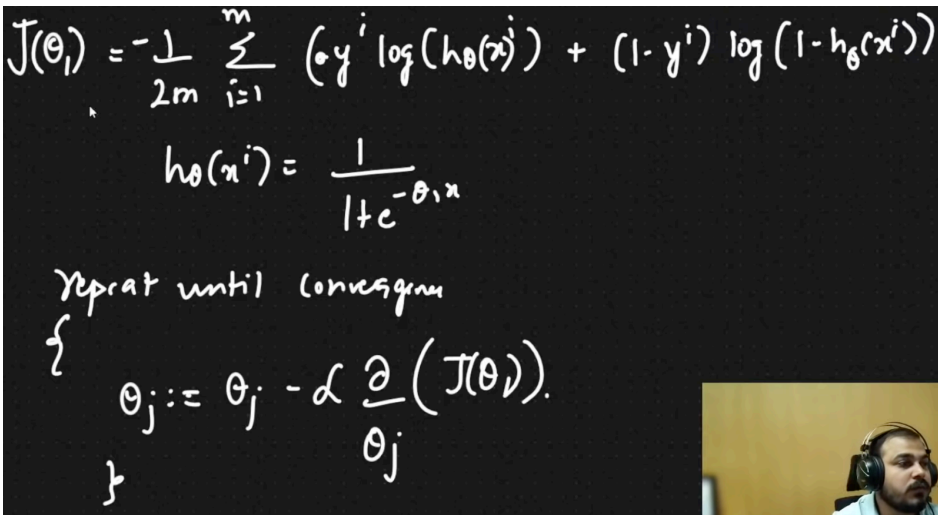
So, this is **Sigmoid** or **Logistic** function.



We can make assumption that is , **g(z) >= 0.5** , When we have **z >=0**

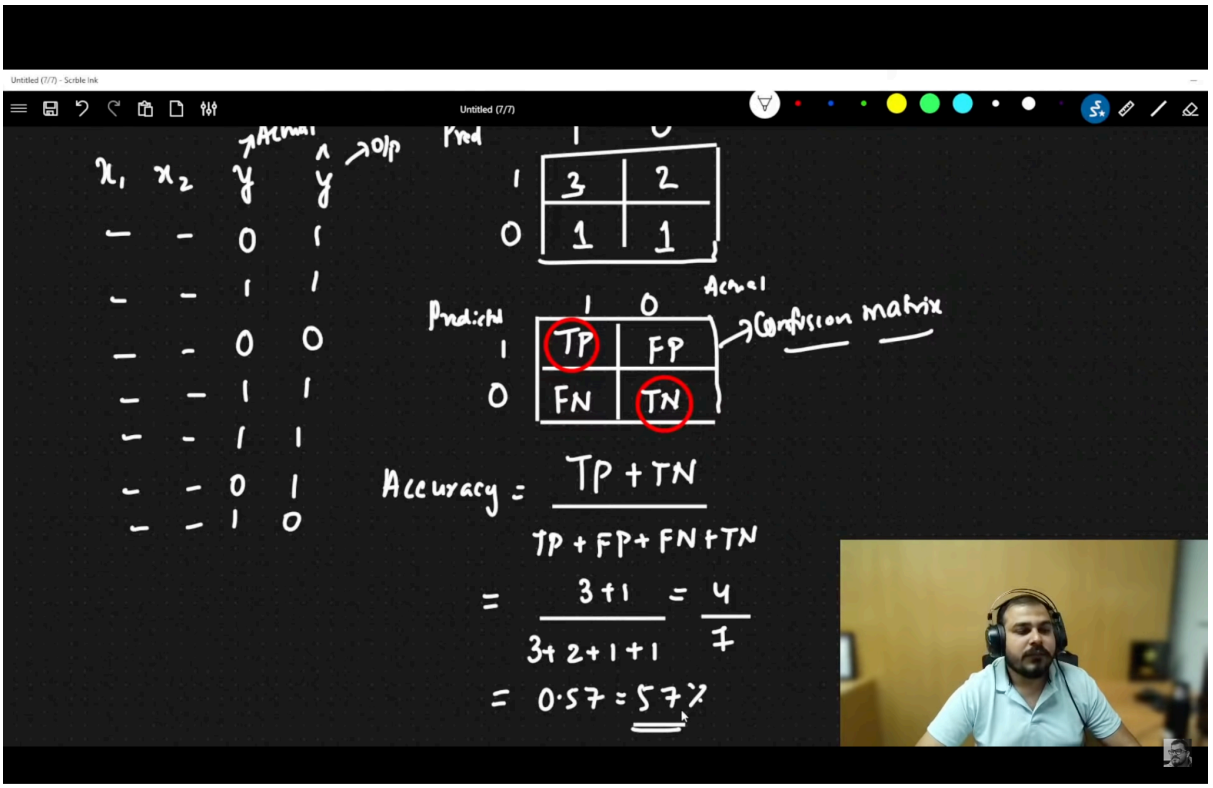
If we use above $h\theta(x)$ equation in linear regression's cost function then we may have to face non-convex function where we get lots of local minima So, there is a different way to handle this problem,





J(θ1) is the cost function , we use convergence algorithm till we get global minima as usual.

Now, how to calculate accuracy with the help of confusion metrix



However, we can't always depend on accuray, for example if we have same 900 data out of 1000 so definitely, we get 90% accuracy. To handle this issue , we use

1) **Precision**
2) **Recall**
3) **F-score**

Precision

From all the **predicted** positive values how many are actually **true** or **Positive.**

Precision = TP / TP+FP

For example, in spame mail identification, it should be identify as spam in that scenario **False positive** should be reduced and **FP** talks in a better way for spam classification, So here our priority is **FP**

Recall

Definition of recall is that from all actual positive values how many predicted true values, so the formula of recall is

Recall = TP / TP+FN , Our target in recall that is we should reduce **False Negative (FN)**

For example, if a person has a cancer model , value should be 1 otherwise it will be beig trouble if it shows 0 ,So here our priority is **FN.**

F-score

We use F-score when positive and negative both are an important. For example, Stock market crash tommorrow, here , from people point of view it's positive as they can sell there stocks, while as per the industry it's negative view. In this type scenario ,we use **F-score.**

So, It's necessary to give focus on both **False Positive (FP)** and **False Negative (FN).**

F-score = (1+ B^2) (Precision * Recall) / B^2 * (Precision + Recall)

Case - 1

If False Positive (FP) and **False Negative (FN)** are an important ,

B =1 , F1-score = (1+ 1) **(Precision * Recall) / (Precision + Recall)**

Case - 2

If False Positive is more important than **False negative,** we use

B= 0.5 , F0.5 - score = (1 + 0.5^2) **(Precision * Recall) / 0.5^2 * (Precision + Recall)**

Case - 3

If False negative is more important than **False positive,** we use

B= 2 , F2 - score = (1 + 2^2) **(Precision * Recall) / 2^2 * (Precision + Recall)**