**DAY 19 AGENDA**

* Linear Regression
* Math Behind Linear Regression
* Error Estimation in Linear Regression
* Steps For Creating a model
* Hands-on Linear Regression in Python
* Model Building and Testing in Python

**Linear Regression:**

1. ML is classified in supervised learning and unsupervised learning.
2. Supervised learning has 2 aspects one is classification and regression.
3. The regression problem is the one where the outcome of the model is numerical values for example stock market, real state prices, etc.
4. The classification problem is the one when the outcome deals with categorical values for example high, low, or the classification in the bank if you are default customer or not.
5. Supervised learning is when you have a target.

**Algorithms to Solve Regression problem:**

1. We use linear regression to solve regression problems as this one gives out very good output and precise outputs.

**The Math behind  Linear Regression :**

1. Line equation y=mx+c is an equation to generate a straight line to represent data for independent and dependent data.
2. Using this line we can have a prediction of future possibilities.
3. Looking at the maths behind the algorithm we can have a better insight as to how it works and how it will react to a specific problem.
4. X here is the input or the independent variable and y it output that is the dependent variable.c is the constant variable and m is the slope.
5. For all the problems we find the m and c values and use them for further forecasting.
6. We can predict the value for y you have accomplished the forecasting.
7. After finding the values of m and c values we can use it and the value of x in the equation to predict future values.
8. To estimate the value of slope there is a formula that has been defined.
9. For the given data we will find out x,y,xy,x2, and summation of x and y,xy, and x2.
10. Using this in the formula we can get the value of x.
11. For finding c we have another formula.
12. Y is the value we have and y’ is the value we will be evaluating.
13. We will plot a graph for the data that has already been provided that is for the values we already know.
14. We will plot the actual and predicted values.
15. After plotting a graph and marking all the points we will get a lie that passes through all the points and the equation of the line is the one we have found out using the math formulae.
16. On the graph, we can check the forecasted value by tracking where the x coordinate meets the line.

**Linear Regression Performance Metrics /Cost Function /Error Estimation :**

1. After understanding the math behind the algorithm we need to look into the performance of the algorithm.
2. Performance is measured based on the error that the algorithm does in forecasting.
3. In the math we have seen the  y is 2 and the predicted y is 2.2 so here we can see there are some errors.
4. Error=y-y’ (in the example we see it is 2-2.2).
5. So at times like in our example, we get the error in negative.
6. To get the total error we have to find the difference between all the data points we have.
7. The total error is the summation(y-y’) that is for all the data point we will sum up the result.
8. We then calculate the mean or the average error that is dividing the total error by the number of outputs.
9. To remove the negative sign we first square the value and then to retain the original value we just take the root of the number.
10. There are possibilities that there can be negative values in the mean error so to remove that we find mean squared error.
11. Mean squared error is the square of the mean error.
12. We can also calculate the mean absolute error by using the modulus to get rid of the sign.
13. We can find the root mean square error by taking the square root of the mean square error.
14. We find the Rsquare Value to properly decide the error percentage of the algorithm.
15. In R2  numerator gives the variance of each point and the denominator gives the total variability.
16. R2   gives out the performance percentage of the algorithm.
17. R2 has a range between 0 to 1.
18. Adjusted R2 has a bit of change in the formula we divide the numerator by (n-p-1) and denominator by (n-1).
19. We use adjusted R2  when we want a better insight or a perfect result of the algorithm.

**Steps for Creating a Model  :**

1. We need to have the data.
2. We need to have a fixed objective.
3. Perform EDA on the data.
4. Try to plot the data with Matplotlib for two features we cannot do it for data with a lot of variables.
5. Data Preprocessing or handling missing values.
6. Handling NaN and NaT values.
7. Find the Nature of data
8. Assign the data to x and y depending on the data where x is always the input.
9. Split the data for training and testing.
10. Call the algorithm from sklearn

**Practical in Python of Simple Linear Regression:**

1. Import numpy as np, pandas as pd,matplotlib.pyplot as plt and seaborn as sns.
2. Import the data using pandas
3. In the following data, we have 2 features Number of claims in the total payment.
4. This is simple linear regression as we have only 2 features to work on.
5. We perform some basic steps.
6. W plot the histogram to make inferences as to when and in which area the mode is not working
7. Find the correlation of the data.
8. Plot the Scatterplot Graph.
9. Plot the pair plot.
10. We can also find the outliers fro the matplotlib boxplot or seaborn boxplot .
11. We will assign the features to X&Y.
12. x=data.iloc[:,:1] (Assigning no of clients to x )
13. y=data.iloc[:,1:] (Assigning total payment to y )
14. Plotting the scatter plot with x and y and then we can add title and xlabel and ylabel to the scatter plot.
15. Split the Record for timing and testing.
16. Import the model from sklearn.
17. From sklearn.model\_selection import train\_test\_split.
18. x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.2,random\_state=0) .2 test size says 20% of data is used for testing andd other for trying ,0 in random state defined the value we wanna pass for random state of model.
19. We check the number of data in training and trying by checking the shape

**Model Building with sklearn:**

1. from sklearn.linear\_model import linearregression
2. lin\_rag=linearregression()
3. We then train the model
4. lin\_red.fit(x\_train,y\_train)
5. lin\_reg.coef\_ gives the slope or m .
6. lin\_reg.intercept\_ gives the intercept or c .
7. plt.scatter(x\_train,y\_train,color=’red’)
8. plt.plot(x\_train,lin\_reg.predict(x\_train),color=green)

**Testing the Model :**

1. ypred-lin\_red.predict(x\_test)
2. ypred  # Returns y’
3. x\_test
4. We can then check with the data to see the difference .
5. We then will calculate all the error.

**Error Estimation in python :**

1. from sklearn.metrics import mean\_squared\_error,r2\_score
2. RMSE=np.sqrt(mean\_squared\_error(y\_test,ypred)
3. rsq=r2\_score(y\_test,ypred)
4. Print both the RMSE and R2 .
5. We will get the performance of the model as the output

**How to perform Forecasting :**

1. unseen\_pred=lin\_reg.predict\_(np.array([[108]]))#We use 108 as we know the value of it so to check we give 108.
2. So by  passing values into the function we can get the unseen values