Mini Project 2

(2019-2020)

On

Customer Churn Prediction

Final Report

Institute of Engineering & Technology

ANKUR OMAR
(171500051)

Supervised By:
Mrs. Harvinder Kour
(Technical Trainer)
Department of Computer Engineering & Applications
GLA University
Mathura-281406, India

INDEX

Page No.

ABSTRACT

This final report documents the amount of work done in the Mini Project during this date. The report first shall give an overview of the tasks completed during this period with technical details. Then the report shall be analyzed. Report shall also elaborate on the future works which are still to be persuaded as an advancement of current work. I have tried my best to keep the report simple yet technically correct.

ABOUT THE PROJECT

In this project we will trying to predict the customer churn(responses) by the help of machine learning technology. The objective of the project is to reducing customer churn by identifying the potential churn customer and take proactive actions to make them stay.

SYSTEM REQUIREMENTS

- Hardware Requirement**:** laptop
- Software Requirement**:** Anaconda – Jupyter, Idle Python (3.7 64 bit)
- Implementation Language: Python

DATASET DESCRIPTION

- The data given is in the form of a comma-separated values files with customer id and their corresponding gender. The training dataset is a csv file of type customer id, churn, gender where the customer id is a unique integer identifying the customer, churn is either 'Yes' or 'No'. Similarly, the test dataset is a csv file of type customer id, gender.

- The data set contains different types of features or columns. the features like customer id, gender, customer services , churn, etc. these features help to identify the customer churn. in our data set, there are 21 columns and approximately 7000 rows.

PROCESS DIAGRAM

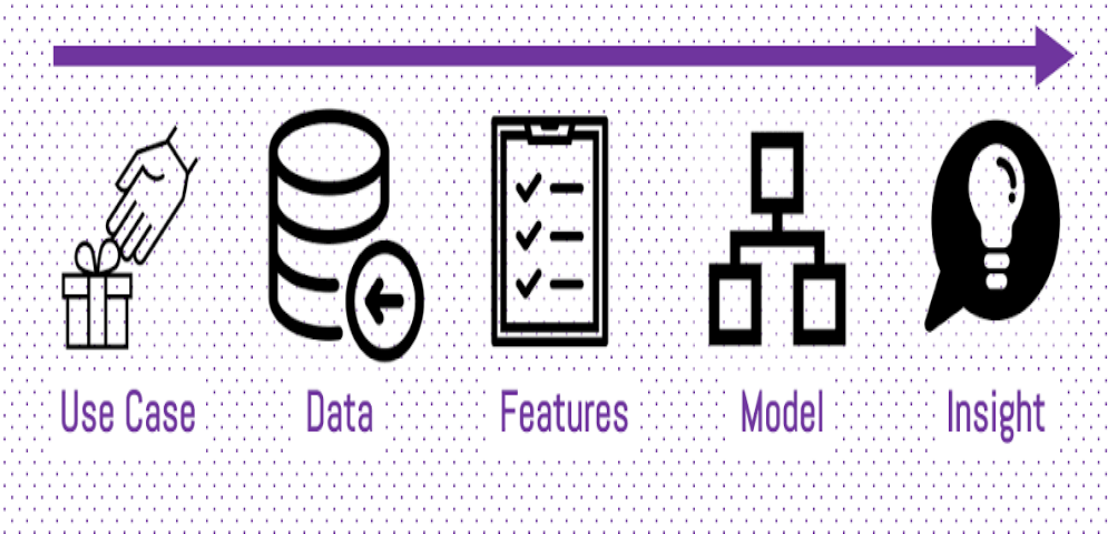


Fig:1

NECESSARY LIBRARIES

- Import numpy
- Import pandas as pd
- Import matplotlib
- Import seaborne
- Import pandas profiling

AREA OF DEVELOPMENT

This project deals with the area of Machine learning technology. The whole Project run on Jupyter Notebook with some important inbuilt python libraries which is help full to implement the machine learning concept. It is also being developed with major machine learning algorithms like Logistic Regression, Naïve byes, K-nearest neobhour,Decision Tree, Random forest .

IMPLEMENTATION

1. First we import the data by the help of pandas library and view the some top of the information given in the dataset.

```
df =pd.read_csv(r"C:\Users\me\Desktop\custumerchurn.csv")
df.head()
```

Output:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSup
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	...	No	
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	...	Yes	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	...	No	
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes	...	Yes	
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	...	No	

2: By using import pandas_profiling library we view the dataset with all columns which have in dataset and check missing values, and also check the data type of each columns. After that we found there is no missing values and the number of categorical columns is more as compare to numeric columns.

Output:-

```
import pandas_profiling
pandas_profiling.ProfileReport(pd.read_csv(r"C:\Users\me\Desktop\custumerchurn.csv"))
```

Dataset statistics

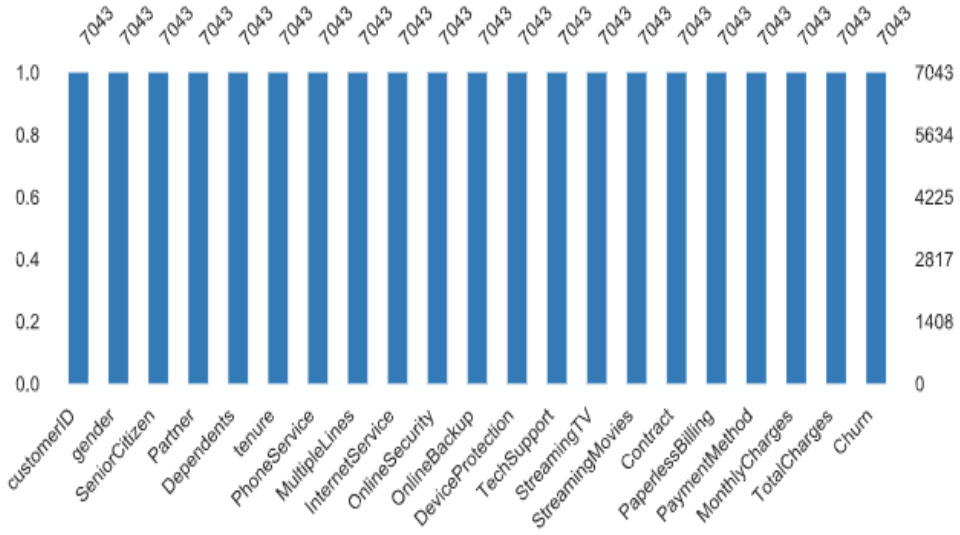
Number of variables	21
Number of observations	7043
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	7.8 MiB
Average record size in memory	1.1 KiB

Variable types

CAT	13
BOOL	6
NUM	2

3: Checking Missing Values:

There is no null or missing values in the dataset. You can see the values in each columns.



4:- Check data type of each columns:

Here you can see the number of object data types is more.

And there are only two integer type columns and only one float data type but here Total Charges column is object but we convert it from object data type to float data type. here object type is categorical features which show a class of ‘YES” and “NO”.

```
In [5]: df.dtypes

Out[5]: customerID          object
        gender              object
        SeniorCitizen        int64
        Partner             object
        Dependents          object
        tenure               int64
        PhoneService        object
        MultipleLines       object
        InternetService     object
        OnlineSecurity      object
        OnlineBackup        object
        DeviceProtection    object
        TechSupport         object
        StreamingTV         object
        StreamingMovies     object
        Contract            object
        PaperlessBilling    object
        PaymentMethod       object
        MonthlyCharges     float64
        TotalCharges        object
        Churn               object
        dtype: object
```

Feature Scaling:-

1:- The first step in feature scaling is to convert our target feature which is “Churn”, in to 0 or 1 form because our machine learning model will not be able to understand the string type values so we need to convert “yes” or “no” type values in to 0 or 1 form.

```
#converting churn value no ar
df.Churn[df.Churn=='No']=0
df.Churn[df.Churn=='Yes']=1
df
```

```
Out[25]: 0       0
         1       0
         2       1
         3       0
         4       1
                ..
         7038    0
         7039    0
         7040    0
         7041    1
         7042    0
         Name: Churn, Length: 7043, dtype: object
```

2:- In columns “OnlineSecurity”, “OnlineBackup”, “DeviceProtection”,”TechSupport”,”StremmingTv”,”StremmingMovie” have some values like “No Internet Service”, we convert this value with “No” in these columns.

```
In [9]: columns =['OnlineSecurity','OnlineBackup','DeviceProtection','TechSupport','StreamingTV','StreamingMovies']
        for i in columns:
            df[i] =df[i].replace({'No internet service':'No'})
```

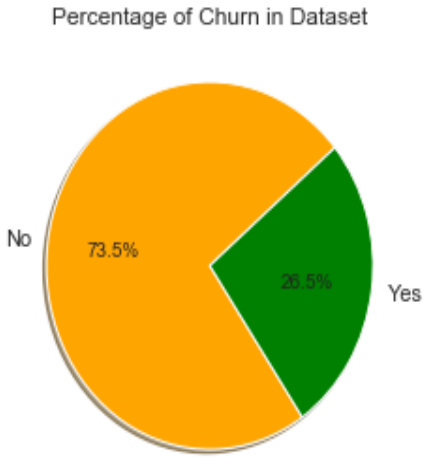
3:- The column “Total Charges” contain some spaces we replace all the spaces with null values and convert “Total Charges” feature to float values.

```
In [10]: df['TotalCharges'] =df['TotalCharges'].replace(" ",np.nan)
         df['TotalCharge'] =df['TotalCharges'].astype(float)
```

DATA VISUALIZATION

1:- First we visualize the output column data in form of graph –

In [12]:
```
sizes = df['Churn'].value_counts(sort = True)
colors = ["orange","green"]
labels =['No',"Yes"]
plt.pie(sizes, labels =labels,colors=colors,
        autopct='%1.1f%%', shadow=True, startangle=400,)
plt.title('Percentage of Churn in Dataset')
plt.show()
```

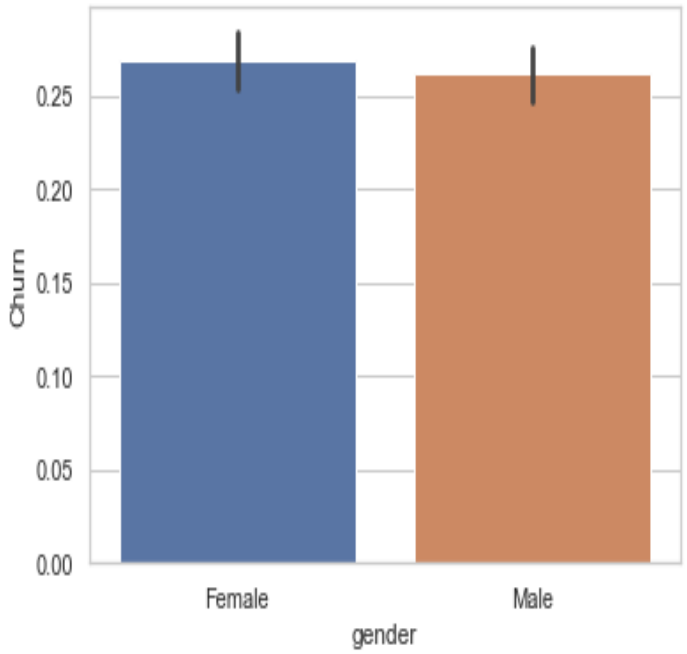


The above show the percentage of churn customer in a form of pie graph. As we can see that the number of customer who are not churned is more as compare to who are churned. There are 73.5% customer are not churn and 26.5% customer who are churn.

2:- After that we generate the graph between gender and churn –

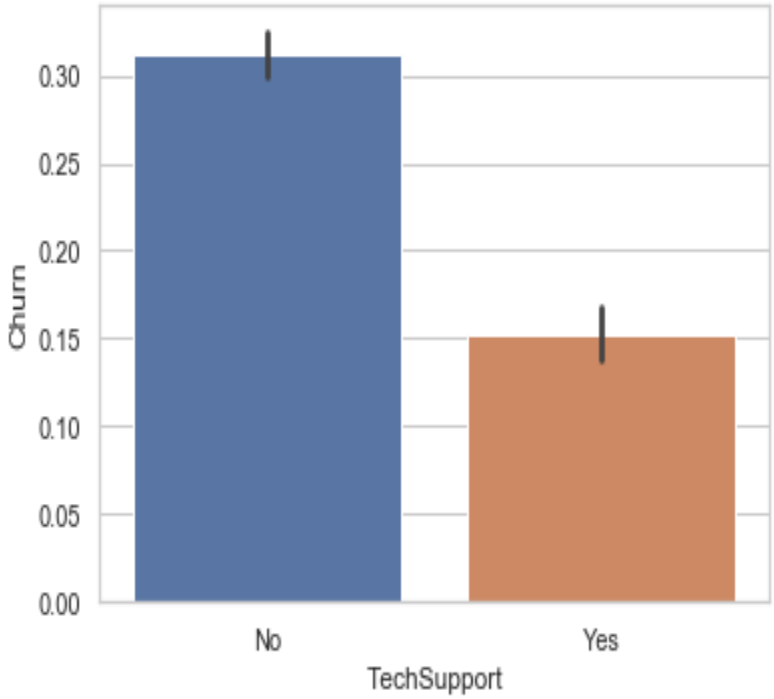
```
In [13]: #Data visulization part
         #churn rate visulisation by gender
         sbn.barplot(x ='gender',y ='Churn',data =df)
```

```
Out[13]: <matplotlib.axes._subplots.AxesSubplot at 0x25b682fe860>
```



3:- we show the graph between churn rate and tech support-

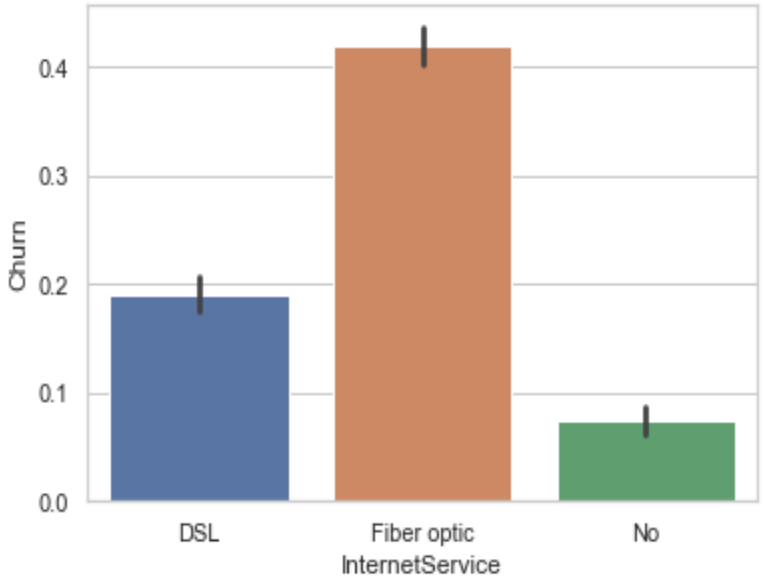
```
In [14]: #churn rate by tech support
         sbn.barplot(x= 'TechSupport',y ='Churn',data =df)

Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0x25b6860eb00>
```



4:- we show the graph between churn rate and internet services-

```
In [15]: # visulization of churn rate  by internet services
         sbn.barplot(x ='InternetService',y ='Churn',data =df)
```

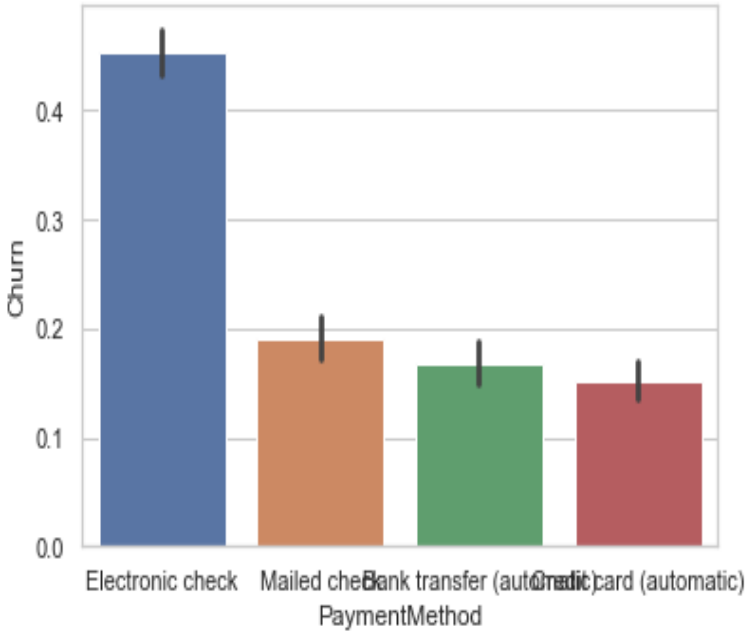
```
Out[15]: <matplotlib.axes._subplots.AxesSubplot at 0x25b68668ac8>
```



5:- we show the graph between churn rate and payment method-

```
In [16]: ## visulization of churn rate  by payment method
         sbn.barplot(x ='PaymentMethod',y ='Churn',data =df)
```

```
Out[16]: <matplotlib.axes._subplots.AxesSubplot at 0x25b686afa58>
```



ONEHOT ENCODING

A one hot encoding allows the representation of categorical data to be more expressive. Many machine learning algorithms cannot work with categorical data directly. The categories must be converted into numbers. This is required for both input and output variables that are categorical. We could use an integer encoding directly, rescaled where needed. This may work for problems where there is a natural ordinal relationship between the categories,

and in turn the integer values, such as labels for temperature ‘cold’, warm’, and ‘hot’.

In our data set there are many features are in form of categories like “yes” and “No” form. So we need to apply the one hot encoding concept in our categorical features. It done by the method get_dummies() it make some dummies features and we drop first feature from dummies features which would made by get_dummies() method.

```
In [18]: #perform onehot encoding by the method get_dummies()
         df =pd.get_dummies(df,columns =['Contract','Dependents','DeviceProtection','gender',
                                         'InternetService','MultipleLines','OnlineBackup','OnlineSecurity','PaperlessBilling',
                                         'Partner','PaymentMethod','PhoneService',
                                         'SeniorCitizen','StreamingMovies','StreamingTV','TechSupport'

         ],drop_first =True)
```

Convert MonthlyCharges and TotalCharges and Tenure in same categorical values range-

```
from sklearn.preprocessing import StandardScaler
stander_Scaler =StandardScaler()
columns_feature_scaling =['MonthlyCharges','TotalCharges','tenure']
df[columns_feature_scaling] =stander_Scaler.fit_transform(df[columns_feature_scaling])
```

	tenure	MonthlyCharges	TotalCharges	Churn	TotalCharge	Contract_One year	Contract_Two year	Dependents_Yes	DeviceProtection_Yes	gender_Male	...
0	-1.28	-1.16	-0.99	0	29.85	0	0	0	0	0	...
1	0.07	-0.26	-0.17	0	1889.50	1	0	0	1	1	...
2	-1.24	-0.36	-0.96	1	108.15	0	0	0	0	1	...
3	0.51	-0.75	-0.20	0	1840.75	1	0	0	1	1	...
4	-1.24	0.20	-0.94	1	151.65	0	0	0	0	0	...

.	PaperlessBilling_Yes	Partner_Yes	PaymentMethod_Credit card (automatic)	PaymentMethod_Electronic check	PaymentMethod_Mailed check	PhoneService_Yes	SeniorCitizen_1	Strea
.	1	1	0	1	0	0	0	
.	0	0	0	0	1	1	0	
.	1	0	0	0	1	1	0	
.	0	0	0	0	0	0	0	
.	1	0	0	1	0	1	0	

Here we have completed the data analysis, cleaning data ,data visualization and handle the missing values in the dataset and we have also completed the one hot encoding process for categorical data..

In further steps we have applied machine learning algorithms and split the data set in the training dataset and the testing dataset. The training set will have targets/labels, while the testing set won't contain these values.

We know that the given dataset in categorical form so here we applied categorical supervised algorithms like-

- Logistic Regression
- K-Nearest Neighbor(KNN) Classifier
- Support Vector Machine(SVM)
- Random Forest Classifier
- Decision Tree Classifier

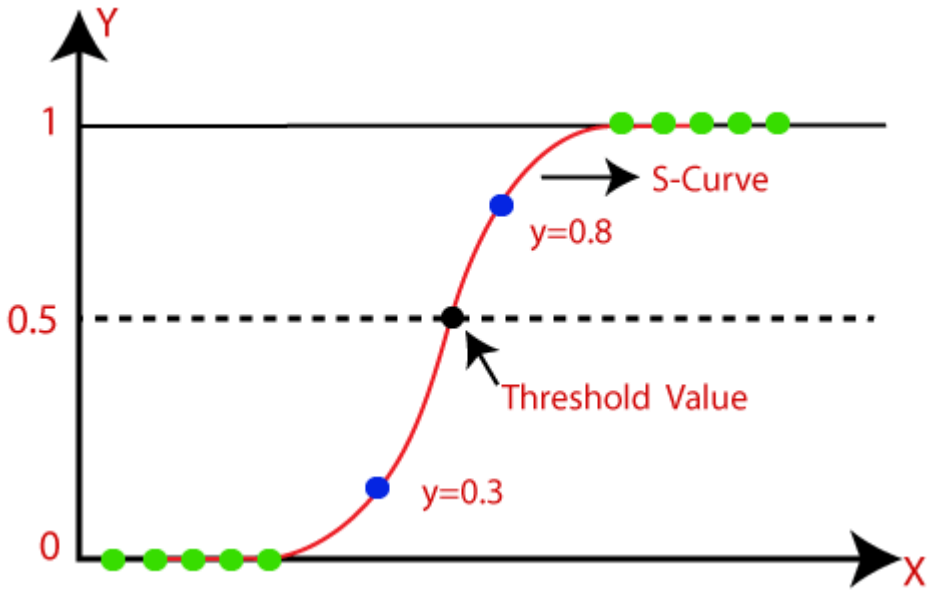
MACHINE LEARNING ALGORITHMS

1.1- LOGISTIC REGRESSION-

- Logistic regression is one of the most popular machine learning algorithms, which comes under the supervised learning technique. It is used for predicting the categorical dependent variable using a given set of independent variable.

- Logistic regression predict the output of a categorical dependent variable. Therefore the outcome must be categorical or discrete value. It can be either ‘yes’ or ‘No’,0 or 1, and ‘true’ or ‘false’ etc. But instead of giving exact value 0 and 1 it gives probalistic value which lies between 0 and 1.

- In logistic regression instead of fitting a regression line we fit as “S” shaped logistic function, which predicts two values 0 and 1 maximum.



Logistic function or Sigmoid function-

The sigmoid function a mathematical function used to map the predicted values to probabilities.it maps any real value in to another with in a range of 0 and 1.

$$f(x) = \frac{1}{1 + e^{-(x)}}$$

Logistic Equation-

$$y = b_0 + b_1 x_1 + b_2 x_2 + b_3 x_3 + \cdots + b_n x_n$$

In Logistic Regression y can be between 0 and 1 only, so for this let's divide the above equation by (1-y)

$\frac{y}{1-y}$; 0 for y= 0, and infinity for y=1

$$log\left[\frac{y}{1-y}\right] = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \cdots + b_nx_n$$

Cost Function- We cannot use the same cost function that we use for linear regression because the Logistic Function will cause the output to be wavy, causing many local optima. In other words, it will not be a convex function.

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \text{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

$$\text{Cost}(h_\theta(x), y) = -\log(h_\theta(x)) \quad \text{if } y = 1$$

$$\text{Cost}(h_\theta(x), y) = -\log(1 - h_\theta(x)) \quad \text{if } y = 0$$

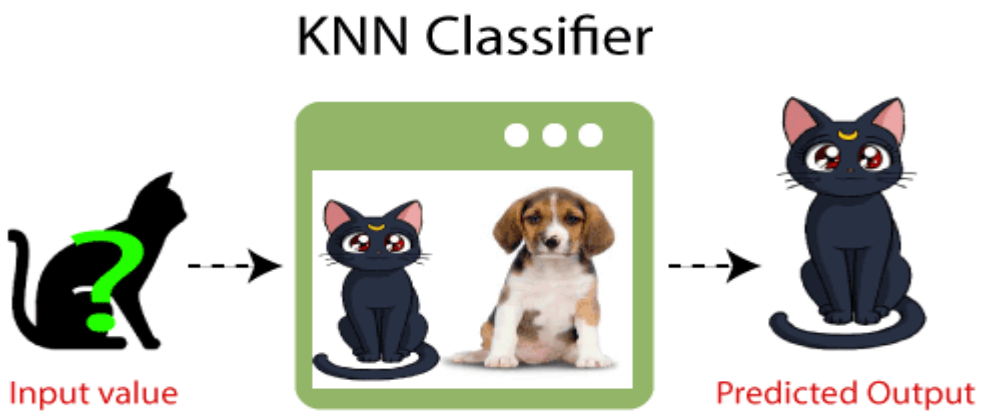
1.2 K- NEAREST NEIBHBOR CLASSIFIER (KNN)-:

- k-nearest neighbor is one of the simplest machine learning algorithm based on supervised learning technique. KNN algorithm stores all the available data and classify a new data point based on a similarity. This means when new data appears it can be easily classified into a well suit category by using K-NN algorithm.

- K-NN algorithm can be used for regression as well as classification but mostly it can be used for classification problem.

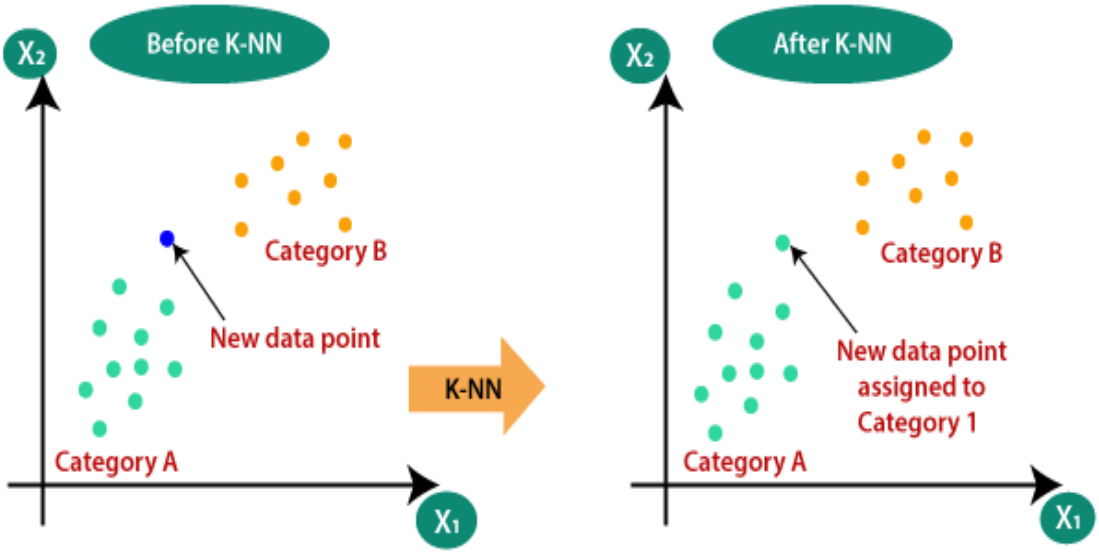
- It is also called lazy learner algorithm because it does not learn from training set immediately instead it stores the data set at time of classification, it performs an action on dataset.

Example- if we take one example to understand the concept of KNN algorithm that is-
Suppose we have a image of creatures which looks similar to cat or dog. But we want to know it is either cat or dog.so for this identification we can use KNN algorithm.

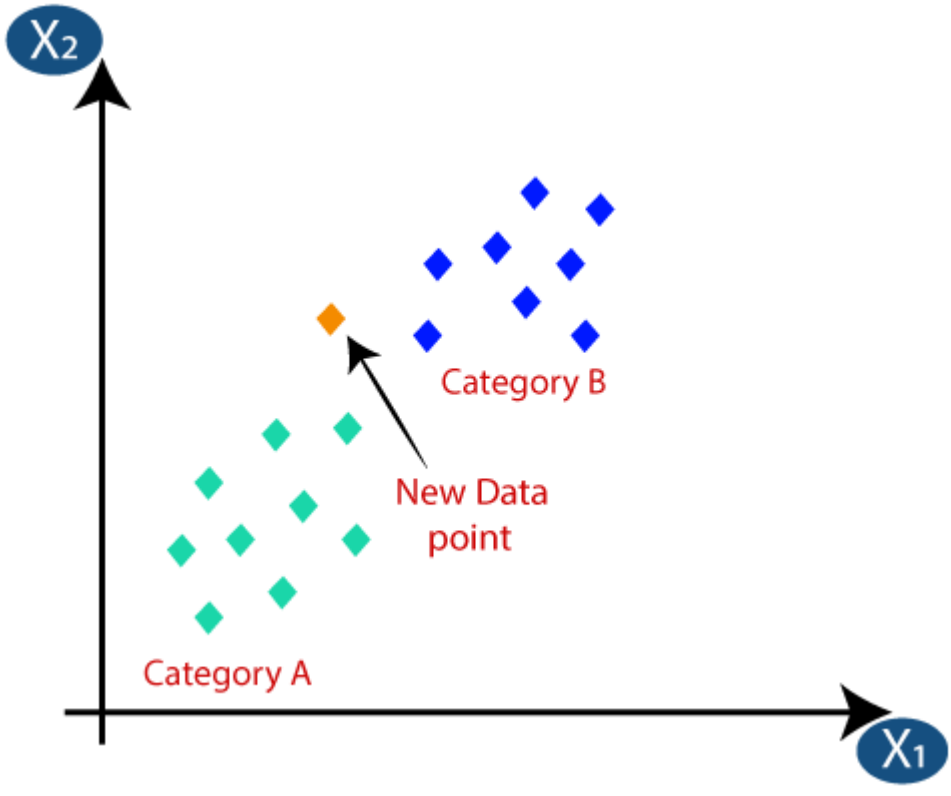


Graph Representation of New Data point-

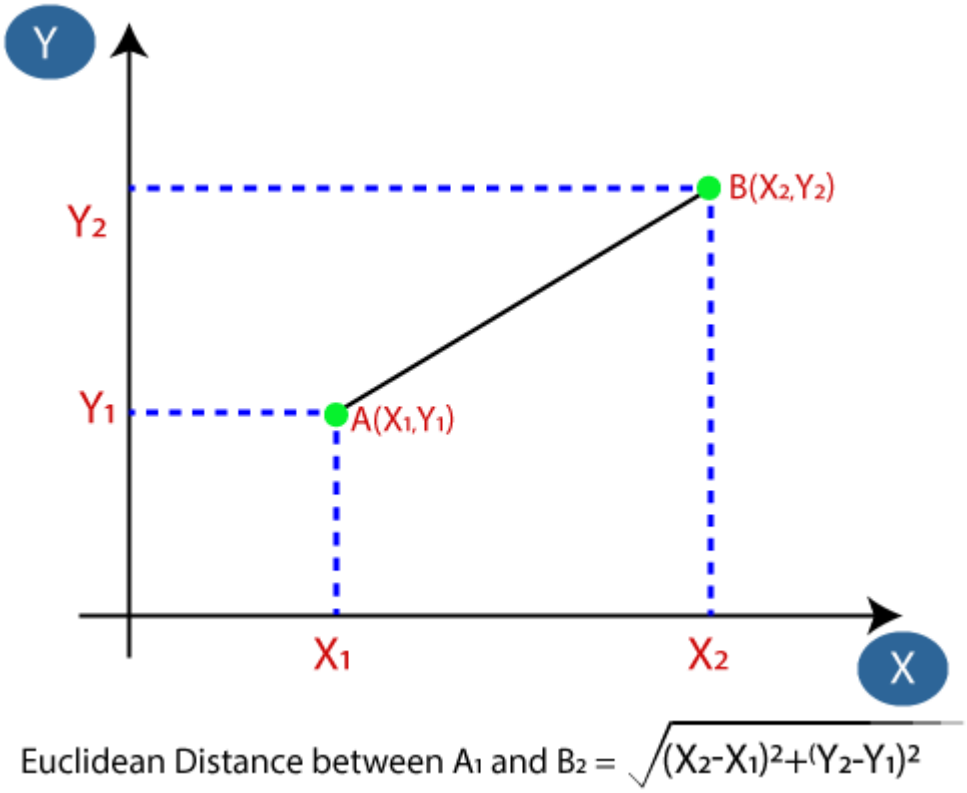
suppose there are two categories that is category A and category B and we have a new data point x1,so this data point lies in which of these categories.to solve this type of problem we need KNN algorithm. With the help of KNN we can easily solve this kind of problem and we can easily identify the class or category for each data points.



Suppose we have a new data point and we need to put in a required category –

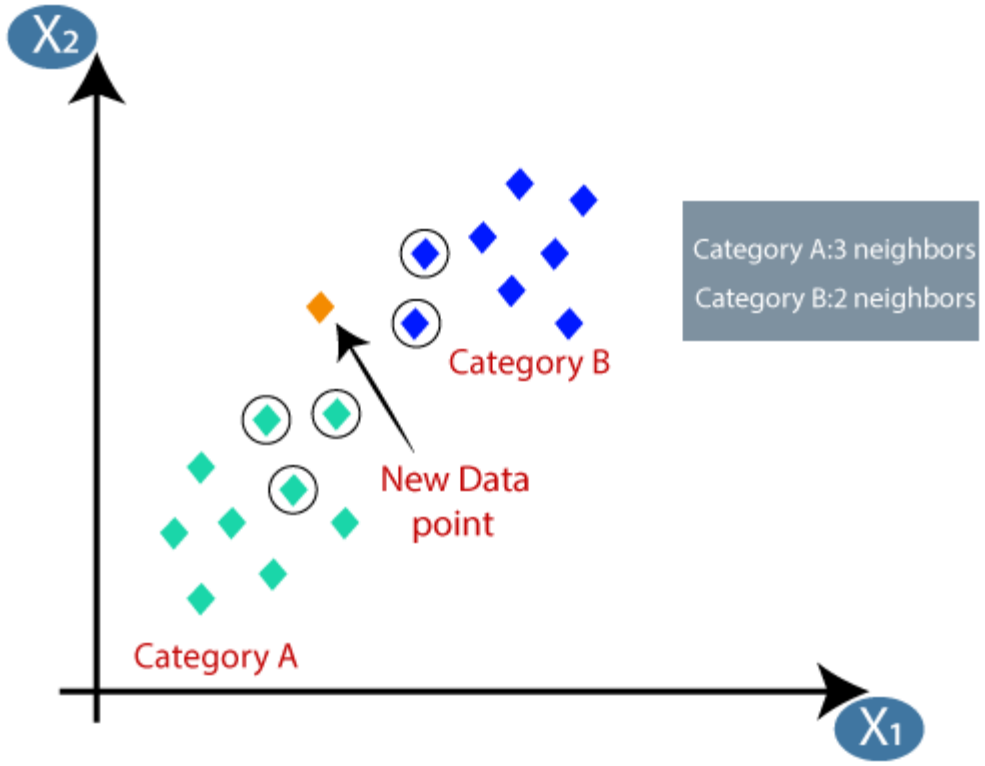


- Firstly we will choose the number of neighbors so we will choose the k =5
- After choose the value of k we will calculate the Euclidean distance between the data points.
- It can be calculated as-



Euclidean Distance between A_1 and B_2 = $\sqrt{(X_2-X_1)^2+(Y_2-Y_1)^2}$

- By calculating the Euclidean distance we got the nearest neighbors, as three nearest neighbors in category A and two nearest neighbors in category

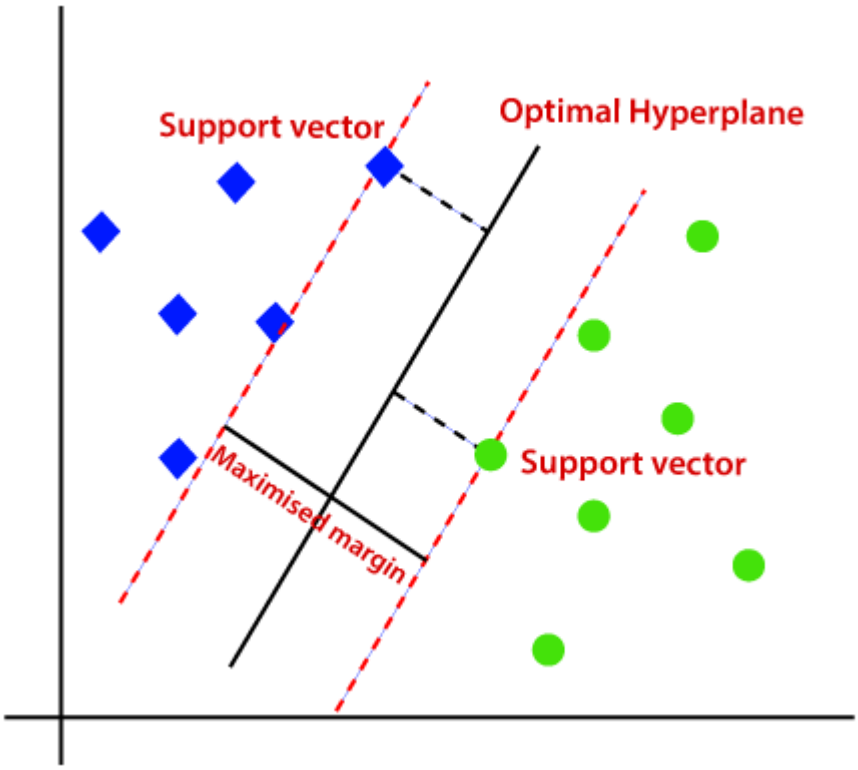


As we can see the three nearest neighbors are from category A so this data point belong to category A.

1.3 SUPPORT VECTOR MACHINE ALGORITHM

- Support vector machine or SVM is one of the most popular supervised algorithms. This is used for both classifications as well as regression problem. But primarily it is used for classification problem.

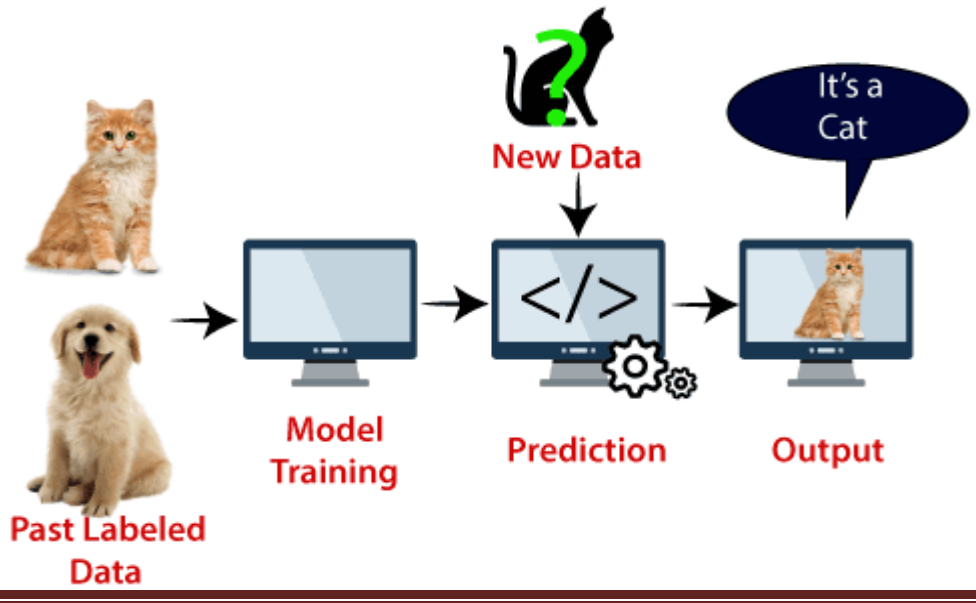
- The goal or target of support vector machine is to create the best line or decision boundary that can segregate n – dimensional space in to classes so that we can easily put the new data point in the correct category in future. This best decision boundary is called hyper plane.



Support Vector:- the data points or vectors that are closest to hyper plane and which affect the position of hyper plane is called as support vector.

EXAMPLE-

SVM can be understood with the example that we have used in the KNN classifier. Suppose we see a strange cat that also has some features of dogs, so if we want a model that can accurately identify whether it is a cat or dog, so such a model can be created by using the SVM algorithm. We will first train our model with lots of images of cats and dogs so that it can learn about different features of cats and dogs, and then we test it with this strange creature. So as support vector creates a decision boundary between these two data (cat and dog) and choose extreme cases (support vectors), it will see the extreme case of cat and dog. On the basis of the support vectors, it will classify it as a cat.



1.4 RANDOM FOREST ALGORITHM

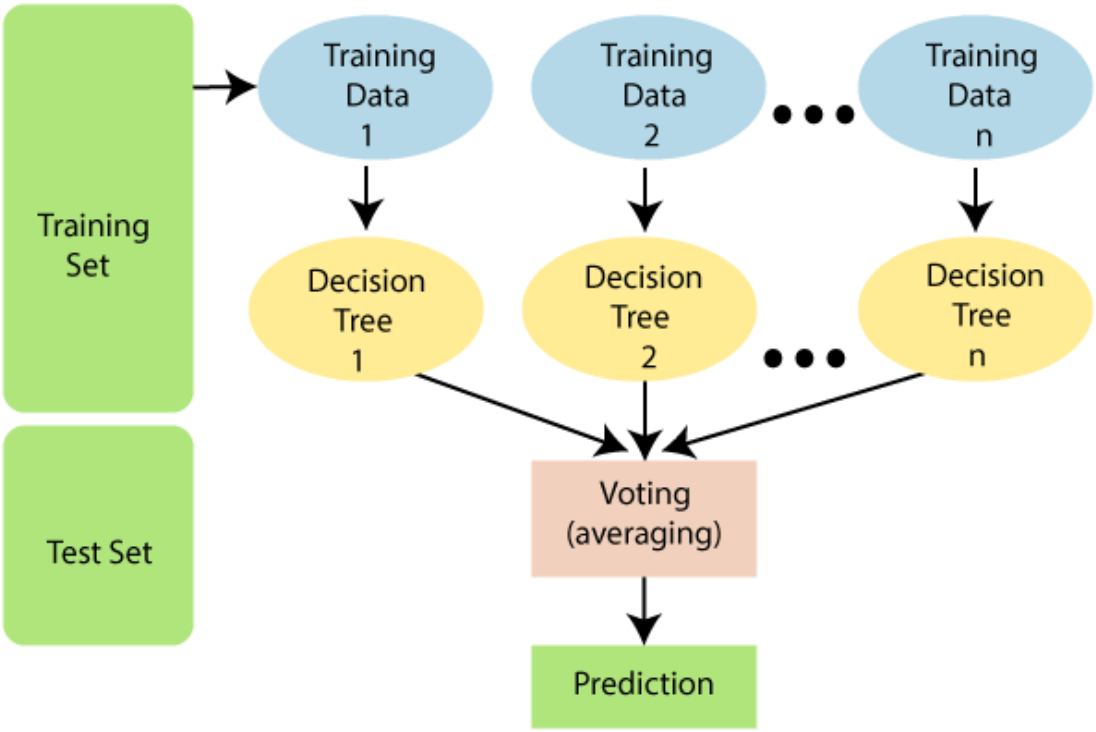
- Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML.

- It is based on the concept of **ensemble learning,** which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

- Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset

- Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of over fitting.



Assumption for Random Forest-

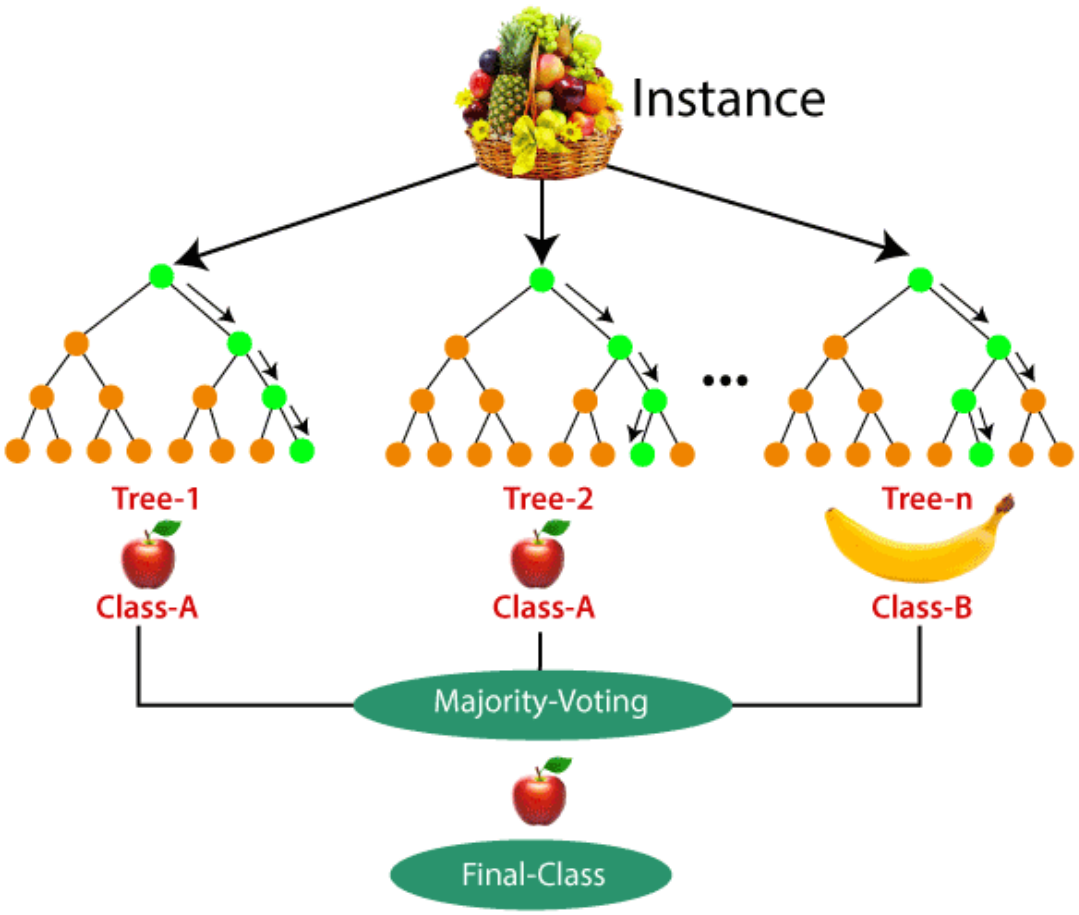
- There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.
- The predictions from each tree must have very low correlations.

Steps To perform Random forest-

1) Select random K data points from the training set.
2) Build the decision trees associated with the selected data points.
3) Choose the number N for decision trees that you want to build.
4) Repeat Step 1 & 2
5) For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

EXAMPLE-

Suppose there is a dataset that contains multiple fruit images. So, this dataset is given to the Random forest classifier. The dataset is divided into subsets and given to each decision tree. During the training phase, each decision tree produces a prediction result, and when a new data point occurs, then based on the majority of results, the Random Forest classifier predicts the final decision.


Instance
Tree-1
Tree-2
Tree-n
Class-A
Class-A
Class-B
Majority-Voting
Final-Class

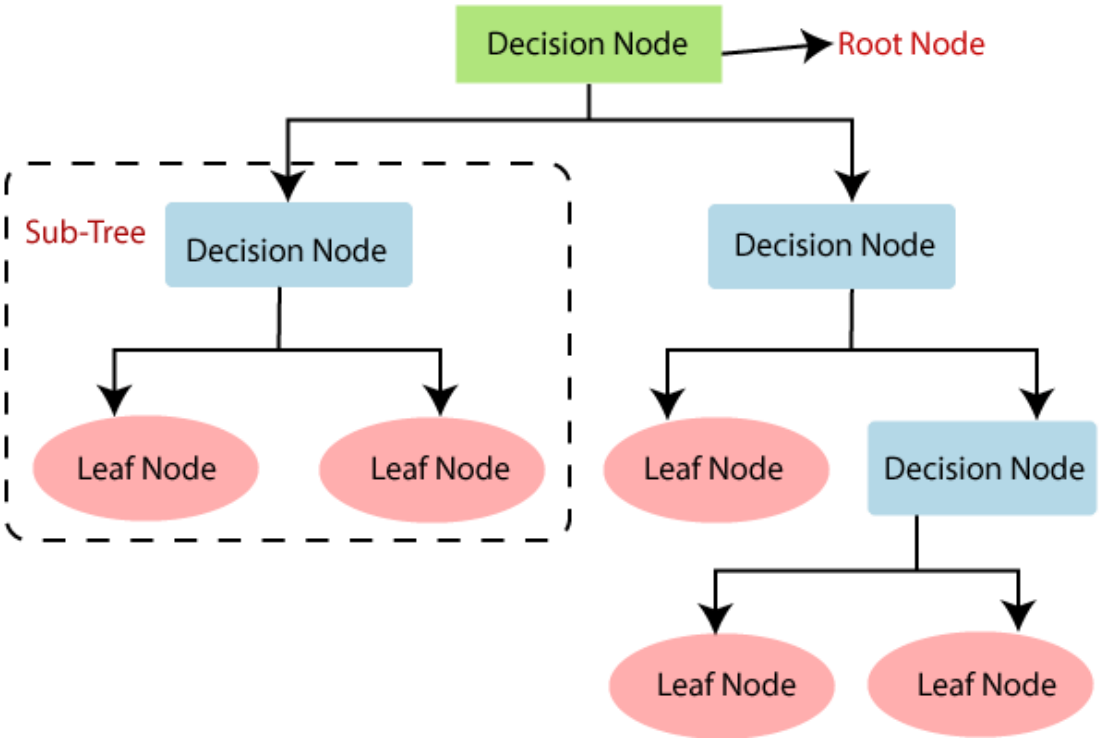
1.5 DECISION TREE CLASSIFICATION ALGORITHM

- Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.

- In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node**.** Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.

- It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.

- A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into sub trees.

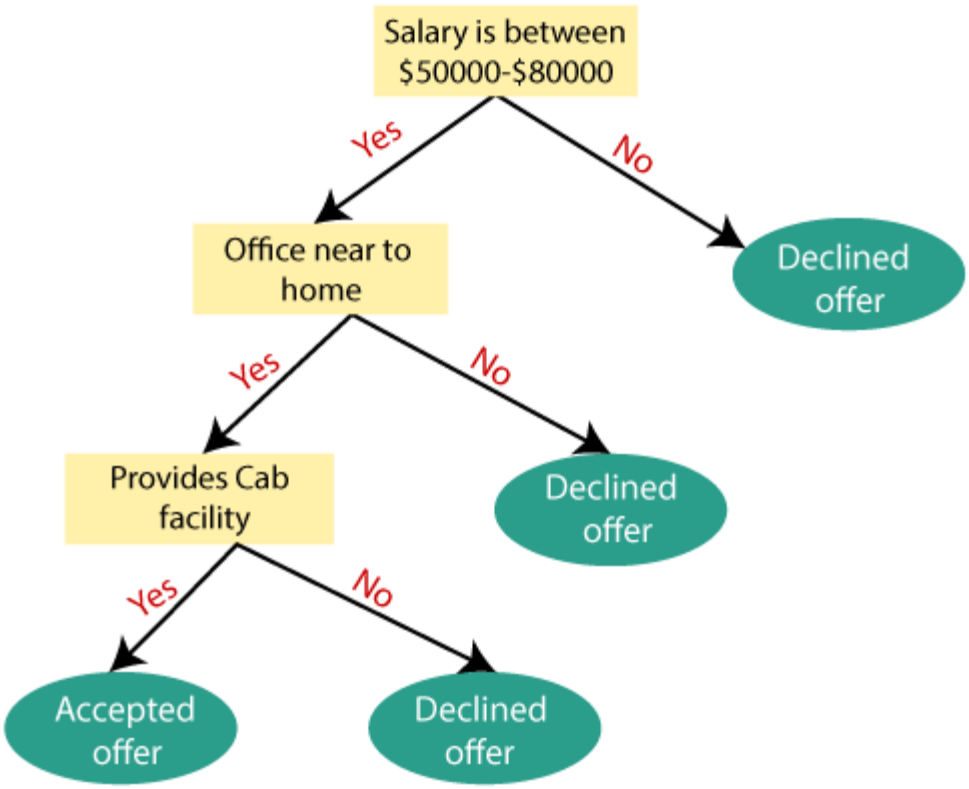


Steps to perform decision tree Algorithm-

1) Begin the tree with the root node, says S, which contains the complete dataset.
2) Find the best attribute in the dataset using Attribute Selection Measure.
3) Divide the S into subsets that contains possible values for the best attributes.
4) Generate the decision tree node, which contains the best attribute.
5) Recursively make new decision trees using the subsets of the dataset created in 3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

Example:

Suppose there is a candidate who has a job offer and wants to decide whether he should accept the offer or Not. So, to solve this problem, the decision tree starts with the root node (Salary attribute). The root node splits further into the next decision node (distance from the office) and one leaf node based on the corresponding labels. The next decision node further gets split into one decision node (Cab facility) and one leaf node. Finally, the decision node splits into two leaf nodes (Accepted offers and Declined offer)



Algorithm Implementation Details With Python code

1.1 LOGISTIC REGRESSION-:

```
y =df['Churn']
X =df.drop(['Churn','customerID'],axis =1)
```

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test =train_test_split(X,y,test_size =0.30,random_state =0)
```

```
In [23]: # here now we start to implement algorithms

         #LOGISTIC REGRESSION

         from sklearn.linear_model import LogisticRegression
         from sklearn import metrics
         l_model =LogisticRegression(random_state =50)
         l_model.fit(x_train,y_train)
         pred =l_model.predict(x_test)
         model_accuracy =round(metrics.accuracy_score(y_test,pred)*100,2)
         print("THE accuracy of logistic regression model is",model_accuracy)

         THE accuracy of logistic regression model is 80.09
```

1.2 Random Forest Classifier

In [26]:
```
#RANDOM FOREST CLASSIFIERS
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics
r_model = RandomForestClassifier(n_estimators = 100, criterion = 'entropy', random_state = 0)
r_model.fit(x_train, y_train)
r_pred = r_model.predict(x_test)
r_accuracy = round(metrics.accuracy_score(y_test, r_pred) * 100, 2)
print("THE accuracy of random forest  model is",r_accuracy)
```

```
THE accuracy of random forest  model is 78.44
```

1.3 Decision Tree Classifier

In [27]:
```
#DECISION TREE CLASSIFIERS
from sklearn.tree import DecisionTreeClassifier
from sklearn import metrics
d_model = DecisionTreeClassifier(criterion = "gini", random_state = 50)
d_model.fit(x_train, y_train)
d_pred = d_model.predict(x_test)
d_accuracy = round(metrics.accuracy_score(y_test, d_pred) * 100, 2)
print("THE accuracy of decision tree model is",d_accuracy)
```

```
THE accuracy of decision tree model is 71.23
```

1.4 Support Vector Machine

In [28]:
```
from sklearn.svm import SVC
from sklearn import metrics
s_model = SVC(kernel='linear', random_state=50, probability=True)
s_model.fit(x_train,y_train)
s_pred = s_model.predict(x_test)
s_accuracy = round(metrics.accuracy_score(y_test, s_pred) * 100, 2)
print("THE accuracy of support vector machine  model is",s_accuracy)
```

```
THE accuracy of support vector machine  model is 80.33
```

1.5 K-Nearest Neighbor Classifier

In [25]:
```
#K-NEAREST NEIGHBORS CLASSIFIERS
from sklearn.neighbors import KNeighborsClassifier
from sklearn import metrics
k_model = KNeighborsClassifier(n_neighbors=5, metric='minkowski', p=2)
k_model.fit(x_train, y_train)
k_pred = k_model.predict(x_test)
knn_accuracy = round(metrics.accuracy_score(y_test, k_pred) * 100, 2)
print("THE accuracy of k-nearest neighbors  model is",knn_accuracy)
```

```
THE accuracy of k-nearest neighbors  model is 76.26
```

Let's compare the accuracy of each model

In [29]:
```
#now we comparison aur model accuracy
import pandas as pd
model_comp ={'Model':['LOGISTIC REGRESSION','K-NEAREST NEIGHBORS CLASSIFIERS','RANDOM FOREST CLASSIFIERS',
                      'DECISION TREE CLASSIFIER'
                           ,'SUPPORT VECTOR MACHINE'  ],'Score':[model_accuracy,knn_accuracy,r_accuracy,d_accuracy,s_accuracy]}
df1 =pd.DataFrame(model_comp)
Model_Compa = df1.sort_values(by='Score', ascending=False)
Model_Compa = Model_Compa.set_index('Model')
Model_Compa.reset_index()
```

Out[29]:

	Model	Score
0	SUPPORT VECTOR MACHINE	80.33
1	LOGISTIC REGRESSION	80.09
2	RANDOM FOREST CLASSIFIERS	78.44
3	K-NEAREST NEIGHBORS CLASSIFIERS	76.26
4	DECISION TREE CLASSIFIER	71.23

Confusion Matrix

In [30]:
```
from sklearn.metrics import confusion_matrix
conf_mat_model = confusion_matrix(y_test,pred)
conf_mat_model
```

Out[30]:
```
array([[1391,  164],
       [ 256,  299]], dtype=int64)
```

FINAL OUTPUT

In [31]:
```
# Predict the probability of Churn of each customer
df['Probability of Churn'] = s_model.predict_proba(df[x_test.columns])[:,1]
```

In [32]:
```
# Create a Dataframe showcasing probability of Churn of each customer
df[['customerID','Probability of Churn']].head(10)
```

Out[32]:

	customerID	Probability of Churn
0	7590-VHVEG	0.45
1	5575-GNVDE	0.06
2	3668-QPYBK	0.21
3	7795-CFOCW	0.05
4	9237-HQITU	0.71
5	9305-CDSKC	0.83
6	1452-KIOVK	0.42
7	6713-OKOMC	0.22
8	7892-POOKP	0.59
9	6388-TABGU	0.01

CONCLUSION

- Customer churn is a major problem and one of the most important concerns for large companies. Due to the direct effect on the revenues of the companies, especially in the telecom field, companies are seeking to develop means to predict potential customer to churn. Therefore, finding factors that increase customer churn is important to take necessary actions to reduce this churn.

- The main contribution of our work is to develop a churn prediction model which assists telecom operators to predict customers who are most likely subject to churn.

- The model developed in this work uses machine learning techniques. The model experimented four algorithms: Decision Tree, Random Forest, Logistic Regression and KNN Algorithm and Support Vector Machine. However, the best results were obtained by applying Support Vector Machine algorithm. This algorithm was used for classification in this churn predictive model.

REFERENCE

- https://www.geeksforgeeks.org/machine-learning/
- https://expertsystem.com/blog/machine-learning/
- https://www.coursera.org/learn/machine-learning
- https://towardsdatascience.com/machine-learning-an-introduction-23b84d51e6d0
- https://www.javatpoint.com/