Intelligent Customer Retention Using Machine Learning For Enhanced Prediction Of Telecom Customer Churn

Done by,

Sriram.B
Selvam.C
Pravin.S
Sabarisan.B

INTELLIGENT CUSTOMER RETENTION USING MACHINE LEARNING FOR ENHANCED PREDICTION OF TELECOM CUSTOMER CHURN

1. Introduction

1.1 Overview

Customer churn is often referred to as customer attrition, or customer defection which is the rate at which the customers are lost. 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. Looking at churn, different reasons trigger customers to terminate their contracts, for example better price offers, more interesting packages, bad service experiences or change of customers' personal situations. Customer churn has become highly important for companies because of increasing competition among companies, increased importance of marketing strategies and conscious behaviour of customers in the recent years. Customers can easily trend toward alternative services. Companies must develop various strategies to prevent these possible trends, depending on the services they provide. During the estimation of possible churns, data from the previous churns might be used. An efficient churn predictive model benefits companies in many ways. Early identification of customers likely to leave may help to build cost effective ways in marketing strategies. Customer retention campaigns might be limited to selected customers but it should cover most of the customer. Incorrect predictions could result in a company losing profits because of the discounts offered to continuous subscribers. Telecommunication industry always suffers from a very high churn rates when one industry offers a better plan than the previous there is a high possibility of the customer churning from the present due to a better plan in such a scenario it is very difficult to avoid losses but through prediction we can keep it to a minimal level. Telecom companies often use customer churn as a key business metrics to predict the number of customers that will leave a telecom service provider. A machine learning model can be used to identity the probable churn customers and then makes the necessary business decisions.

Requirements

The business requirements for a machine learning model to predict whether the customer will churn or not on customer information, to minimise the number of false positives (customer that predicted as loyal but churn) and false negatives (customer predicted to be churn which could have stayed loyal). Provide an explanation for the model's decision, for better decision making in order to gain more profitability.

literature survey

As the data is increasing daily due to digitization in the banking sector, people want to apply for loans through the internet. Machine Learning (ML), as a typical method for information investigation, has gotten more consideration increasingly. Individuals of various businesses are utilising ML calculations to take care of the issues dependent on their industry information. Telecom companies often use customer churn as a key business metrics topredict the number of customers that will leave a telecom service provider. A machine learning model can be used to identity the probable churn customers and then makes the necessary business decisions

Impact

Social Impact:- Proposed model can help improve the overall customer experience and service quality. Companies can also make better decisions about how to retain their customers.

Business Model/ Impact :- This product can generate revenue using a product based model, where the system can be sold as a product to the telecom companies. This product can also be used for subscription based model.

Collect The Dataset

There are many popular open sources for collecting the data. Eg: kaggle.com, UCI repository, etc.

In this project we have used .csv data. This data is downloaded from kaggle.com. Please refer to the link given below to download the dataset.

Link: https://www.kaggle.com/shrutimechlearn/churn-modelling

As the dataset is downloaded. Let us read and understand the data properly with the help of some visualisation techniques and some analysing techniques.

Note: There are a number of techniques for understanding the data. But here we have used some of it. In an additional way, you can use multiple techniques.

Importing the libraries

Import the necessary libraries as shown in the image.

```
#import necessary libraries
import pandas as pd
import numpy as np
import pickle
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import sklearn
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import  RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.model_selection import RandomizedSearchCV
import imblearn
from imblearn.over_sampling import SMOTE
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, f1_score
```

Read the dataset

Our dataset format might be in .csv, excel files, .txt, .json, etc. We can read the dataset with the help of pandas

In pandas we have a function called read_csv() to read the dataset. As a parameter we have to give the directory of the csv file

```
#import dataset
data = pd.read_csv(r"C:\Users\Shivani_SB\OneDrive\Desktop\Telecom churn modelling-updated\data\DataSet.csv")
data
```

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupport	StreamingTV
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	...	No	No	No
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	...	Yes	No	No
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	...	No	No	No
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes	...	Yes	Yes	No
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	...	No	No	No
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes	DSL	Yes	...	Yes	Yes	Yes
7039	2234-XADUH	Female	0	Yes	Yes	72	Yes	Yes	Fiber optic	No	...	Yes	No	Yes
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phone service	DSL	Yes	...	No	No	No

Data Preparation

As we have understood how the data is, let's pre-process the collected data.

The download data set is not suitable for training the machine learning model as it might have so much randomness so we need to clean the dataset properly in order to fetch good results. This activity includes the following steps.

- ✓ Handling missing values
- ✓ Handling categorical data
- ✓ Handling Imbalance Data

Handling Missing Values

Let’s find the shape of our dataset first. To find the shape of our data, the df.shape method is used. To find the data type, data.info() function is used.

```
data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 20 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   gender            7043 non-null   object 
 1   SeniorCitizen     7043 non-null   int64  
 2   Partner           7043 non-null   object 
 3   Dependents        7043 non-null   object 
 4   tenure            7043 non-null   int64  
 5   PhoneService      7043 non-null   object 
 6   MultipleLines     7043 non-null   object 
 7   InternetService   7043 non-null   object 
 8   OnlineSecurity    7043 non-null   object 
 9   OnlineBackup      7043 non-null   object 
 10  DeviceProtection  7043 non-null   object 
 11  TechSupport       7043 non-null   object 
 12  StreamingTV       7043 non-null   object 
 13  StreamingMovies   7043 non-null   object 
 14  Contract          7043 non-null   object 
 15  PaperlessBilling  7043 non-null   object 
 16  PaymentMethod     7043 non-null   object 
 17  MonthlyCharges    7043 non-null   float64
 18  TotalCharges      7043 non-null   object 
 19  Churn             7043 non-null   object 
dtypes: float64(1), int64(2), object(17)
memory usage: 1.1+ MB
```

- For checking the null values, df.isnull() function is used. To sum those null values we use .sum() function. From the below image we found that there are no null values present in our dataset. So we can skip handling the missing values step.

```
#checking for null values
data.TotalCharges = pd.to_numeric(data.TotalCharges, errors='coerce')
data.isnull().any()
```

```
gender              False
SeniorCitizen       False
Partner             False
Dependents          False
tenure              False
PhoneService        False
MultipleLines       False
InternetService     False
OnlineSecurity      False
OnlineBackup        False
DeviceProtection    False
TechSupport         False
StreamingTV         False
StreamingMovies     False
Contract            False
PaperlessBilling    False
PaymentMethod       False
MonthlyCharges      False
TotalCharges         True
Churn               False
dtype: bool
```

- From the above code of analysis, we can infer that column TotalCharges is having the missing values, we need to treat them in a required way.

```
data["TotalCharges"].fillna(data["TotalCharges"].median() , inplace =True)
```

```
data.isnull().sum()
```

```
gender              0
SeniorCitizen       0
Partner             0
Dependents          0
tenure              0
PhoneService        0
MultipleLines       0
InternetService     0
OnlineSecurity      0
OnlineBackup        0
DeviceProtection    0
TechSupport         0
StreamingTV         0
StreamingMovies     0
Contract            0
PaperlessBilling    0
PaymentMethod       0
MonthlyCharges      0
TotalCharges        0
Churn               0
dtype: int64
```

We will fill in the missing values in the TotalCharges column by median as it's a numercal column and then again we checked for null values to see if there is any null value left.

Handling Categorical Values

As we can see our dataset has categorical data we must convert the categorical data to integer encoding or binary encoding.

To convert the categorical features into numerical features we use encoding techniques. There are several techniques but in our project we are using manual encoding with the help of list comprehension.

Label Encoding

Label Encoding refers to converting the labels into numeric form so as to convert it into the machine-readable form. Machine learning algorithms can then decide in a better way on how those labels must be operated. It is an important pre-processing step for the structured dataset in supervised learning.

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
data["gender"] = le.fit_transform(data["gender"])
data["Partner"] = le.fit_transform(data["Partner"])
data["Dependents"] = le.fit_transform(data["Dependents"])
data["PhoneService"] = le.fit_transform(data["PhoneService"])
data["MultipleLines"] = le.fit_transform(data["MultipleLines"])
data["InternetService"] = le.fit_transform(data["InternetService"])
data["OnlineSecurity"] = le.fit_transform(data["OnlineSecurity"])
data["OnlineBackup"] = le.fit_transform(data["OnlineBackup"])
data["DeviceProtection"] = le.fit_transform(data["DeviceProtection"])
data["TechSupport"] = le.fit_transform(data["TechSupport"])
data["StreamingTV"] = le.fit_transform(data["StreamingTV"])
data["StreamingMovies"] = le.fit_transform(data["StreamingMovies"])
data["Contract"] = le.fit_transform(data["Contract"])
data["PaperlessBilling"] = le.fit_transform(data["PaperlessBilling"])
data["PaymentMethod"] = le.fit_transform(data["PaymentMethod"])
data["Churn"] = le.fit_transform(data["Churn"])
```

Data after label encoding

```
data.head()
```

	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	OnlineBa
0	0	0	1	0	1	0	1	0	0	2
1	1	0	0	0	34	1	0	0	2	0
2	1	0	0	0	2	1	0	0	2	2
3	1	0	0	0	45	0	1	0	2	0
4	0	0	0	0	2	1	0	1	0	0

All the data is converted into numerical values.

Splitting the Dataset into Dependent and Independent variable

- The independent variable in the dataset would be considered as 'x' and gender,SeniorCitizen,Partner,Dependents,tenure,PhoneService,MultipleLines,InternetService,OnlineSecurity,OnlineBackup,DeviceProtection,TechSupport,StreamingTV,StreamingMovies,Contract,PaperlessBilling,PaymentMethod,MonthlyCharges,TotalCharges columns would be considered as independent variable.
- The dependent variable in the dataset would be considered as 'y' and the 'Churn' column is considered as dependent variable.

Now we will split the data of independent variables,

```
x= data.iloc[:,0:19].values
y= data.iloc[:,19:20].values
```

From the above code **“:”** indicates that you are considering all the rows in the dataset and **“0:18”** indicates that you are considering columns 0 to 8 such as sex, job and purpose as input values and assigning them to variable x. In the same way in second line **“:”** indicates you are considering all the rows and **“18:19”** indicates that you are considering only last column as output value and assigning them to variable y.

After splitting we see the data as below

X

```
x

array([[0.0000e+00, 0.0000e+00, 1.0000e+00, ..., 2.0000e+00, 2.9850e+01,
        2.9850e+01],
       [1.0000e+00, 0.0000e+00, 0.0000e+00, ..., 3.0000e+00, 5.6950e+01,
        1.8895e+03],
       [1.0000e+00, 0.0000e+00, 0.0000e+00, ..., 3.0000e+00, 5.3850e+01,
        1.0815e+02],
       ...,
       [0.0000e+00, 0.0000e+00, 1.0000e+00, ..., 2.0000e+00, 2.9600e+01,
        3.4645e+02],
       [1.0000e+00, 1.0000e+00, 1.0000e+00, ..., 3.0000e+00, 7.4400e+01,
        3.0660e+02],
       [1.0000e+00, 0.0000e+00, 0.0000e+00, ..., 0.0000e+00, 1.0565e+02,
        6.8445e+03]])
```

Y

```
y

array([[0],
       [0],
       [1],
       ...,
       [0],
       [1],
       [0]], dtype=int64)
```

One Hot Encoding

Sometimes in datasets, we encounter columns that contain numbers of no specific order of preference. The data in the column usually denotes a category or value of the category and also when the data in the column is label encoded. This confuses the machine learning model, to avoid this, the data in the column should be One Hot encoded.

One Hot Encoding –It refers to splitting the column which contains numerical categorical data to many columns depending on the number of categories present in that column. Each column contains “0” or “1” corresponding to which column it has been placed.

```python
from sklearn.preprocessing import OneHotEncoder
one = OneHotEncoder()
a= one.fit_transform(x[:,6:7]).toarray()
b= one.fit_transform(x[:,7:8]).toarray()
c= one.fit_transform(x[:,8:9]).toarray()
d= one.fit_transform(x[:,9:10]).toarray()
e= one.fit_transform(x[:,10:11]).toarray()
f= one.fit_transform(x[:,11:12]).toarray()
g= one.fit_transform(x[:,12:13]).toarray()
h= one.fit_transform(x[:,13:14]).toarray()
i= one.fit_transform(x[:,14:15]).toarray()
j= one.fit_transform(x[:,16:17]).toarray()
x=np.delete(x,[6,7,8,9,10,11,12,13,14,16],axis=1)
x=np.concatenate((a,b,c,d,e,f,g,h,i,j,x),axis=1)
```

Handling Imbalance Data

Data Balancing is one of the most important step, which need to be performed for classification models, because when we train our model on imbalanced dataset ,we will get biassed results, which means our model is able to predict only one class element.

For Balancing the data we are using the SMOTE Method.

SMOTE: Synthetic minority over sampling technique, which will create new synthetic data points for under class as per the requirements given by us using KNN method.

```python
from imblearn.over_sampling import SMOTE

smt = SMOTE()

x_resample, y_resample = smt.fit_resample(x,y)

x_resample
```

```
array([[0.00000000e+00, 0.00000000e+00, 1.00000000e+00, ...,
        2.00000000e+00, 2.98500000e+01, 2.98500000e+01],
       [1.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
        3.00000000e+00, 5.69500000e+01, 1.88950000e+03],
       [1.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
        3.00000000e+00, 5.38500000e+01, 1.08150000e+02],
       ...,
       [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
        3.00000000e+00, 2.02307905e+01, 2.02307905e+01],
       [1.00000000e+00, 0.00000000e+00, 6.76069757e-01, ...,
        3.23930243e-01, 9.00059277e+01, 3.69766940e+03],
       [0.00000000e+00, 3.89455378e-01, 1.00000000e+00, ...,
        2.00000000e+00, 9.63258517e+01, 3.21144455e+03]])
```

```
y_resample
```

```
array([0, 0, 1, ..., 1, 1, 1])
```

```
x.shape, x_resample.shape
```

```
((7043, 19), (10348, 19))
```

```
y.shape, y_resample.shape
```

```
((7043, 1), (10348,))
```

From the above picture, we can infer that ,previously our dataset had 492 class 1, and 192 class items, after applying smote technique on the dataset the size has been changed for minority class

1.2 purpose

Predicting Customer Churn

Churn prediction means detecting which customers are likely to leave a service or to cancel a subscription to a service. It is a critical prediction for many businesses because acquiring new clients often costs more than retaining existing ones. Once you can identify those customers that are at risk of cancelling, you should know exactly what marketing action to take for each individual customer to maximise the chances that the customer will remain.

Different customers exhibit different behaviours and preferences, so they cancel their subscriptions for various reasons. It is critical, therefore, to proactively communicate with each of them in order to retain them in your customer list. You need to know which marketing action will be the most effective for each and every customer, and when it will be most effective.

Why is it so important?

Customer churn is a common problem across businesses in many sectors. If you want to grow as a company, **you have to invest in acquiring new clients**. Every time a client leaves, it represents a significant investment lost. Both time and effort need to be channelled into replacing them. Being able to predict when a client is likely to leave, and offer them incentives to stay, can offer huge savings to a business.

As a result, understanding what keeps customers engaged is extremely valuable knowledge, as it can help you to develop your retention strategies, and to roll out operational practices aimed at keeping customers from walking out the door.

Predicting churn is a fact of life for any subscription business, and even slight fluctuations in churn can have a significant impact on your bottom line. We need to know: "Is this customer going to leave us within X months?" Yes or No? It is a binary classification task.

What are the main challenges?

Churn prediction modelling techniques attempt to understand the precise customer behaviours and attributes that signal the risk and timing of customers leaving. It's not a walk-in-the-park task so I mention just four points to consider.

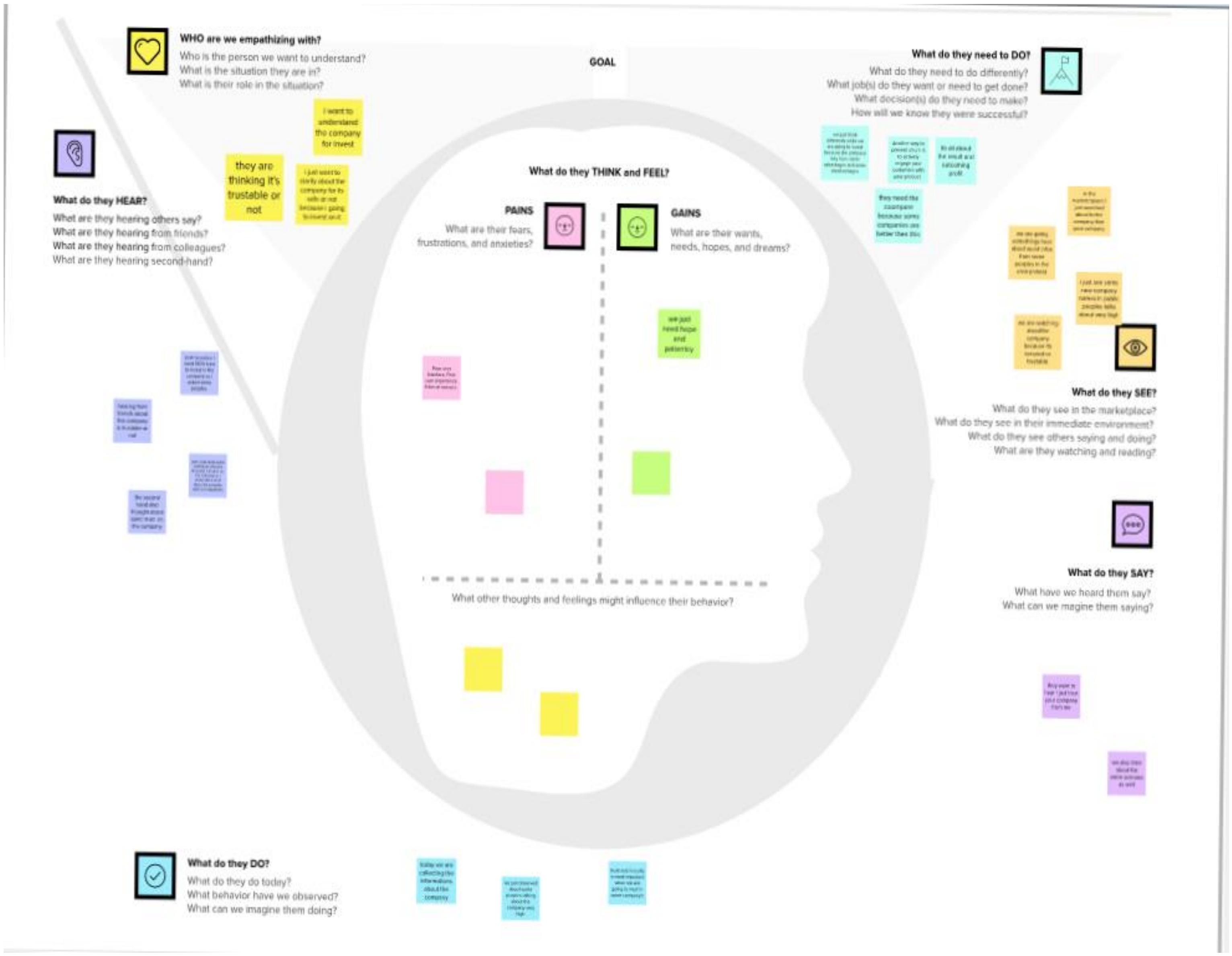
- ✓ To succeed at retaining customers who are ready to abandon your business, Marketers & Customer Success experts must be able to predict in advance which customers are going to churn and set up a plan of marketing actions that will have the greatest retention impact on each customer. The key here is to to be proactive and engage with these customers. While simple in theory, the realities involved with achieving this "proactive retention" goal are extremely challenging.
- ✓ The accuracy of the technique is critical to the success of any proactive retention efforts. If the Marketer is unaware of a customer about to churn, no action will be taken to retain that customer.
- ✓ Special retention-focused offers or incentives may be provided to happy, active customers, resulting in reduced revenues for no good reason.

- ✓ Your churn prediction model should rely on (almost) real-time data to quantify the risk of churning, not on static data. Although you will be able to identify a certain percentage of at-risk customers with even static data, your predictions will be inaccurate.

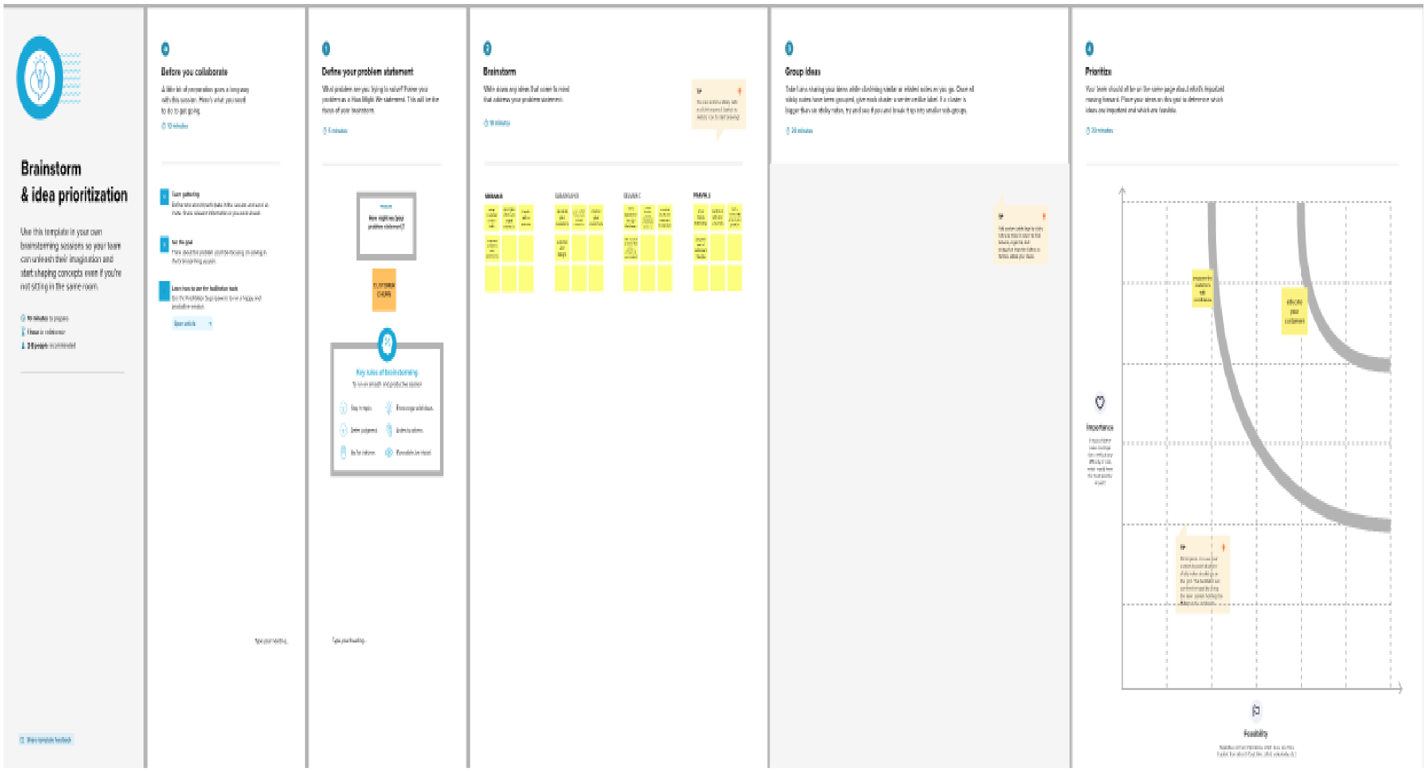
Bringing it all together, predicting customer churn is important. Effective action can be taken to retain the customer before it is too late. The ability to predict that a customer is at high risk of churning while there is still time to do something about it represents a huge additional potential revenue source for every online business.

2.Problem definition& design thinking

2.1 Empathy map



2.2 Ideation & Brainstorming Map



3. Result

The Best Model

Tthe best model after comparing its performance using different evaluation metrics means selecting the model with the highest performance and saving its weights and configuration. This can be useful in avoiding the need to retrain the model every time it is needed and also to be able to use it in the future.

```
classifier.save("telcom_churn.h5")
```

Integrate With Web Framework

In this section, we will be building a web application that is integrated to the model we built. A UI is provided for the uses where he has to enter the values for predictions. The enter values are given to the saved model and prediction is showcased on the UI.
This section has the following tasks

Building Html Pages

For this project create two HTML files namely ,

- base.html
- index.html
- predyes.html
- predno.html

and save them in the complete folder.

Build Python Code

Import the libraries

```
from flask import Flask, render_template, request
import keras
from keras.models import  load_model
```

Load the saved model. Importing the flask module in the project is mandatory. An object of Flask class is our WSGI application. Flask constructor takes the name of the current module (__name__) as argument.

```
app = Flask(__name__)
model = load_model("telcom_churn.h5")
```

Render HTML page

```
@app.route('/') # rendering the html template
def home():
    return render_template('home.html')
```

Here we will be using a declared constructor to route to the HTML page which we have created earlier.

In the above example, '/' URL is bound with the home.html function. Hence, when the home page of the web server is opened in the browser, the html page will be rendered. Whenever you enter the values from the html page the values can be retrieved using POST Method.

Retrieves the value from UI

Here we are routing our app to predict() function. This function retrieves all the values from the HTML page using Post request. That is stored in an array. This array is passed to the model.predict() function. This function returns the prediction. And this prediction value will be rendered to the text that we have mentioned in the submit.html page earlier.

Main Function

```
@app.route('/')
def helloworld():
    return render_template("base.html")
@app.route('/assesment')
def prediction():
    return render_template("index.html")

@app.route('/predict', methods = ['POST'])
def admin():
    a= request.form["gender"]
    if (a == 'f'):
        a=0
    if (a == 'm'):
        a=1
    b= request.form["srcitizen"]
    if (b == 'n'):
        b=0
    if (b == 'y'):
        b=1
    c= request.form["partner"]
    if (c == 'n'):
        c=0
    if (c == 'y'):
        c=1
    d= request.form["dependents"]
    if (d == 'n'):
        d=0
    if (d == 'y'):
        d=1
    e= request.form["tenure"]
    f= request.form["phservices"]
    if (f == 'n'):
        f=0
    if (f == 'y'):
        f=1
    g= request.form["multi"]
    if (g == 'n'):
```

```
if (g == 'n'):
    g1,g2,g3=1,0,0
if (g == 'nps'):
    g1,g2,g3=0,1,0
if (g == 'y'):
    g1,g2,g3=0,0,1
h= request.form["is"]
if (h == 'dsl'):
    h1,h2,h3=1,0,0
if (h == 'fo'):
    h1,h2,h3=0,1,0
if (h == 'n'):
    h1,h2,h3=0,0,1
i= request.form["os"]
if (i == 'n'):
    i1,i2,i3=1,0,0
if (i == 'nis'):
    i1,i2,i3=0,1,0
if (i == 'y'):
    i1,i2,i3=0,0,1
j= request.form["ob"]
if (j == 'n'):
    j1,j2,j3=1,0,0
if (j == 'nis'):
    j1,j2,j3=0,1,0
if (j == 'y'):
    j1,j2,j3=0,0,1
k= request.form["dp"]
if (k == 'n'):
    k1,k2,k3=1,0,0
if (k == 'nis'):
    k1,k2,k3=0,1,0
if (k == 'y'):
    k1,k2,k3=0,0,1
l= request.form["ts"]
if (l == 'n'):
    l1,l2,l3=1,0,0
```

```
    l1,l2,l3=1,0,0
if (l == 'nis'):
    l1,l2,l3=0,1,0
if (l == 'y'):
    l1,l2,l3=0,0,1
m= request.form["stv"]
if (m == 'n'):
    m1,m2,m3=1,0,0
if (m == 'nis'):
    m1,m2,m3=0,1,0
if (m == 'y'):
    m1,m2,m3=0,0,1
n= request.form["smv"]
if (n == 'n'):
    n1,n2,n3=1,0,0
if (n == 'nis'):
    n1,n2,n3=0,1,0
if (n == 'y'):
    n1,n2,n3=0,0,1
o= request.form["contract"]
if (o == 'mtm'):
    o1,o2,o3=1,0,0
if (o == 'oyr'):
    o1,o2,o3=0,1,0
if (o == 'tyrs'):
    o1,o2,o3=0,0,1
p= request.form["pmt"]
if (p == 'ec'):
    p1,p2,p3,p4=1,0,0,0
if (p == 'mail'):
    p1,p2,p3,p4=0,1,0,0
if (p == 'bt'):
    p1,p2,p3,p4=0,0,1,0
if (p == 'cc'):
    p1,p2,p3,p4=0,0,0,1
q= request.form["plb"]
if (q == 'n'):
```

```
q= request.form["plb"]
if (q == 'n'):
    q=0
if (q == 'y'):
    q=1
r= request.form["mcharges"]
s= request.form["tcharges"]

t=[[int(g1),int(g2),int(g3),int(h1),int(h2),int(h3),int(i1),int(i2),int(i3),int(j1
print(t)
x = model.predict(t)
print(x[0])
if (x[[0]] <=0.5):
    y ="No"
    return render_template("predno.html", z = y)

if (x[[0]] >= 0.5):
    y ="Yes"
    return render_template("predyes.html", z = y)
```

Run The Web Application

- Open anaconda prompt from the start menu
- Navigate to the folder where your python script is.
- Now type “python app.py” command
- Navigate to the localhost where you can view your web page.
- Click on the predict button from the top left corner, enter the inputs, click on the submit button, and see the result/prediction on the web.

```
(base) C:\Users\Shivani_SB\OneDrive\Desktop\Telecom churn modelling-updated\flask app>python app.py
2023-01-26 00:46:27.532503: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow bir
h oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-c
 AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with watchdog (windowsapi)
2023-01-26 00:46:34.072445: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow bir
h oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-c
 AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
 * Debugger is active!
 * Debugger PIN: 109-979-709
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Now,Go the web browser and write the localhost url (http://127.0.0.1:5000) to get the below result

TELECOM CUSTOMER CHURN PREDICTION

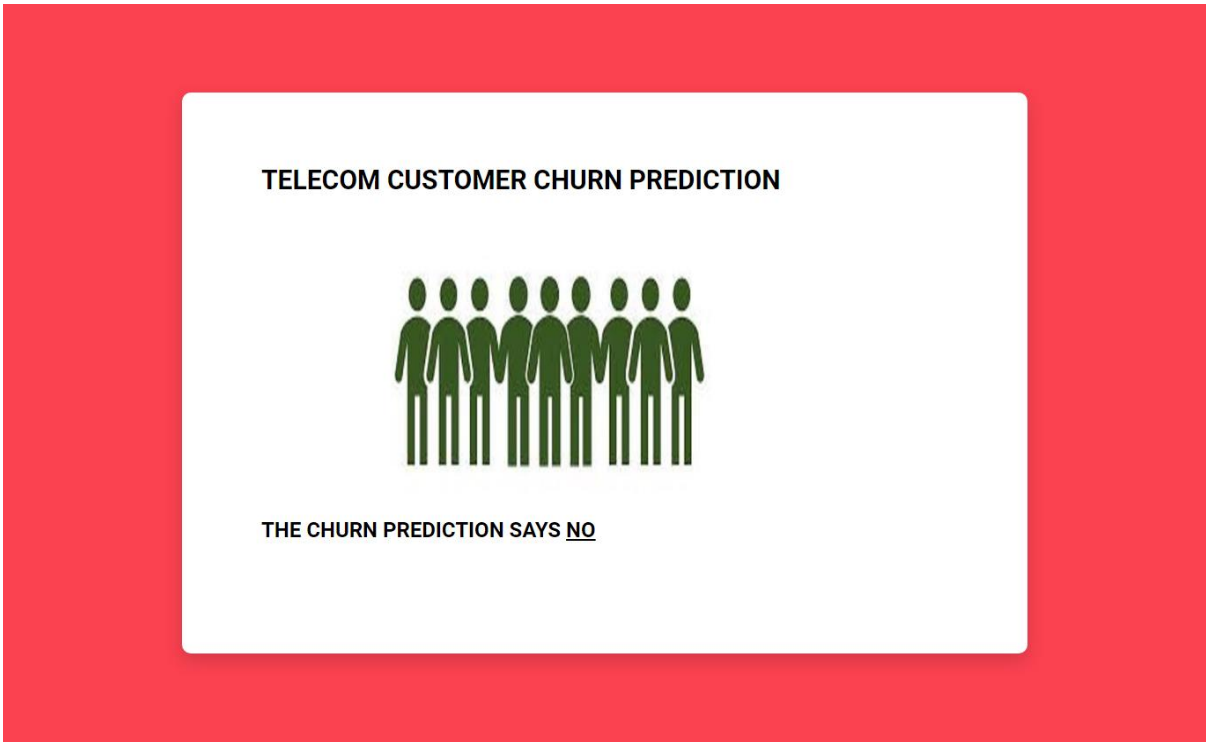
Customer churn has become highly important for companies because of increasing competition among companies, increased importance of marketing strategies and conscious behaviour of customers in the recent years. Customers can easily trend toward alternative services. Companies must develop various strategies to prevent these possible trends, depending on the services they provide. During the estimation of possible churns, data from the previous churns might be used. An efficient churn predictive model benefits companies in many ways. Early identification of customers likely to leave may help to build cost effective ways in marketing strategies. Customer retention campaigns might be limited to selected customers but it should cover most of the customer. Incorrect predictions could result in a company losing profits because of the discounts offered to continuous subscribers.

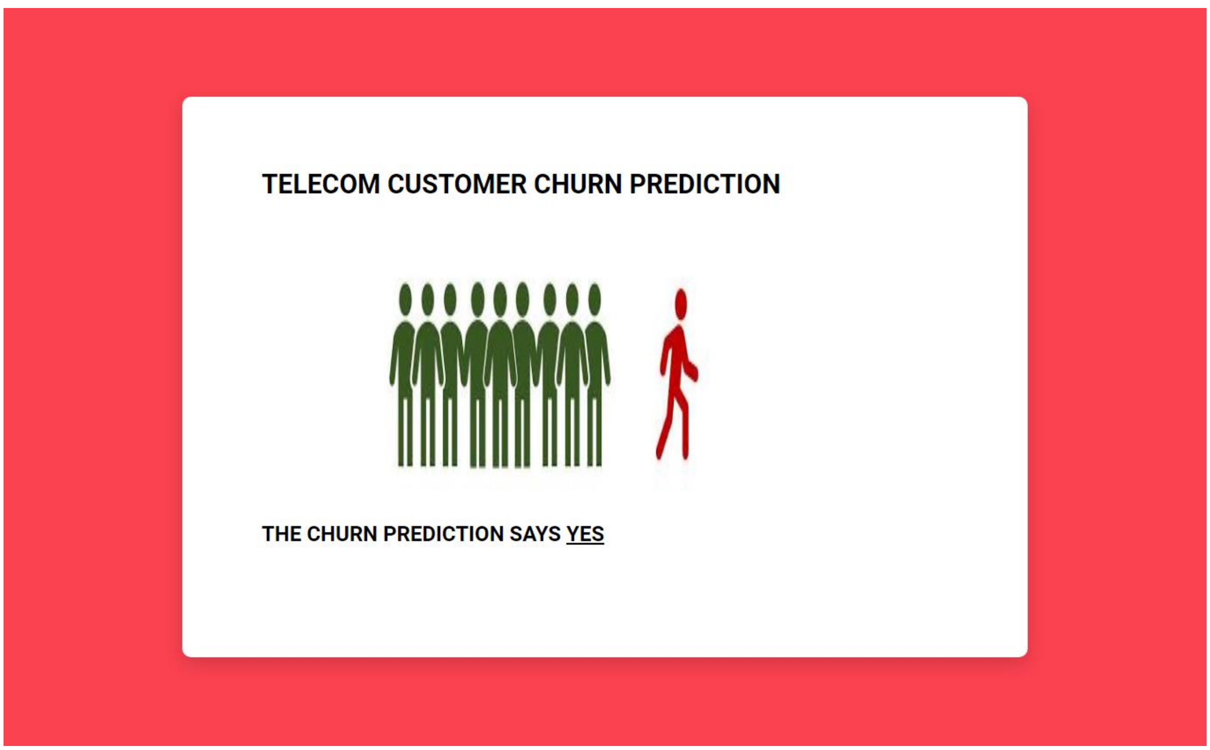
Click me to continue with prediction

PREDICTION FORM

Gender	Yes
Yes	Yes
3	Yes
No Phone service	DSL
No	Yes
No	No
Yes	Yes
Month to Month	Yes
Bank Transfer(Automatic)	39.5
39.5	

Submit


TELECOM CUSTOMER CHURN PREDICTION
THE CHURN PREDICTION SAYS NO


TELECOM CUSTOMER CHURN PREDICTION
THE CHURN PREDICTION SAYS YES

4. Advantages and Disadvantages

Advantages

- The churn rate shows the value of a product or service to a company's customers .If the churn rate is consistently high ,try to identify what may be causing customers to leave .You can do this by conducting surveys or focus groups or asking for customer reviews .Once you've completed the market research ,you could develop strategies to make improvements to the product or service & persuade customers to renew their subscriptions.
- The churn rate is also a useful measure of a company's fiscal health .By calculating the metric monthly or annually ,you can monitor performance and observe any improvements or fluctuations .It can also help you predict the company's financial performance in the future

Disadvantages

- There are some disadvantages to consider when analyzing the churn rate. While it may be useful to know the number of customers that have left the business ,the churn rate cannot provide details of who those customers are .For example newly subscribed customers may be more likely to churn than more established clients .In addition ,if the company experiences a sudden increase in growth ,a higher churn rate may follow.
- The churn rate is unable to differentiate between the varying services a business may offer .There may be different levels of subscriptions services ,each with distinct costs and features .A company might provide a free trial which means those customers may not be providing any revenue during this time. Calculating the company's recurring revenue loss may be more helpful than the customer churn rate in this instance .A more accurate measure of business development is the growth rate, which can help you discover the number of new subscriptions the company received in a specific time frame.

5. Applications

- **Collecting the data** – First, they need to pull in data from various systems, such as their CRM, call center, ERP, etc. These systems provide a piece of the information needed to have a holistic view of their customers. By combining all the data, companies will be able to better identify the customers most at risk of attrition.
- **Access to streaming data** – The next issue is that the data needs to be as close to real-time as possible. Having data that is a few days or weeks old means that the customer base may have changed in that timeframe. The more recent the data, the more likely the prediction of which customers are "at-risk" of attrition is accurate.
- **Identifying a relevant offer** – Having the data and training a machine learning (ML) model on the data to identify which customers are at risk is only part of the solution. The other part of that solution is determining what is the next step to keep that customer. Is there a special offer, training, or promotion? Telling a customer service representative that a specific customer is at risk of churning without providing them offers that will address the customer's issues won't make a difference to the churn rate.

6. Conclusion

Customer churn is often referred to as customer attrition, or customer defection which is the rate at which the customers are lost. 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. Looking at churn, different reasons trigger customers to terminate their contracts, for example better price offers, more interesting packages, bad service experiences or change of customers' personal situations. Customer churn has become highly important for companies because of increasing competition among companies, increased importance of marketing strategies and conscious behavior of customers in the recent years. Customers can easily trend toward alternative services. Companies must develop various strategies to prevent these possible trends, depending on the services they provide. During the estimation of possible churns, data from the previous churns might be used. An efficient churn predictive model benefits companies in many ways. Early identification of customers likely to leave may help to build cost effective ways in marketing strategies.

7. Future Scope

The importance of this type of research in the telecom market is to help companies make more profit. It has become known that predicting churn is one of the most important sources of income to telecom companies. Hence, this research aimed to build a system that predicts the churn of customers in SyriaTel telecom company. These prediction models need to achieve high AUC values. To test and train the model, the sample data is divided into 70% for training and 30% for testing. We chose to perform cross-validation with 10-folds for validation and hyperparameter optimization. We have applied feature engineering, effective feature transformation and selection approach to make the features ready for machine learning algorithms. In addition, we encountered another problem: the data was not balanced. Only about 5% of the entries represent customers' churn. This problem was solved by undersampling or using trees algorithms not affected by this problem. Four tree based algorithms were chosen because of their diversity and applicability in this type of prediction. These algorithms are Decision Tree, Random Forest, GBM tree algorithm, and XGBOOST algorithm. The method of preparation and selection of features and entering the mobile social network features had the biggest impact on the success of this model, since the value of AUC in SyriaTel reached 93.301%. XGBOOST tree model achieved the best results in all measurements. The AUC value was 93.301%. The GBM algorithm comes in the second place and the random forest and Decision Tree came third and fourth regarding AUC values. We have evaluated the models by fitting a new dataset related to different periods and without any proactive action from marketing, XGBOOST also gave the best result with 89% AUC. The decrease in result could be due to the non-stationary data model phenomenon, so the model needs training each period of time.

8.Appendix

Source Code

```
#importing and building the Decision tree model
def logreg(x_train,x_test,y_train,y_test):
    lr = LogisticRegression(random_state=0)
    lr.fit(x_train,y_train)
    y_lr_tr = lr.predict(x_train)
    print(accuracy_score(y_lr_tr,y_train))
    yPred_lr = lr.predict(x_test)
    print(accuracy_score(yPred_lr,y_test))
    print("***Logistic Regression***")
    print("Confusion_Matrix")
    print(confusion_matrix(y_test,yPred_lr))
    print("Classification Report")
    print(classification_report(y_test,yPred_lr))
```

```
#printing the train accuracy and test accuracy respectively
logreg(x_train,x_test,y_train,y_test)
```

```
0.7734960135298381
0.7734299516908213
***Logistic Regression***
Confusion_Matrix
[[754 279]
 [190 847]]
Classification Report
              precision    recall  f1-score   support

           0       0.80      0.73      0.76      1033
           1       0.75      0.82      0.78      1037

    accuracy                           0.77      2070
   macro avg       0.78      0.77      0.77      2070
weighted avg       0.78      0.77      0.77      2070
```

```python
#importing and building the Decision tree model
def decisionTree(x_train,x_test,y_train,y_test):
    dtc = DecisionTreeClassifier(criterion="entropy",random_state=0)
    dtc.fit(x_train,y_train)
    y_dt_tr = dtc.predict(x_train)
    print(accuracy_score(y_dt_tr,y_train))
    yPred_dt = dtc.predict(x_test)
    print(accuracy_score(yPred_dt,y_test))
    print("***Decision Tree***")
    print("Confusion_Matrix")
    print(confusion_matrix(y_test,yPred_dt))
    print("Classification Report")
    print(classification_report(y_test,yPred_dt))
```

```python
#printing the train accuracy and test accuracy respectively
decisionTree(x_train,x_test,y_train,y_test)
```

```
0.9981879681082387
0.6067632850241546
***Decision Tree***
Confusion_Matrix
[[ 242  791]
 [  23 1014]]
Classification Report
              precision    recall  f1-score   support

           0       0.91      0.23      0.37      1033
           1       0.56      0.98      0.71      1037

    accuracy                           0.61      2070
   macro avg       0.74      0.61      0.54      2070
```

```python
#importing and building the random forest model
def RandomForest(x_tarin,x_test,y_train,y_test):
    rf = RandomForestClassifier(criterion="entropy",n_estimators=10,random_state=0)
    rf.fit(x_train,y_train)
    y_rf_tr = rf.predict(x_train)
    print(accuracy_score(y_rf_tr,y_train))
    yPred_rf = rf.predict(x_test)
    print(accuracy_score(yPred_rf,y_test))
    print("***Random Forest***")
    print("Confusion_Matrix")
    print(confusion_matrix(y_test,yPred_rf))
    print("Classification Report")
    print(classification_report(y_test,yPred_rf))
```

```python
#printing the train accuracy and test accuracy respectively
RandomForest(x_train,x_test,y_train,y_test)
```

```
0.9886446001449626
0.7536231884057971
***Random Forest***
Confusion_Matrix
[[563 470]
 [ 40 997]]
Classification Report
              precision    recall  f1-score   support

           0       0.93      0.55      0.69      1033
           1       0.68      0.96      0.80      1037

    accuracy                           0.75      2070
   macro avg       0.81      0.75      0.74      2070
weighted avg       0.81      0.75      0.74      2070
```

```python
#importing and building the KNN model
def KNN(x_train,x_test,y_train,y_test):
    knn = KNeighborsClassifier()
    knn.fit(x_train,y_train)
    y_knn_tr = knn.predict(x_train)
    print(accuracy_score(y_knn_tr,y_train))
    yPred_knn = knn.predict(x_test)
    print(accuracy_score(yPred_knn,y_test))
    print("***KNN***")
    print("Confusion_Matrix")
    print(confusion_matrix(y_test,yPred_knn))
    print("Classification Report")
    print(classification_report(y_test,yPred_knn))
```

```python
#printing the train accuracy and test accuracy respectively
KNN(x_train,x_test,y_train,y_test)
```

```
0.8570910848030925
0.7913043478260869
***KNN***
Confusion_Matrix
[[730 303]
 [129 908]]
Classification Report
              precision    recall  f1-score   support

           0       0.85      0.71      0.77      1033
           1       0.75      0.88      0.81      1037

    accuracy                           0.79      2070
   macro avg       0.80      0.79      0.79      2070
weighted avg       0.80      0.79      0.79      2070
```

```python
#importing and building the random forest model
def svm(x_tarin,x_test,y_train,y_test):
    svm = SVC(kernel = "linear")
    svm.fit(x_train,y_train)
    y_svm_tr = svm.predict(x_train)
    print(accuracy_score(y_svm_tr,y_train))
    yPred_svm = svm.predict(x_test)
    print(accuracy_score(yPred_svm,y_test))
    print("***Support Vector Machine***")
    print("Confusion_Matrix")
    print(confusion_matrix(y_test,yPred_svm))
    print("Classification Report")
    print(classification_report(y_test,yPred_svm))
```

```python
#printing the train accuracy and test accuracy respectively
svm(x_train,x_test,y_train,y_test)
```

```
0.7628654264315052
0.7555555555555555
***Support Vector Machine***
Confusion_Matrix
[[719 314]
 [192 845]]
Classification Report
              precision    recall  f1-score   support

           0       0.79      0.70      0.74      1033
           1       0.73      0.81      0.77      1037

    accuracy                           0.76      2070
   macro avg       0.76      0.76      0.75      2070
weighted avg       0.76      0.76      0.75      2070
```

ANN Model

```
# Importing the Keras libraries and packages
import keras
from keras.models import Sequential
from keras.layers import Dense
```

```
# Initialising the ANN
classifier = Sequential()
```

```
# Adding the input layer and the first hidden layer
classifier.add(Dense(units=30, activation='relu', input_dim=40))
```

```
# Adding the second hidden layer
classifier.add(Dense(units=30, activation='relu'))
```

```
# Adding the output layer
classifier.add(Dense(units=1, activation='sigmoid'))
```

```
# Compiling the ANN
classifier.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

```
# Fitting the ANN to the Training set
model_history = classifier.fit(x_train, y_train, batch_size=10, validation_split=0.33, epochs=200)
```

```
Epoch 1/200
555/555 [==============================] - 4s 3ms/step - loss: 0.5017 - accuracy: 0.7494 - val_loss: 0.4688 - val_accuracy: 0.7756
Epoch 2/200
555/555 [==============================] - 2s 3ms/step - loss: 0.4535 - accuracy: 0.7815 - val_loss: 0.4627 - val_accuracy: 0.7782
Epoch 3/200
555/555 [==============================] - 1s 3ms/step - loss: 0.4424 - accuracy: 0.7865 - val_loss: 0.4691 - val_accuracy: 0.7778
Epoch 4/200
555/555 [==============================] - 1s 2ms/step - loss: 0.4325 - accuracy: 0.7950 - val_loss: 0.4541 - val_accuracy: 0.7917
Epoch 5/200
555/555 [==============================] - 1s 2ms/step - loss: 0.4239 - accuracy: 0.8002 - val_loss: 0.4536 - val_accuracy: 0.7892
Epoch 6/200
555/555 [==============================] - 1s 3ms/step - loss: 0.4146 - accuracy: 0.8078 - val_loss: 0.4564 - val_accuracy: 0.7936
Epoch 7/200
555/555 [==============================] - 1s 2ms/step - loss: 0.4058 - accuracy: 0.8100 - val_loss: 0.4551 - val_accuracy: 0.7921
Epoch 8/200
555/555 [==============================] - 1s 2ms/step - loss: 0.3999 - accuracy: 0.8150 - val_loss: 0.4510 - val_accuracy: 0.7943
```

```
Epoch 195/200
555/555 [==============================] - 2s 3ms/step - loss: 0.1564 - accuracy: 0.9335 - val_loss: 0.7783 - val_accuracy: 0.8093
Epoch 196/200
555/555 [==============================] - 2s 3ms/step - loss: 0.1514 - accuracy: 0.9347 - val_loss: 0.7982 - val_accuracy: 0.7994
Epoch 197/200
555/555 [==============================] - 2s 3ms/step - loss: 0.1549 - accuracy: 0.9327 - val_loss: 0.8319 - val_accuracy: 0.7917
Epoch 198/200
555/555 [==============================] - 2s 3ms/step - loss: 0.1593 - accuracy: 0.9320 - val_loss: 0.7693 - val_accuracy: 0.8130
Epoch 199/200
555/555 [==============================] - 2s 3ms/step - loss: 0.1535 - accuracy: 0.9362 - val_loss: 0.7646 - val_accuracy: 0.8089
Epoch 200/200
555/555 [==============================] - 1s 3ms/step - loss: 0.1544 - accuracy: 0.9356 - val_loss: 0.7744 - val_accuracy: 0.8115
```

```
ann_pred = classifier.predict(x_test)
ann_pred = (ann_pred>0.5)
ann_pred
```

```
65/65 [==============================] - 0s 2ms/step
array([[False],
       [False],
       [ True],
       ...,
       [False],
       [False],
       [False]])
```

```
print(accuracy_score(ann_pred,y_test))
print("***ANN Model***")
print("Confusion_Matrix")
print(confusion_matrix(y_test,ann_pred))
print("Classification Report")
print(classification_report(y_test,ann_pred))
```

```
0.8067632850241546
***ANN Model***
Confusion_Matrix
[[840 193]
 [207 830]]
Classification Report
              precision    recall  f1-score   support

           0       0.80      0.81      0.81      1033
           1       0.81      0.80      0.81      1037

    accuracy                           0.81      2070
   macro avg       0.81      0.81      0.81      2070
```

```
#testing on random input values
lr = LogisticRegression(random_state=0)
lr.fit(x_train,y_train)
print("Predicting on random input")
lr_pred_own = lr.predict(sc.transform([[0,0,1,1,0,0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,1,0,0,1,1,0,0,456,1,0,3245,4567]]))
print("output is: ",lr_pred_own)
```

```
Predicting on random input
output is:  [0]
```

```
#testing on random input values
dtc = DecisionTreeClassifier(criterion="entropy",random_state=0)
dtc.fit(x_train,y_train)
print("Predicting on random input")
dtc_pred_own = dtc.predict(sc.transform([[0,0,1,1,0,0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,1,0,0,1,1,0,0,456,1,0,3245,4567]]))
print("output is: ",dtc_pred_own)
```

```
Predicting on random input
output is:  [0]
```

```
#testing on random input values
rf = RandomForestClassifier(criterion="entropy",n_estimators=10,random_state=0)
rf.fit(x_train,y_train)
print("Predicting on random input")
rf_pred_own = rf.predict(sc.transform([[0,0,1,1,0,0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,1,0,0,1,1,0,0,456,1,0,3245,4567]]))
print("output is: ",rf_pred_own)
```

```
Predicting on random input
output is:  [0]
```

```
#testing on random input values
svc = SVC(kernel = "linear")
svc.fit(x_train,y_train)
print("Predicting on random input")
svm_pred_own = svc.predict(sc.transform([[0,0,1,1,0,0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,1,0,0,1,1,0,0,456,1,0,3245,4567]]))
print("output is: ",svm_pred_own)
```

```
Predicting on random input
output is:  [0]
```

```
#testing on random input values
knn = KNeighborsClassifier()
knn.fit(x_train,y_train)
print("Predicting on random input")
knn_pred_own = knn.predict(sc.transform([[0,0,1,1,0,0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,1,0,0,1,1,0,0,456,1,0,3245,4567]]))
print("output is: ",knn_pred_own)
```

```
Predicting on random input
output is:  [0]
```

```
#testing on random input values
print("Predicting on random input")
ann_pred_own = classifier.predict(sc.transform([[0,0,1,1,0,0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,1,0,0,1,1,0,0,456,1,0,3245,4567]]))
print(ann_pred_own)
ann_pred_own = (ann_pred_own>0.5)
print("output is: ",ann_pred_own)
```

```
Predicting on random input
1/1 [==============================] - 0s 24ms/step
[[1.]]
output is:  [[ True]]
```