INTELLIGENT CUSTOMER RETENTION

INTRODUCTION :

To build customer loyalty and maximize profitability, intelligent techniques for predicting customer churning behaviors are becoming more than necessary in customer relationship management (CRM), especially in the highly competitive wireless telecommunications industry. A study by KPMG about customer defection in the UK reported that 44 percent of UK consumers changed at least one of their key product or service suppliers in 2004 [1]. Analysts have estimated that churning cost wireless service providers in North America and Europe more than four billion US dollars a year [2]. In addition, analysts agreed that acquiring a new customer was many times more expensive than retaining an existing one.

Besides predicting customer defection, proposing appropriate retention policies to retain profitable churners is another significant issue. So far, most of the researchers focus on analyzing customers' behavior and predicting who are likely to churn [3],
[4], [5]. There have been, however, no specific researches working on how policies can be made in accord with the churners. Knowing that the reasons of churn are distinct for different people, so are retention policies, by analyzing the specific characteristics of each churner we should be able to provide suitable policy to retain him.

In this paper, we describe an intelligent system for customer retention by suggesting appropriate retention policies for possible churners according to their specific characteristics. In addition to predicting churners, our system deals with customer retention by first constructing an ontology, which contains comprehensive retention policies of various incentives and recommendations to deal with most possible causes of customer defection. It then digs out hidden correlations among the attribute values of the churners, trying to find out the knowledge of how retention policies are related to churn attribute clusters built upon the hidden correlations,which often reveal why a customer defects. The knowledge is then used to automatically propose retention policies for valuable churners. Note that the retention policy ontology we constructed not only can support the construction of specific retention policies but also can help general retention policy design and analysis.

Retention Policy Ontology and Historical Services Database

Our system runs in two modes: the learning mode and the application mode. In the learning mode, it learns a churn predictive model and constructs a retention policy model; while in the application mode, it uses the above models to predict whether a customer defects and to propose proper retention policies to retain a potential, valuable churner. Before we further describe how each mode works, we first introduce the construction of our retention policy ontology and the structure of the exemplified historical customer services database.

PROBLEM DEFINITION AND DESIGN THINKING:

Empaty Map:

Template

Empathy map canvas

Use this framework to empathize with a customer, user, or any person who is affected by a team's work. Document and discuss your observations and note your assumptions to gain more empathy for the people you serve.

Originally created by Dave Gray at XPLANE®

Share template feedback

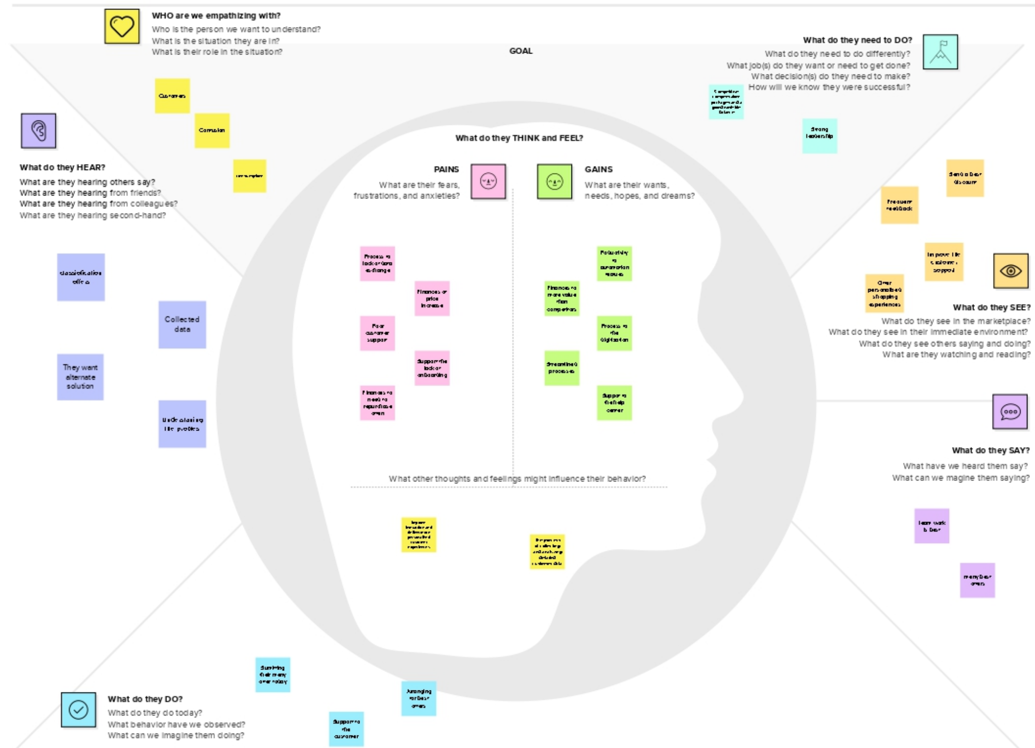
Need some inspiration?

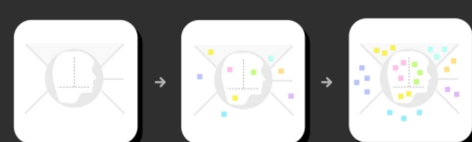
See a finished version of this template to kickstart your work.

Open example →

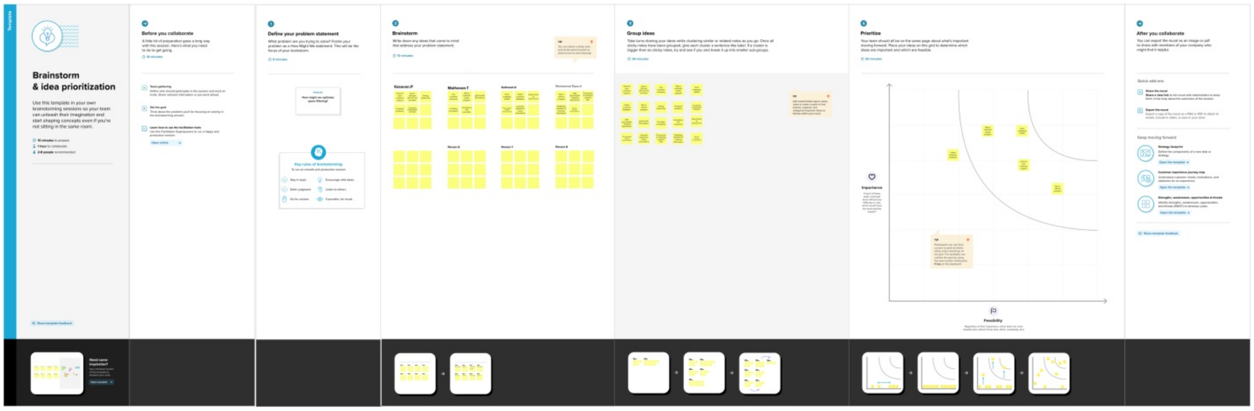
Develop shared understanding and empathy

Summarize the data you have gathered related to the people that are impacted by your work. It will help you generate ideas, prioritize features, or discuss decisions.





Ideation & Brainstoming Map:



RESULT:

TELECOM CUSTOMER CHUN PREDICTION

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 on big data platform and builds a new way of features' engineering and selection. In order to measure the performance of the model, the Area Under Curve (AUC) standard measure is adopted, and the AUC value obtained is 93.3%. Another main contribution is to use customer social network in the prediction model by extracting Social Network Analysis (SNA) features. The use of SNA enhanced the performance of the model from 84 to 93.3% against AUC standard. The model was prepared and tested through Spark environment by working on a large dataset created by transforming big raw data provided by SyriaTel telecom company. The dataset contained all customers' information over 9 months, and was used to train, test, and evaluate the system at SyriaTel. The model experimented four algorithms: Decision Tree, Random Forest, Gradient Boosted Machine Tree "GBM" and Extreme Gradient Boosting "XGBOOST". However, the best results were obtained by applying XGBOOST algorithm. This algorithm was used for classification in this churn predictive model.

clik me to continue with prediction [clik me to continue with prediction]

Fill the form for prediction

Gender

Male

Married Status

No

Dependents

1

Education

Not graduate

Self employed

Yes

Credit History

1

Property Area

Semiruban

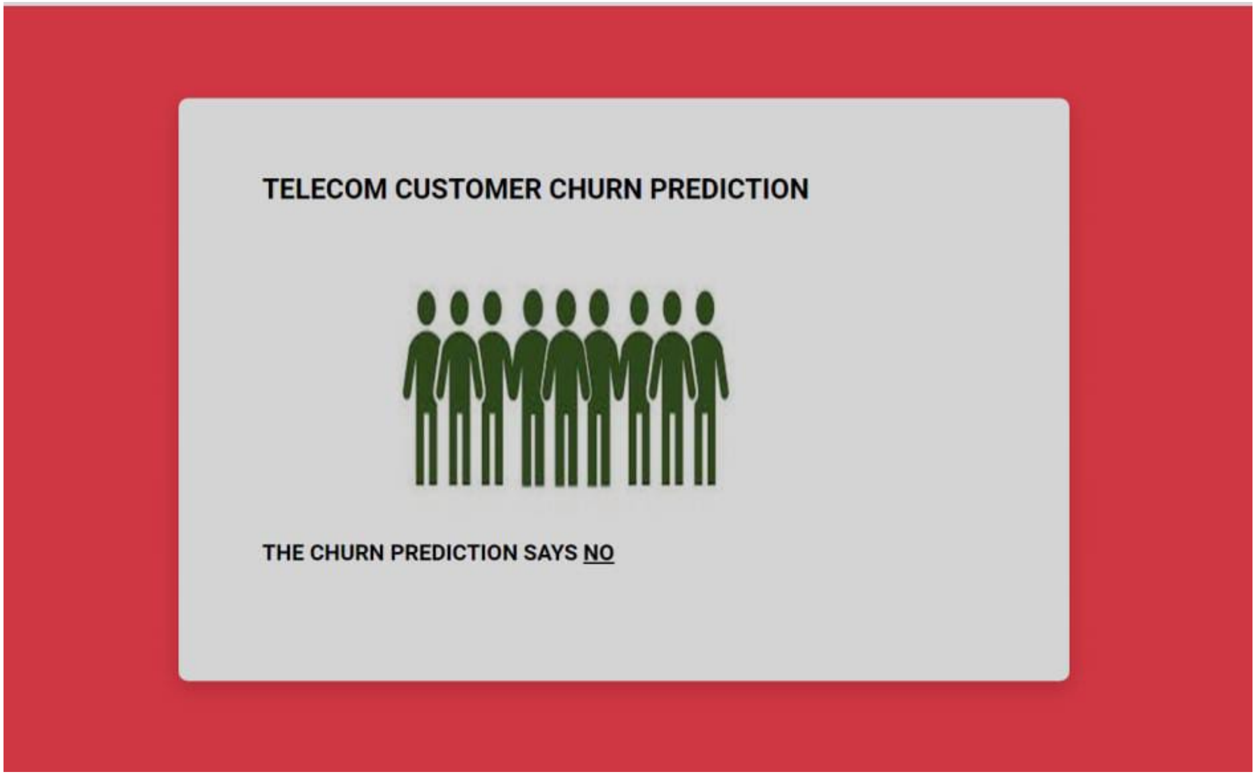
Enter Applicant Income

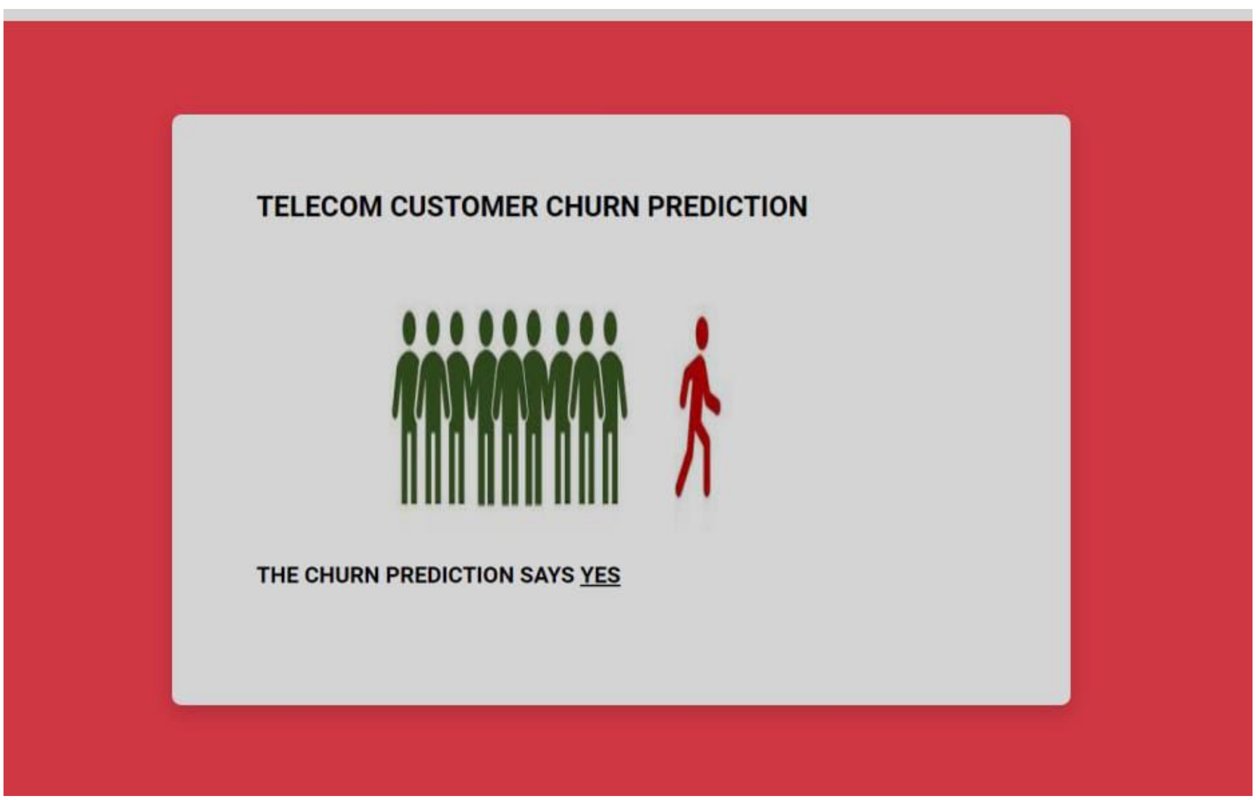
Enter Loan Amount

Enter Co_Applicant Income

Enter Loan Amount term

Submit


TELECOM CUSTOMER CHURN PREDICTION
THE CHURN PREDICTION SAYS NO


TELECOM CUSTOMER CHURN PREDICTION
THE CHURN PREDICTION SAYS YES

ADVANTAGES:

- **Cheaper than Acquisition:** Foremost advantage of customer retention for every organization is that saves the cost involved in acquiring new customers. Customer retention is five times more economical than the acquisition process and thereby a most cost-effective method of maintaining a customer base. Several kinds of research have been conducted in the past which has favored retention over the acquisition process.
- **Loyal customers yield higher profits:** Companies are able to yield higher profits by maintaining loyal customers. These customers make more frequent purchases and spend large in every transaction with their brand. Loyal customers do tend to shift frequently even if their brand raise price, they continue for long. However, every company need to be cautious while raising their prices as end goal is lifetime revenue not just today's revenue.
- **More word of mouth referrals:** Customer retention enable companies in getting large amount of referrals easily. Loyal customers serve as an important source of new business as they bring in large number of customers by referring them brand products. Peoples get attracted toward brand easily through referral of their friends and relatives despite various online and mobile marketing.
- **Easy up-selling and cross-selling:** Business find it easier in up-selling and cross-selling its products to group of loyal customers. Once people are satisfied with the services of particular brand; they like to explore more of its stuff. Customer don't go for much analysis and comparison of brand products while making purchase decisions if they are assured of its quality.
- **Loyal customers are more forgiving:** Another major advantage of customer retention is that it enables in getting better co-operation from people. Loyal customers are more forgiving and they support their brand even if they get a poor service experience. Business get better support from its customers in its hard times that ensure its long-term continuity.
- **Better communication with customers:** Customer retention is an effective tool available to business for establishing proper communication network with customer. Customer that engages with business for long term shares their valuable feedback with their brand. They communicate information regarding people's demand and expectation from business that enables it in formulating effective policies.

DISADVANTAGES:

- **Large investment in terms of price and time:** Customer retention can prove expensive for business in the way that it involves large investment both in terms of price and time. It requires huge cost for running loyalty programs in order to retain customers for a longer period. Business need to sacrifice their profit by offering several discount and cashback offers to its audience.
- **Require concerted commitment and Business Culture:** Every business organization for attaining better customer retention rate should ensure a concerted commitment and proper culture. Every member working at distinct hierarchy of organization should focuses on providing best services to their customers.
- **New customers may be overlooked:** Organization making efforts for attaining efficient customer retention rate may not focus on needs of new customers. There may be chances of new clients being overlooked by brand in a hoard to satisfy its existing customers. Unsatisfied customers may spread negative piece of information about the brand in society.

APPLICATION:

This paper proposes an intelligent system for handling the customer retention task, which is getting important due to keen competition among companies in many modern industries. Taking wireless telecommunication industry as a target of research, our system first learns an optimized churn predictive model from a historical services database by the decision tree-based technique to support the prediction of defection probability of customers.

We then construct a retention policy model which maps clusters of churn attributes to retention policies structured in a retention ontology. The retention policy model supports automatic proposing of suitable retention policies to retain a possible churner provided that he or she is a valuable subscriber. Our experiment shows the learned churn predictive model has around 85% of correctness in tenfold cross-validation. And a preliminary test on proposing suitable package plans shows the retention policy model works equally well as a commercial website.

The fact that our system can automatically propose proper retention policies for possible churners according to their specific characteristics is new and important in customer retention study.

The retention policy ontology is the key component in our system [6], [7]. To develop the domain ontology, we have done a survey on a variety of policies about CRM in lots of industries and academic papers, from which we analyzed the reasons for subscriber defection, the categories of retention policies, and the potential meaning of each policy [2], [8], [9], [10]. Based on these professional suggestions, we have collected comprehensive retention policies for most types of possible churners.

CONCLUSION:

Like all the other marketing processes, customer retention strategies are subject to continuous change and improvement. There is no right or wrong approach, you just need to know your audience and build your programs around customer insights. When implemented correctly, these tactics will help you retain customers for a longer time, while also significantly increasing your sales.

Keep in mind the Pareto principle: 20% of your customers are responsible for 80% of your sales.

Treat them right, personalize your communication, show your appreciation on special occasions and always give them a reason to choose you over any other competitor. It's the safest and most effective way of creating strong relationships that stand the test of time.

And also the shortest road to increased customer retention rates.

FUTUE SCOPE:

E-commerce Retention Rate is often overlooked, but it's an important metric. Since customer acquisition is the most expensive thing an online business has to do, profits depend on how you can profit from each customer after you acquire them.

While most companies traditionally spend more money on customer acquisition because they view it as a quick and effective way of increasing revenue, customer retention often is faster and, on average, costs up to seven times less than customer acquisition. Selling to customers with whom you already have a relationship is often a more effective way of growing revenue because companies don't need to attract, educate, and convert new ones.

Companies that shift their focus to customer retention often find it to be a more efficient process because they are marketing to customers who already have expressed an interest in the products and are engaged with the brand, making it easier to capitalize on their experiences with the company. In fact, retention is a more sustainable business model that is a key to sustainable growth.

With rising customer acquisition costs, businesses need to innovate and assume a proactive role in retaining customer. So you need to do everything in your power to convince your customers to keep coming back after their first purchase.

If your Retention Rate is low you're putting extra pressure on acquisition channels to bring ever more customers through the door. Putting pressure on acquisition usually ends up driving up customer acquisition costs and profitability down.

In ecommerce, your goal should be to maximize Average Order Value (how much each customer spends in one purchase) and promote repeat purchases (or Ecommerce Retention Rate).

APPENDIX:

Source Code:

```
#import necessary libraies

import pandas as pd

import numpy as np

import pickle

import matplotlib.pyplot as plt

import IPython

print(IPython.sys_info())

%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.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

from google.colab import drive
```

```
drive.mount('/content/drive')


with open('/content/drive/My Drive/Colab Notebooks/Intelligent Customer
Retention/data/Dataset.csv','r') as dataset:
  data = pd.read_csv(dataset)
data.info()
#checking for null values
data.TotalCharges = pd.to_numeric(data.TotalCharges, errors='coerce')
data.isnull().any()
data["TotalCharges"].fillna(data ["TotalCharges"].median(), inplace=True)
data.isnull().sum()
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["SteamingMovies"]=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"])

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

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

print(x)

print(y)

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)

pip install imblearn

from imblearn.over_sampling import SMOTE

smt = SMOTE()

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

```
#x_resample

#y_resample

#x.shape,x_resample.shape

#y.shape,y_resample.shape

data.describe()

plt.figure(figsize=(12,5))

plt.subplot(1,2,1)

sns.distplot(data["tenure"])

plt.subplot(1,2,2)

sns.distplot(data["MonthlyCharges"])

plt.figure(figsize=(12,5))

plt.subplot(1,2,1)

sns.countplot(data["gender"])

plt.subplot(1,2,2)

sns.countplot(data["Dependents"])

sns.barplot(x="Churn",y="MonthlyCharges",data=data)

sns.heatmap(data.corr(), annot=True)

sns.pairplot(data=data, markers=["^","v"], palette="inferno")

from sklearn.model_selection import train_test_split

#x_train,x_test,y_train,y_test=train_test_split(x_resample,y_resample,test_size=0.2,random_state=0)

from sklearn.preprocessing import StandardScaler

#sc = StandardScaler()

#x_train = sc.fit_transform(x_train)

#x_test = sc.fit_transform(x_test)

#x_train.shape
```

```
#printing the train accuacy and test accuracy respectively
#logreg(x_train,x_test,y_train,y_test)
#impoting 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_train))
  print("***Decision Tree***")
  print("Confution_Matrix")
  print(confution_matrix(y_test,y_pred_dt))
  print("Classification Report")
  print(classification_report(y_test,y_pred_dt))
#printing the train accuracy and test accuracy respectively
decisionTree(x_train,x_test,y_train,y_test)
#importing and building the random forest model
def andomForest(x_train,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(ypred_rf,y_train))
  ypred_rf=rf.predict(x_test)
  print(accuracy_score(ypred_rf,y_train))
```

```
  print("***Random Forest***")
  print("Confution_Matrix")
  print(confution_matrix(y_test,y_pred_rf))
  print("Classification Report")
  print(classification_report(y_test,y_pred_rf))
#importing and building the KNN model
def KNN(x_train,x_test,y_tain,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_train))
  print("***KNN***")
  print("Confution_Matrix")
  print(confution_matrix(y_test,y_pred_knn))
  print("Classification Report")
  print(classification_report(y_test,y_pred_knn))
#pinting the tain accuracy and test acccuacy respectively
KNN(x_train,x_test,y_train,y_test)
#printing the train accuracy and test accuracy respectively
svm(x_train,x_test,y_train,y_test)
# Impoting the Keas libraries and package
impot keras
from keras.models import Sequential
```

```
from keras.layers import Dense

classifier=sequential()

classsifier.add(Dense(units=30,activation='relu',input_dim=40))

classsifier.add(Dense(units=30,activation='relu'))

model_history=classifier.fit(x_train,y_train,batch_size=10,validation_split=0.33,epochs=200)

ann_pred=classifier.predict(x_test)

ann_pred=(ann_pred>0.5)

ann_pred

print(accuracy_score(ann_pred,y_test))

print("***ANN Model***")

print("Confution_Matrix")

print(confsion_matrix(y_test,ann_pred))

print("Classification Report")

print(classification_report)(y_test,ann_pred))

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)

dtc=DecisionTreeClassifier(criterion="entropy",random_state=0)

dtc.fit(x_train,y_train)

print("Prediciting 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)
```

```
rf=RandomForestClassifier(criterion="entropy",n_estimaters=10,random_state=0)

rf.fit(x_train,y_train)

print("Prediciting 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)

svc=SVC(kernel="linear")

svc.fit(x_train,y_train)

print("Prediciting 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)

knn=KNeighborsClassifier()

knn.fit(x_train,y_train)

print("Prediciting 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)

print("Prediciting 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>.5)

print("output is:",ann_pred_own)

def compareModel(x_train,x_test,y_train,y_test):

  logreg(x_train,x_test,y_train,y_test)

  print('-'*100)
```

```
  decisionTree(x_train,x_test,y_train,y_test)
  print('-'*100)
  RandomForest(x_train,x_test,y_train,y_test)
  print('-'*100)
  svm(x_train,x_test,y_train,y_test)
  print('-'*100)
  KNN(x_train,x_test,y_train,y_test)
  print('-'*100)
compareModel(x_train,x_test,y_train,y_test)
print(accuacy_score(ann_pred,y_test))
print("***ANN Model***")
print("Confution_Matrix")
print(confution_matrix(y_test,ann_pred))
print("Classification Report")
print(classification_report(y_test,ann_pred))
y_rf=model.predict(x_train)
print(accuracy_score(y_rf,y_train))
ypred_rfcv=model.predict(x_test)
print(accuracy_score(ypred_rfcv,y_test))
print("***Random Forset after Hyperparameter tuning***")
print("Confusion_Matrix")
print(confusion_matrix(y_test,ypred_rfcv))
print("Classification Report")
print(classification_report(y_test,ypred_rfcv))
print("Prediction on random input")
```

rfcv_pred_own = model.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: ",rfcv_pred_own)

classifier.save("telcom_churn.h5")

APP.PY:

```
from flask import Flask, render_template, request
import keras
from keras.models import load_model
@app.route('/') # rendering the html template
def home():
     return render_template('home.html')
@app.route('/') # rendering the html template
def home():
     return render_template('home.html')
@app.route('/')
def hellowold():
    return render_template("base.html")
@app.route('/assesment')
def prediction():
    return render_template("index.html")

@app.route('/predict',methods = ['POST'])
def admin():
    a= request.from["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
    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=0,1,0
    if (j=='nis'):
        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
   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=0,0,1
    o=request.form["contracct"]
 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=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),int(j2),int(j3),int(k1),int(k2),int(k3)]]
```