Intelligent Customer Retention: Using Machine Learning for Enhanced Prediction of Telecom Customer Churn

Team ID	**:**	**NM2023TMID31995**
Team Lead Name	:	VIJAYALAKSHMI R
Team Members	:	RUBINI V
		PARVEEN K
		PUSHPARANI R
		MONISHA L

1. INTRODUCTION

1.1 Overview

This project discusses the issue of customer churn, which is the rate at which customers are lost by companies. It is a major concern for large companies, particularly in the telecom industry, due to its direct impact on revenues. Companies are seeking to predict potential churn by analyzing data from previous churns, which may be caused by various reasons such as better price offers, more interesting packages, bad service experiences or change in customers' personal situations. The project explores the importance of customer churn for companies in the current competitive market and the need for efficient churn predictive models to build cost-effective marketing strategies and prevent losses. The use of machine learning models in the telecom industry to identify probable churn customers and make necessary business decisions is also discussed. The project emphasizes the importance of early identification of customers likely to leave to limit losses and retain customers.

Key words: Random Forest, flight price prediction, KNN, decision trees.

1.2 Purpose

<u>Bisiness problem:</u>

The business problem addressed in this project is the issue of customer churn, which is the rate at which customers are lost by companies, particularly in the telecom industry. The problem is significant as it directly impacts the revenues of companies and is caused by various factors such as better price offers, more interesting packages, bad service experiences or change in customers' personal situations. Companies are seeking to predict potential churn by analyzing data from previous churns and developing efficient churn predictive models to build cost-effective marketing strategies and prevent losses. The aim is to retain customers and limit the number of customers likely to leave, thereby increasing the company's revenue and maintaining its market position in the face of stiff competition.

telecom industry. It is challenging to predict which customers are likely to leave, as there may be various reasons behind the churn. The reasons could be related to better price offers, more attractive packages from competitors, bad service experiences, or changes in customers' personal situations. As a result, companies need to develop efficient churn predictive models that can analyze data from previous churns and identify patterns and trends that can help predict potential churn.

The development of churn predictive models is crucial for companies as it can help them build cost-effective marketing strategies to retain customers and limit losses. By identifying customers who are likely to leave, companies can tailor their marketing campaigns to target these customers with personalized offers, rewards, or discounts that will encourage them to stay. Additionally, the development of churn predictive models can help companies optimize their marketing budgets by focusing their retention efforts on customers who are most likely to leave.

The ultimate goal of solving the customer churn problem is to increase the company's revenue and maintain its market position in the face of stiff competition. Companies that can retain their customers are more likely to succeed and grow in the long run, as retaining customers is much less expensive than acquiring new ones. Therefore, the development of efficient churn predictive models is essential for companies to stay competitive and achieve their business goals.

1.3 Business requirement:

The business requirement for addressing the problem of customer churn is to develop efficient churn predictive models that can analyze customer data and predict potential churn. Companies need to collect customer data such as past purchase history, loyalty program status, demographic information, and service usage patterns to develop personalized pricing and offers that can attract and retain customers.

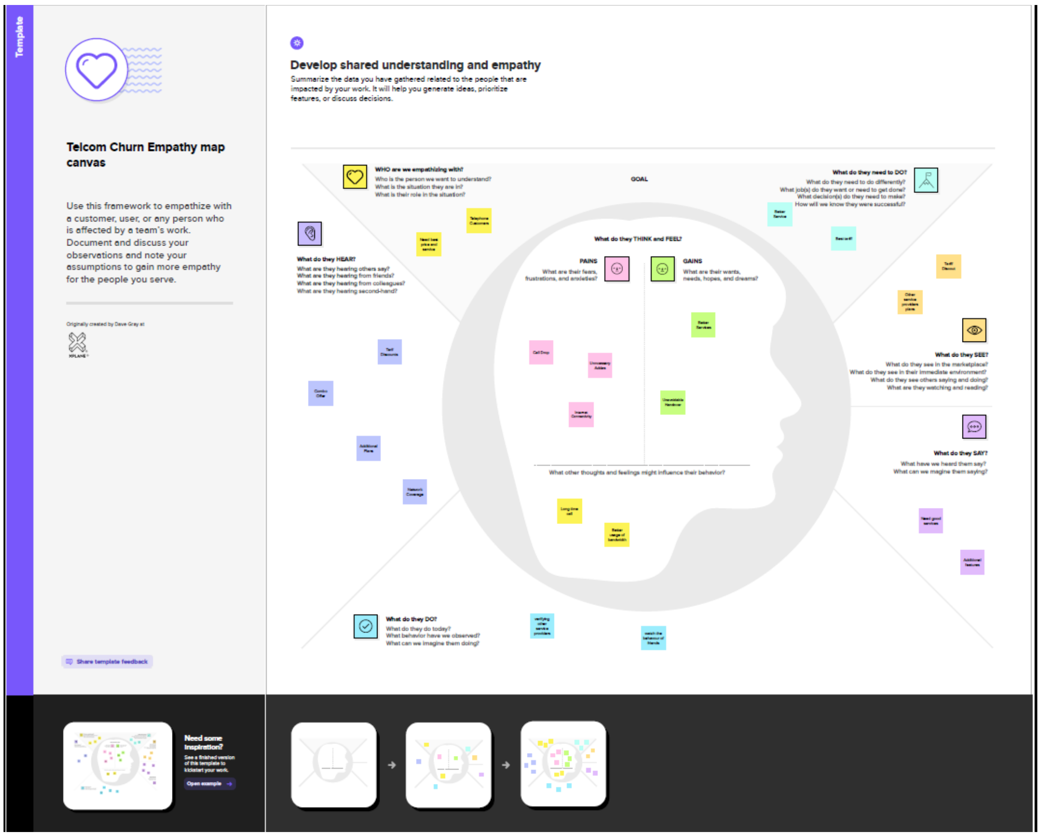
The churn predictive model should be able to analyze the customer data and identify patterns and trends that are indicative of potential churn. The model should be able to generate accurate predictions and alert the company when customers are at risk of leaving, allowing the company to take timely action to retain the customer.

The churn predictive model should also be scalable, able to analyze large amounts of data quickly, and provide actionable insights to the company. The model should be easy to use and integrate with existing customer relationship management systems, allowing companies to streamline their retention efforts.

Overall, the business requirement for addressing the problem of customer churn is to develop an efficient churn predictive model that can help companies retain customers, build cost-effective marketing strategies, optimize marketing budgets, and ultimately increase revenue and maintain market position

2. PROBLEM DEFINITION & DESIGN THINKING

2.1 Empathy Map



Template

Telcom Churn 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

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.

WHO are we empathizing with?
Who is the person we want to understand?
What is the situation they are in?
What is their role in the situation?

GOAL

What do they need to DO?
What do they need to do differently?
What job(s) do they want or need to get done?
What decision(s) do they need to make?
How will we know they were successful?

What do they HEAR?
What are they hearing others say?
What are they hearing from friends?
What are they hearing from colleagues?
What are they hearing second-hand?

What do they THINK and FEEL?

PAINS
What are their fears, frustrations, and anxieties?

GAINS
What are their wants, needs, hopes, and dreams?

What other thoughts and feelings might influence their behavior?

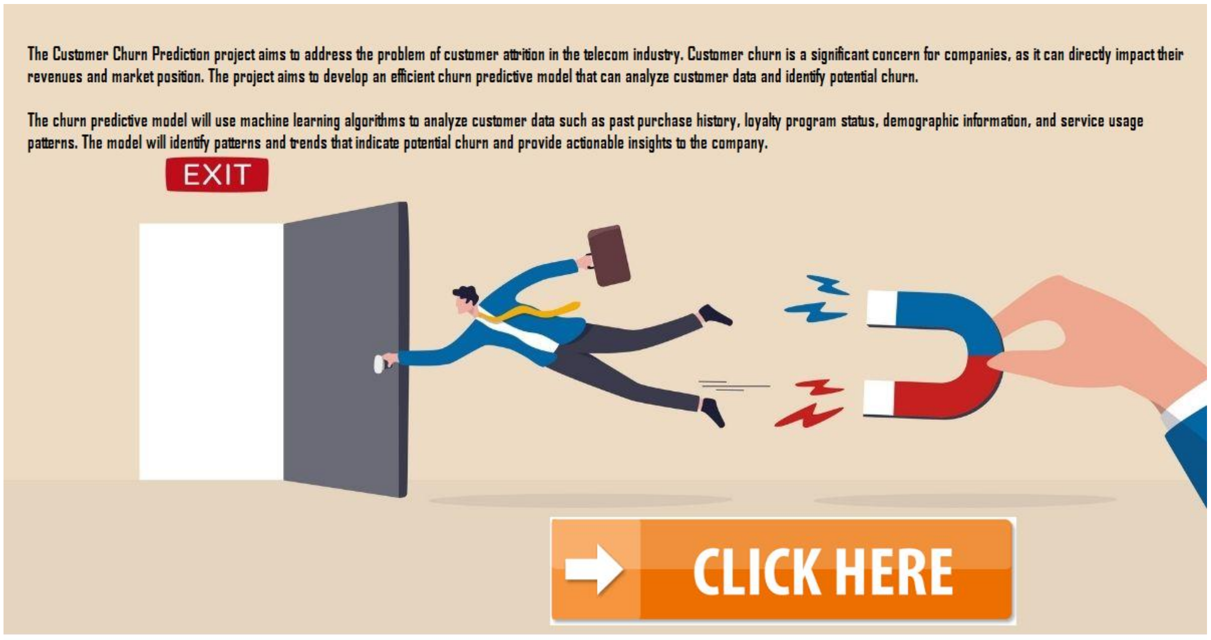
What do they SEE?
What do they see in the marketplace?
What do they see in their immediate environment?
What do they see others saying and doing?
What are they watching and reading?

What do they SAY?
What have we heard them say?
What can we imagine them saying?

What do they DO?
What do they do today?
What behavior have we observed?
What can we imagine them doing?

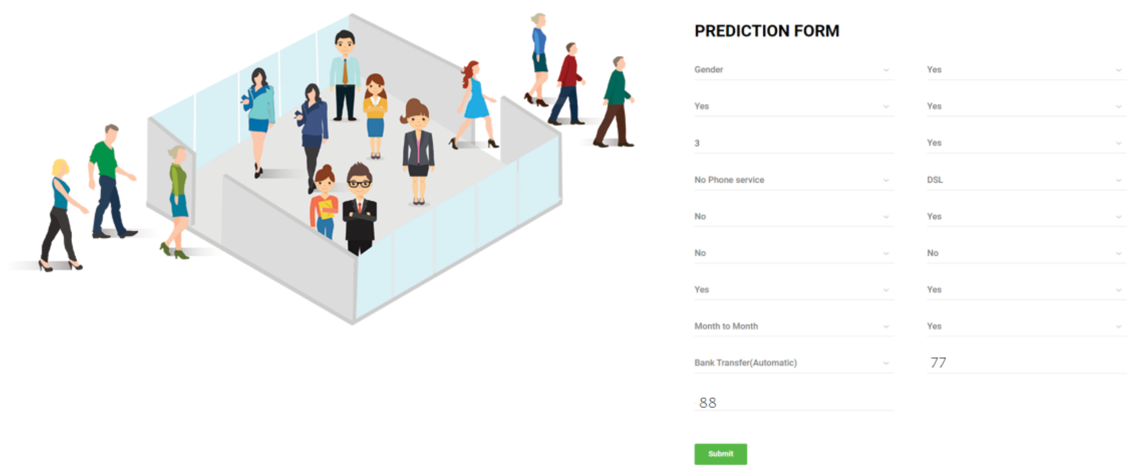
2.2 Ideation & Brainstorming map

3. RESULT



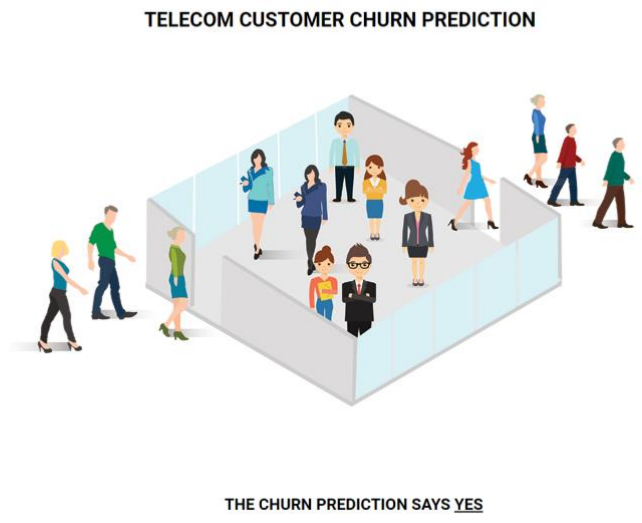
The Customer Churn Prediction project aims to address the problem of customer attrition in the telecom industry. Customer churn is a significant concern for companies, as it can directly impact their revenues and market position. The project aims to develop an efficient churn predictive model that can analyze customer data and identify potential churn.

The churn predictive model will use machine learning algorithms to analyze customer data such as past purchase history, loyalty program status, demographic information, and service usage patterns. The model will identify patterns and trends that indicate potential churn and provide actionable insights to the company.



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)	77
88	


TELECOM CUSTOMER CHURN PREDICTION
THE CHURN PREDICTION SAYS YES

4. ADVANTAGE AND DISADVANTAGE

Advantages

- Increased customer satisfaction: Personalized pricing and offers can enhance the travel experience by providing customers with tailored options that meet their individual needs and preferences.
- Improved revenue: Optimized pricing strategies can increase revenue and profitability by maximizing the value of each customer transaction.
- Enhanced loyalty: Personalized pricing and offers can help build stronger customer relationships by showing customers that the airline values their business and is willing to provide customized services.
- Better marketing: Analyzing customer data can provide airlines with insights into customer behavior and preferences, allowing them to develop targeted marketing campaigns and promotions that engage customers more effectively.
- Competitive advantage: By providing personalized pricing and offers, airlines can differentiate themselves from their competitors and improve their market position.

Disadvantages of using flight price prediction methods to personalize pricing and offers include:

- Privacy concerns: Customers may be hesitant to share personal information with airlines, particularly if they feel that their data is not being used appropriately or ethically.
- Accuracy: Predictive models may not always be accurate, and there is a risk of mispricing or misidentifying customer preferences.
- Complexity: Developing and implementing a flight price prediction system can be complex and requires significant resources, including expertise in data analytics and machine learning.
- Regulatory compliance: The use of customer data is subject to regulatory compliance requirements, which can be complex and time-consuming to navigate.

- Customer dissatisfaction: Personalized pricing and offers may not always be perceived as fair or transparent, leading to customer dissatisfaction and potential reputational damage for the airline.

5. APPLICATION

- Finance: In the finance industry, churn predictive models can help banks and financial institutions identify customers who are likely to close their accounts or stop using their services. This can help banks retain customers by offering them better interest rates, loan terms, or other promotions.

- Healthcare: In the healthcare industry, churn predictive models can help insurance companies identify customers who are likely to switch to a different provider or stop paying their premiums. This can help insurance companies retain customers by offering them better coverage or more affordable plans.

- Retail: In the retail industry, churn predictive models can help companies identify customers who are likely to stop shopping at their stores or switch to a different brand. This can help companies retain customers by offering them personalized promotions, loyalty rewards, or other incentives.

- Subscription services: Churn predictive models are especially useful for subscription-based services, such as streaming platforms or software-as-a-service (SaaS) companies. These models can help companies identify customers who are likely to cancel their subscriptions and offer them personalized deals or incentives to stay subscribed.

6. CONCLUSION

In conclusion, churn predictive modeling is a powerful tool that can help companies identify and retain their most valuable customers. By analyzing customer data and predicting potential churn, companies can take proactive measures to retain customers, optimize their marketing strategies, and gain a competitive advantage. Churn predictive models can help companies build cost-effective retention strategies that are tailored to individual customers, resulting in higher customer satisfaction, lower customer acquisition costs, and increased revenues.

7. FUTURE SCOPE

In the future, churn predictive modeling is expected to become even more sophisticated and accurate, as companies continue to gather and analyze larger amounts of customer data. Machine learning algorithms and artificial intelligence will play an increasingly important role in churn prediction, allowing companies to make more precise and personalized predictions. Additionally, as more industries move towards subscription-based business models, churn predictive models will become even more essential for retaining customers and ensuring long-term business success. Companies will need to stay ahead of the curve by investing in advanced analytics tools, data management systems, and machine learning technologies to stay competitive in a rapidly evolving business landscape.

8. APPENDIX

CODE

```
                    # -*- coding: utf-8 -*-
"""Telephone_churn.ipynb

Automatically generated by Colaboratory.

Original file is located at
   https://colab.research.google.com/drive/1UO2ryH_uszryAewspDF8n-arSGF5sXei
"""

# Commented out IPython magic to ensure Python compatibility.
import pandas as pd
import numpy as np
import pickle
from  google.colab import files
import io
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

data=files.upload()

data=pd.read_csv("/content/WA_Fn-UseC_-Telco-Customer-Churn.csv")
df=data
data

data = data.iloc[:,1:]
data.info()

data.TotalCharges=pd.to_numeric(data.TotalCharges,errors='coerce')
data.isnull().any()

data.TotalCharges.fillna(data.TotalCharges.mean(), 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["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.head()

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

x

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)

x

y

from imblearn.over_sampling import SMOTE
sm = SMOTE()
x_resampled, y_resampled = sm.fit_resample(x, y)

x_resampled
```

```
y_resampled

x.shape,x_resampled.shape

y.shape,y_resampled.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(df.corr(),annot=True)

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

from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x_resampled,y_resampled,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

def logreg(x_train,x_test,y_train,y_test):
  lr=LogisticRegression(random_state=0)
  lr.fit(x_train,y_train)
  ylt=lr.predict(x_train)
  print("Accuracy Score :",accuracy_score(ylt,y_train))
  yPred_lr=lr.predict(x_test)
  print("Accuracy Test :",accuracy_score(yPred_lr,y_test))
  print("Logistic Regression")
  print("Confusion Matrix")
  print(confusion_matrix(y_test, yPred_lr))
  print("Classification Reprot")
  print(classification_report(y_test, yPred_lr))

logreg(x_train,x_test,y_train,y_test)
```

```
def dTree(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 :",accuracy_score(y_dt_tr,y_train))
  yPred_dt=dtc.predict(x_test)
  print("Accuracy Test :",accuracy_score(yPred_dt,y_test))
  print("Decsion Tree")
  print("Confusion Matrix")
  print(confusion_matrix(y_test, yPred_dt))
  print("Classification Reprot")
  print(classification_report(y_test, yPred_dt))

dTree(x_train,x_test,y_train,y_test)

def RandomForest(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 :",accuracy_score(y_rf_tr,y_train))
  yPred_rf=rf.predict(x_test)
  print("Accuracy Test :",accuracy_score(yPred_rf,y_test))
  print("Random Forest")
  print("Confusion Matrix")
  print(confusion_matrix(y_test, yPred_rf))
  print("Classification Reprot")
  print(classification_report(y_test, yPred_rf))

RandomForest(x_train,x_test,y_train,y_test)

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 :",accuracy_score(y_knn_tr,y_train))
  yPred_knn=knn.predict(x_test)
  print("Accuracy Test :",accuracy_score(yPred_knn,y_test))
  print("KNN")
  print("Confusion Matrix")
  print(confusion_matrix(y_test, yPred_knn))
  print("Classification Reprot")
  print(classification_report(y_test, yPred_knn))

KNN(x_train,x_test,y_train,y_test)

def SVM(x_train,x_test,y_train,y_test):
  svm=KNeighborsClassifier()
  svm.fit(x_train,y_train)
  y_svm_tr=svm.predict(x_train)
  print("Accuracy Score :",accuracy_score(y_svm_tr,y_train))
  yPred_svm=svm.predict(x_test)
  print("Accuracy Test :",accuracy_score(yPred_svm,y_test))
  print("SVM")
  print("Confusion Matrix")
  print(confusion_matrix(y_test, yPred_svm))
  print("Classification Reprot")
  print(classification_report(y_test, yPred_svm))

SVM(x_train,x_test,y_train,y_test)

import keras
from keras.models import Sequential
from keras.layers import Dense
classifier=Sequential()
classifier.add(Dense(units=30, activation="relu",input_dim=40))
classifier.add(Dense(units=30, activation="relu"))
classifier.add(Dense(units=1, activation="sigmoid"))
classifier.compile(optimizer="adam",loss="binary_crossentropy",metrics=['accuracy'])

model_histroty=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 Test :",accuracy_score(ann_pred,y_test))
print("ANN model")
print("Confusion Matrix")
print(confusion_matrix(y_test, ann_pred))
print("Classification Reprot")
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("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)

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)

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]]))
ann_pred_own=(ann_pred_own>0.5)
print("output is :", ann_pred_own)

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)

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)

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)

def comp_mod(x_train,x_test,y_train,y_test):
  logreg(x_train,x_test,y_train,y_test)
  print("-"*100)
  dTree(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)

comp_mod(x_train,x_test,y_train,y_test)

model = RandomForestClassifier()
model.fit(x_train, y_train)
y_rf=model.predict(x_train)
print(accuracy_score(y_rf,y_train))
y_pred_rcv=model.predict(x_test)
print(accuracy_score(y_pred_rcv,y_test))
print("**Random Forest after Hyperparameter tuning**")
print("Confusion Matrix")
print(confusion_matrix(y_test, y_pred_rcv))
print("Classification Reprot")
print(classification_report(y_test, y_pred_rcv))
print("Predicting 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("telecom_churn.h5")
```