# INTELLIGENT CUSTOMER RETENTION

Team size:4

**Team Leader:** **SUVEDHA M**

Team Members:

1. **KAVIYARASI M**
2. **NISHANSI S**
3. **THAMARAISELVI S**
4. **PRIYANGA M**

# INTRODUCTION

**Customer retention definition: a company's ability to turn customers into repeat buyers and prevent them from switching to a competitor. Customer retention indicates whether your product and the quality of your service please your existing customers. 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**

## 1.1 OVERVIEW

Customer retention refers to a business strategy that aims to retain as many customers as possible and improves customer relationships. Your customer retention rate, or how many loyal customers you can keep over a specified period, is directly affected by your customer churn (how many customers lost) and your customer acquisition (how many new customers join).

As you might expect, there's more than one way of going about building customer retention strategies — by and large, any process that promotes customer loyalty can be classed as a customer retention activity.

## 1.2 PURPOSE

### 1. Offer a seamless online experience (Amazon)

One of the most basic customer retention examples is meeting customer expectations. And customers today expect online experiences that are on-par with or better than, in-person experiences. In fact, 65 percent of customers want to buy from companies that offer quick and easy online transactions, according to our Trends Report. And 49 percent gave Amazon the highest marks for service for that reason. Are there pain points in your online experience? How can you make things easy for customers?

### 2. Make every customer feel like a VIP customer (Four Seasons)

Luxury hotels are known for their heritage of high-touch, exclusive customer service. The Four Seasons is able to expand that feeling of luxury to every customer through its combination of technology and white glove service. Guests can use Four Seasons Chat to message staff through channels such as WhatsApp for any inquiry or service, including requests for restaurant recommendations and reservations, ordering room service, arrival or early checkout and even ordering a private jet.

### 3. Build empathic customer relationships (Zappos)

If there is one thing the pandemic showed us, it is that empathy is key to building lasting customer relationships. In fact, 49 percent of customers want agents to be empathetic, according to our Trends Report. During the pandemic, Zappos started a hotline where customers could call or chat with its support team about anything, even the best Netflix shows.

### 4. Be proactive (Dollar Shave Club)

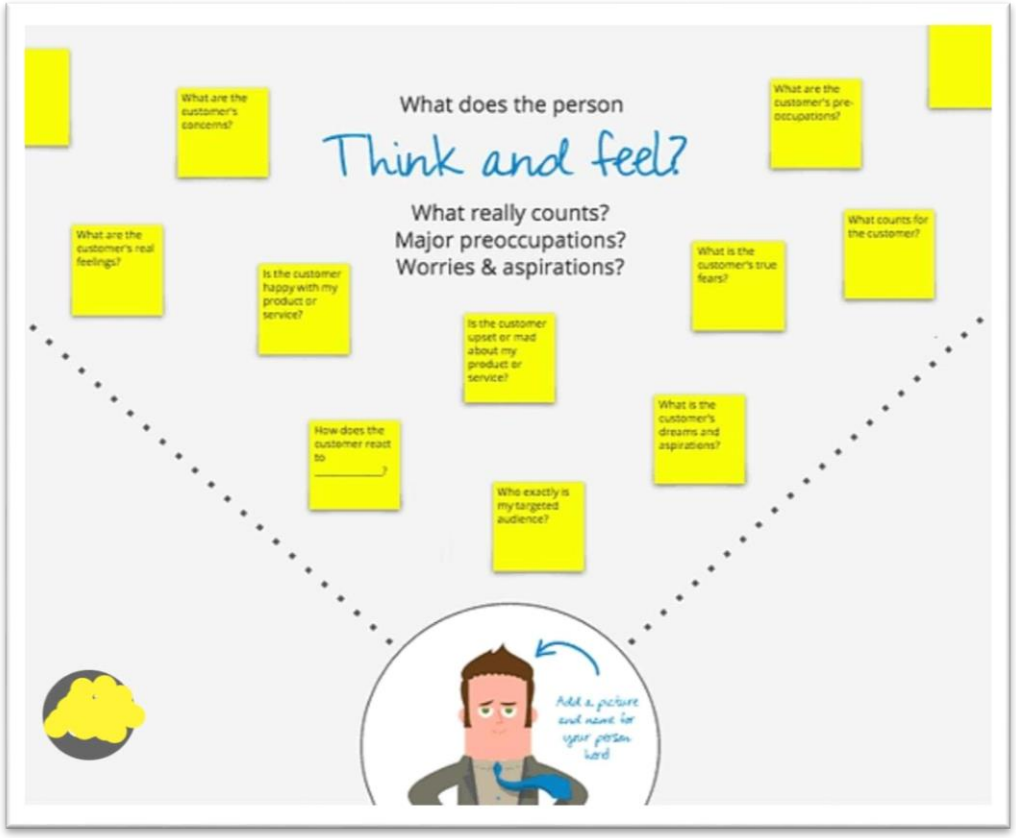
Customers expect brands to anticipate their needs and get in front of issues before they even happen. That is why proactive service is so important in retaining customers. Dollar Shave Club welcomes website visitors with a chatbot to answer common questions before a customer has to reach out to customer support or abandons their cart.

### 5. Support causes your customers to care about (Bombas)

54 percent of customers want to buy from companies that prioritise diversity, equity and inclusion in their communities and workplaces and 63 percent want to buy from companies that are socially responsible, according to our Trends Report. Bombas donates a clothing item to a homeless shelter or homelessness-related charity with every purchased.

# PROBLEM DEFINITION &DESIGN THINKING

## 2.1 EMPATHY MAP




## ADVANTAGES:

**1.Reduce your cost of acquiring new customers.**

**2.Develop better products products,faster aging gracefully.**

**3.Reduce customer churn that weighs down sales growth.**

**4.Experiment safely with customers who are open to change.**

## DISADVANTAGES

**1.Large investment in terms of price and time.**

**2.Require concerted commitment and Business Culture.**

## APPLICATION

For small businesses to do well, it is important that they hold on to their existing clientele.

While Indian small businesses have traditionally been great at customer retention, the highly competitive environment of the present times requires them to adopt a strategic approach. They can do this by:

1.Recognising the long-term value of each customer

2.Using technology wisely

3.Keeping track of events of interest to important clients

4.Watching out for cues that might indicate a client is preparing to leave

5.Customising offers for the most valuable clients

Using these strategies will ensure both stability of revenue and the chances of growth, thanks to the references provided by a loyal clientele.

## RESULT

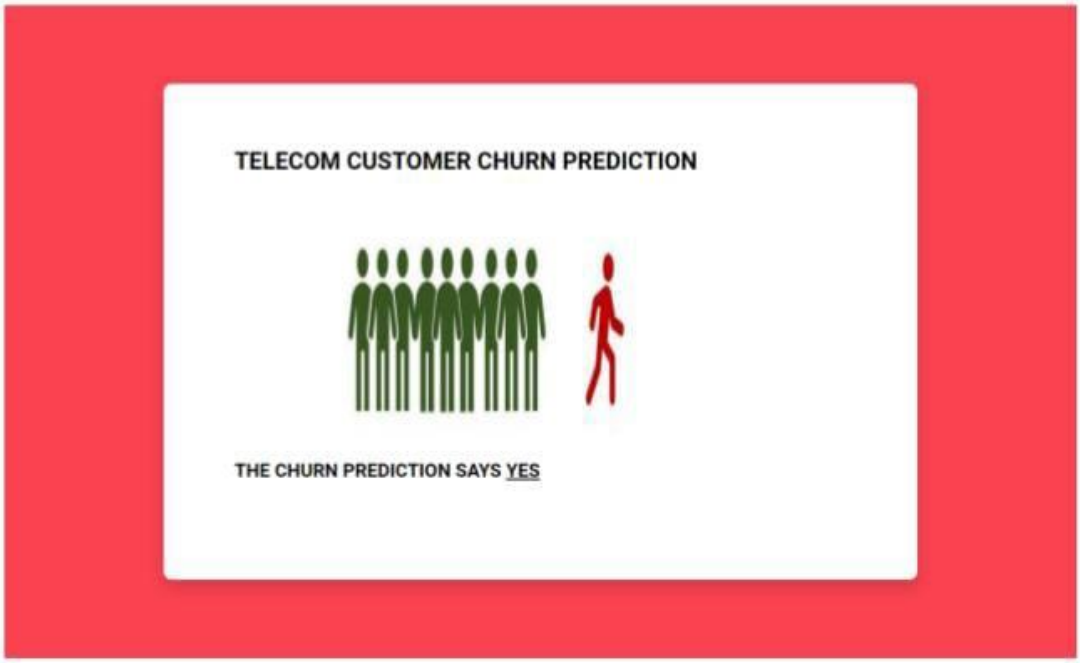
## PREDICTION FORM

| | |
|---|---|
| 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**




## CONCLUSION

The customer retention definition in marketing is the process of engaging existing customers to continue buying products or services from your business. It's different from customer acquisition or lead generation because you've already converted the customer at least once.

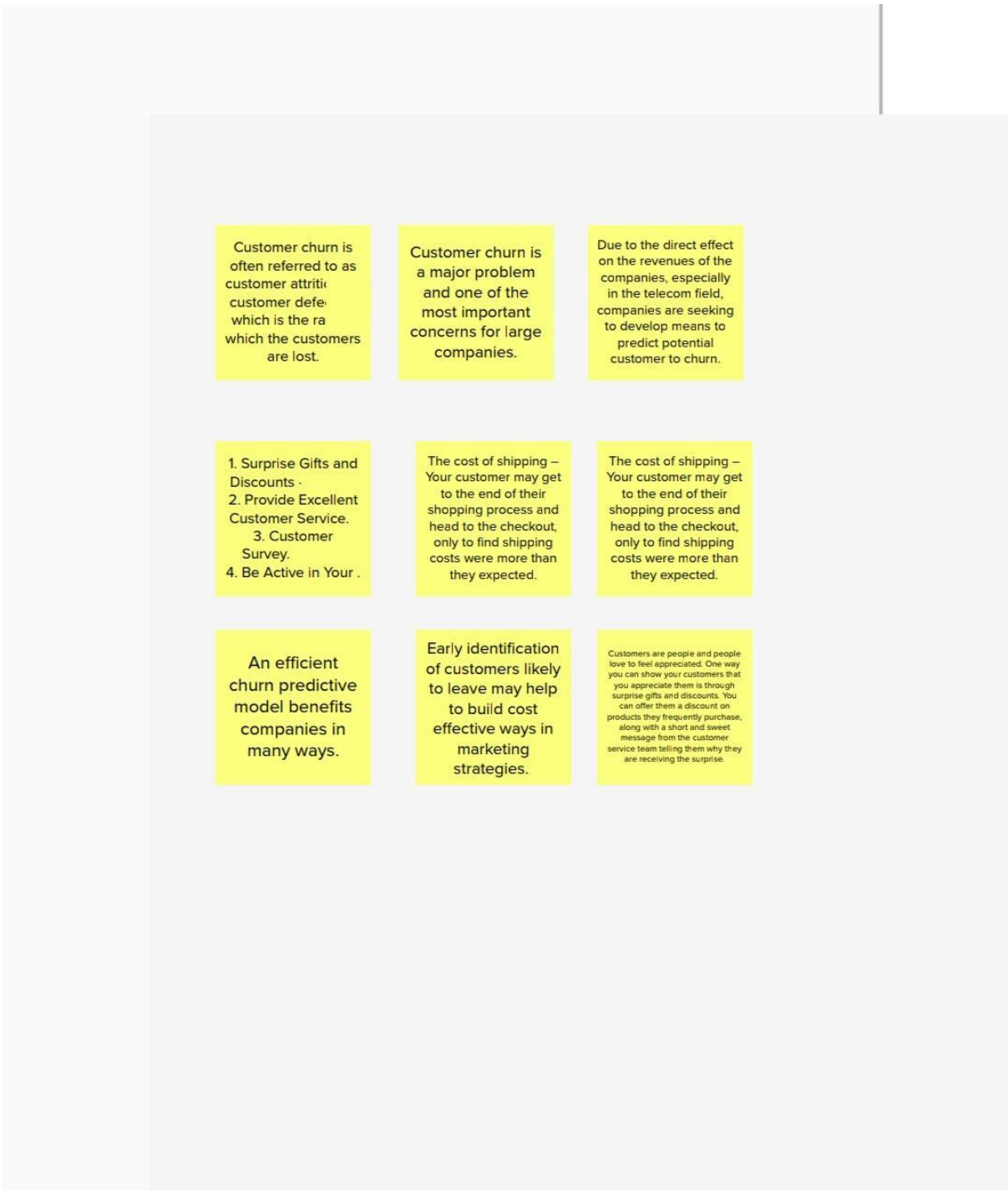
The best customer retention tactics enable you to form lasting relationships with consumers who will become loyal to your brand. They might even spread the word within their own circles of influence, which can turn them into brand ambassadors.

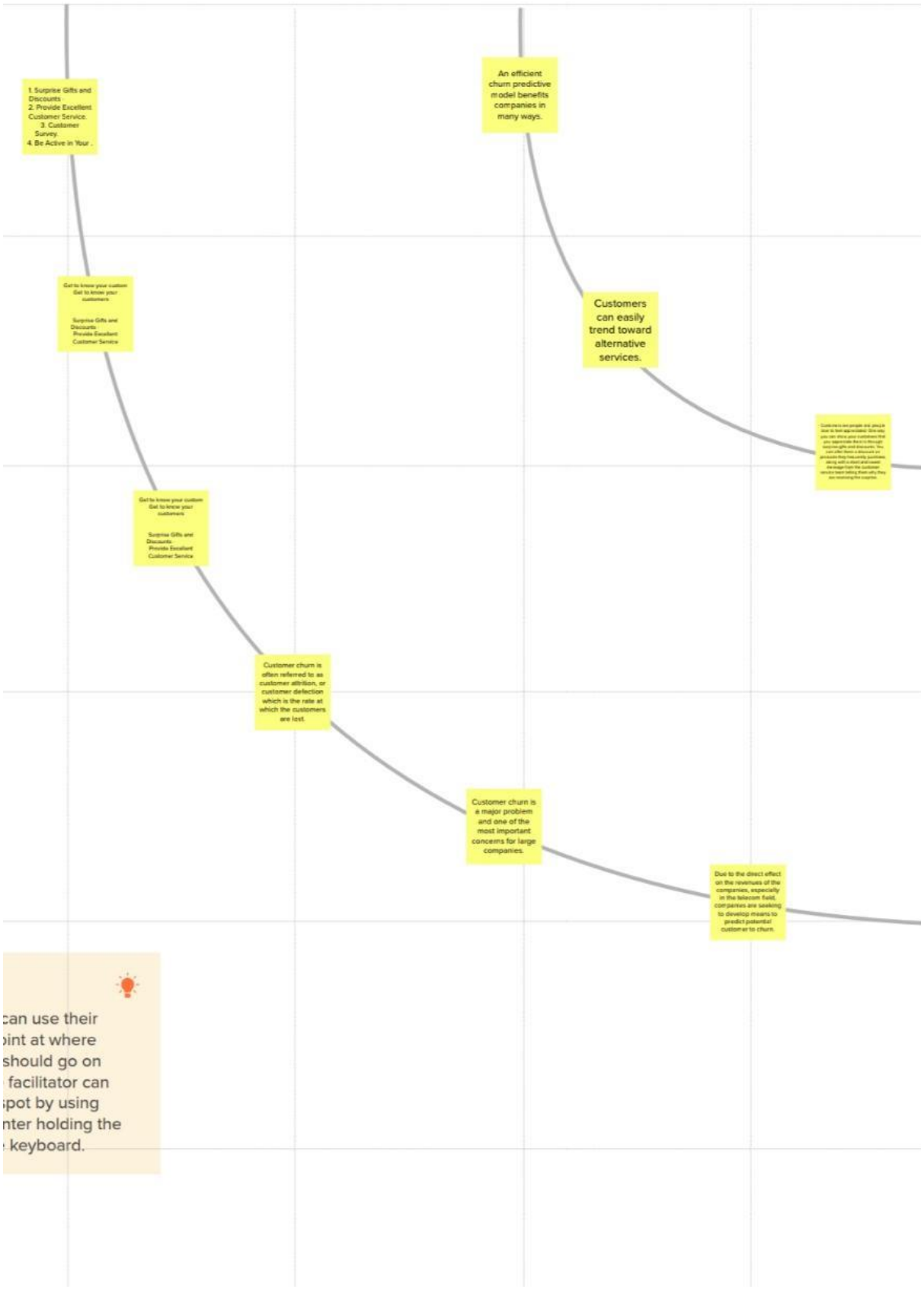
## FUTURE SCOPE

Large client base is the backbone of any successful business. This is especially true for small businesses, where the costs involved for winning a new customer makes customer retention extremely important.

**“It costs less to keep your current customers happy than to gain new ones.”**

## IDCATION & BRAINSTROMING




1. Surprise Gifts and Discounts
2. Provide Excellent Customer Service.
3. Customer Survey.
4. Be Active in Your .
An efficient churn predictive model benefits companies in many ways.
Get to know your custom Get to know your customers
Surprise Gifts and Discounts Provide Excellent Customer Service
Customers can easily trend toward alternative services.
Get to know your custom Get to know your customers
Surprise Gifts and Discounts Provide Excellent Customer Service
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.
can use their
oint at where
should go on
facilitator can
spot by using
nter holding the
keyboard.

# APPENDIX

## SOURCE CODE

```
#import necessary libraries

import pandas as pd

importnumpy as np

import pickle

importmatplotlib.pyplot as plt

%matplotlib inline

importseaborn as sns

importsklearn

fromsklearn.preprocessing import LabelEncoder, OneHotEncoder

fromsklearn.linear_model import LogisticRegression

fromsklearn.tree import DecisionTreeClassifier

fromsklearn.ensemble import RandomForestClassifier

fromsklearn.neighbors import KNeighborsClassifier

fromsklearn.svm import SVC

fromsklearn.model_selection import RandomizedSearchCV

importimblearn

fromimblearn.over_sampling import SMOTE

fromsklearn.model_selection import train_test_split

fromsklearn.preprocessing import StandardScaler

fromsklearn.metrics import accuracy_score, classification_report,
confusion matrix, f1_score
data = pd.read_csv(r"C:\Users\Shivani_SB\OneDrive\Desktop\Telecom churn
modelling-updated\data\Dataset.csv") data
data.info()
data.TotalCharges = pd.to_numeric(data.TotalCharges, errors='coerce')

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

data.isnull().sum()
fromsklearn.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
fromsklearn.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)
fromimblearn.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")
fromsklearn.model selection import train_test_split
x_train,x_test,y_train,y_testtrain_test_split(x_resample,y resample,
test_size = 0.2, random state = 0)
fromsklearn.preprocessing import StandardScaler SC Standardscaler()
x_train = sc.fit_transform(x_train) x_testsc.fit_transform(x_test)

x_train.shape
#importing and building the Decision tree model
deflogreg(x_train,x_test,y_train,y_test):

Ir = LogisticRegression (random_state=0) 1r.fit(x_train,y_train)

N print(accuracy_score (yPred_lr,y_test))
print(classification_report(y_test,yPred_1r))

y_lr_trlr.predict(x_train)

print(accuracy_score (y_lr_tr,y_train))

yPred_1r = lr.predict(x_test)

print("**Logistic Regression**")
```

```
print("Confusion_Matrix")

print(confusion_matrix (y_test,yPred_1r))

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)
#importing and building the Decision tree model

defdecisionTree(x_train,x_test,y_train,y_test):

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

dtc.fit(x_train,y_train)

y_dt_trdtc.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))

#printing the train accuracy and test accuracy respectively

decisionTree(x_train,x_test,y_train,y_test)

#importing and building the random forest model

defRandomForest(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))

#printing the train accuracy and test accuracy respectively

RandomForest(x_train,x_test,y_train,y_test)

#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))

#printing the train accuracy and test accuracy respectively

KNN(x_train,x_test,y_train,y_test)

#importing and building the random forest model

defsvm(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))
```

```
#printing the train accuracy and test accuracy respectively

svm(x_train,x_test,y_train,y_test)

[] # Importing the Keras libraries and packages

importkeras

fromkeras.models import Sequential

fromkeras.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)

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("Confusion_Matrix")

print(confusion matrix(y_test,ann_pred))

print("Classification Report")

print(classification_report(y_test,ann_pred))

#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)

#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)

#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)

#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)

#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)

#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_owne.5)

print("output is: ",ann_pred_own)

defcompareModel(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(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))

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 Forest after Hyperparameter tuning***")

print("Confusion_Matrix")

print(confusion_matrix(y_test,yPred_rfcv))

print("Classification Report")

print(classification_report(y_test,yPred_rfcv))

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("telcom_churn.h5")

from flask import Flask, render_template, request

importkeras

fromkeras.models import load_model

app = Flask(__name__)

model = load_model("telcom_churn.h5")
@app.route('/')

def home():

returnrender_template("home.html")

@app.route('/')

defhelloworld():

returnrender_template("base.html")

@app.route('/assesment')

def prediction():
```

```
returnrender_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'):
i1,12,13=0,1,0
if (i == 'y'):
i1,i2,13=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
1= 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=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(12),int
(13),int(j1)

print(t)

x = model.predict(t)

print(x[0])

if (x[[0]] <=0.5):
```

```
y ="No"
returnrender_template("predno.html", z = y)
if (x[[0]] >= 0.5):
y ="Yes"
returnrender_template("predyes.html", z = y)
```