



Government of Tamil Nadu

Naan Muthalvan - Project-Based Experiential Learning

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

Submitted by

Team ID: NM2023TMID22182

M.PRIYADHARSHINI - (20326ER025)

S.POOMAYIL - (20326ER022)

P.POORNIMA – (20326ER023)

G.PRADEEPA – (20326ER024)

Under the guidance of
Mrs. J.SUKANYA MCA., M.Phil.,
Assistant Professor

PG and Research Department of Computer Science



M.V.MUTHIAH GOVERNMENT ART SCOLLEGE FOR WOMEN

(Affiliated To Mother Teresa Women's University, Kodaikanal)

Reaccredited with "A" Grade by NAAC

DINDIGUL-624001.

APRIL-2023

M.V.MUTHIAH GOVERNMENT ARTS COLLEGE FOR WOMEN
(Affiliated to Mother Teresa Women's University, Kodaikanal)
Reaccredited with "A" Grade by NAAC
Dindigul-624001



PG & RESEARCH DEPARTMENT OF COMPUTER SCIENCE

BONAFIDE CERTIFICATE

This is to certify that this is a bonafide record of the project entitled, **"INTELLIGENT CUSTOMER RETENTION: USING MACHINE LEARNING FOR ENHANCED PREDICTION OF TELOCOM CUSTOMER CHURN"** done by **Ms.M.PRIYADHARSHINI(20326ER025)** **Ms.S.POOMAYIL(20326ER022)**, **Ms.P.POORNIMA(20326ER023)** and **Ms.G.PRADEEPA(20326ER024)**. This is submitted in partial fulfillment for the award of the degree of **Bachelor of Science in Computer Science in M.V.MUTHIAH GOVERNMENT ARTS COLLEGE FOR WOMEN,DINDIGUL** during the period of December 2022 to April 2023.

Project Mentor(s)

Head of the Department

Submitted for viva-voce Examination held on 11.04.2023

TABLE OF CONTENTS

S.No	Contents	Page No.
1	INTRODUCTION	1
	1.1 Overview	
	1.2 Purpose	
2	PROBLEM DEFINITION & DESIGN THINKING	6
	2.1 Empathy Map	
	2.2 Ideation & Brainstorming Map	
3	RESULT	9
4	ADVANTAGES & DISADVANTAGES	14
5	APPLICATIONS	16
6	CONCLUSION	17
7	FUTURE SCOPE	18
8	APPENDIX	19
	A. Source Code	

1. Introduction

1.1 Overview

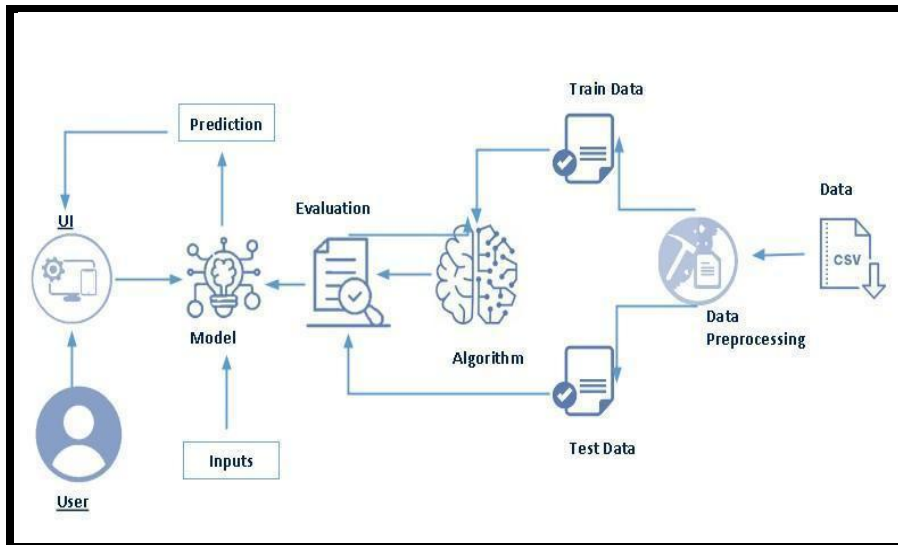
Customer churn is often referred to as customer attrition, or customer defection which is the rate at which the customers are lost. Customer churn is a major problem and one of the most important concerns for large companies. Due to the direct effect on the revenues of the companies, especially in the 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.

Technical Architecture



Project Flow

- User interacts with the UI to enter the input.
- Entered input is analysed by the model which is integrated.
- Once model analyses the input the prediction is showcased on the UI

To accomplish this, we have to complete all the activities listed below,

- Define Problem / Problem Understanding
 - Specify the business problem
 - Business requirements
 - Literature Survey
 - Social or Business Impact.

- Data Collection & Preparation
 - Collect the dataset
 - Data Preparation
- Exploratory Data Analysis
 - Descriptive statistical
 - Visual Analysis
- Model Building
 - Training the model in multiple algorithms
 - Testing the model
- Performance Testing & Hyperparameter Tuning
 - Testing model with multiple evaluation metrics
 - Comparing model accuracy before & after applying hyperparameter tuning
- Model Deployment
 - Save the best model
 - Integrate with Web Framework
- Project Demonstration & Documentation
 - Record explanation Video for project end to end solution
 - Project Documentation-Step by step project development procedure

Project Structure

- A python file called app.py for server side scripting.
- We need the model which is saved and the saved model in this content is churn.pkl
- Templates folder which contains base.HTML file, index.HTML file, predyes.HTML , predno.HTML file.
- Static folder which contains css folder which contains main.css , js folder which contains global.js , images folder and vendor folder.

1.2 Purpose

"Gather data: Telecom companies already have access to a vast amount of data, including customer demographics, usage patterns, and call history. This data can be combined with external data sources such as social media activity, online browsing history, and customer feedback to build a more comprehensive picture of each customer.

Prepare and preprocess data: The data must be prepared and preprocessed to ensure that it is clean and in the correct format for analysis. This process may include removing duplicates, filling in missing data, and transforming data into numerical or categorical variables that can be used for machine learning algorithms.

Feature engineering: Feature engineering involves selecting the most important variables for predicting customer churn. Telecom companies can use machine learning techniques to identify which variables are most predictive of churn and create new variables that can enhance prediction.

Train the machine learning model: The telecom company can use various machine learning algorithms, such as logistic regression, decision trees, and neural networks, to train the model. The model can be trained using a labeled dataset that includes information on which customers have churned and which customers have not.

Validate the model: Once the model is trained, it needs to be validated using a hold-out dataset. This dataset should be separate from the training dataset and should not be used in the model-building process. The validation dataset allows the telecom company to see how well the model performs on new data and can help identify any overfitting issues.

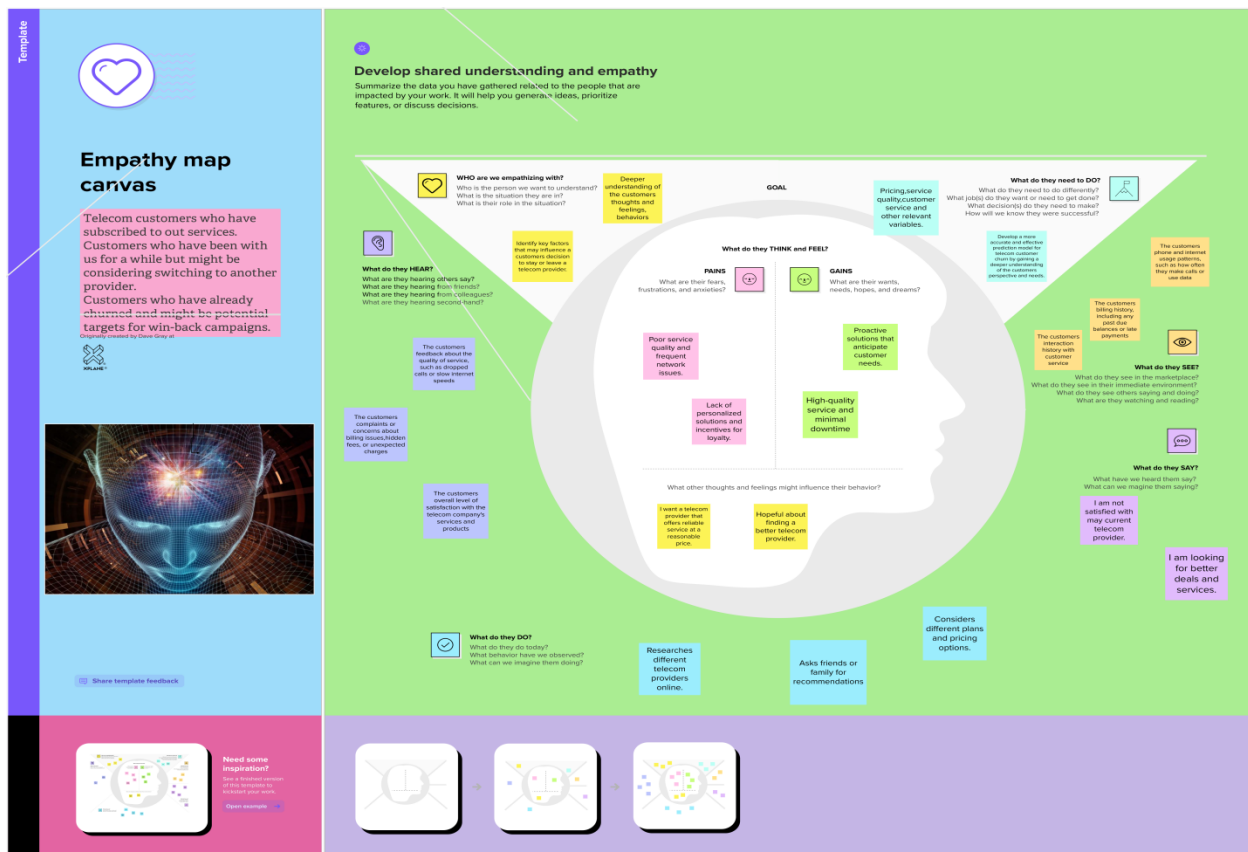
Deploy the model: After the model has been validated, it can be deployed in production to predict customer churn. The model can be used to identify customers who are at risk of leaving and provide them with personalized offers to encourage them to stay.

2. Problem Definitions & Design Thinking

2.1 Empathy Map

Customer Empathy Map is an amazing resource that helps companies achieve greater control in customer service. They can be used on its own or in combination with other CX devices, including customer journey maps, customer persona, service blueprints, motivation matrix, and mind maps.

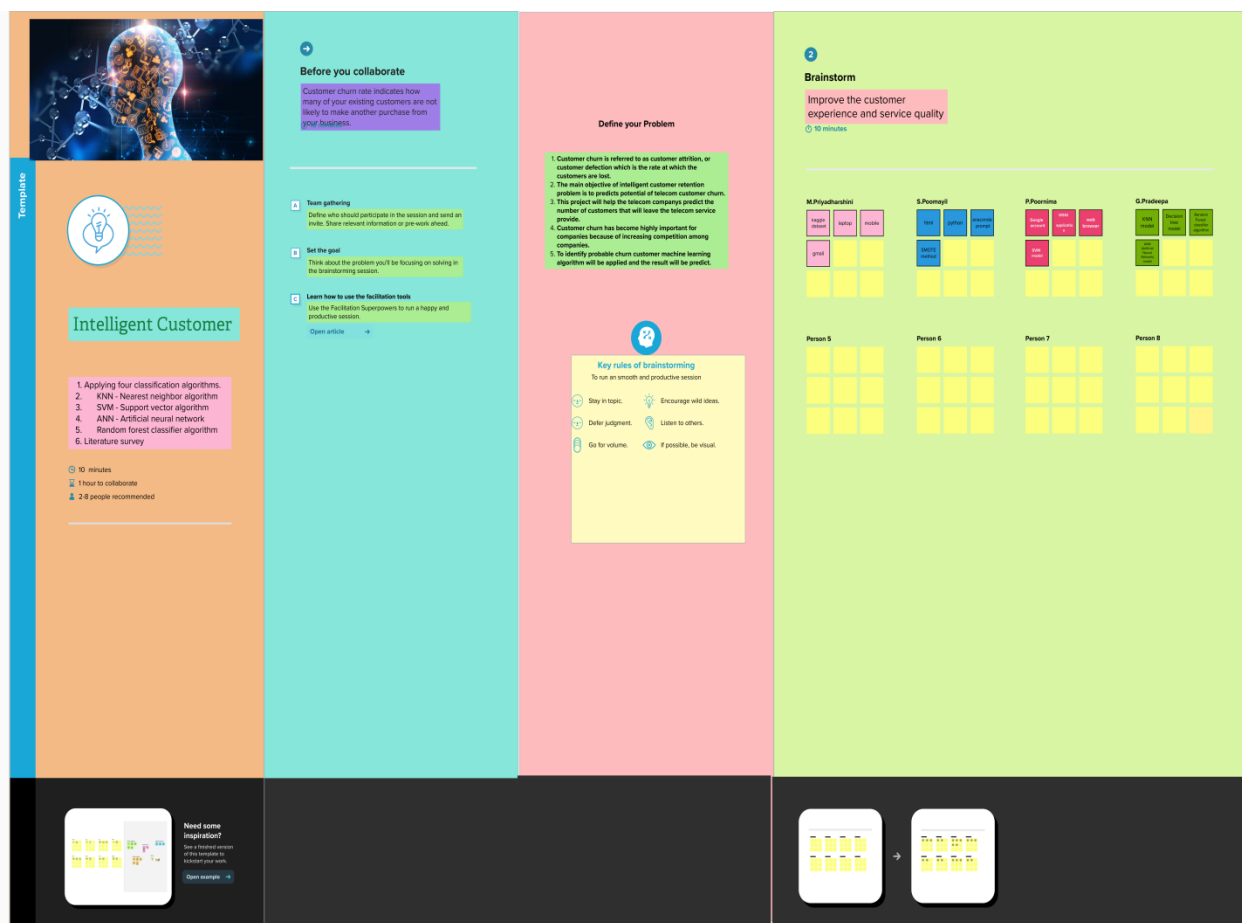
This human-centred design tool helps you gain a great deal of insight and understanding of how to get into the hearts and minds of your customers. You significantly increase the individual emotional intelligence when you use a customer empathy map. This gives you the ability to increase your organizational emotional intelligence.



2.2 Ideation & Brainstorming Map

One of the primary reasons to create a mind map is to allow yourself and your team members to have a proper platform from which everyone can easily brainstorm different ideas. As illustrated in the following brainstorming mind map, you can create different subtopics or subphrases for brainstorming on a different idea, like understanding the objective, noting down whom all are going to attend the brainstorming session, adding the questions that someone might ask, analyze how the brainstorming session went, implement the learnings, and more.

Creating such brainstorming mind maps helps companies or business owners note down all the important aspects of any meeting so they can easily execute it to see the results.



Group ideas

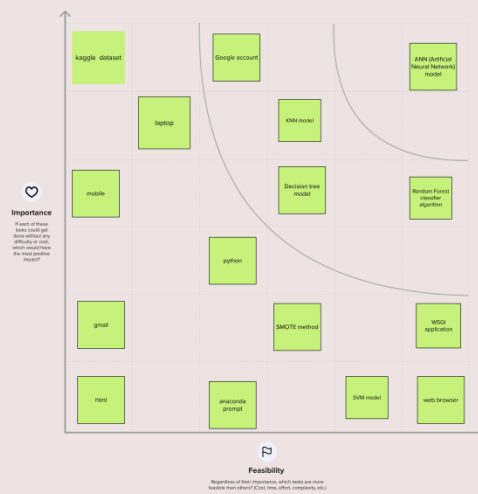
1. Be proactive with communication.
2. User interacts with the UI to enter the input.
3. Entered input is analysed by the model which is integrated.
4. Once model analyses the input the prediction is showcased on Th UI.
5. Offer incentives.



4

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes






We can export the mural as pdf to share, it is helpful to getting information.

Quick add-ons

- A Share the mural!**
Share a view link to the mural with stakeholders to keep them in the loop about the outcomes of the session.
- B Export the mural**
Export a copy of the mural as a PNG or PDF to attach to emails, include in slides, or save in your drive.

Keep moving forward

-  **Strategy blueprint**
Define the components of a new idea or strategy.
[Open the template →](#)
 -  **Customer experience journey map**
Understand customer needs, motivations, and obstacles for an experience.
[Open the template →](#)
 -  **Strengths, weaknesses, opportunities & threats**
Identify strengths, weaknesses, opportunities, and threats (SWOT) to develop a plan.
[Open the template →](#)

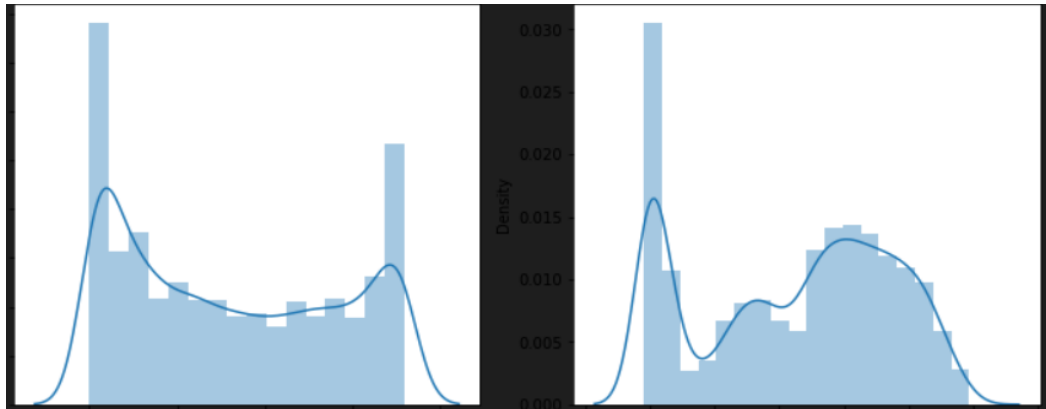
[Share template feedback](#)



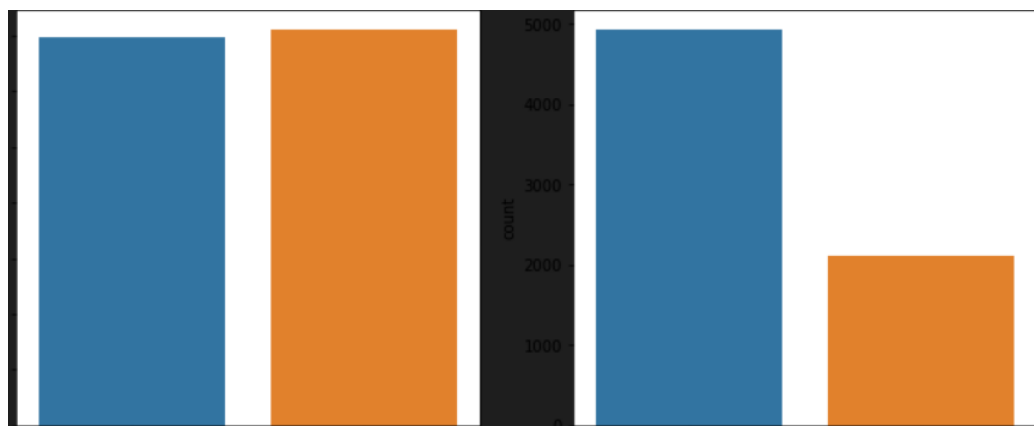
3. Result

- 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.

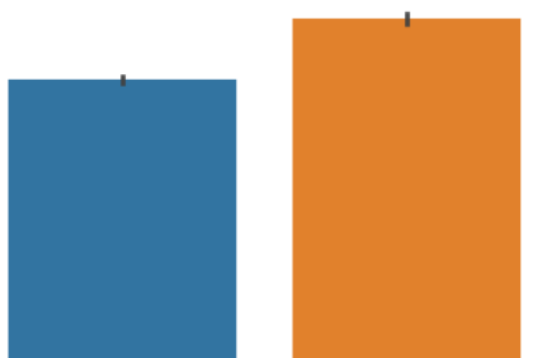
Univariate Analysis



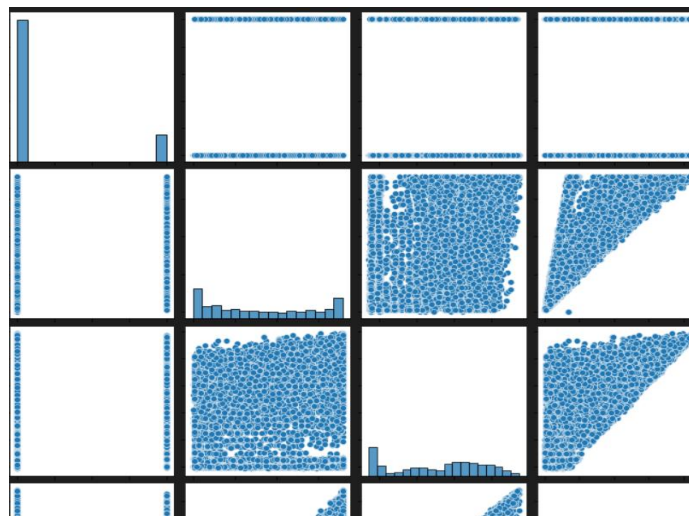
Countplot



Bivariate Analysis



Multivariate Analysis



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 & Disadvantages

Advantages

Cheaper than Acquisition: 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.

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: 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.

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.

5. Applications

1. Allow App User segmentations
2. Use in-App push notifications
3. In-app referral
4. Go for Smart Surveys
5. Deliver Contents per location
6. Incorporate Smart Analytics
7. Smart Contracts
8. Stir Loyalty
9. Optimizing in-app journeys
10. Incentivize with special features

6. Conclusion

Conclusion In this paper, the customer retention system based on intelligence was introduced. The Machine Learning \Data Mining techniques provided a powerful tool to predict customer churn. By using the rules to predict customers churn, the system will help telecom companies understand customer churn risk and get the list of customers deserved to perform company's retention strategies. Overall, the system is helpful in customizing marketing communications and customer treatment programs to optimally time their marketing

7. Future Scope

The implementation of an intelligent customer retention system using machine learning for enhanced prediction of telecom customer churn can have a significant impact on reducing customer churn rates and improving overall customer satisfaction.

By leveraging machine learning algorithms and analyzing customer data, telecom companies can accurately predict which customers are at risk of churning and take proactive measures to retain them. This can include targeted marketing campaigns, personalized offers, and improved customer service.

The success of this project will ultimately depend on the quality of the data used and the accuracy of the machine learning algorithms employed. It is important to continuously monitor and adjust the algorithms to ensure they remain effective over time.

Overall, the use of machine learning for customer retention is a promising area that has the potential to revolutionize the way telecom companies approach customer churn. By investing in this technology, companies can improve customer loyalty, increase revenue, and gain a competitive advantage in the market.

8. Appendix

A. Source Code

```
#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

#import dataset
data=pd.read_csv(r"/content/Telco_Cust_Churn.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()

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[:,1:20].values
y=data.iloc[:,20:21].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)

```

```
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"])
```

```
import matplotlib.pyplot as plt  
import seaborn as sns  
import pandas as pd  
#loading the dataset 'tips'  
df=pd.read_csv(r"/content/sample_data/Telco_Cust_Churn.csv")  
#plotting the graph  
plt.figure(figsize=(12,5))  
plt.subplot(1,2,1)  
sns.countplot(x='gender',data=df)  
plt.subplot(1,2,2)  
sns.countplot(x='Dependents',data=df)  
plt.show()
```

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

```
sns.heatmap(data.corr(),annot=true)
```

```
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
```

```
#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)

```

```

#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))

```

```

#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 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(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 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
def SVM(x_train,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 and building the random forest model
def SVM(x_train,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 train 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 in 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("Classiification 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([[0,0,1,1,0,0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,1,0,0,456,1,
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([[0,0,1,1,0,0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,1,0,0,456,
1,0,3425,4567]]))
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("Prediction on random input")
rf_pred_own =
rf.predict(sc.transform([[0,0,1,1,0,0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,1,0,0,456,1,
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("Prediction on random input")
svm_pred_own =
svc.predict(sc.transform([[0,0,1,1,0,0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,1,0,0,456,
1,0,3245,4567]]))
print("output is:",svm_pred_own)

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

from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

# define the random forest classifier model
model = RandomForestClassifier()

# define the hyperparameters to tune
params = {
    'n_estimators': [50, 100, 200],
    'max_depth': [5, 10, 20],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}

# perform grid search cross-validation to find the best hyperparameters
grid_search = GridSearchCV(model, params, cv=5)
grid_search.fit(x_train, y_train)

# print the best hyperparameters found by grid search
print("Best hyperparameters:", grid_search.best_params_)

# get the best model from grid search
model = grid_search.best_estimator_

# evaluate the model on the training set
y_rf = model.predict(x_train)
print("Training set accuracy:", accuracy_score(y_rf, y_train))

# evaluate the model on the test set

```

```

yPred_rfcv = model.predict(x_test)
print("Test set accuracy:", accuracy_score(yPred_rfcv, y_test))

# print the confusion matrix and classification report for the test set
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))

# use the model to predict on a new input
rfcv_pred_own =
model.predict(sc.transform([[0,0,1,1,0,0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,1,0,0,4
56,1,0,3245,4567]]))
print("Output is:", rfcv_pred_own)

classifier.save("telcom_churn.h5")

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

app = Flask(__name__)
model = load_model("telcom_churn.h5")@app.route('/')# rendering the html template
def home():
    return render_template("home.html")

#testing on random input values
print("Prediction on random input")
ann_pred_own =
classifier.predict(sc.tarnform([[0,0,1,1,0,0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,1,0,0,1,1,0,0,
456,1,10,3245,4567]]))

print(ann_pred_own)
ann_pred_own = (ann_pred_own)
print("output is:",ann_pred_own)

@app.route('/')
def helloworld():
    return rendder_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["scritcion"]
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'):

@app.route('/')
def helloworld():
    return rendder_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["scritcion"]
    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'):

@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["scrition"]
    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'):

11,12,13=1,0,0
if(1=='nis'):
    11,12,13=0,1,0
if(1=='y'):
    11,12,13=0,0,1
m= request.form["stv"]
if(m=='n'):
    m1,m2,m3=1,0,0
if(m=='y'):
    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=0,0,1
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=1,0,0
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'):

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=0,0,0
if(h=='fo'):
    h1,h2,h3=0,0,1
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=0,0,1
if(i=='y'):
    i1,i2,i3=0,0,1
j= request.form["ob"]
if(j=='n'):
    j1,j2,j3=1,0,0
if(j=='y'):
    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

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)

```