

ARIGNAR ANNA GOVERNMENT ARTS COLLEGE

VILLUPURAM – 605 602.

INTELENT CUSTOMER RETENTION:USING MACHINE LEARNING FOR
ENHANCED PREDICTION OF TELECOM CUSTOMER CHURN

Team Leader: KIRUTHIKA.D

Team Members:

1. LAVANYA.K
2. LENIN.E
3. PREMKUMAR.B

1. INTRODUCTION

1.1 Overview:

Intelligent customer retention refers to the use of advanced technologies and data analytics to enhance customer retention strategies. It involves the analysis of customer data to identify patterns and behaviors, which can be used to develop targeted retention strategies that are designed to keep customers engaged and loyal to a brand or company.

Some of the key components of intelligent customer retention include the use of machine learning algorithms to analyze customer data, the use of predictive analytics to identify customers who are at risk of churn, and the development of personalized marketing campaigns that are tailored to each customer's unique needs and preferences.

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 identify the probable churn customers and then makes the necessary business decisions.

1.2purpose

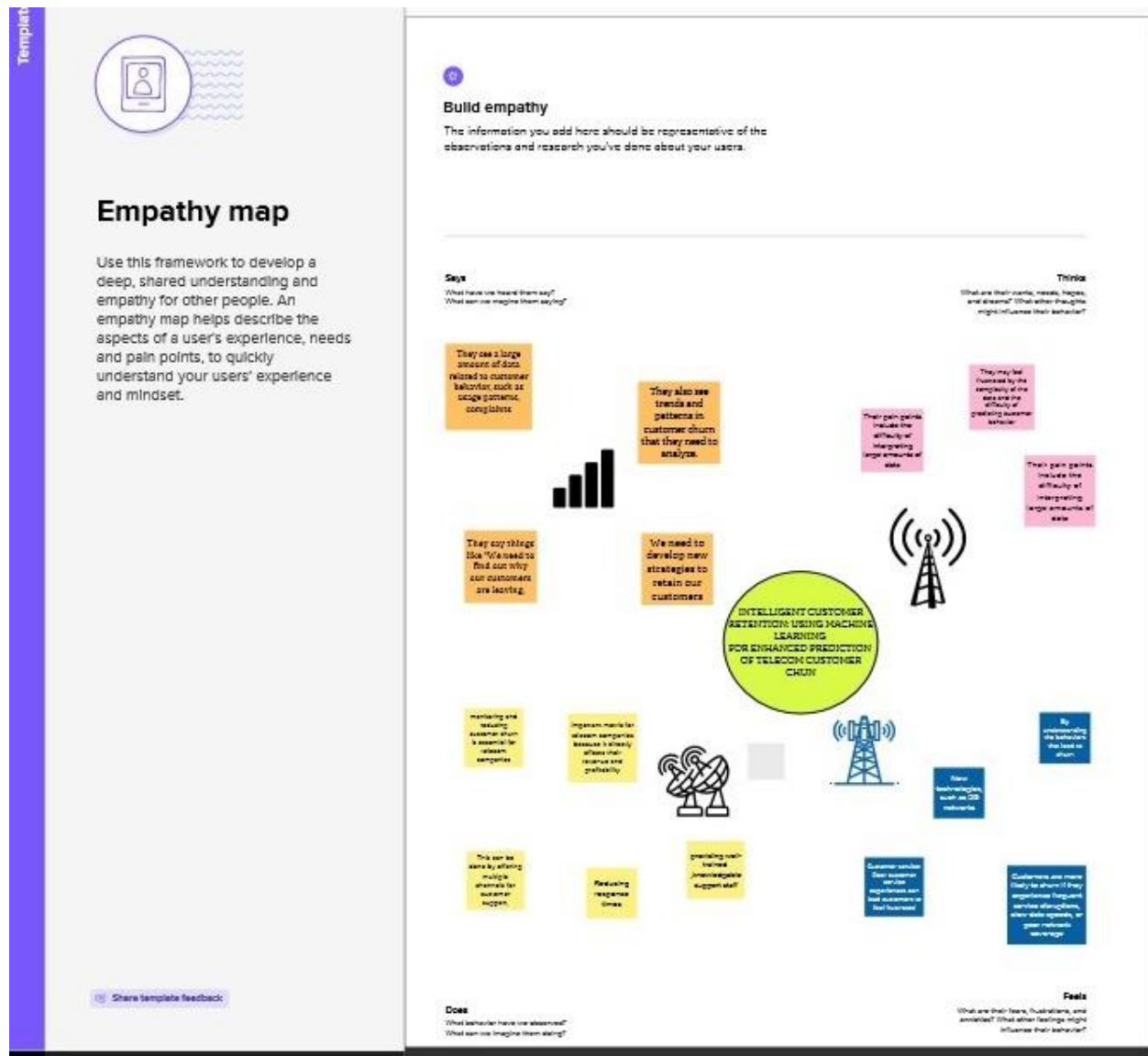
purpose of customer retention is to maintain a loyal and satisfied customer base that continues to do business with a company over time. Retaining customers is important for several reasons, including:

- **Cost-effectiveness:** It is generally more cost-effective to retain existing customers than to acquire new ones. Retaining customers reduces marketing and advertising costs and can also lead to increased sales and profits.
- **Brand reputation:** Customers who are satisfied with a company's products or services are more likely to recommend it to others, which can lead to positive word-of-mouth advertising and a strong brand reputation.

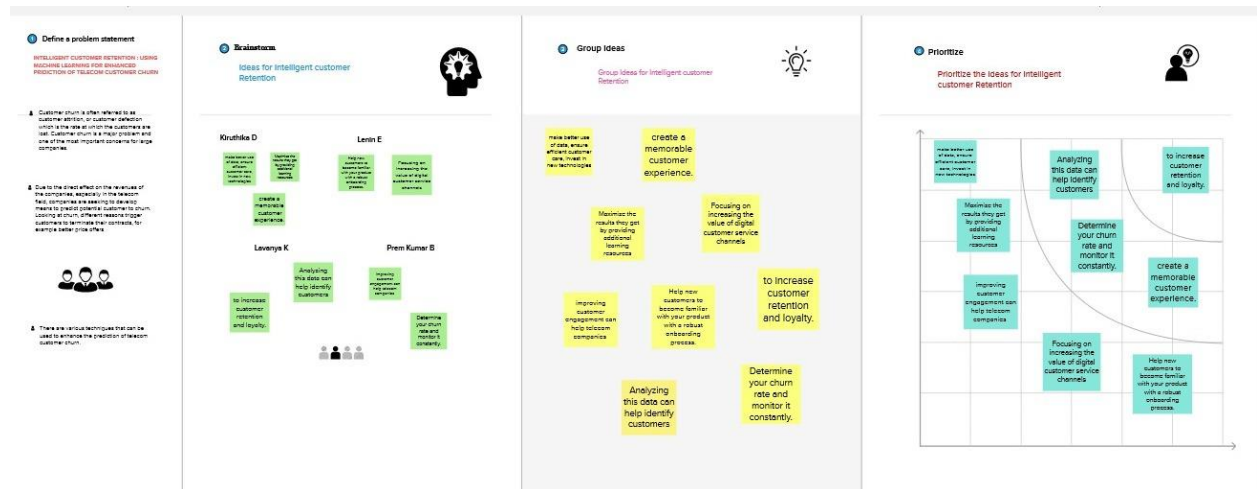
- Competitive advantage: A loyal customer base can provide a competitive advantage for a company, as customers are less likely to switch to a competitor product or service.
- Long-term revenue: Retaining customers can lead to long-term revenue streams, as loyal customers are more likely to make repeat purchases and may also be more likely to purchase additional products or services from the company.
- In summary, the purpose of customer retention is to build long-term relationships with customers, which can lead to increased profitability, positive brand reputation, and a competitive advantage in the marketplace.

2. PROBLEM DEFINITION AND DESIGN THINKING

2.1 Empathy map



2.2 Ideation&BrainstromingMap



3. ADVANTAGES AND DISADVANTAGES

Advantages:

- **Increased revenue:** Retaining existing customers is generally more cost-effective than acquiring new ones. According to research, it can cost up to five times more to attract a new customer than to retain an existing one. By focusing on customer retention, businesses can increase their revenue and profits.
- **Repeat business:** Retained customers are more likely to make repeat purchases from a business. This repeat business not only generates revenue, but it also helps to build long-term relationships with customers, leading to increased loyalty and advocacy.
- **Lower marketing costs:** By focusing on customer retention, businesses can reduce their marketing costs as they do not need to invest as much in customer acquisition. This can help to improve profit margins and overall business efficiency.
- **Increased referrals:** Satisfied, loyal customers are more likely to refer their friends and family to a business. This can help to increase the customer base and generate new business through word-of-mouth marketing.

- **Improved customer lifetime value:** Customer lifetime value (CLV) refers to the total value a customer brings to a business over their lifetime. Retained customers typically have a higher CLV as they make more purchases and are more likely to refer others to the business. This can help to improve the overall financial performance of the business.

Disadvantages:

- While customer retention is generally considered to be a key factor in the success of a business, there are also some potential disadvantages to consider:
- **Increased cost:** Retaining customers often requires ongoing investment in customer service, marketing, and loyalty programs, which can be costly for businesses, especially small ones.
- **Limited growth:** Focusing too heavily on retaining existing customers may limit a business's ability to acquire new customers and expand its market share.
- **Reduced innovation:** When businesses become too focused on keeping their existing customers happy, they may become less willing to take risks and try new things, which could stifle innovation and growth.

- **Stagnation:** Over time, customer needs and preferences may change, and if a business is too focused on retaining its existing customers, it may not adapt quickly enough to these changes, which could lead to stagnation.
- **Dependence:** If a business becomes too reliant on a small number of loyal customers, it may be vulnerable to fluctuations in their purchasing behavior or changes in the market, which could put the business at risk.

4. APPLICATIONS

- ✓ **Retail:** Retail businesses can use customer retention strategies to encourage repeat purchases and increase customer loyalty. For example, offering loyalty programs, personalized recommendations, and exclusive promotions to existing customers can incentivize them to return to the store.
- ✓ **E-commerce:** E-commerce businesses can use customer retention strategies to increase customer engagement and reduce churn rates. For example, sending personalized product recommendations based on a customer's purchase history, offering free shipping for repeat customers, and providing exceptional customer service can help keep customers coming back.
- ✓ **Hospitality:** Hotels and restaurants can use customer retention strategies to improve guest satisfaction and increase repeat visits. For example, offering loyalty programs, personalized recommendations, and special discounts to regular guests can incentivize them to continue booking stays or making reservations.
- ✓ **Subscription services:** Subscription-based businesses can use customer retention strategies to reduce churn rates and increase customer lifetime

value. For example, offering exclusive content or early access to new products to long-term subscribers, or providing personalized recommendations based on their viewing history, can encourage them to continue using the service.

- ✓ Overall, customer retention is a critical aspect of any business's success. By implementing effective customer retention strategies, businesses can improve customer satisfaction, increase revenue, and build long-term relationships with their customers.

5. CONCLUSION

In conclusion, customer retention is essential for the long-term success of any business. By prioritizing the needs and expectations of their customers and implementing effective retention strategies, businesses can build strong relationships with their customers and ensure a steady revenue stream. Customer retention is a crucial aspect of any business as it helps to maintain a stable customer base and increase revenue over time. Retaining customers is more cost-effective than acquiring new ones, and loyal customers can also become brand advocates who bring in more business through positive word-of-mouth.

To achieve customer retention, businesses need to focus on building strong relationships with their customers by providing exceptional customer service, personalized experiences, and continuously meeting their needs and expectations. Regular communication and engagement with customers can also help build trust and loyalty.

Some effective strategies for customer retention include offering loyalty programs, providing exclusive discounts and promotions, and creating a seamless and convenient customer experience across all channels.

Businesses should also analyze customer data and feedback to identify areas for improvement and tailor their retention strategies accordingly.

In conclusion, customer retention is essential for the long-term success of any business. By prioritizing the needs and expectations of their customers and implementing effective retention strategies, businesses can build strong relationships with their customers and ensure a steady revenue stream.

6. FUTURE SCOPE

The future scope of customer retention is vast, and it is expected to become increasingly important for businesses as customer expectations continue to evolve. Here are some possible directions that customer retention may take in the future.

- **Personalization:** The use of advanced data analytics and machine learning algorithms can enable businesses to create highly personalized experiences for each customer. This can help to improve customer satisfaction and loyalty, as well as increase the likelihood of repeat purchases.
- **Omnichannel approach:** In the future, businesses are likely to focus on creating a seamless experience across multiple channels, including social media, mobile apps, and brick-and-mortar stores. This will require a holistic approach to customer engagement, where businesses will need to ensure that customers receive consistent messaging and experiences across all touchpoints.
- **Emotional engagement:** Customer retention strategies are likely to focus more on emotional engagement in the future, rather than

just transactional interactions. This means that businesses will need to invest in building relationships with customers, creating meaningful experiences, and demonstrating that they truly care about their customers' needs.

- **Sustainability:** With increasing awareness of environmental issues, businesses that prioritize sustainability and social responsibility are likely to be more successful in retaining customers in the future. Customers are more likely to stay loyal to companies that align with their values and make a positive impact on the world . Overall, the future of customer retention is likely to be characterized by a deeper understanding of customer needs and preferences, greater personalization and emotional engagement, and a focus on sustainability and social responsibility .

MILE STONE2:DATA COLLECTION &PREPARATION:

Process of preparing raw data so that it is suitable for further processing and analysis.

ACTIVITY1.1:Importing the libraries

Import the necessary libraries as shown in the image.

```
Help
+ Code + Text

import pandas as pd
import numpy as np
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.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
```

ACTIVITY 1.2: Read the data set

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

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

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

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	Multiplelines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupport	StreamingTV	Strea
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	...	No	No	No	
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	...	Yes	No	No	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	...	No	No	No	
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes	...	Yes	Yes	No	
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	...	No	No	No	

5 rows x 21 columns

ACTIVITY2:Data preparation

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

The download data set is not suitable for training the machine learning model as it might have so much randomness so we need to clean the

dataset properly in order to fetch good results. This activity includes the following steps.

- Handling missing value
- Handling categorical dat
- Handling Imbalance Data

ACTIVITY 2.1 Handling missing values

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

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

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



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

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

ACTIVITY2.2: Handling Categorical Values

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

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

Label Encoding.

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

```

from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()

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

```

Data after label encoding

[] data.head()

	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	OnlineBackup	DeviceProtection	TechSupport	StreamingTV	StreamingMovies
0	0	0	1	0	1	0	1	0	0	2	0	0	0	0
1	1	0	0	0	34	1	0	0	2	0	2	0	0	0
2	1	0	0	0	2	1	0	0	2	2	0	0	0	0
3	1	0	0	0	45	0	1	0	2	0	2	2	0	0
4	0	0	0	0	2	1	0	1	0	0	0	0	0	0

Splitting the Dataset into Dependent and Independent variable

Let's split our dataset into independent and dependent variables. 1. The independent variable in the dataset would be considered as 'x' and gender,SeniorCitizen,Partner,Dependents,tenure,PhoneService,MultipleLines,InternetService,OnlineSecurity,OnlineBackup,DeviceProtection,TechSupport,StreamingTV,StreamingMovies,Contract,PaperlessBilling,PaymentMethod,MonthlyCharges ,TotalCharges columns would be considered as independent

variable. 2. The dependent variable in the dataset would be considered as 'y' and the 'Churn' column is considered as dependent variable.

Now we will split the data of independent variables.

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

After splitting we see the data as below.

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

Y

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

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


```
[ ] 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)
```

ACTIVITY 2.3: Handling Imbalance Data

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

For Balancing the data we are using the SMOTE Method.

```
[ ] x_resample
```

```
array([[0.00000000e+00, 1.00000000e+00, 0.00000000e+00, ...,
        1.00000000e+00, 2.98500000e+01, 2.98500000e+01],
       [1.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
        0.00000000e+00, 5.69500000e+01, 1.88950000e+03],
       [1.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
        1.00000000e+00, 5.38500000e+01, 1.08150000e+02],
       ...,
       [0.00000000e+00, 0.00000000e+00, 1.00000000e+00, ...,
        0.00000000e+00, 8.13885654e+01, 3.32649207e+02],
       [4.24511674e-01, 0.00000000e+00, 5.75488326e-01, ...,
        1.00000000e+00, 8.36760207e+01, 3.39727798e+02],
       [0.00000000e+00, 0.00000000e+00, 1.00000000e+00, ...,
        1.00000000e+00, 9.52513236e+01, 9.52513236e+01]])
```

```
[ ] y_resample
```

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

```
[ ] x.shape,x_resample.shape
```

```
((7043, 40), (10348, 40))
```

```
[ ] y.shape,y_resample.shape
```

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

```
[ ] data.describe()
```

	SeniorCitizen	tenure	MonthlyCharges
count	7043.000000	7043.000000	7043.000000
mean	0.162147	32.371149	64.761692
std	0.368612	24.559481	30.090047
min	0.000000	0.000000	18.250000
25%	0.000000	9.000000	35.500000
50%	0.000000	29.000000	70.350000
75%	0.000000	55.000000	89.850000
max	1.000000	72.000000	118.750000

MILESTONE3: Exploratory Data Analysis

ACTIVITY1: Descriptive statistical

Descriptive analysis is to study the basic features of data with the statistical process. Here pandas has a worthy function called describe. With this describe function we can understand the unique, top and frequent values of categorical features. And we can find mean, std, min, max and percentile values of continuous features.

```
[ ] data.describe()
```

	SeniorCitizen	tenure	MonthlyCharges
count	7043.000000	7043.000000	7043.000000
mean	0.162147	32.371149	64.761692
std	0.368612	24.559481	30.090047
min	0.000000	0.000000	18.250000
25%	0.000000	9.000000	35.500000
50%	0.000000	29.000000	70.350000
75%	0.000000	55.000000	89.850000
max	1.000000	72.000000	118.750000

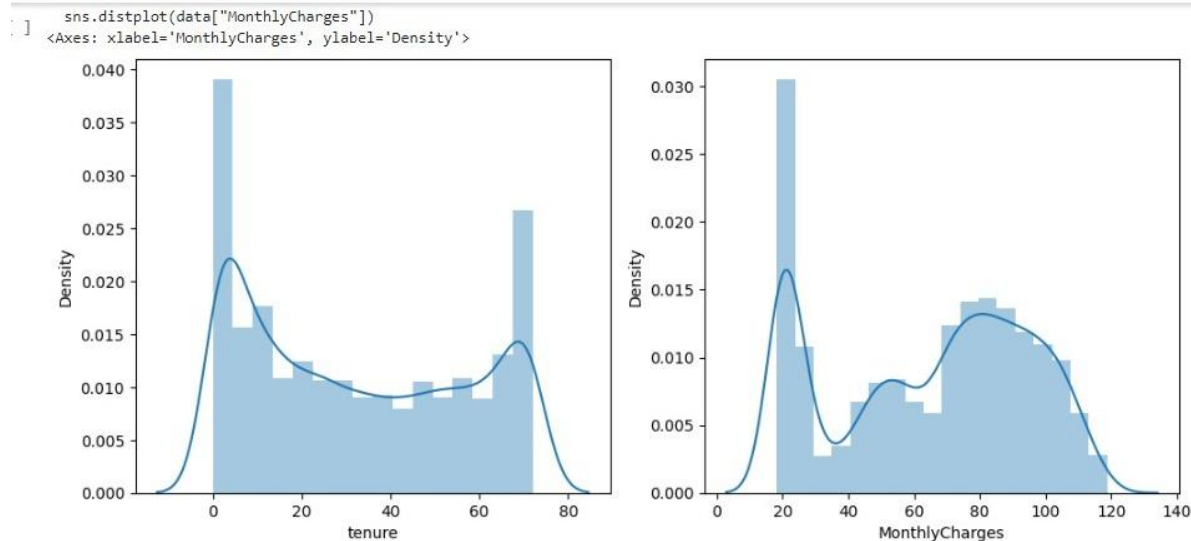
Activity 2: Visual analysis

Visual analysis is the process of using visual representations, such as charts, plots, and graphs, to explore and understand data. It is a way to quickly identify patterns, trends, and outliers in the data, which can help to gain insights and make informed decisions.

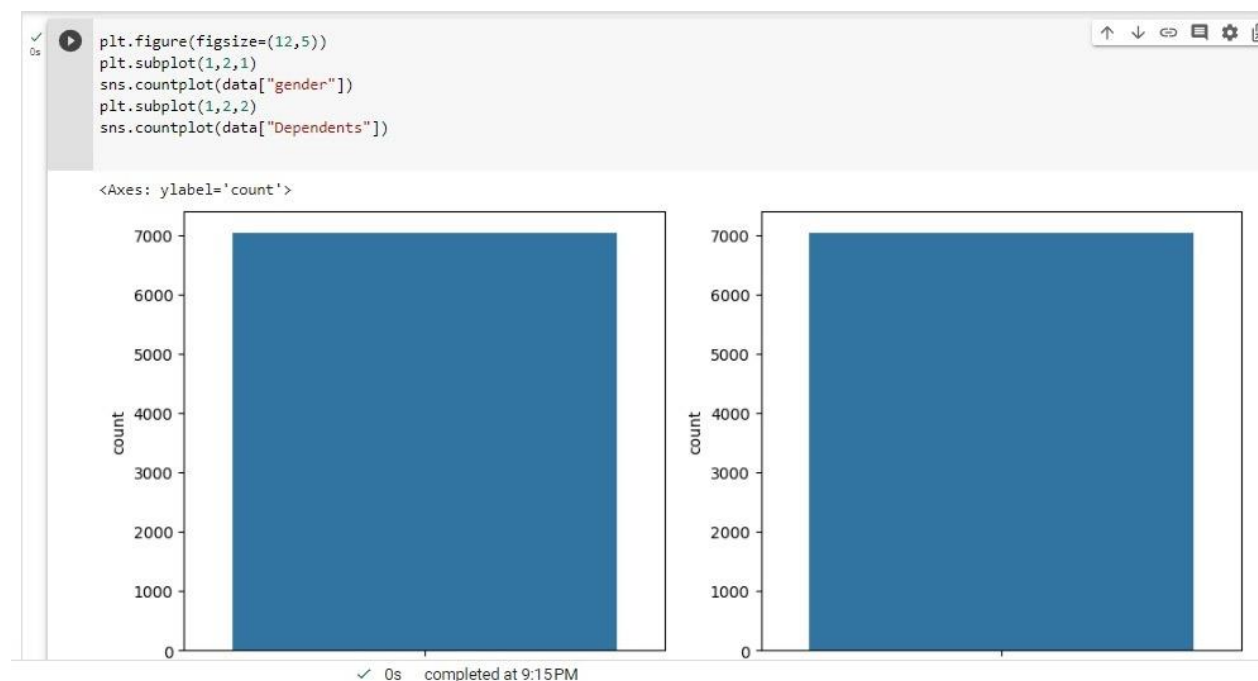
Activity 2.1: Univariate analysis

In simple words, univariate analysis is understanding the data with a single feature. Here we have displayed two different graphs such as distplot and countplot.

- The Seaborn package provides a wonderful function `distplot`. With the help of `distplot`, we can find the distribution of the feature. To make multiple graphs in a single plot, we use `subplot`.



Countplot :A count plot can be thought of as a histogram across a categorical, instead of quantitative, variable. The basic API and options are identical to those for `barplot()` , so you can compare counts across nested variables.

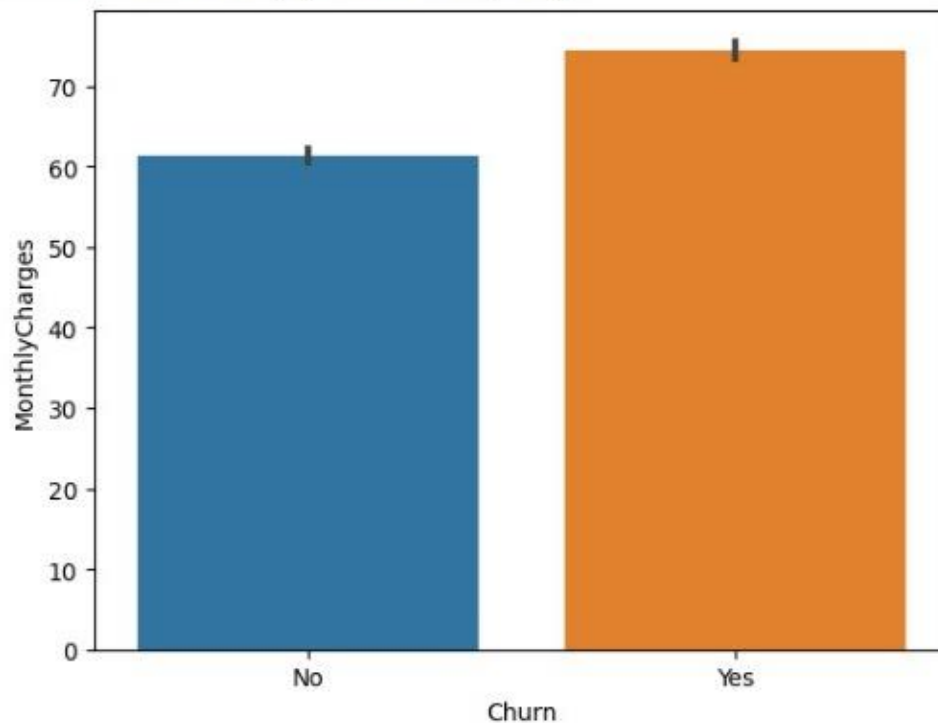


ACTIVITY2.2:Bivariate analysis

+ Code + Text

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

<Axes: xlabel='Churn', ylabel='MonthlyCharges'>



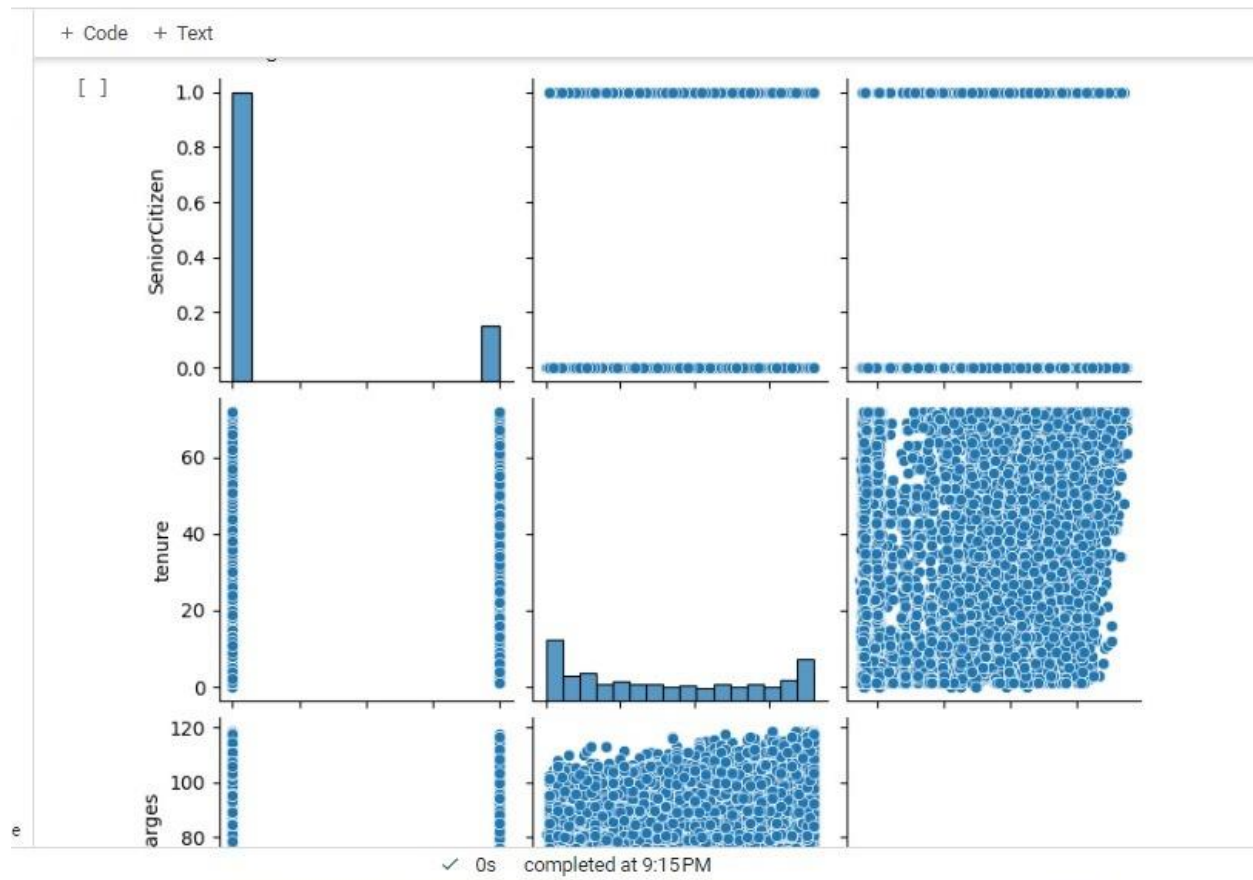
ACTIVITY 2.3: Multivariate analysis

In simple words, multivariate analysis is to find the relation between multiple features. Here we have used a swarm plot from the seaborn package

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

<Axes: >





Split&scan the data

Now let's split the Dataset into train and test sets.

Scaling is one the important process, we have to perform on the dataset, because of data measures in different ranges can leads to mislead in prediction

Models such as KNN, Logistic regression need scaled data, as they follow distance based method and Gradient Descent concept.

```
from sklearn.preprocessing import StandardScaler
[ ] 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

(8278, 40)
```


MILESTONE4: Model Building

ACTIVITY1: Training the model in multiple algorithms Now our data is cleaned and it's time to build the model. We can train our data on different algorithms. For this project we are applying four classification algorithms. The best model is saved based on its performance.

ACTIVITY1.2: Logistic Regression Model

Logistic regression estimates the probability of an event occurring, such as voted or didn't vote, based on a given dataset of independent variables. Since the outcome is a probability, the dependent variable is bounded between 0 and 1.

```
[ ] 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("***LogisticRegression***")  
    print("Confusion_Matrix")  
    print(confusion_matrix(y_test,yPred_lr))  
    print("Classification Report")  
    print(classification_report(y_test,yPred_lr))
```

```
[ ] logreg(x_train,x_test,y_train,y_test)  
  
0.7702343561246678  
0.7797101449275362  
***LogisticRegression***  
Confusion_Matrix  
[[760 273]  
 [183 854]]  
Classification Report  
              precision    recall  f1-score   support  
  
     0       0.81       0.74       0.77       1033  
     1       0.76       0.82       0.79       1037  
  
    accuracy          0.78          0.78          0.78       2070  
   macro avg          0.78          0.78          0.78       2070  
  weighted avg          0.78          0.78          0.78       2070
```

ACTIVITY1.2: Decision tree model

A function named `decisionTree` is created and train and test data are passed as the parameters. Inside the function, `DecisionTreeClassifier` algorithm is initialised and training data is passed to the model with the `.fit()` function.

```
[ ] 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("Confusio_ Matrix")
    print(confusion_matrix(y_test,yPred_dt))
    print("classification Report")
```

```
[ ] decisionTree(x_train,x_test,y_train,y_test)

0.9981879681082387
0.6202898550724638
***Decision Tree***
Confusio_ Matrix
[[287 746]
 [ 40 997]]
classification Report
```

	precision	recall	f1-score	support
0	0.88	0.28	0.42	1033
1	0.57	0.96	0.72	1037
accuracy			0.62	2070
macro avg	0.72	0.62	0.57	2070
weighted avg	0.72	0.62	0.57	2070

ACTIVITY 1.3: Random forest model

A function named `randomForest` is created and train and test data are passed as the parameters. Inside the function, `RandomForestClassifier` algorithm is initialised and training data is passed to the model with `.fit()` function.

```
[ ] def RandomForest(x_train,x_test,y_train,y_test):
    rf = RandomForestClassifier(criterion="entropy", 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("Confusio_ Matrix")
    print(confusion_matrix(y_test,yPred_rf))
    print("classification Report")
```

```
[ ] RandomForest(x_train,x_test,y_train,y_test)
```

```
0.9981879681082387
0.7386473429951691
***Random forest***
Confusio_ Matrix
[[533 500]
 [ 41 996]]
classification Report
```

	precision	recall	f1-score	support
0	0.93	0.52	0.66	1033
1	0.67	0.96	0.79	1037
accuracy			0.74	2070
macro avg	0.80	0.74	0.72	2070
weighted avg	0.80	0.74	0.73	2070

✓ 0s completed at 9:15PM

ACTIVITY1.3: KNN model

A function named KNN is created and train and test data are passed as the parameters. Inside the function, KNeighborsClassifier algorithm is initialised and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in new variable. For evaluating the model, confusion matrix and classification report is done.

```
[ ] from sklearn.neighbors import KNeighborsClassifier
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")
```

✓ 0s completed at 9:15 PM

```
[ ] KNN(x_train,x_test,y_train,y_test)

0.8514133848755738
0.7932367149758454
***KNN***
Confusion Matrix
[[730 303]
 [125 912]]
classification Report
              precision    recall  f1-score   support

     0       0.85         0.71         0.77        1033
     1       0.75         0.88         0.81        1037

 accuracy          0.79         2070
```

✓ 0s completed at 9:15 PM

ACTIVITY 1.4: SVM model

“Support Vector Machine” (SVM) is a supervised machine learning algorithm that can be used for both classification or regression challenges. However, it is mostly used in classification problems.

```
[ ] 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))
```

ACTIVITY1.5: ANN model

Building and training an Artificial Neural Network (ANN) using the Keras library with TensorFlow as the backend. The ANN is initialised as an instance of the Sequential class, which is a linear stack of layers.

```
[ ] 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'])
```

```
X + Code + Text HAM Disk
[ ] Epoch 1/200
    555/555 [=====] - 2s 4ms/step - loss: 0.1514 - accuracy: 0.9353 - val_loss: 0.7712 - val_accuracy: 0.8060
    Epoch 2/200
    555/555 [=====] - 2s 3ms/step - loss: 0.1498 - accuracy: 0.9389 - val_loss: 0.7392 - val_accuracy: 0.8067
    Epoch 3/200
    555/555 [=====] - 2s 3ms/step - loss: 0.1550 - accuracy: 0.9364 - val_loss: 0.7460 - val_accuracy: 0.8093
    Epoch 4/200
    555/555 [=====] - 2s 3ms/step - loss: 0.1516 - accuracy: 0.9365 - val_loss: 0.7505 - val_accuracy: 0.8064
    Epoch 5/200
    555/555 [=====] - 2s 3ms/step - loss: 0.1471 - accuracy: 0.9354 - val_loss: 0.7635 - val_accuracy: 0.8031
    Epoch 6/200
    555/555 [=====] - 1s 2ms/step - loss: 0.1468 - accuracy: 0.9378 - val_loss: 0.7742 - val_accuracy: 0.8133
    Epoch 7/200
    555/555 [=====] - 2s 3ms/step - loss: 0.1453 - accuracy: 0.9354 - val_loss: 0.7900 - val_accuracy: 0.8045
    Epoch 8/200
    555/555 [=====] - 2s 4ms/step - loss: 0.1484 - accuracy: 0.9374 - val_loss: 0.7968 - val_accuracy: 0.8137
    Epoch 9/200
    555/555 [=====] - 2s 3ms/step - loss: 0.1475 - accuracy: 0.9344 - val_loss: 0.7970 - val_accuracy: 0.7939
    Epoch 10/200
```

```

555/555 [=====] - 2s 3ms/step - loss: 0.1427 - accuracy: 0.9400 - val_loss: 0.7740 - val_accuracy: 0.8047
Epoch 23/200
555/555 [=====] - 2s 3ms/step - loss: 0.1493 - accuracy: 0.9333 - val_loss: 0.7946 - val_accuracy: 0.8060
Epoch 24/200
555/555 [=====] - 1s 2ms/step - loss: 0.1457 - accuracy: 0.9374 - val_loss: 0.7864 - val_accuracy: 0.8031
Epoch 25/200
555/555 [=====] - 1s 2ms/step - loss: 0.1424 - accuracy: 0.9401 - val_loss: 0.8043 - val_accuracy: 0.8130
Epoch 26/200
555/555 [=====] - 2s 3ms/step - loss: 0.1449 - accuracy: 0.9401 - val_loss: 0.7968 - val_accuracy: 0.8130
Epoch 27/200
555/555 [=====] - 2s 3ms/step - loss: 0.1426 - accuracy: 0.9398 - val_loss: 0.8017 - val_accuracy: 0.8115
Epoch 28/200
555/555 [=====] - 1s 2ms/step - loss: 0.1410 - accuracy: 0.9428 - val_loss: 0.8093 - val_accuracy: 0.7998
Epoch 29/200
555/555 [=====] - 1s 2ms/step - loss: 0.1419 - accuracy: 0.9403 - val_loss: 0.7992 - val_accuracy: 0.8082

```

```

[ ] ann_pred = Classifier.predict(x_train)
    ann_pred = (ann_pred>0.5)
    ann_pred

```

```

259/259 [=====] - 1s 1ms/step
array([[ True],
       [False],
       [False],
       ...,
       [False],
       [False],
       [False]])

```

ACTIVITY 2: Testing the model

```

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

```

Predicting on random input
Output is: [0]

```

▶ 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([[0,0,1,1,0,0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,456,1,0,3245,4567]]))
    print("Output is:",dtc_pred_own)

```

☞ Predicting on random input
Output is: [1]

```
[ ] 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([[0,0,1,1,0,0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,1,0,0,456,1,0,3245,4567]]))
print("Output is:",rf_pred_own)
```

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

```
[ ] svc = SVC(kernel="linear")
svc.fit(x_train,y_train)
print("Predicting on random input")
svc_pred_own = svc.predict(svc.transform([[0,0,1,1,0,0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,456,1,0,3245,4567]]))
print("Output is:",svc_pred_own)
```

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

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

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

MILE STONE5:Performance testing& Hyperparameter Tuning

ACTIVITY 1: Testing model with multiple evaluation metrics

Multiple evaluation metrics means evaluating the model's performance on a test set using different performance measures. This can provide a more comprehensive understanding of the model's strengths and weaknesses. We are using evaluation metrics for classification tasks including accuracy, precision, recall, support and F1-score.

ACTIVITY1.1: Compare the model

For comparing the above four models, the `compareModel` function is defined.


```
[ ] from sklearn import svm
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)
```

MILESTONE 6: Model Deployment

ACTIVITY 1: Save the best model

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

```
[ ] Classifier.save("telcom_churn.h5")
```

ACTIVITY 2: Integrate with Web Framework

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

This section has the following tasks

- Building HTML Page
- Building server side script
- Run the web application

ACTIVITY2.1: Building Html Pages:

For this project create two HTML files namely

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

and save them in the templates folder.

ACTIVITY2.2: Build Python code:

Import the libraries.

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

Load the saved model. Importing the flask module in the project is mandatory.

An object of Flask class is our WSGI application. Flask constructor takes the name of the current module (`__name__`) as argument.

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

5 1 HTML

Render HTML page:

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

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

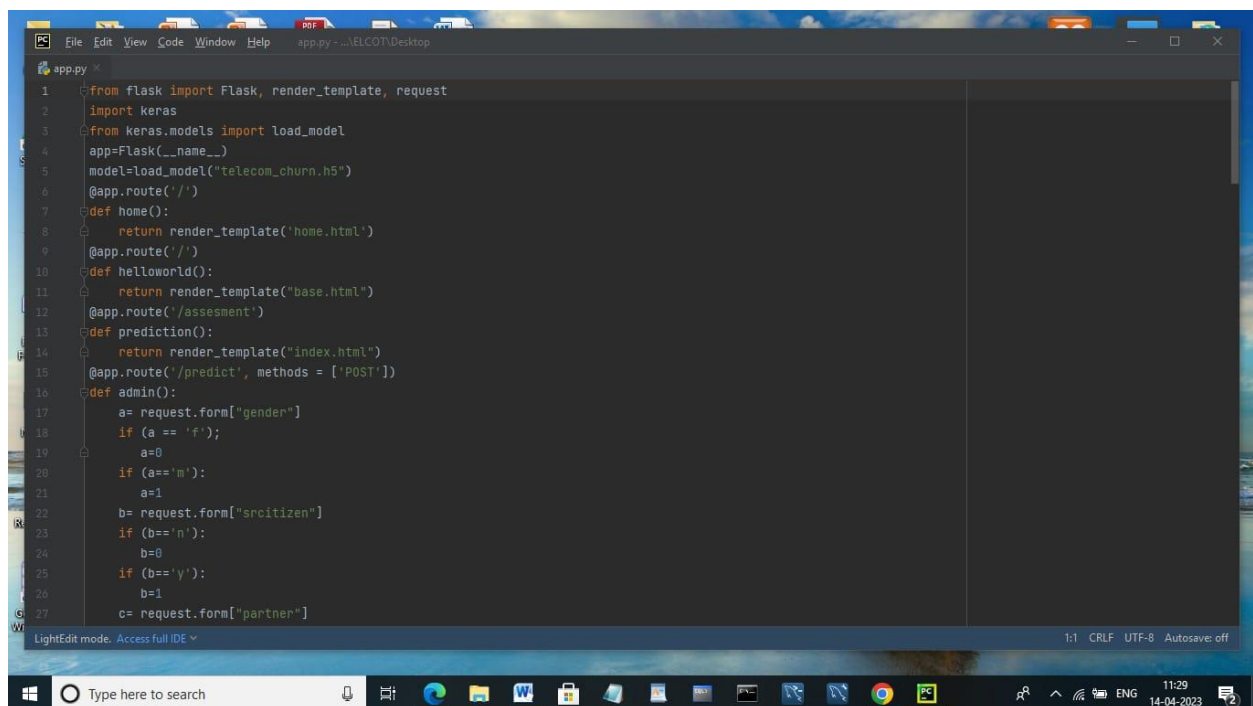
In the above example, '/' URL is bound with the home.html function.

Hence, when the home page of the web server is opened in the browser, the html page will be rendered. Whenever you enter the values from the html page the values can be retrieved using POST Method.

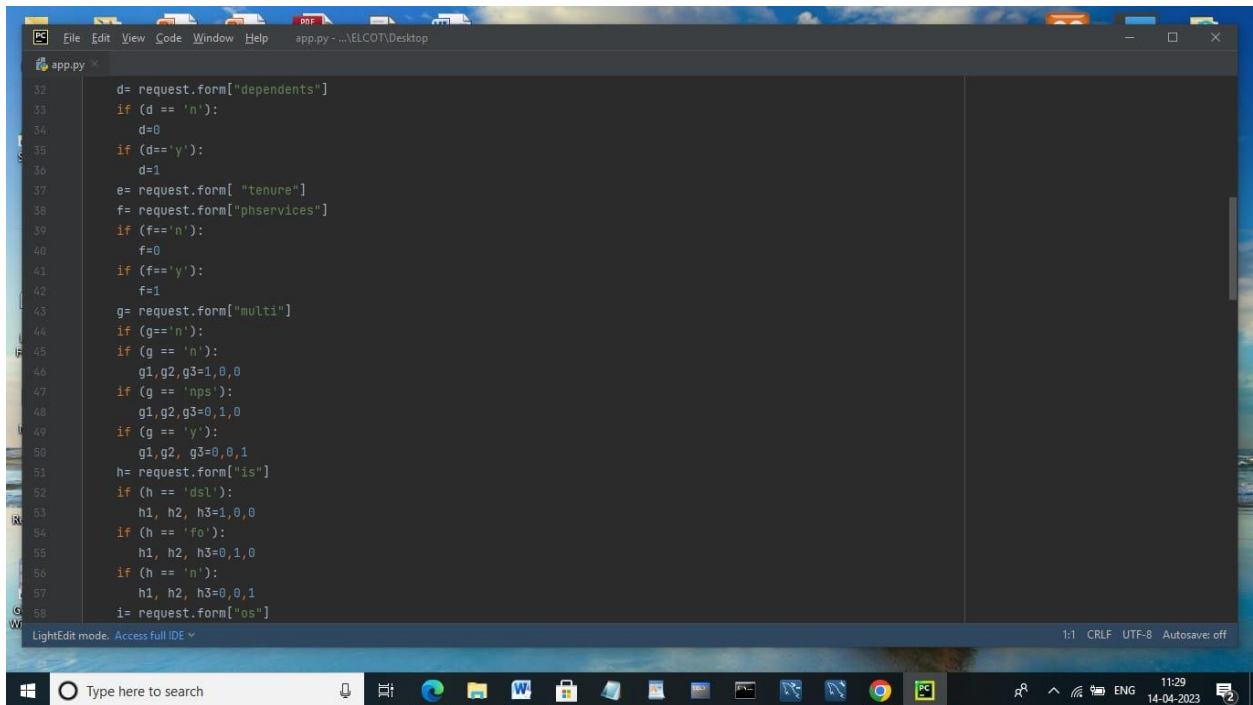
Retrieves the value from UI:

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

Main function:

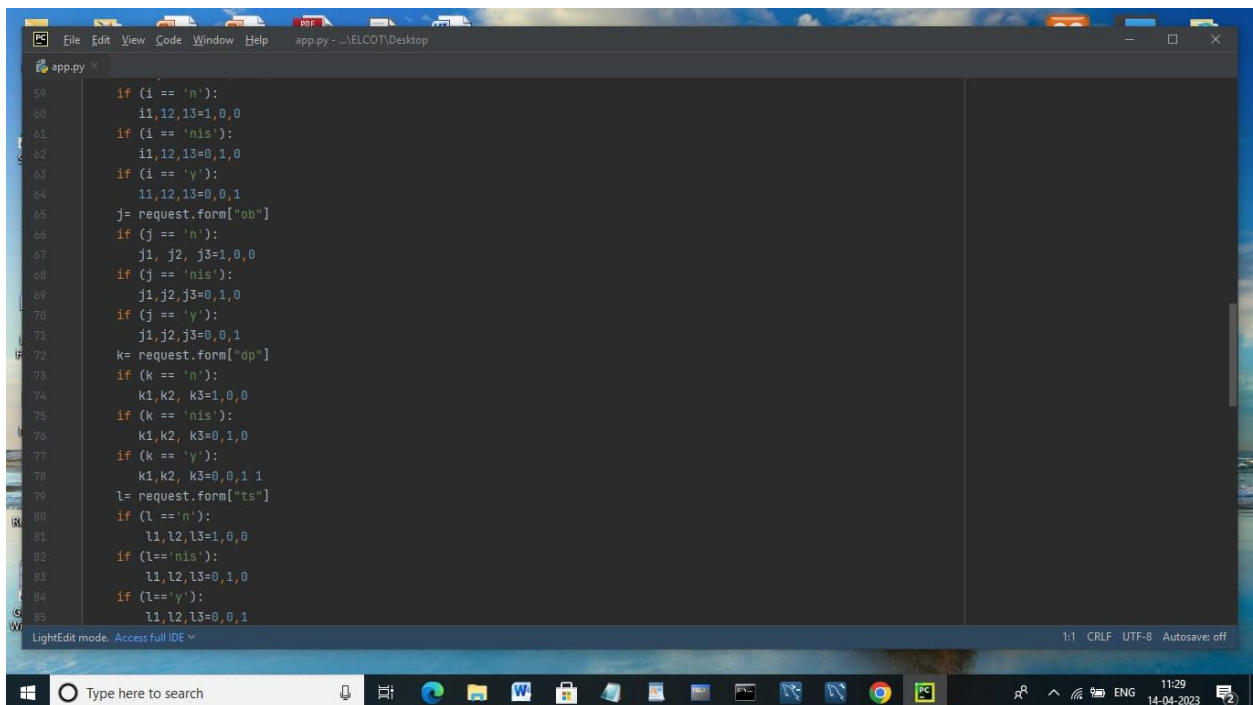
A screenshot of a code editor window titled 'app.py - ...\ALCOT\Desktop'. The code is written in Python and uses the Flask framework. It imports Flask, render_template, request, and keras. A Flask app is created, and a Keras model is loaded from 'telecom_churn.h5'. The code defines several routes: a home route at '/' that renders 'home.html', a helloworld route at '/' that renders 'base.html', an assesment route at '/assesment' that renders 'index.html', and a predict route at '/predict' that handles POST requests. The predict function uses request.form to get values for 'gender', 'srcitizen', and 'partner', and then calls model.predict() with these values. The code is shown in a dark-themed editor with line numbers on the left. The Windows taskbar is visible at the bottom, showing the time as 11:29 on 14-04-2023.

```
1 from flask import Flask, render_template, request
2 import keras
3 from keras.models import load_model
4 app=Flask(__name__)
5 model=load_model("telecom_churn.h5")
6 @app.route('/')
7 def home():
8     return render_template('home.html')
9 @app.route('/')
10 def helloworld():
11     return render_template("base.html")
12 @app.route('/assesment')
13 def prediction():
14     return render_template("index.html")
15 @app.route('/predict', methods = ['POST'])
16 def admin():
17     a= request.form["gender"]
18     if (a == 'f');
19         a=0
20     if (a=='m'):
21         a=1
22     b= request.form["srcitizen"]
23     if (b=='n'):
24         b=0
25     if (b=='y'):
26         b=1
27     c= request.form["partner"]
```



```
32 d= request.form["dependents"]
33 if (d == 'n'):
34     d=0
35     if (d=='y'):
36         d=1
37 e= request.form[ "tenure"]
38 f= request.form["phservices"]
39 if (f=='n'):
40     f=0
41     if (f=='y'):
42         f=1
43 g= request.form["multi"]
44 if (g=='n'):
45     if (g == 'n'):
46         g1,g2,g3=1,0,0
47     if (g == 'nps'):
48         g1,g2,g3=0,1,0
49     if (g == 'y'):
50         g1,g2,g3=0,0,1
51 h= request.form["is"]
52 if (h == 'ds1'):
53     h1,h2,h3=1,0,0
54     if (h == 'fo'):
55         h1,h2,h3=0,1,0
56     if (h == 'n'):
57         h1,h2,h3=0,0,1
58 i= request.form["os"]
```

LightEdit mode. Access full IDE ▾ 1:1 CRLF UTF-8 Autosave: off



```
59 if (i == 'n'):
60     i1,i2,i3=1,0,0
61 if (i == 'nis'):
62     i1,i2,i3=0,1,0
63 if (i == 'y'):
64     i1,i2,i3=0,0,1
65 j= request.form["ob"]
66 if (j == 'n'):
67     j1,j2,j3=1,0,0
68     if (j == 'nis'):
69         j1,j2,j3=0,1,0
70     if (j == 'y'):
71         j1,j2,j3=0,0,1
72 k= request.form["ap"]
73 if (k == 'n'):
74     k1,k2,k3=1,0,0
75     if (k == 'nis'):
76         k1,k2,k3=0,1,0
77     if (k == 'y'):
78         k1,k2,k3=0,0,1
79 l= request.form["ts"]
80 if (l == 'n'):
81     l1,l2,l3=1,0,0
82     if (l=='nis'):
83         l1,l2,l3=0,1,0
84     if (l=='y'):
85         l1,l2,l3=0,0,1
```

LightEdit mode. Access full IDE ▾ 1:1 CRLF UTF-8 Autosave: off

ACTIVITY 2.3: Run the web application

- Open anaconda prompt from the start menu.

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

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

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

TELECOM CUSTOMER CHURN PREDICTION



THE CHURN PREDICTION SAYS YES