Article

Customer Churn Analysis

Problem Definition

In this article, we are going to analyze and predict customer churn prediction using Machine Learning, and exploratory data analysis techniques. with the help of features, we are going to predict the customer churn

About the Dataset

- customerID
- gender
- SeniorCitizen
- Partner
- Dependents
- tenure
- PhoneService
- MultipleLines
- InternetService
- OnlineSecurity
- OnlineBackup
- DeviceProtection
- TechSupport
- StreamingTV
- StreamingMovies
- Contract
- PaperlessBilling
- PaymentMethod
- MonthlyCharges
- TotalCharges
- Churn

1. Data Reading

Importing libraries

```
import pandas as pd
from pandas.plotting import scatter_matrix
import matplotlib.pyplot as plt
import pickle
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
import seaborn as sns
from sklearn.linear_model import LinearRegression
import statsmodels.formula.api as smf
import numpy as np
```

2. Exploratory Data Analysis and Data Cleaning

Using the shape method for checking the size of rows and columns in the dataset

```
data.shape
(7043, 21)
```

We have 7043 rows and 21 column in our dataset

Checking some more information regarding the dataset

```
data.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 21 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   customerID        7043 non-null   object
 1   gender            7043 non-null   object
 2   SeniorCitizen     7043 non-null   int64
 3   Partner           7043 non-null   object
 4   Dependents        7043 non-null   object
 5   tenure            7043 non-null   int64
 6   PhoneService      7043 non-null   object
 7   MultipleLines     7043 non-null   object
 8   InternetService   7043 non-null   object
 9   OnlineSecurity    7043 non-null   object
```

```
 10  OnlineBackup      7043 non-null   object
 11  DeviceProtection  7043 non-null   object
 12  TechSupport       7043 non-null   object
 13  StreamingTV       7043 non-null   object
 14  StreamingMovies   7043 non-null   object
 15  Contract          7043 non-null   object
 16  PaperlessBilling  7043 non-null   object
 17  PaymentMethod     7043 non-null   object
 18  MonthlyCharges    7043 non-null   float64
 19  TotalCharges      7043 non-null   object
 20  Churn             7043 non-null   object
dtypes: float64(1), int64(2), object(18)
memory usage: 1.1+ MB
```

Checking if dataset contain any Null value

data.isnull().sum()

```
customerID          0
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
```

Using describe function to study data

data.describe()

	SeniorCitizen	tenure	MonthlyCharges
count	7043.000000	7043.000000	7043.000000
mean	0.162147	32.371149	64.761692

	SeniorCitizen	tenure	MonthlyCharges
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

Dropping some irrelevant column

data.drop(columns ="customerID" , axis =1 , inplace = True)

Changing column into Numerical form

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

Checking changes

data.info()

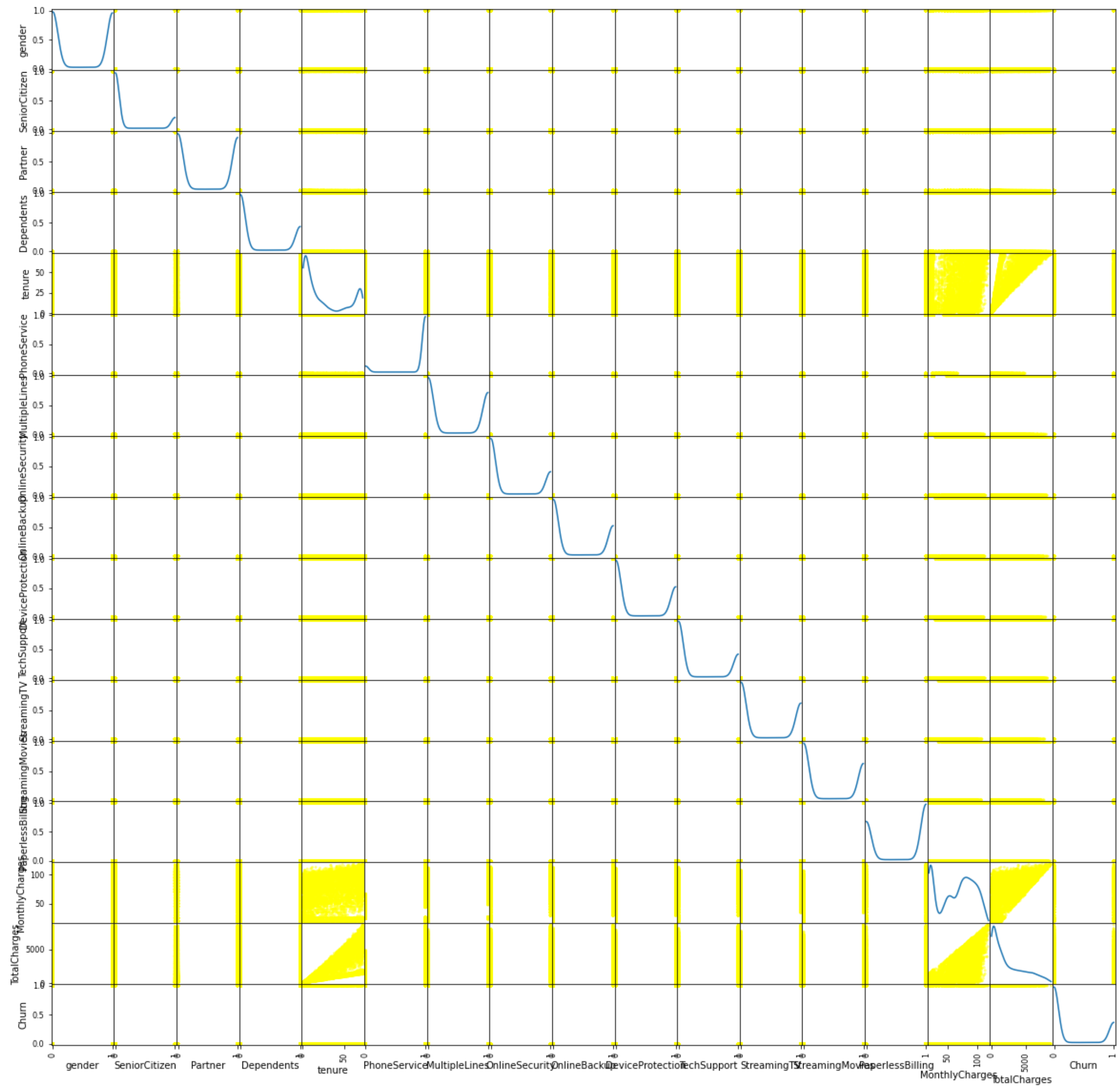
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 20 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   gender            7043 non-null   object
 1   SeniorCitizen     7043 non-null   int64
 2   Partner           7043 non-null   object
 3   Dependents        7043 non-null   object
 4   tenure            7043 non-null   int64
 5   PhoneService      7043 non-null   object
 6   MultipleLines     7043 non-null   object
 7   InternetService   7043 non-null   object
 8   OnlineSecurity    7043 non-null   object
 9   OnlineBackup      7043 non-null   object
 10  DeviceProtection  7043 non-null   object
 11  TechSupport       7043 non-null   object
 12  StreamingTV       7043 non-null   object
 13  StreamingMovies   7043 non-null   object
 14  Contract          7043 non-null   object
 15  PaperlessBilling  7043 non-null   object
 16  PaymentMethod     7043 non-null   object
```

```
 17  MonthlyCharges    7043 non-null   float64
 18  TotalCharges      7032 non-null   float64
 19  Churn             7043 non-null   object
dtypes: float64(2), int64(2), object(16)
memory usage: 1.1+ MB
```

Replacing data with numerical data by using .replace method

```
data.replace("No phone service","No" , inplace =True)
data.replace("No internet service","No" , inplace =True)
data.replace({'Partner':{'No':0,'Yes':1},'Dependents':{'No':0,'Yes':1},'PhoneService':{'No':0,'Yes':1},'MultipleLines':{'No':0,'Yes':1},'OnlineSecurity':{"No":0,"Yes":1},

'OnlineBackup':{'No':0,'Yes':1},'DeviceProtection':{'No':0,'Yes':1},'TechSupport':{'No':0,'Yes':1},'StreamingTV':{'No':0,'Yes':1},

'StreamingMovies':{'No':0,'Yes':1},'PaperlessBilling':{'No':0,'Yes':1},'Churn':{'No':0,'Yes':1},'gender':{'Male':0,'Female':1}},inplace =True)
```

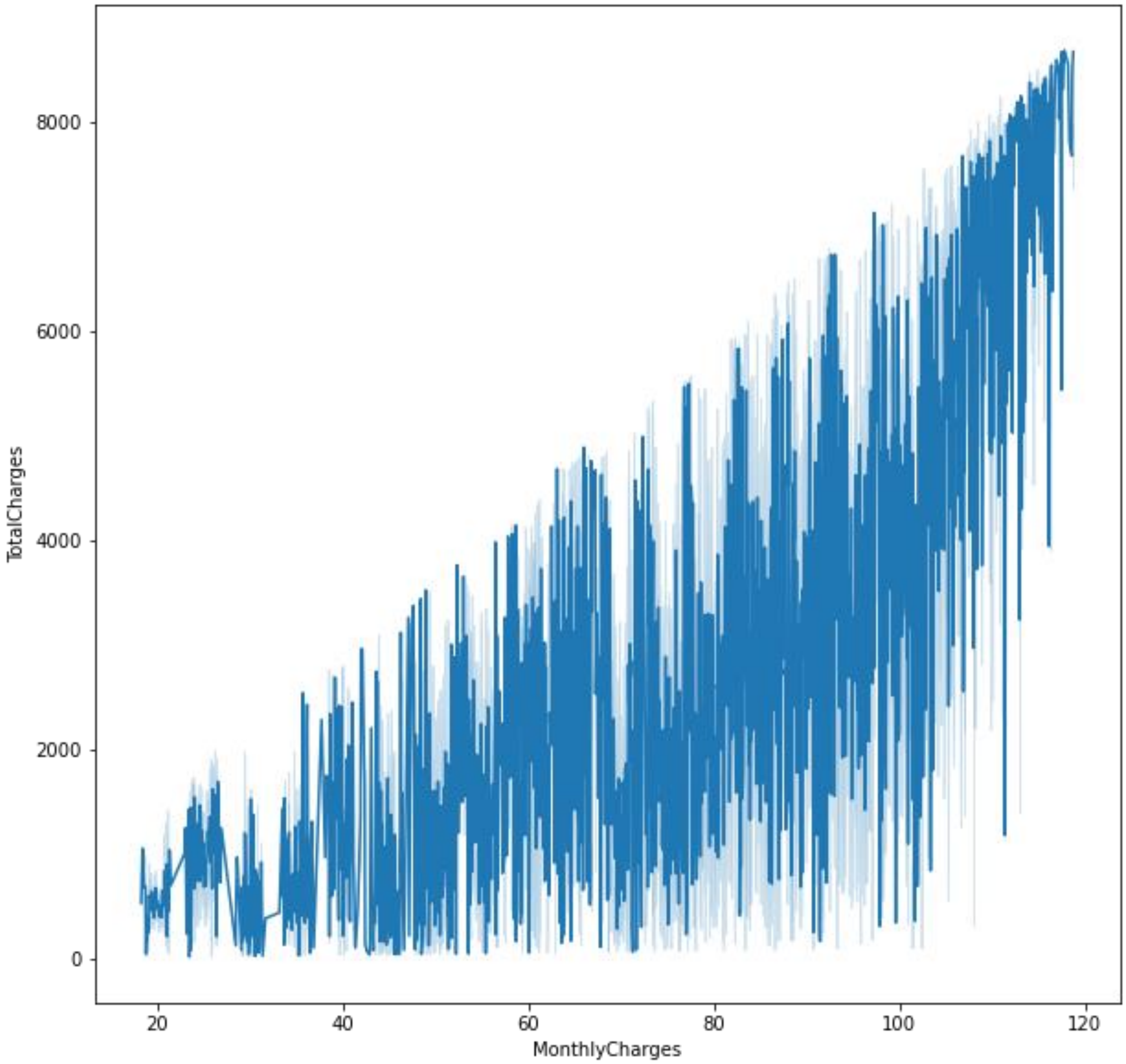
Using scatter matrix for better visualization



Making dummies of some columns and storing all data in a new variable

data1 =pd.get_dummies(data=data, columns =['InternetService','Contract','PaymentMethod'])

Using line plot for understanding the relation between Monthly charges and Total charges



Importing libraries for model building

from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score,confusion_matrix,roc_curve,roc_auc_score
from sklearn.metrics import classification_report
from sklearn.preprocessing import MinMaxScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier

Scaling the data using min max scaler

```
scale_colum =["tenure","MonthlyCharges","TotalCharges",]
scaler = MinMaxScaler()
data1[scale_colum] =scaler.fit_transform(data1[scale_colum])
```

separating the features and target variable

```
x =data1.drop("Churn" ,axis =1)
y =data1["Churn"]
```

Using test train split for splitting the dataset

```
x_train,x_test,y_train,y_test= train_test_split(x,y, test_size=0.2 , random_state =5)
print(x_train,y_train)
```

Using Logistic regression for prediction
Using .fit method to train the model

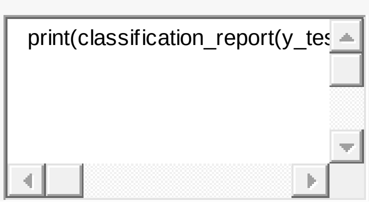
```
lr =LogisticRegression()
lr.fit(x_train,y_train)

y_pred =lr.predict(x_test)
y_pred
array([0, 0, 0, ..., 1, 1, 0], dtype=int64)
```

Checking the accuracy and confusion matrix

```
print("Accuracy =",accuracy_score(y_test,y_pred))
print("Confusion matrix =",confusion_matrix(y_test,y_pred))
Accuracy = 0.7945984363894811
Confusion matrix = [[890 109]
 [180 228]]
```

Printing classification report

```
print(classification_report(y_test,y_pred))
```



```
              precision    recall  f1-score   support

           0       0.83      0.89      0.86       999
           1       0.68      0.56      0.61       408

    accuracy                           0.79      1407
   macro avg       0.75      0.72      0.74      1407
weighted avg       0.79      0.79      0.79      1407
```

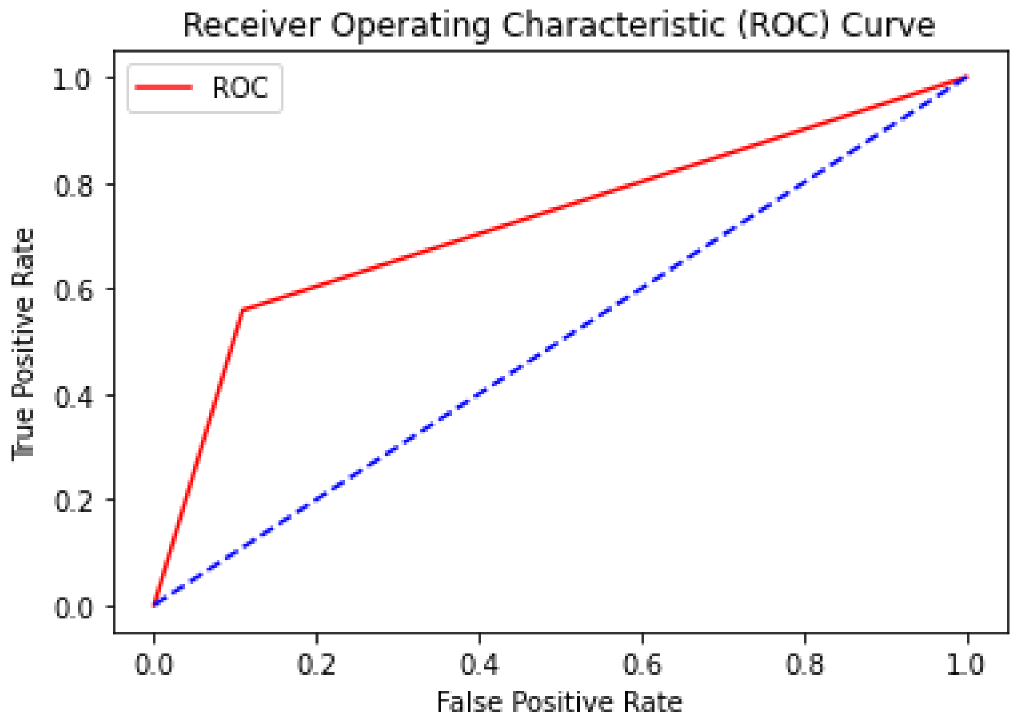
Using ROC curve

```
# ROC Curve
fpr,tpr,threshold =roc_curve(y_test,y_pred)


print("Threshold           :",threshold)
print("True Psitive Rate   :",tpr)
print("False Positive Rate :",fpr)
```

```
Threshold           : [2 1 0]
True Psitive Rate   : [0.         0.55882353 1.        ]
False Positive Rate : [0.         0.10910911 1.        ]
```

Plotting ROC curve

```
plt.plot(fpr,tpr,color="r" , label ="ROC")
plt.plot([0,1],[0,1], color ="b" ,linestyle ="--")
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("Receiver Operating Characteristic (ROC) Curve")
plt.legend()
plt.show()
```



Checking AUC score

AUC curve

auc_score =roc_auc_score(y_test,y_pred)
print(auc_score)
```
0.7248572101513279
```

Fitting data into different model for better results

Using Decision Tree Classifier

dtc =DecisionTreeClassifier()
dtc.fit(x_train,y_train)

y_predi =dtc.predict(x_test)
y_predi
```
array([0, 0, 1, ..., 1, 1, 0], dtype=int64)
```

checking accuracy and confusion matrix

```
print("Accuracy =",accuracy_score(y_test,y_predi))
print("Confusion matrix =",confusion_matrix(y_test,y_predi))
Accuracy = 0.7043354655294953
Confusion matrix = [[784 215]
 [201 207]]
```

```
print(classification_report(y_test,y_predi))

              precision    recall  f1-score   support

           0       0.80      0.78      0.79       999
           1       0.49      0.51      0.50       408

    accuracy                           0.70      1407
   macro avg       0.64      0.65      0.64      1407
weighted avg       0.71      0.70      0.71      1407
```

using Random Forest Classifier

```
rf =RandomForestClassifier()
rf.fit(x_train,y_train)
```

```
y_predict =rf.predict(x_test)
y_predict
array([0, 0, 0, ..., 1, 1, 1], dtype=int64)
```

```
print("Accuracy =",accuracy_score(y_test,y_predict))
print("Confusion matrix =",confusion_matrix(y_test,y_predict))
```

```
Accuracy = 0.7718550106609808
Confusion matrix = [[891 108]
 [213 195]]
```

```
print(classification_report(y_test,y_predict))
              precision    recall  f1-score   support

           0       0.81      0.89      0.85       999
           1       0.64      0.48      0.55       408

    accuracy                           0.77      1407
   macro avg       0.73      0.68      0.70      1407
weighted avg       0.76      0.77      0.76      1407
```

conclusion

In this article, we saw how to apply Different libraries to choose the best machine learning algorithm for the task at hand.
We analyzed the dataset and then find the null values, information regarding the dataset removed all the Null values and then used some methods to clean the data. and build a machine learning model further.
We have tried different machine learning models