Blog/Article

CUSTOMER CHURN ANALYSIS

A project with help of the Flip Robo to understand Telecom customer churn with the aim of building and comparing several customer churn prediction models.

Submitted by - Jay Pandey

Email – jpandey943@gmail.com

Abstract:

In very competitive service industries, especially the telecoms industry, customer churn causes enormous worry. The goal of this study was to create a predictive churn model that could foretell which customers will leave.

The trend of customer attrition can be identified with the aid of churn analysis. Improved customer retention rates show the effect of churn analytics efforts. Through the analytics toolbox, the appropriate churn analysis insights will assist you in comprehending three crucial points: Why do consumers churn? What are the main causes of consumer dissatisfaction?

Problem Definition:

IBM Sample Dataset and Customer Churn analysis:

Customers churn in all telecom firms. When consumers or subscribers stop doing business with a company, this is referred to as customer churn. Customers in the telecom sector have a number of service providers to pick from and can actively switch between them. Since it is relatively expensive to gain new consumers, it presents a challenge for telecommunications businesses that wish to keep their current clientele. A churn label indicating if a client terminated their membership was given to us together with cleaned user activity data (features).

Our analysis may reveal how customer churn is related to other attributes, identify potential causes, and offer suggestions for improving customer retention rates.

The vast amounts of customer data that have been gathered can be leveraged to develop churn prediction algorithms for this issue. By identifying that group, a company can concentrate its marketing efforts on the part of its client base that is most likely to leave. The telecoms business must focus on preventing customer churn because there are little obstacles to switching providers.

We will examine customer data from IBM Sample Data Sets in order to create and assess multiple customer churn prediction models.

Data Analysis:

Data preparation, importing necessary libraries and dataset

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')

from sklearn.model_selection import train_test_split
```

```
##Importing dataset
data = pd.read_csv("https://raw.githubusercontent.com/dsrscientist/DSData/master/Telecom_customer_churn.csv")
```

Overview of dataset

```
data.head()
```

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupport	StreamingTV
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 × 21 columns

```
##Shape
data.shape
```

```
(7043, 21)
```

Dataset has 7043 rows and 21 columns

```
##Checking columns
data.columns
```

```
Index(['customerID', 'gender', 'SeniorCitizen', 'Partner', 'Dependents',
       'tenure', 'PhoneService', 'MultipleLines', 'InternetService',
       'OnlineSecurity', 'OnlineBackup', 'DeviceProtection', 'TechSupport',
       'StreamingTV', 'StreamingMovies', 'Contract', 'PaperlessBilling',
       'PaymentMethod', 'MonthlyCharges', 'TotalCharges', 'Churn'],
      dtype='object')
```

```python
data.info()
```

```
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
```

Observations:

There is one float value among the 21 columns, 18 object types, and two int datatypes. Churn is the only target variable. Each entry in this case uses a distinct value for customerID. Drop this column later. The category variable "SeniorCitizen" has two possible values: 0 and 1. Let's make it an object datatype.

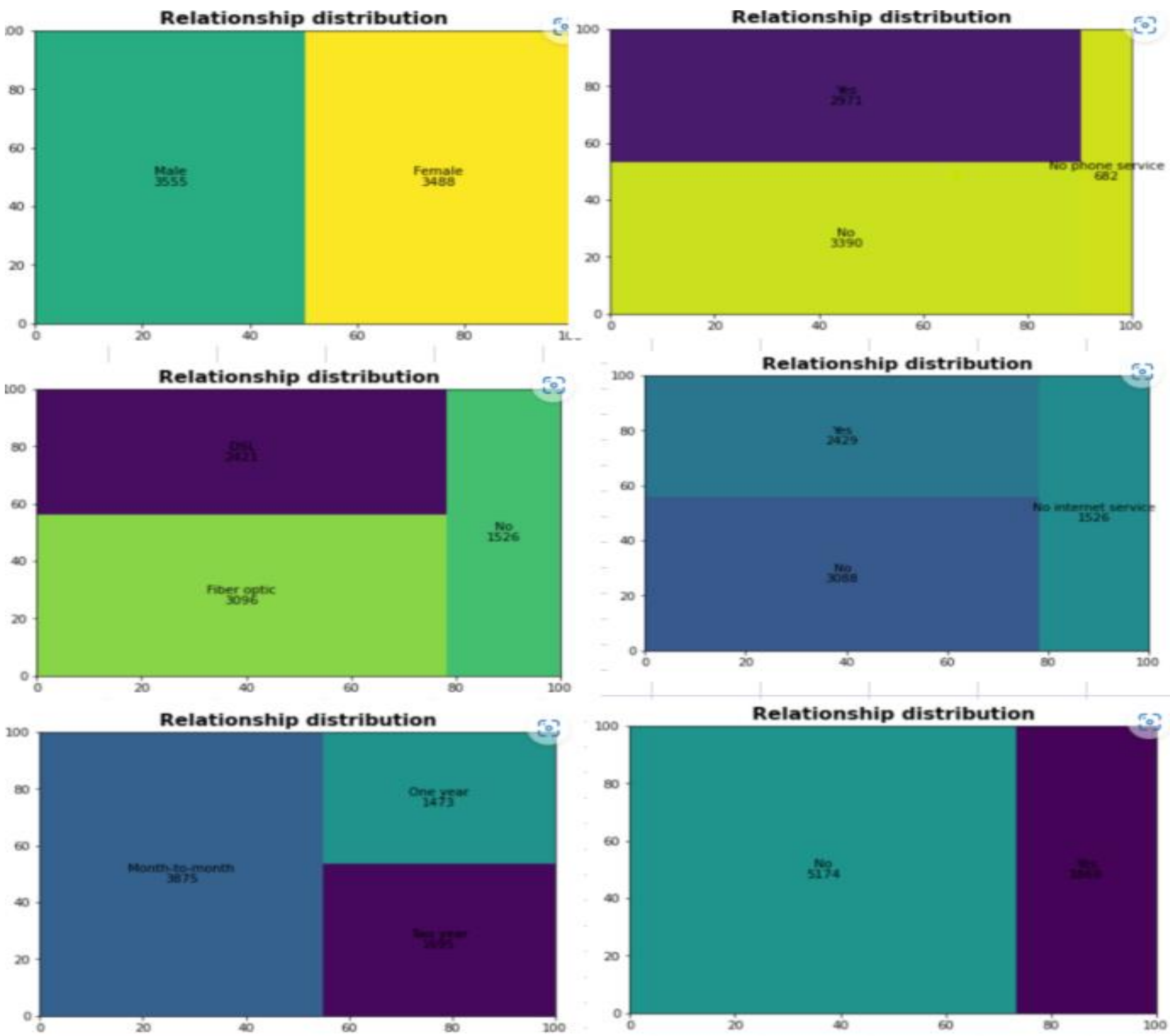
```python
# Converting datatype into float

data['TotalCharges']= data['TotalCharges'].astype(float)
```

Exploratory Data Analysis

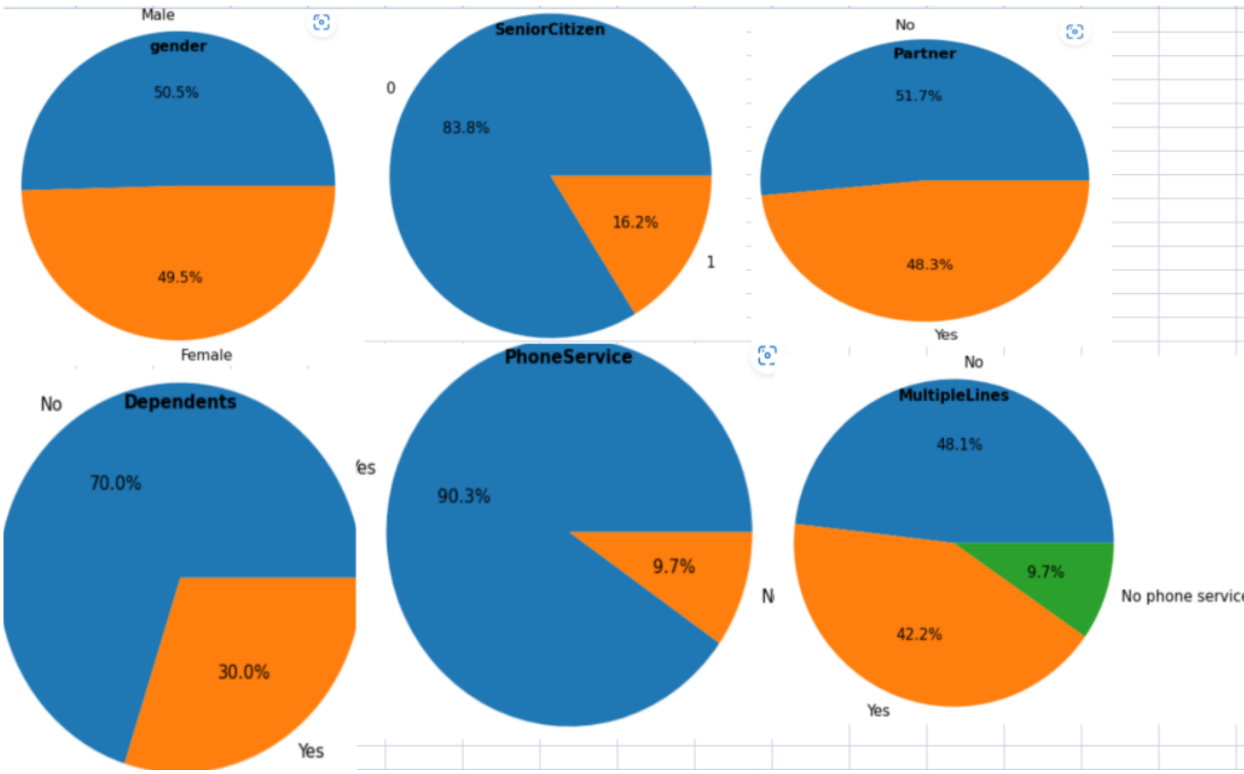
In statistics, exploratory data analysis is a method of examining data sets to highlight their key features, frequently utilizing statistical graphics and other techniques for data visualization.

EDA is a process that uses a variety of procedures or methods to format our dataset in the right way so that we may reach our actual goal. In this EDA, we analyze the complete dataset by utilizing a variety of tools and Python modules.

Relationship Distribution among given data



Categorical Data Analysis



Most used electronic check

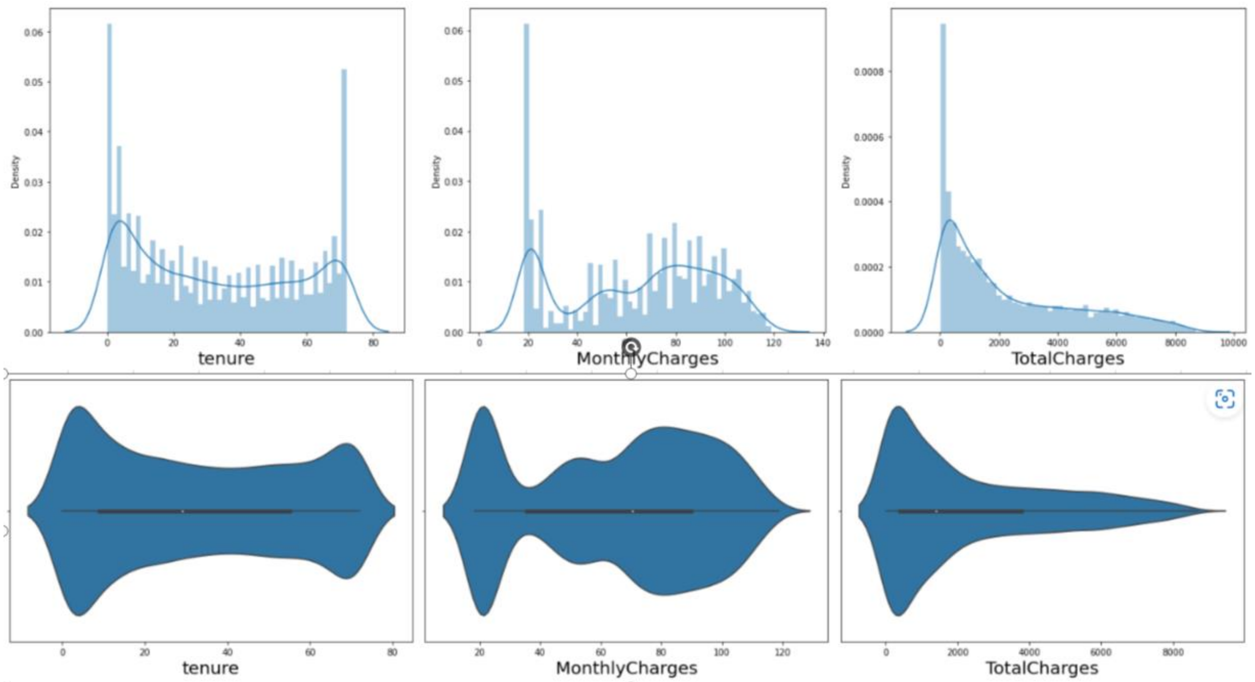
55% customer prefer month to month contract compare to other.

50% customer are having partners

30% customer have dependents on them

16% customer are Senior citizen

Numerical Data Analysis



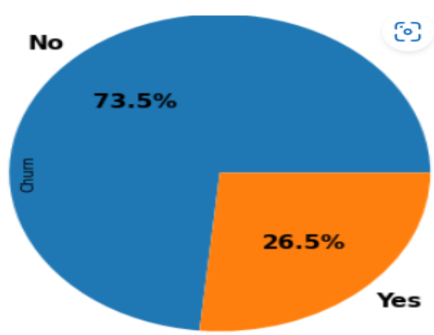
All the data have right skewness

Avg range of age is 0-70

Monthly charges range is 20-120

0 value is present in TotalCharges column

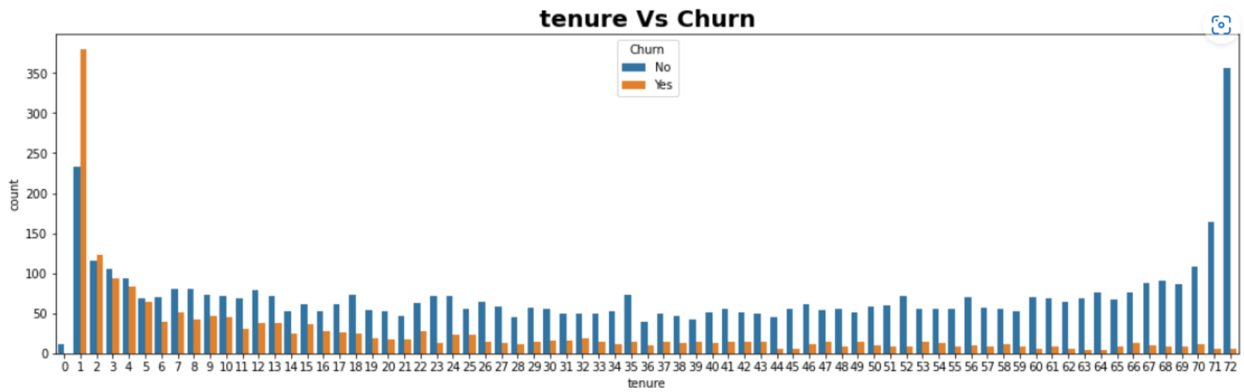
Analysis of Target variable



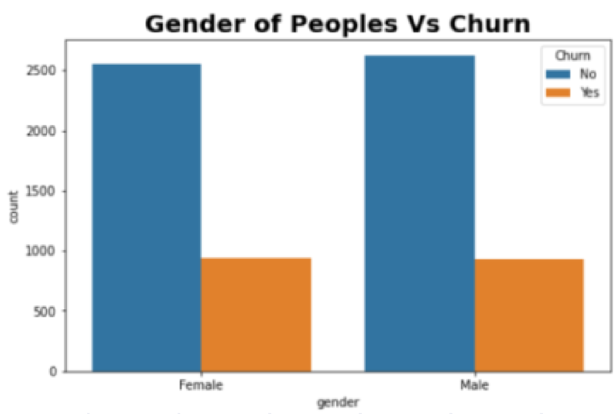
26.5 % customers Churn in last month.

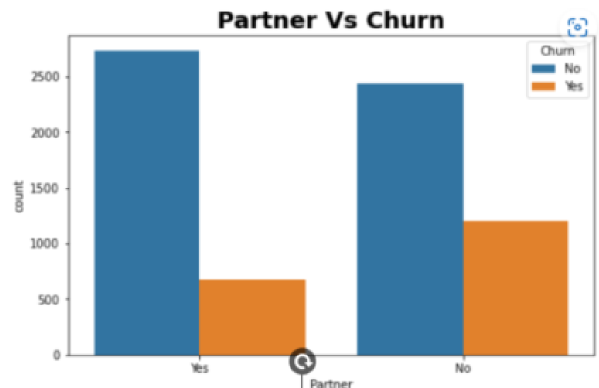
73.5 % customers choose to continue the service in last month.

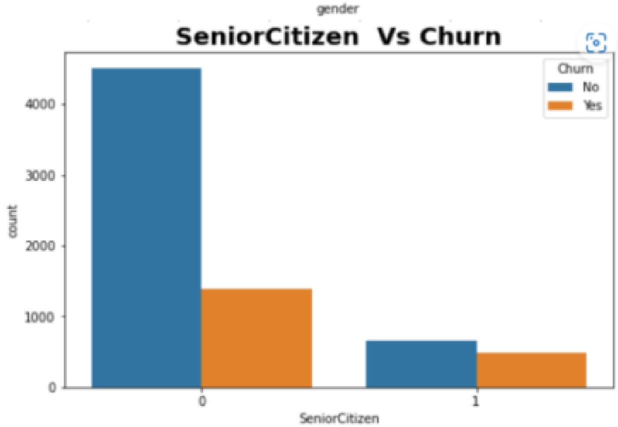
Features on target variable

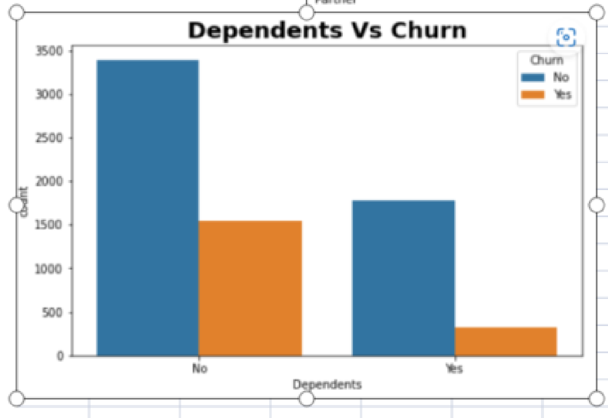


No clear relationship between tenure



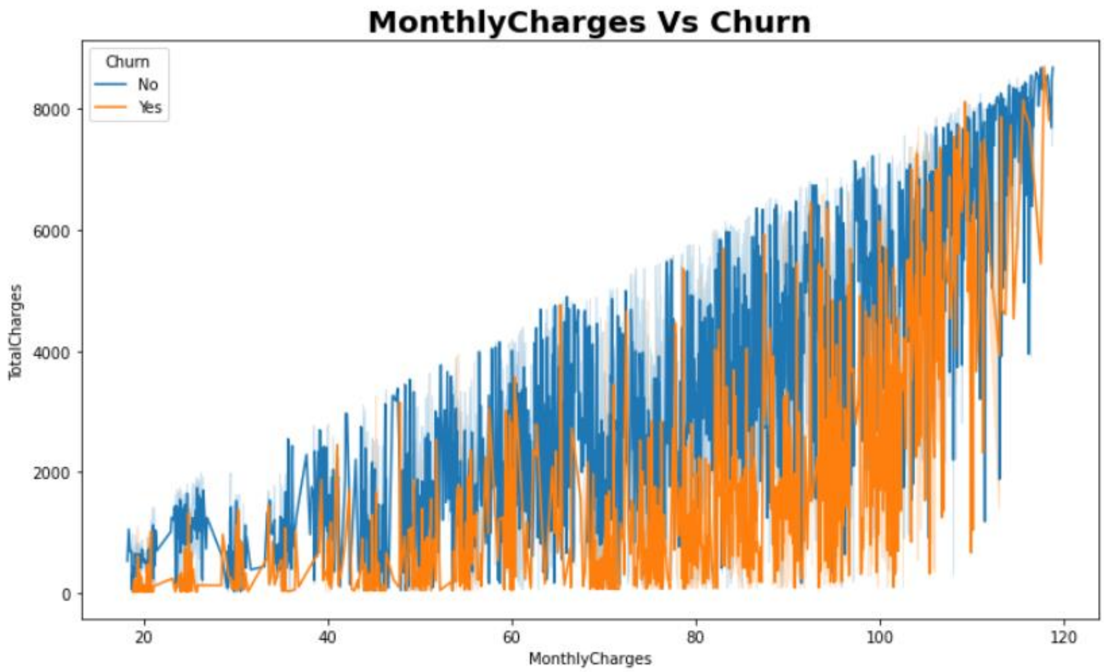






30% customers who doesn't have dependents are tendency to Churn. For all dependent customers around 85 % customers are more tendency to Churn.

Customer having Partner have less tendency to Churn. The customer not having partner have more tendency to Churn with respect to the customer who have their partner.



If MonthlyCharges is high, then the customers are more tendence to choose churn compare to rest. if TotalCharges is high, then the customers are more tendence to choose churn compare to rest.

Pre-Processing Pipeline:

The stage of feature engineering is critical for creating a machine learning model. Initiatives involving machine learning may succeed or fail. What sets them apart? The most important factor is without a doubt the features that are used.

The pipeline guarantees that the preprocessing steps will only be carried out using training data (or training folds in cross-validation). 3. It ensures that your data is consistently preprocessed in the same manner. This is crucial, for instance, if a categorical feature in the test set has a category that does not appear in the training set.

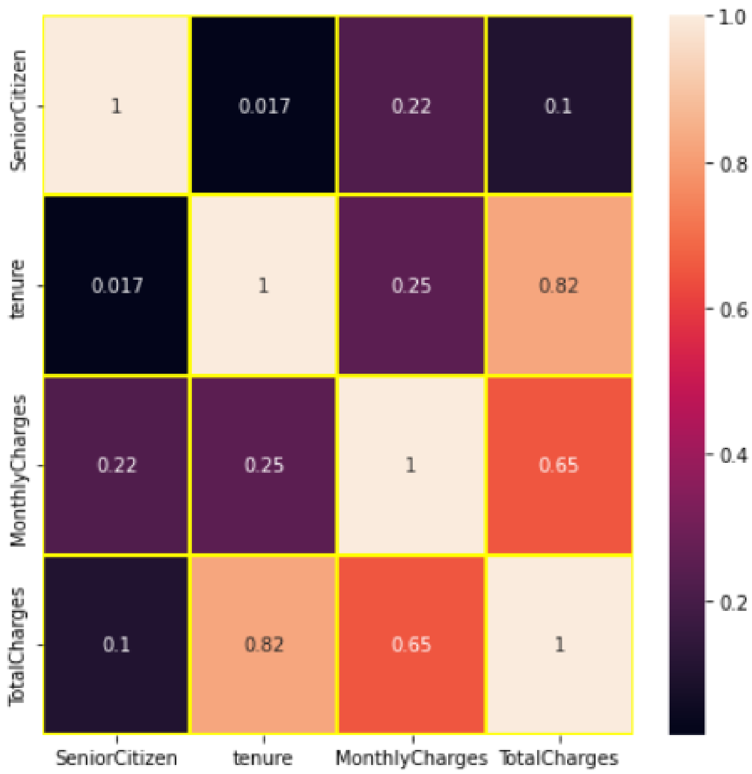
Correlation:

Correlation is high between total charges and tenure. But as we have only 3 numerical features at this time, let's encode the categorical features.

```
data.corr().T
```

	SeniorCitizen	tenure	MonthlyCharges	TotalCharges
SeniorCitizen	1.000000	0.016567	0.220173	0.102395
tenure	0.016567	1.000000	0.247900	0.824757
MonthlyCharges	0.220173	0.247900	1.000000	0.650468
TotalCharges	0.102395	0.824757	0.650468	1.000000

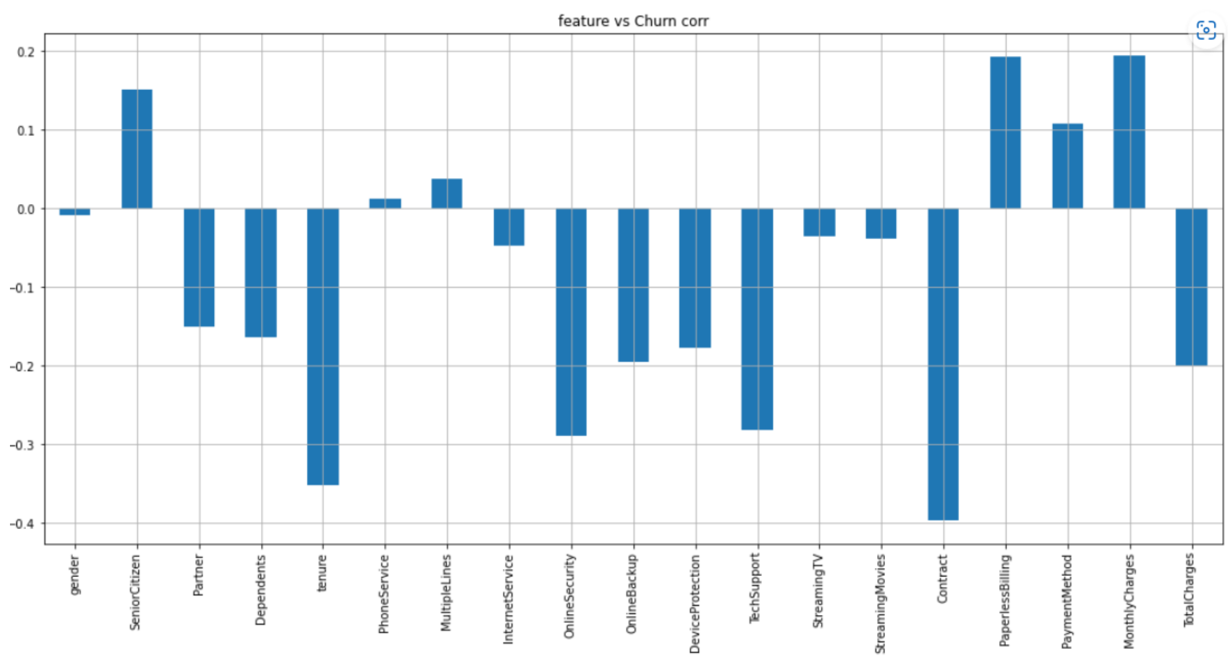
```
plt.subplots(figsize=(7,7))
sns.heatmap(data.corr(), annot= True, linecolor= "yellow",  linewidths= 2)
plt.show()
```



Categorical features Encoding

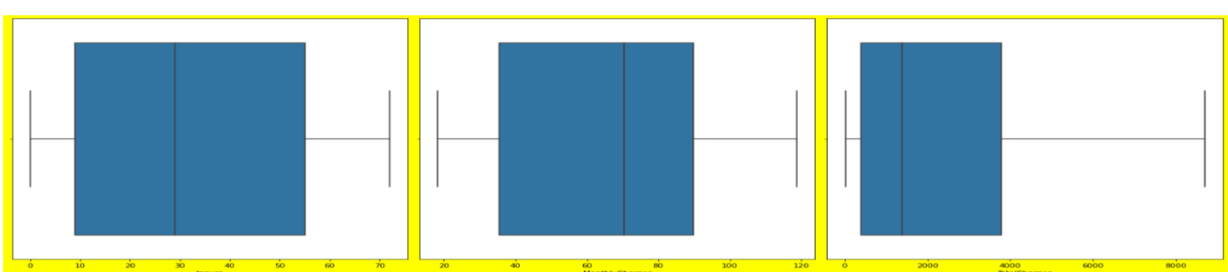
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
for i in cat_features:
    data[i] = le.fit_transform(data[i])
data.head()
```

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



Churn has a highly negative relationship with Contract. In other hand, paperless billing and monthly charges are positively correlated with churn. All the features are correlated with each other

Outliers



Skewness:

```
data[['tenure', 'MonthlyCharges', 'TotalCharges']].skew().sort_values()
```

```
MonthlyCharges   -0.220524
TotalCharges     -0.144899
tenure            0.239540
dtype: float64
```

Data Balancing:

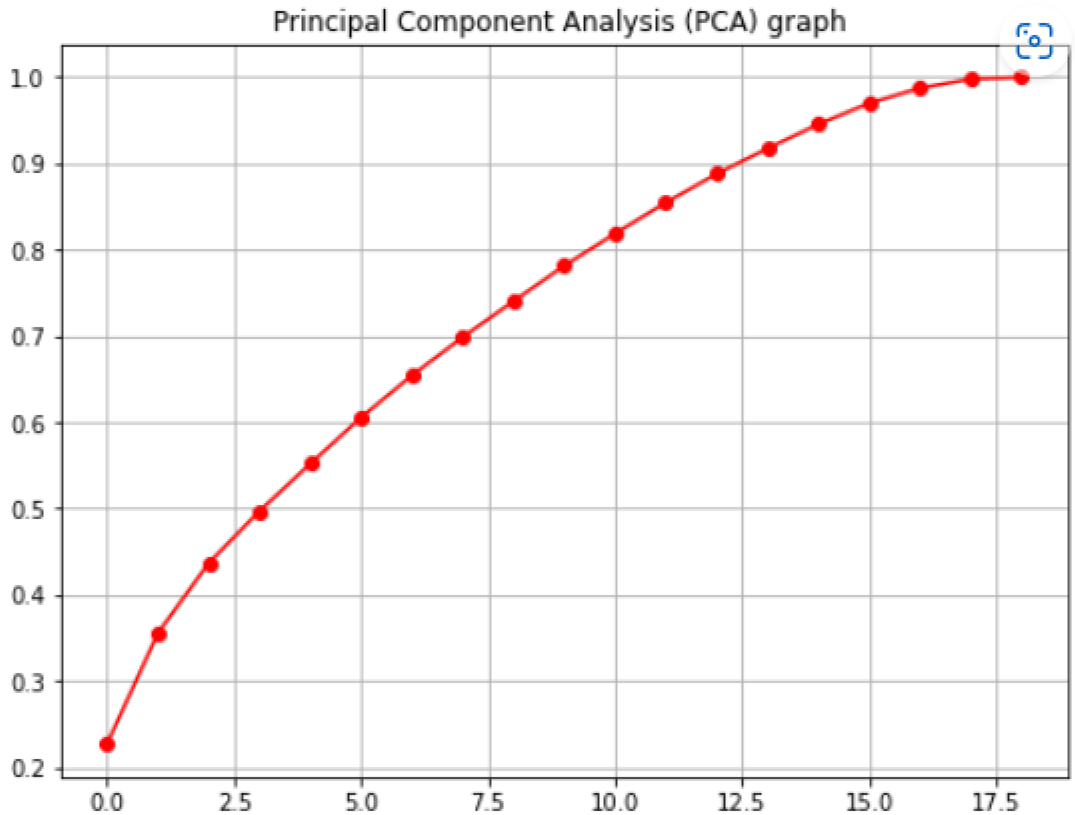
```
from imblearn import under_sampling
from imblearn import over_sampling
from imblearn.over_sampling import SMOTE
```

```
from imblearn.over_sampling import SMOTE
ovrs = SMOTE()

# Splitting data in target and features
x = data.drop(['Churn'], axis =1)
y = data['Churn']
```

Multicollinearity:

```
from statsmodels.stats.outliers_influence import variance_inflation_factor

vif= pd.DataFrame()
vif["VIF"]= [variance_inflation_factor(data.values,i)for i in range(data.shape[1])]
vif["Features"] = data.columns
vif
```

	VIF	Features
0	1.992203	gender
1	1.372640	SeniorCitizen
2	2.821218	Partner
3	1.961200	Dependents
4	13.497891	tenure
5	16.014903	PhoneService
6	2.756853	MultipleLines
7	4.478147	InternetService
8	2.287594	OnlineSecurity
9	2.445350	OnlineBackup
10	2.627903	DeviceProtection
11	2.412647	TechSupport
12	3.235594	StreamingTV
13	3.256673	StreamingMovies
14	4.209950	Contract
15	2.924748	PaperlessBilling
16	3.516126	PaymentMethod
17	18.118004	MonthlyCharges
18	4.806274	TotalCharges
19	1.937378	Churn



95% variance gives the first 14 component.

<u>Building Machine Learning Models:</u>

A mathematical representation of the results of the training process is known as a machine learning model. The study of various algorithms that may develop a model automatically through practice and historical data is known as machine learning. A machine learning model is comparable to software created for computers that can identify patterns or behaviors based on past experience or data. A machine learning (ML) model that captures the patterns found in the training data is produced by the learning algorithm after it analyses the training data for patterns.

Using different Classification ML Models

```
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score, confusion_matrix,classification_report,f1_score

from sklearn.linear_model import LogisticRegression

acc_max=0
random_max=0
for i in range(400, 1500):
    x_train,x_test,y_train,y_test = train_test_split(x_scale_new,y,test_size = 0.25, random_state=i)
    log= LogisticRegression()
    log.fit(x_train,y_train)
    y_pred=log.predict(x_test)
    acc= accuracy_score(y_test,y_pred)
    if acc>acc_max:
        acc_max=acc
        random_max=i

print('Best accuracy is', acc_max ,'on Random_state', random_max)
```

```
Best accuracy is 0.8074990336296869 on Random_state 1168
```

The best accuracy on Random_state=1168

<u>Logistic Regression</u>

```
accu score :  0.8001546192500967
cof_mat:
  [[ 997  295]
 [ 222 1073]]
classification report:
               precision    recall  f1-score   support

           0       0.82      0.77      0.79      1292
           1       0.78      0.83      0.81      1295

    accuracy                           0.80      2587
   macro avg       0.80      0.80      0.80      2587
weighted avg       0.80      0.80      0.80      2587

-----------
-----------
training score :  0.778636773611648
testing score :  0.8001546192500967
```

After Hyper Parameter Tuning

```
training score :  0.7795387192372117
testing score :  0.7974487823734054
```

R2 score not improved after using gridsearchCV

DecisionTreeClassifier

```
classification report:

              precision    recall  f1-score   support

           0       0.78      0.74      0.76      1292
           1       0.75      0.79      0.77      1295

    accuracy                           0.77      2587
   macro avg       0.77      0.77      0.77      2587
weighted avg       0.77      0.77      0.77      2587

-----------
-----------
training score :  0.9985826568741142
testing score :  0.7669114804793197
```

After Hyper Parameter Tuning

```
training score :  0.8062105398788816
testing score :  0.7816003092385002
```

The difference between training score, testing score is also decreased. Accuracy score is slightly improved after using GridSearchCV with DecisionTreeClassifier

GradientBoostingClassifier

```
classification report:

              precision    recall  f1-score   support

           0       0.84      0.78      0.81      1292
           1       0.79      0.85      0.82      1295

    accuracy                           0.81      2587
   macro avg       0.82      0.81      0.81      2587
weighted avg       0.82      0.81      0.81      2587

-----------
-----------
training score :  0.8218013142636258
testing score :  0.8148434480092771
```

After Hyper Parameter Tuning

```
training score :  0.8218013142636258
testing score :  0.8140703517587939
```

No improvement in Accuracy score, training score, testing score after using GridSearchCV with GradientBoostingClassifier

RandomForestClassifier

```
classification report:               precision    recall  f1-score   support

           0       0.82      0.82      0.82      1292
           1       0.82      0.82      0.82      1295

    accuracy                           0.82      2587
   macro avg       0.82      0.82      0.82      2587
weighted avg       0.82      0.82      0.82      2587

-----------
-----------
training score :  0.9985826568741142
testing score :  0.8233475067645922
```

After Hyper Parameter Tuning

```
training score :  0.960185543100116
testing score :  0.8237340548898338
```

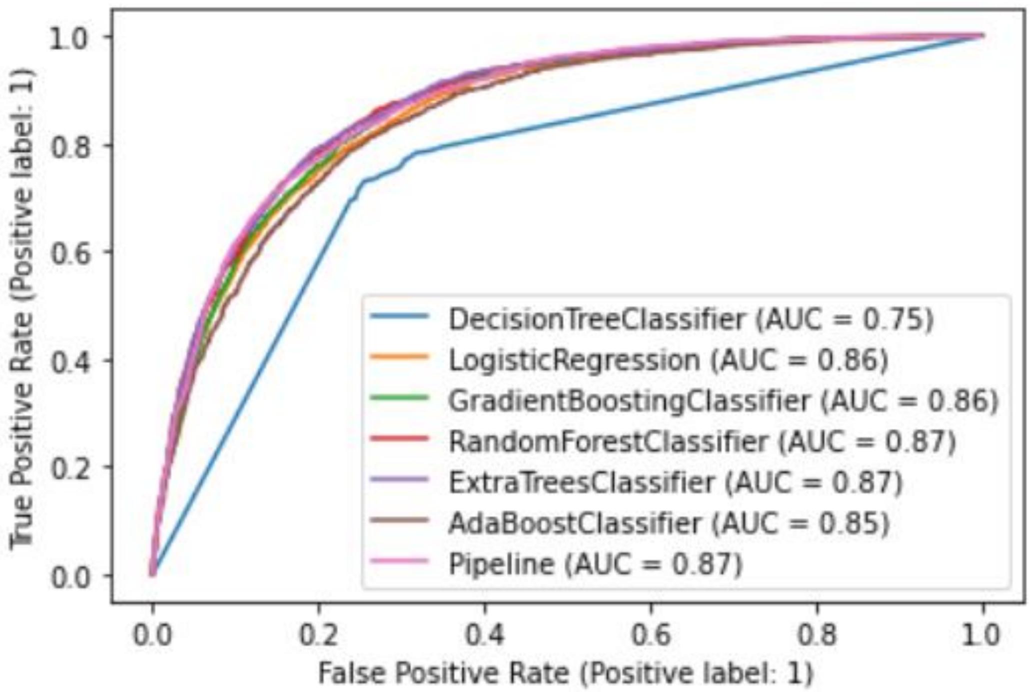
Accuracy score is improved after using GridSearchCV with RandomForestClassifier

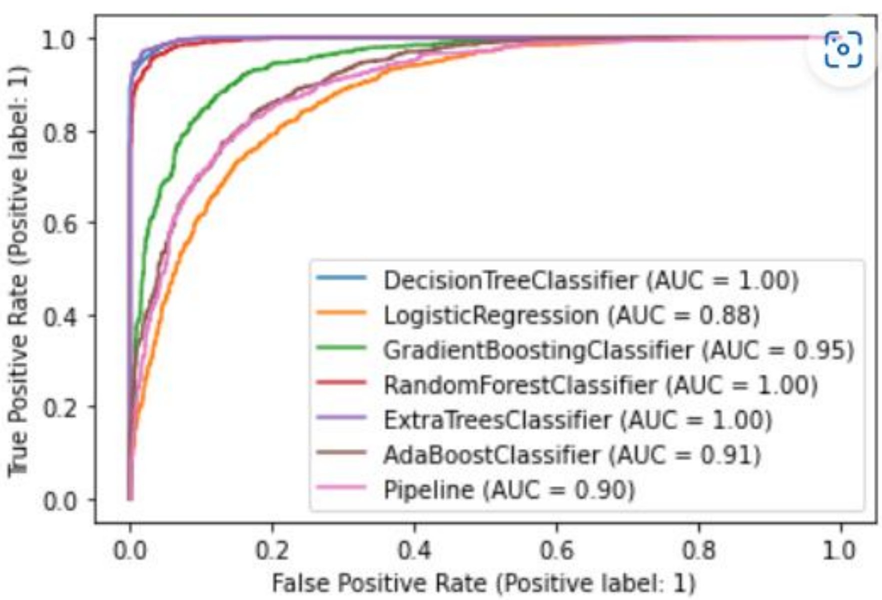
Random Forest Classifier gives the best score.

Cross Validation

```python
from sklearn.model_selection import cross_val_score
svm_best = Pipeline([('PCA', PCA()),
                    ('SVM', SVC(C=7, gamma=0.01, kernel = "rbf"))])

all_models = [log, grid_clf_best, grid_gbdt_best, grid_rf_best, grid_etc_best, grid_ada_best, svm_best ]

for i in all_models:
    cvscore = cross_val_score(i, x_scale,y, cv =7)
    print('Cross Validation Score of :',i)
    print("\n Cross Validation Score : " ,cvscore)
    print("\nMean CV Score :",cvscore.mean())
    print("\nStd deviation :",cvscore.std())
    print("\n-----------")
    print("-----------")
```

ROC AUC Curve

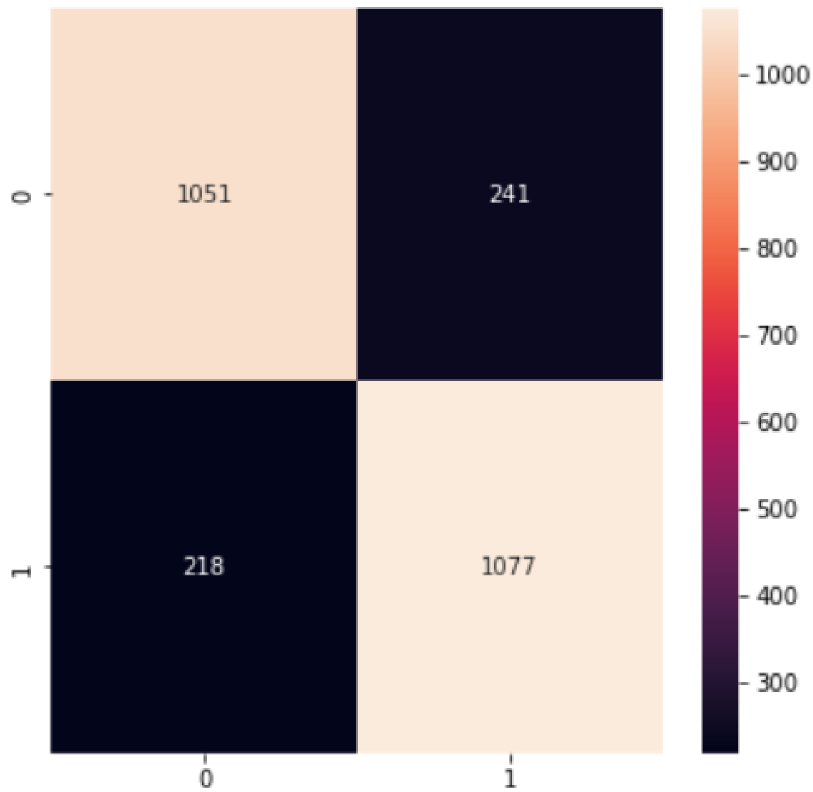




Randomforest is the final model for this dataset

Confusion Matrix:

```
conf = confusion_matrix (y_test, y_pred)

fig , ax = plt.subplots(figsize=(6,6))
sns.heatmap(conf, annot = True, fmt = ".0f")
plt.show()
```



We will save the final model using the Pickle library.

```
import pickle
pickle.dump(grid_etc_best, open("Customer_Churn_Classification_model", "wb"))
load_Customer_Churn_Classification_model= pickle.load(open("Customer_Churn_Classification_model", "rb"))
```

Concluding Remarks:

High churn rates and significant churning loss have harmed the telecommunications sector. To keep the telecommunications business from facing difficulties, effective procedures must be devised and current ones improved. In this post, we explored numerous prediction models and compared their various quality indicators. We discovered that the accuracy achieved using SVM Classifier is significantly higher than the accuracy achieved with logistic regression, proving the effectiveness of decision trees as a method.

1. The majority of customers who have a propensity to leave have no online security.
2. Customers have a greater inclination to churn if Monthly Charges are larger than
3. Different feature engineering approaches are applied to the data, including data balance, outlier removal, label encoding, feature selection, and PCA.
4. The ideal model for this particular dataset is Random Forest.