In [43]:
```
import pandas as pd
```

In [44]:
```
data=pd.read_csv("/home/placement/Desktop/venkatesh/TelecomCustomerChurn.csv")
```

In [45]:
```
data.describe()
```

Out[45]:
```
       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
```

In [19]:
```
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
```

In [46]:
```
data.isna().sum()
```

Out[46]:
```
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
```

In [47]:
```
data.dtypes
```

Out[47]:
```
customerID           object
gender               object
SeniorCitizen         int64
Partner              object
Dependents           object
tenure                int64
PhoneService         object
MultipleLines        object
InternetService      object
OnlineSecurity       object
OnlineBackup         object
DeviceProtection     object
TechSupport          object
StreamingTV          object
StreamingMovies      object
Contract             object
PaperlessBilling     object
PaymentMethod        object
MonthlyCharges      float64
TotalCharges         object
Churn                object
dtype: object
```

In [22]:
```
data['TotalCharges']=pd.to_numeric(data['TotalCharges'],errors='coerce')
# or_use_this==> data['TotalCharges']=data['TotalCharges'].replace('',np.nan).astype(float).values
```

In [32]:
```
data.isna().sum()
```

Out[32]:
```
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        11
Churn                0
dtype: int64
```

In [23]:

```
data.dtypes
```

Out[23]:

```
customerID           object
gender               object
SeniorCitizen         int64
Partner              object
Dependents           object
tenure                int64
PhoneService         object
MultipleLines        object
InternetService      object
OnlineSecurity       object
OnlineBackup         object
DeviceProtection     object
TechSupport          object
StreamingTV          object
StreamingMovies      object
Contract             object
PaperlessBilling     object
PaymentMethod        object
MonthlyCharges      float64
TotalCharges        float64
Churn                object
dtype: object
```

In [33]:

```
data['TotalCharges']=data['TotalCharges'].fillna(data['TotalCharges'].median())
```

In [ ]:

```
#for backup
#databackup=data.copy()
```

In [35]:

```
X=data.drop(['customerID','Churn'],axis=1)
y=data['Churn']
```

In [36]:

```
#data['SeniorCitizen']=data['SeniorCitizen'].map{0:'No',1:'Yes'}
#data['TotalCharges']=data['TotalCharges'].fillna(data['TotalCharges'].median())
```

In [37]:
```
X=pd.get_dummies(X)
```

In [38]:
```
X.head(10)
```

Out[38]:

	SeniorCitizen	tenure	MonthlyCharges	TotalCharges	gender_Female	gender_Male	Partner_No	Partner_Yes	Dependents_No	Dependents_Yes
0	0	1	29.85	29.85	1	0	0	1	1	0
1	0	34	56.95	1889.50	0	1	1	0	1	0
2	0	2	53.85	108.15	0	1	1	0	1	0
3	0	45	42.30	1840.75	0	1	1	0	1	0
4	0	2	70.70	151.65	1	0	1	0	1	0
5	0	8	99.65	820.50	1	0	1	0	1	0
6	0	22	89.10	1949.40	0	1	1	0	0	1
7	0	10	29.75	301.90	1	0	1	0	1	0
8	0	28	104.80	3046.05	1	0	0	1	1	0
9	0	62	56.15	3487.95	0	1	1	0	0	1

10 rows × 45 columns

```
In [49]: list(X)
```

```
Out[49]: ['SeniorCitizen',
          'tenure',
          'MonthlyCharges',
          'TotalCharges',
          'gender_Female',
          'gender_Male',
          'Partner_No',
          'Partner_Yes',
          'Dependents_No',
          'Dependents_Yes',
          'PhoneService_No',
          'PhoneService_Yes',
          'MultipleLines_No',
          'MultipleLines_No phone service',
          'MultipleLines_Yes',
          'InternetService_DSL',
          'InternetService_Fiber optic',
          'InternetService_No',
          'OnlineSecurity_No',
          'OnlineSecurity_No internet service',
          'OnlineSecurity_Yes',
          'OnlineBackup_No',
          'OnlineBackup_No internet service',
          'OnlineBackup_Yes',
          'DeviceProtection_No',
          'DeviceProtection_No internet service',
          'DeviceProtection_Yes',
          'TechSupport_No',
          'TechSupport_No internet service',
          'TechSupport_Yes',
          'StreamingTV_No',
          'StreamingTV_No internet service',
          'StreamingTV_Yes',
          'StreamingMovies_No',
          'StreamingMovies_No internet service',
          'StreamingMovies_Yes',
          'Contract_Month-to-month',
          'Contract_One year',
          'Contract_Two year',
          'PaperlessBilling_No',
          'PaperlessBilling_Yes',
          'PaymentMethod_Bank transfer (automatic)',
          'PaymentMethod_Credit card (automatic)',
          'PaymentMethod_Electronic check',
          'PaymentMethod_Mailed check']
```

In [39]:

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.33,random_state=42)
```

In [40]:

```
from sklearn.model_selection import GridSearchCV #GridSearchCV is for parameter tuning
from sklearn.ensemble import RandomForestClassifier
cls=RandomForestClassifier()
n_estimators=[25,50,75,100,125,150,175,200] #number of decision trees in the forest, default = 100
criterion=['gini','entropy'] #criteria for choosing nodes default = 'gini'
max_depth=[3,5,10] #maximum number of nodes in a tree default = None (it will go till all possible nodes)
parameters={'n_estimators': n_estimators,'criterion':criterion,'max_depth':max_depth} #this will undergo 8*2
RFC_cls = GridSearchCV(cls, parameters)
RFC_cls.fit(X_train,y_train)
```

Out[40]:

```
GridSearchCV(estimator=RandomForestClassifier(),
             param_grid={'criterion': ['gini', 'entropy'],
                         'max_depth': [3, 5, 10],
                         'n_estimators': [25, 50, 75, 100, 125, 150, 175, 200]})
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [50]:

```
RFC_cls.best_params_
```

Out[50]:

```
{'criterion': 'entropy', 'max_depth': 10, 'n_estimators': 125}
```

In [51]:

```
cls=RandomForestClassifier(n_estimators=125,criterion='entropy',max_depth=10)
```

In [52]:

```
cls.fit(X_train,y_train)
```

Out[52]:

```
RandomForestClassifier(criterion='entropy', max_depth=10, n_estimators=125)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [53]:

```
rfy_pred=cls.predict(X_test)
```

In [54]:

```
rfy_pred
```

Out[54]:

```
array(['Yes', 'No', 'No', ..., 'Yes', 'No', 'No'], dtype=object)
```

In [55]:

```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,rfy_pred)
```

Out[55]:

```
array([[1547,  150],
       [ 299,  329]])
```

In [56]:

```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,rfy_pred)
```

Out[56]:

```
0.8068817204301075
```

In [ ]:

```

```