In [20]:
```
import pandas as pd
import numpy as np
```

In [21]:
```
import warnings
warnings.filterwarnings("ignore")
```

In [22]:
```
data=pd.read_csv("/home/placement/Downloads/TelecomCustomerChurn.csv")
```

In [23]:
```
data['TotalCharges']=pd.to_numeric(data['TotalCharges'],errors='coerce')
```

In [24]:
```
data.describe()
```

Out[24]:

	SeniorCitizen	tenure	MonthlyCharges	TotalCharges
count	7043.000000	7043.000000	7043.000000	7032.000000
mean	0.162147	32.371149	64.761692	2283.300441
std	0.368612	24.559481	30.090047	2266.771362
min	0.000000	0.000000	18.250000	18.800000
25%	0.000000	9.000000	35.500000	401.450000
50%	0.000000	29.000000	70.350000	1397.475000
75%	0.000000	55.000000	89.850000	3794.737500
max	1.000000	72.000000	118.750000	8684.800000

In [25]:
```
data.isna().sum()
```

Out[25]:
```
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 [26]:
```
data1=data.fillna(data.median())
```

In [27]:
```
data1.isna().sum()
```

Out[27]:
```
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 [28]:
```
data1.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   float64
 20  Churn             7043 non-null   object 
dtypes: float64(2), int64(2), object(17)
memory usage: 1.1+ MB
```

In [29]:
```
data.dtypes
```

Out[29]:
```
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 [30]:
```
list(data1)
```

Out[30]:
```
['customerID',
 'gender',
 'SeniorCitizen',
 'Partner',
 'Dependents',
 'tenure',
 'PhoneService',
 'MultipleLines',
 'InternetService',
 'OnlineSecurity',
 'OnlineBackup',
 'DeviceProtection',
 'TechSupport',
 'StreamingTV',
 'StreamingMovies',
 'Contract',
 'PaperlessBilling',
 'PaymentMethod',
 'MonthlyCharges',
 'TotalCharges',
 'Churn']
```

In [31]:
```
data1.shape
```

Out[31]:
```
(7043, 21)
```

In [32]:
```
data2=data1.drop(['customerID','customerID','SeniorCitizen','Partner','Dependents','PhoneService','OnlineSec
```

In [33]:
```
data2
```

Out[33]:

	gender	tenure	MultipleLines	InternetService	TechSupport	Contract	MonthlyCharges	TotalCharges	Churn
0	Female	1	No phone service	DSL	No	Month-to-month	29.85	29.85	No
1	Male	34	No	DSL	No	One year	56.95	1889.50	No
2	Male	2	No	DSL	No	Month-to-month	53.85	108.15	Yes
3	Male	45	No phone service	DSL	Yes	One year	42.30	1840.75	No
4	Female	2	No	Fiber optic	No	Month-to-month	70.70	151.65	Yes
...	...	...	...	...	...	...	...	...	...
7038	Male	24	Yes	DSL	Yes	One year	84.80	1990.50	No
7039	Female	72	Yes	Fiber optic	No	One year	103.20	7362.90	No
7040	Female	11	No phone service	DSL	No	Month-to-month	29.60	346.45	No
7041	Male	4	Yes	Fiber optic	No	Month-to-month	74.40	306.60	Yes
7042	Male	66	No	Fiber optic	Yes	Two year	105.65	6844.50	No

7043 rows × 9 columns

In [34]:
```
data['TotalCharges']=data['TotalCharges'].fillna(data['TotalCharges'].median())
```

In [35]:
```
data.isna().sum()
```

Out[35]:
```
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 [36]:
```
data2['Churn']=data2['Churn'].map({'Yes':1,'No':0})
```

In [37]:
```
data2
```

Out[37]:

	gender	tenure	MultipleLines	InternetService	TechSupport	Contract	MonthlyCharges	TotalCharges	Churn
0	Female	1	No phone service	DSL	No	Month-to-month	29.85	29.85	0
1	Male	34	No	DSL	No	One year	56.95	1889.50	0
2	Male	2	No	DSL	No	Month-to-month	53.85	108.15	1
3	Male	45	No phone service	DSL	Yes	One year	42.30	1840.75	0
4	Female	2	No	Fiber optic	No	Month-to-month	70.70	151.65	1
...	...	...	...	...	...	...	...	...	...
7038	Male	24	Yes	DSL	Yes	One year	84.80	1990.50	0
7039	Female	72	Yes	Fiber optic	No	One year	103.20	7362.90	0
7040	Female	11	No phone service	DSL	No	Month-to-month	29.60	346.45	0
7041	Male	4	Yes	Fiber optic	No	Month-to-month	74.40	306.60	1
7042	Male	66	No	Fiber optic	Yes	Two year	105.65	6844.50	0

7043 rows × 9 columns

In [38]:
```
databackup=data.copy()
```

In [39]:
```
x=data.drop(['customerID','Churn'],axis=1)
y=data['Churn']
```

In [40]:
```
x=pd.get_dummies(x)
```

In [41]:
```
x
```

Out[41]:

	SeniorCitizen	tenure	MonthlyCharges	TotalCharges	gender_Female	gender_Male	Partner_No	Partner_Yes	Dependents_No	Dependents_Y
0	0	1	29.85	29.85	1	0	0	1	1	
1	0	34	56.95	1889.50	0	1	1	0	1	
2	0	2	53.85	108.15	0	1	1	0	1	
3	0	45	42.30	1840.75	0	1	1	0	1	
4	0	2	70.70	151.65	1	0	1	0	1	
...	...	...	...	...	...	...	...	...	...	
7038	0	24	84.80	1990.50	0	1	0	1	0	
7039	0	72	103.20	7362.90	1	0	0	1	0	
7040	0	11	29.60	346.45	1	0	0	1	0	
7041	1	4	74.40	306.60	0	1	0	1	1	
7042	0	66	105.65	6844.50	0	1	1	0	1	

7043 rows × 45 columns

In [42]:
```
data['TotalCharges']=data['TotalCharges'].fillna(data['TotalCharges'].median())
```

In [43]:
```
data.isna().sum()
```

Out[43]:
```
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 [44]:
```
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 [45]:
```
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[45]:
```
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 [46]:
```
RFC_cls.best_params_
```

Out[46]:
```
{'criterion': 'entropy', 'max_depth': 10, 'n_estimators': 50}
```

In [56]:
```
cls=RandomForestClassifier(n_estimators=100,criterion='entropy',max_depth=10)
```

In [57]:
```
cls.fit(x_train,y_train)
```

Out[57]:
```
RandomForestClassifier(criterion='entropy', max_depth=10)
```
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 [58]:
```
rfy_pred=cls.predict(x_test)
rfy_pred
```

Out[58]:
```
array(['Yes', 'No', 'No', ..., 'Yes', 'No', 'No'], dtype=object)
```

In [60]:
```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,rfy_pred)
```

Out[60]:
```
array([[1549,  148],
       [ 301,  327]])
```

In [62]:
```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,rfy_pred)
```

Out[62]:
```
0.8068817204301075
```

In [ ]:
```

```