In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
data=pd.read_csv("/home/placement/Downloads/TelecomCustomerChurn.csv")
```

In [3]:
```
data.describe()
```

Out[3]:

	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 [4]:
```
data.head()
```

Out[4]:

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

5 rows × 21 columns

In [5]:
```
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 [6]:
```
data.shape
```

Out[6]: (7043, 21)

In [7]:
```
list(data)
```

Out[7]:
```
['customerID',
 'gender',
 'SeniorCitizen',
 'Partner',
 'Dependents',
 'tenure',
 'PhoneService',
 'MultipleLines',
 'InternetService',
 'OnlineSecurity',
 'OnlineBackup',
 'DeviceProtection',
 'TechSupport',
 'StreamingTV',
 'StreamingMovies',
 'Contract',
 'PaperlessBilling',
 'PaymentMethod',
 'MonthlyCharges',
 'TotalCharges',
 'Churn']
```

In [8]:
```
data.isna().sum()
```

Out[8]:
```
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 [9]:
```
data['TotalCharges']=pd.to_numeric(data['TotalCharges'],errors='coerce')
```

In [10]:
```
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      7032 non-null   float64
 20  Churn             7043 non-null   object 
dtypes: float64(2), int64(2), object(17)
memory usage: 1.1+ MB
```

In [11]:
```
data['TotalCharges']=data['TotalCharges'].fillna(data['TotalCharges'].median())
```

In [12]:
```
data.isna().sum()
```

Out[12]:
```
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 [13]:
```
data['SeniorCitizen']=data['SeniorCitizen'].map({0:'No',1:'Yes'})
```

In [14]:
```
x=data.drop(['customerID','Churn'],axis=1)
y=data['Churn']
```

In [15]:
```
x=pd.get_dummies(x)
```

In [16]:
```
x.head()
```

Out[16]:

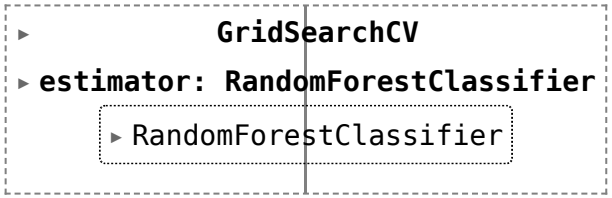
	tenure	MonthlyCharges	TotalCharges	gender_Female	gender_Male	SeniorCitizen_No	SeniorCitizen_Yes	Partner_No	Partner_Yes	Dependents
0	1	29.85	29.85	1	0	1	0	0	1	
1	34	56.95	1889.50	0	1	1	0	1	0	
2	2	53.85	108.15	0	1	1	0	1	0	
3	45	42.30	1840.75	0	1	1	0	1	0	
4	2	70.70	151.65	1	0	1	0	1	0	

5 rows × 46 columns

In [17]:
```
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 [18]:
```
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[18]:



In [19]:
```
data
```

Out[19]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DevicePro
0	7590-VHVEG	Female	No	Yes	No	1	No	No phone service	DSL	No	...	
1	5575-GNVDE	Male	No	No	No	34	Yes	No	DSL	Yes	...	
2	3668-QPYBK	Male	No	No	No	2	Yes	No	DSL	Yes	...	
3	7795-CFOCW	Male	No	No	No	45	No	No phone service	DSL	Yes	...	
4	9237-HQITU	Female	No	No	No	2	Yes	No	Fiber optic	No	...	
...	...	...	...	...	...	...	...	...	...	...	...	
7038	6840-RESVB	Male	No	Yes	Yes	24	Yes	Yes	DSL	Yes	...	
7039	2234-XADUH	Female	No	Yes	Yes	72	Yes	Yes	Fiber optic	No	...	
7040	4801-JZAZL	Female	No	Yes	Yes	11	No	No phone service	DSL	Yes	...	
7041	8361-LTMKD	Male	Yes	Yes	No	4	Yes	Yes	Fiber optic	No	...	
7042	3186-AJIEK	Male	No	No	No	66	Yes	No	Fiber optic	Yes	...	

7043 rows × 21 columns

In [20]:
```
RFC_cls.best_params_
```

Out[20]:
```
{'criterion': 'gini', 'max_depth': 10, 'n_estimators': 150}
```

In [21]:
```
cls=RandomForestClassifier(n_estimators=100,criterion='entropy',max_depth=10)
```

In [22]:
```
cls.fit(x_train,y_train)
```

Out[22]:
```
RandomForestClassifier
RandomForestClassifier(criterion='entropy', max_depth=10)
```

In [23]:
```
rfy_pred=cls.predict(x_test)
```

In [24]:
```
rfy_pred
```

Out[24]:
```
array(['Yes', 'No', 'No', ..., 'Yes', 'No', 'No'], dtype=object)
```

In [25]:
```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,rfy_pred)
```

Out[25]:
```
array([[1549,  148],
       [ 292,  336]])
```

In [26]:
```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,rfy_pred)
```

Out[26]:
```
0.810752688172043
```

In [27]:
```
from sklearn.linear_model import LogisticRegression
classifier= LogisticRegression()
classifier.fit(x_train,y_train)
```

Out[27]:
```
LogisticRegression
LogisticRegression()
```

In [28]:
```
y_pred=classifier.predict(x_test)
```

In [29]:
```
y_pred
```

Out[29]:
```
array(['Yes', 'No', 'No', ..., 'Yes', 'No', 'No'], dtype=object)
```

In [30]:
```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,y_pred)
```

Out[30]:
```
array([[1538,  159],
       [ 279,  349]])
```

In [31]:
```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

Out[31]:
```
0.8116129032258065
```

In [ ]:
```

```