In [53]:
```
import pandas as pd
```

In [54]:
```
data = pd.read_csv("/home/placement/Downloads/TelecomCustomerChurn.csv")
```

In [55]:
```
data.describe()
```

Out[55]:

	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 [56]:

```
data.head()
```

Out[56]:

	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 [58]:

```
data['TotalCharges'] = pd.to_numeric(data['TotalCharges'],errors='coerce')
data.dtypes
```

Out[58]:

```
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 [59]:

```
data.isna().sum()
```

Out[59]:

```
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 [62]:

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

In [63]:

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

```
In [64]: data.tail(20)
```

Out[64]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DevicePro
7023	1035-IPQPU	Female	Yes	Yes	No	63	Yes	Yes	Fiber optic	No	...	
7024	7398-LXGYX	Male	No	Yes	No	44	Yes	Yes	Fiber optic	Yes	...	
7025	2823-LKABH	Female	No	No	No	18	Yes	Yes	Fiber optic	No	...	
7026	8775-CEBBJ	Female	No	No	No	9	Yes	No	DSL	No	...	
7027	0550-DCXLH	Male	No	No	No	13	Yes	No	DSL	No	...	
7028	9281-CEDRU	Female	No	Yes	No	68	Yes	No	DSL	No	...	
7029	2235-DWLJU	Female	Yes	No	No	6	No	No phone service	DSL	No	...	
7030	0871-OPBXW	Female	No	No	No	2	Yes	No	No	No internet service	...	No
7031	3605-JISKB	Male	Yes	Yes	No	55	Yes	Yes	DSL	Yes	...	
7032	6894-LFHLY	Male	Yes	No	No	1	Yes	Yes	Fiber optic	No	...	
7033	9767-FFLEM	Male	No	No	No	38	Yes	No	Fiber optic	No	...	
7034	0639-TSIQW	Female	No	No	No	67	Yes	Yes	Fiber optic	Yes	...	
7035	8456-QDAVC	Male	No	No	No	19	Yes	No	Fiber optic	No	...	
7036	7750-EYXWZ	Female	No	No	No	12	No	No phone service	DSL	No	...	
7037	2569-WGERO	Female	No	No	No	72	Yes	No	No	No internet service	...	No
7038	6840-RESVB	Male	No	Yes	Yes	24	Yes	Yes	DSL	Yes	...	

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DevicePro
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	...	

20 rows × 21 columns

In [65]:

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

In [67]:

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

In [68]:

```
x=pd.get_dummies(x)
```

In [69]:
```
x.head()
```

Out[69]:

	tenure	MonthlyCharges	TotalCharges	gender_Female	gender_Male	SeniorCitizen_No	SeniorCitizen_Yes	Partner_No	Partner_Yes	Dependent
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 [70]:
```
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 [72]:
```
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[72]:
```
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 [73]: RFC_cls.best_params_

Out[73]: {'criterion': 'gini', 'max_depth': 10, 'n_estimators': 150}

In [74]: cls=RandomForestClassifier(n_estimators=150,criterion='entropy',max_depth=10)

In [75]: cls.fit(x_train,y_train)

Out[75]: RandomForestClassifier(criterion='entropy', max_depth=10, n_estimators=150)
```

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 [78]: rfy_pred=cls.predict(x_test)

In [79]: rfy_pred

Out[79]: array(['Yes', 'No', 'No', ..., 'Yes', 'No', 'No'], dtype=object)

In [80]: from sklearn.metrics import confusion_matrix
         confusion_matrix(y_test,rfy_pred)

Out[80]: array([[1541,  156],
                [ 292,  336]])

In [81]: from sklearn.metrics import accuracy_score
         accuracy_score(y_test,rfy_pred)

Out[81]: 0.8073118279569892
```

In [85]:

```
import warnings
warnings.filterwarnings('ignore')
from sklearn.linear_model import LogisticRegression
classifier=LogisticRegression()
classifier.fit(x_train,y_train)
```

Out[85]:

```
LogisticRegression()
```

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 [ ]:

```
im
```