

```
In [31]: import pandas as pd
import warnings
warnings.filterwarnings('ignore')
data=pd.read_csv("/home/placement/Downloads/TelecomCustomerChurn.csv")
```

```
In [32]: data.describe()
```

```
Out[32]:
```

	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 [33]:

```
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 [34]:

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

```
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   11  
Churn          0  
dtype: int64
```

```
In [36]: data1=data.fillna(data.median())
```

```
In [37]: data1.isna().sum()
```

```
Out[37]: 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 [38]: `list(data1)`

Out[38]: ['customerID',
'gender',
'SeniorCitizen',
'Partner',
'Dependents',
'tenure',
'PhoneService',
'MultipleLines',
'InternetService',
'OnlineSecurity',
'OnlineBackup',
'DeviceProtection',
'TechSupport',
'StreamingTV',
'StreamingMovies',
'Contract',
'PaperlessBilling',
'PaymentMethod',
'MonthlyCharges',
'TotalCharges',
'Churn']

In [39]: `data1.shape`

Out[39]: (7043, 21)

In [55]: `data2=data1.drop(['customerID'],axis=1)`

In [56]: data2

Out[56]:

	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	OnlineBackup	DeviceProte
0	Female	0	Yes	No	1	No	No phone service	DSL	No	Yes	
1	Male	0	No	No	34	Yes	No	DSL	Yes	No	
2	Male	0	No	No	2	Yes	No	DSL	Yes	Yes	
3	Male	0	No	No	45	No	No phone service	DSL	Yes	No	
4	Female	0	No	No	2	Yes	No	Fiber optic	No	No	
...	
7038	Male	0	Yes	Yes	24	Yes	Yes	DSL	Yes	No	
7039	Female	0	Yes	Yes	72	Yes	Yes	Fiber optic	No	Yes	
7040	Female	0	Yes	Yes	11	No	No phone service	DSL	Yes	No	
7041	Male	1	Yes	No	4	Yes	Yes	Fiber optic	No	No	
7042	Male	0	No	No	66	Yes	No	Fiber optic	Yes	No	

7043 rows × 20 columns

In [42]: data2['Churn']=data2['Churn'].map({'Yes':1,'No':0})

In [43]: data2

Out[43]:

	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	OnlineBackup	DeviceProte
0	Female	0	Yes	No	1	No	No phone service	DSL	No	Yes	
1	Male	0	No	No	34	Yes	No	DSL	Yes	No	
2	Male	0	No	No	2	Yes	No	DSL	Yes	Yes	
3	Male	0	No	No	45	No	No phone service	DSL	Yes	No	
4	Female	0	No	No	2	Yes	No	Fiber optic	No	No	
...	
7038	Male	0	Yes	Yes	24	Yes	Yes	DSL	Yes	No	
7039	Female	0	Yes	Yes	72	Yes	Yes	Fiber optic	No	Yes	
7040	Female	0	Yes	Yes	11	No	No phone service	DSL	Yes	No	
7041	Male	1	Yes	No	4	Yes	Yes	Fiber optic	No	No	
7042	Male	0	No	No	66	Yes	No	Fiber optic	Yes	No	

7043 rows × 20 columns

```
In [62]: x=data1.drop(['customerID','Churn'],axis=1)
y=data1['Churn']
```

```
In [63]: x=pd.get_dummies(x)
x
```

Out[63]:

	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 [64]: `x.head()`

Out[64]:

	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 rows × 11 columns

In [65]: `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 [66]: `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*`
`RFC_cls = GridSearchCV(cls, parameters)`
`RFC_cls.fit(x_train,y_train)`

Out[66]: `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 [67]: x_train.isna().sum()
```

```
Out[67]: SeniorCitizen      0
         tenure            0
         MonthlyCharges    0
         TotalCharges      0
         gender_Female     0
         gender_Male       0
         Partner_No        0
         Partner_Yes       0
         Dependents_No     0
         Dependents_Yes    0
         PhoneService_No   0
         PhoneService_Yes  0
         MultipleLines_No  0
         MultipleLines_No phone service 0
         MultipleLines_Yes 0
         InternetService_DSL 0
         InternetService_Fiber optic    0
         InternetService_No 0
         OnlineSecurity_No  0
         OnlineSecurity_No internet service 0
         OnlineSecurity_Yes 0
         OnlineBackup_No    0
         OnlineBackup_No internet service 0
         OnlineBackup_Yes   0
         DeviceProtection_No 0
         DeviceProtection_No internet service 0
         DeviceProtection_Yes 0
         TechSupport_No      0
         TechSupport_No internet service 0
         TechSupport_Yes     0
         StreamingTV_No      0
         StreamingTV_No internet service 0
         StreamingTV_Yes     0
         StreamingMovies_No  0
         StreamingMovies_No internet service 0
         StreamingMovies_Yes 0
         Contract_Month-to-month 0
         Contract_One year   0
```

```
Contract_Two year      0
PaperlessBilling_No    0
PaperlessBilling_Yes   0
PaymentMethod_Bank transfer (automatic) 0
PaymentMethod_Credit card (automatic)    0
PaymentMethod_Electronic check           0
PaymentMethod_Mailed check               0
dtype: int64
```

```
In [68]: RFC_cls.best_params_
```

```
Out[68]: {'criterion': 'entropy', 'max_depth': 10, 'n_estimators': 25}
```

```
In [71]: cls=RandomForestClassifier(n_estimators=200,criterion='entropy',max_depth=10)
```

```
In [72]: cls.fit(x_train,y_train)
```

```
Out[72]: RandomForestClassifier(criterion='entropy', max_depth=10, n_estimators=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]: rfy_pred=cls.predict(x_test)
```

```
In [74]: rfy_pred
```

```
Out[74]: array(['Yes', 'No', 'No', ..., 'Yes', 'No', 'No'], dtype=object)
```

```
In [76]: from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,rfy_pred)
```

```
Out[76]: array([[1548,  149],
               [ 303,  325]])
```

```
In [77]: from sklearn.metrics import accuracy_score
accuracy_score(y_test,rfy_pred)
```

```
Out[77]: 0.8055913978494623
```

```
In [78]: from sklearn.linear_model import LogisticRegression
classifier=LogisticRegression()
classifier.fit(x_train,y_train)
```

Out[78]: 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 [79]: y_pred=classifier.predict(x_test)
y_pred
```

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

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

Out[80]: array([[1526, 171],
 [266, 362]])

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

Out[81]: 0.8120430107526881

In []: