

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

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

```
In [3]: sai.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 [4]: list(sai)
```

```
Out[4]: ['customerID',  
        'gender',  
        'SeniorCitizen',  
        'Partner',  
        'Dependents',  
        'tenure',  
        'PhoneService',  
        'MultipleLines',  
        'InternetService',  
        'OnlineSecurity',  
        'OnlineBackup',  
        'DeviceProtection',  
        'TechSupport',  
        'StreamingTV',  
        'StreamingMovies',  
        'Contract',  
        'PaperlessBilling',  
        'PaymentMethod',  
        'MonthlyCharges',  
        'TotalCharges',  
        'Churn']
```

```
In [5]: sai.head()
```

Out[5]:

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

5 rows × 21 columns



```
In [6]: sai.describe()
```

Out[6]:

	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 [7]: sai=sai.drop("customerID",axis=1)
```

```
In [8]: sai
```

Out[8]:

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

7043 rows × 20 columns

```
In [9]: sai['TotalCharges']=pd.to_numeric(sai['TotalCharges'],errors='coerce')
```

```
In [10]: sai.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 20 columns):
#   Column                Non-Null Count  Dtype
---  -
0   gender                7043 non-null   object
1   SeniorCitizen         7043 non-null   int64
2   Partner               7043 non-null   object
3   Dependents            7043 non-null   object
4   tenure                7043 non-null   int64
5   PhoneService          7043 non-null   object
6   MultipleLines         7043 non-null   object
7   InternetService       7043 non-null   object
8   OnlineSecurity        7043 non-null   object
9   OnlineBackup          7043 non-null   object
10  DeviceProtection      7043 non-null   object
11  TechSupport           7043 non-null   object
12  StreamingTV           7043 non-null   object
13  StreamingMovies       7043 non-null   object
14  Contract              7043 non-null   object
15  PaperlessBilling      7043 non-null   object
16  PaymentMethod         7043 non-null   object
17  MonthlyCharges        7043 non-null   float64
18  TotalCharges          7032 non-null   float64
19  Churn                 7043 non-null   object
dtypes: float64(2), int64(2), object(16)
memory usage: 1.1+ MB
```

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

```
In [12]: sai
```

Out[12]:

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

7043 rows × 20 columns

```
In [13]: sai["SeniorCitizen"]=sai["SeniorCitizen"].map({0:"No",1:"Yes"})
```

```
In [14]: sai
```

Out[14]:

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

7043 rows × 20 columns

```
In [15]: x=sai.drop(['Churn'],axis=1)
```

```
In [16]: y=sai['Churn']
```

```
In [17]: x=pd.get_dummies(x,dtype=int)
```

```
In [18]: x.head()
```

Out[18]:

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

5 rows × 46 columns




```
In [19]: x.isna().sum()
```

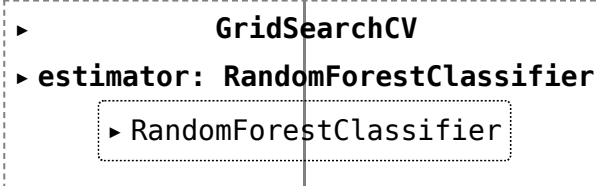
```
Out[19]: tenure                                0
MonthlyCharges                               0
TotalCharges                                 0
gender_Female                                0
gender_Male                                  0
SeniorCitizen_No                             0
SeniorCitizen_Yes                             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 [20]: 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 [21]: 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*3 = 48 i
RFC_cls = GridSearchCV(cls, parameters)
RFC_cls.fit(x_train,y_train)
```

```
Out[21]:
```



```
  ▶ GridSearchCV
  ▶ estimator: RandomForestClassifier
    ▶ RandomForestClassifier
```

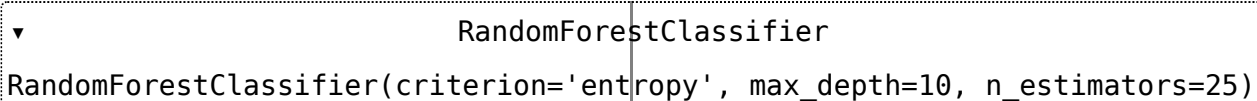
```
In [22]: RFC_cls.best_params_
```

```
Out[22]: {'criterion': 'gini', 'max_depth': 10, 'n_estimators': 150}
```

```
In [23]: cls=RandomForestClassifier(n_estimators=25,criterion='entropy',max_depth=10)
```

```
In [24]: cls.fit(x_train,y_train)
```

```
Out[24]:
```



```
  ▼ RandomForestClassifier
  RandomForestClassifier(criterion='entropy', max_depth=10, n_estimators=25)
```

```
In [25]: rfy_pred=cls.predict(x_test)
```

```
In [26]: rfy_pred
```

```
Out[26]: array(['Yes', 'No', 'No', ..., 'Yes', 'No', 'No'], dtype=object)
```

```
In [27]: from sklearn.metrics import confusion_matrix  
confusion_matrix(y_test,rfy_pred)
```

```
Out[27]: array([[1531,  166],  
               [ 293,  335]])
```

```
In [28]: from sklearn.metrics import accuracy_score  
accuracy_score(y_test,rfy_pred)#EFFICENCY OF THE CONFUSION MATRIX
```

```
Out[28]: 0.8025806451612904
```

```
In [ ]:
```

```
In [ ]:
```