

```
In [178]: import pandas as pd
import warnings
warnings.filterwarnings("ignore")
```

```
In [179]: data=pd.read_csv("/home/placement/Desktop/divyasri/TelecomCustomerChurn.csv")
```

```
In [180]: data.describe()
```

Out[180]:

	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 [181]: 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 [182]: list(data)
```

```
Out[182]: ['customerID',  
           'gender',  
           'SeniorCitizen',  
           'Partner',  
           'Dependents',  
           'tenure',  
           'PhoneService',  
           'MultipleLines',  
           'InternetService',  
           'OnlineSecurity',  
           'OnlineBackup',  
           'DeviceProtection',  
           'TechSupport',  
           'StreamingTV',  
           'StreamingMovies',  
           'Contract',  
           'PaperlessBilling',  
           'PaymentMethod',  
           'MonthlyCharges',  
           'TotalCharges',  
           'Churn']
```

```
In [183]: data['TotalCharges']=pd.to_numeric(data['TotalCharges'],errors='coerce')
```

```
In [184]: data.dtypes
```

```
Out[184]: 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 [185]: data.isna().sum()
```

```
Out[185]: 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 [186]: data.shape
```

```
Out[186]: (7043, 21)
```

```
In [187]: data.backup=data.copy()
```

```
In [188]: x=data.drop(['customerID','Churn'],axis=1)
y=data['Churn']
```

```
In [189]: data=data.fillna(data.median())
```

In [190]: data

Out[190]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	Device
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	...	
...	
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes	DSL	Yes	...	
7039	2234-XADUH	Female	0	Yes	Yes	72	Yes	Yes	Fiber optic	No	...	
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phone service	DSL	Yes	...	
7041	8361-LTMKD	Male	1	Yes	No	4	Yes	Yes	Fiber optic	No	...	
7042	3186-AJIEK	Male	0	No	No	66	Yes	No	Fiber optic	Yes	...	

7043 rows × 21 columns

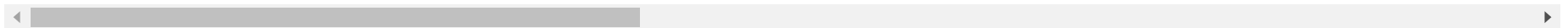
In [191]: x=pd.get_dummies(x)

In [192]: x

Out[192]:

	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	
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

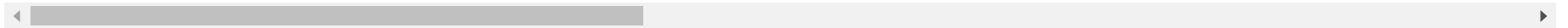


```
In [193]: x.head()
```

```
Out[193]:
```

	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 [205]: x['TotalCharges'] = x['TotalCharges'].fillna(x['TotalCharges'].median())
```


In [195]:

x

Out[195]:

	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	
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 [196]: from sklearn.model_selection import train_test_split #splitting of training and testing
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.33,random_state=42)
```

```
In [197]: y_train.head(5)
```

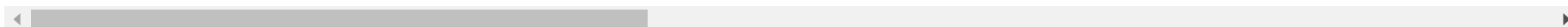
```
Out[197]: 298      No
          3318    Yes
          5586    No
          6654    Yes
          5362    No
          Name: Churn, dtype: object
```

```
In [198]: x_train.head(5)
```

```
Out[198]:
```

	SeniorCitizen	tenure	MonthlyCharges	TotalCharges	gender_Female	gender_Male	Partner_No	Partner_Yes	Dependents_No	Dependents_Yes
298	0	40	74.55	3015.75	0	1	0	1	0	
3318	0	10	29.50	255.25	0	1	1	0	1	
5586	0	27	19.15	501.35	1	0	1	0	1	
6654	0	7	86.50	582.50	1	0	0	1	1	
5362	0	65	24.75	1715.10	0	1	0	1	0	

5 rows × 45 columns



```
In [199]: y_test.head(5)
```

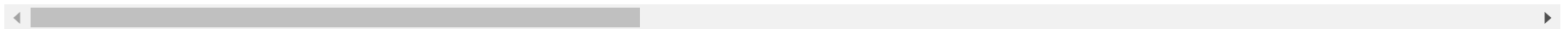
```
Out[199]: 185      Yes
          2715    No
          3825    No
          1807    Yes
          132     No
          Name: Churn, dtype: object
```

```
In [200]: x_test.head(5)
```

```
Out[200]:
```

	SeniorCitizen	tenure	MonthlyCharges	TotalCharges	gender_Female	gender_Male	Partner_No	Partner_Yes	Dependents_No	Dependents_Yes
185	0	1	24.80	24.80	1	0	0	1	1	
2715	0	41	25.25	996.45	0	1	1	0	1	
3825	0	52	19.35	1031.70	1	0	0	1	0	
1807	0	1	76.35	76.35	1	0	1	0	1	
132	0	67	50.55	3260.10	0	1	1	0	1	

5 rows × 11 columns



```
In [201]: x.isna().sum()
```

```
Out[201]: 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 [204]: 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[204]:
└─ GridSearchCV
  └─ estimator: RandomForestClassifier
    └─ RandomForestClassifier

```

```

In [208]: RFC_cls.best_params_

```

```

Out[208]: {'criterion': 'entropy', 'max_depth': 10, 'n_estimators': 100}

```

```

In [213]: cls=RandomForestClassifier(n_estimators=100,criterion='entropy',max_depth=10)

```

```

In [214]: cls.fit(x_train,y_train)

```

```

Out[214]:
└─ RandomForestClassifier
  RandomForestClassifier(criterion='entropy', max_depth=10)

```

```
In [215]: rfy_pred=cls.predict(x_test)
```

```
In [216]: rfy_pred
```

```
Out[216]: array(['Yes', 'No', 'No', ..., 'Yes', 'No', 'No'], dtype=object)
```

```
In [217]: from sklearn.metrics import confusion_matrix #confusion_matrix  
confusion_matrix(y_test,rfy_pred)
```

```
Out[217]: array([[1552, 145],  
                [ 304, 324]])
```

```
In [218]: from sklearn.metrics import accuracy_score  
accuracy_score(y_test,rfy_pred)
```

```
Out[218]: 0.8068817204301075
```

```
In [219]: from sklearn.linear_model import LogisticRegression #logistic regression  
classifier=LogisticRegression()  
classifier.fit(x_train,y_train)
```

```
Out[219]: 

▼ LogisticRegression



LogisticRegression()


```

```
In [220]: y_pred=classifier.predict(x_test)  
y_pred
```

```
Out[220]: array(['Yes', 'No', 'No', ..., 'Yes', 'No', 'No'], dtype=object)
```

```
In [221]: from sklearn.metrics import confusion_matrix #confusion_matrix  
confusion_matrix(y_test,y_pred)
```

```
Out[221]: array([[1526, 171],  
                [ 266, 362]])
```

```
In [222]: from sklearn.metrics import accuracy_score  
accuracy_score(y_test,y_pred)
```

```
Out[222]: 0.8120430107526881
```

```
In [ ]:
```