```
In [27]: import pandas as pd
```

```
In [28]: data = pd.read_csv("/home/placement/Downloads/TelecomCustomerChurn.csv")
```

```
In [29]: data.describe()
```

```
Out[29]:
         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 [30]:

```
data.head()
```

Out[30]:

	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 [31]:

```
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 [32]:

```
data.shape
```

Out[32]:

```
(7043, 21)
```

In [33]:

```
list(data)
```

Out[33]:

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

In [34]:

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

In [35]:
```
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 [36]:

```
data1=data.drop(['customerID' ,'StreamingTV','StreamingMovies','Partner','SeniorCitizen','Dependents','Phone
data1
```

Out[36]:

	gender	tenure	MultipleLines	InternetService	DeviceProtection	TechSupport	Contract	MonthlyCharges	TotalCharges	Churn
0	Female	1	No phone service	DSL	No	No	Month-to-month	29.85	29.85	No
1	Male	34	No	DSL	Yes	No	One year	56.95	1889.50	No
2	Male	2	No	DSL	No	No	Month-to-month	53.85	108.15	Yes
3	Male	45	No phone service	DSL	Yes	Yes	One year	42.30	1840.75	No
4	Female	2	No	Fiber optic	No	No	Month-to-month	70.70	151.65	Yes
...	...	...	...	...	...	...	...	...	...	...
7038	Male	24	Yes	DSL	Yes	Yes	One year	84.80	1990.50	No
7039	Female	72	Yes	Fiber optic	Yes	No	One year	103.20	7362.90	No
7040	Female	11	No phone service	DSL	No	No	Month-to-month	29.60	346.45	No
7041	Male	4	Yes	Fiber optic	No	No	Month-to-month	74.40	306.60	Yes
7042	Male	66	No	Fiber optic	Yes	Yes	Two year	105.65	6844.50	No

7043 rows × 10 columns

In [37]:

```
data2=data1.fillna(data1.median())
```

```
/tmp/ipykernel_5199/3414091449.py:1: FutureWarning: The default value of numeric_only in DataFrame.median i
s deprecated. In a future version, it will default to False. In addition, specifying 'numeric_only=None' is
deprecated. Select only valid columns or specify the value of numeric_only to silence this warning.
  data2=data1.fillna(data1.median())
```

In [38]:

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

Out[38]:

	gender	tenure	MultipleLines	InternetService	DeviceProtection	TechSupport	Contract	MonthlyCharges	TotalCharges	Churn
0	Female	1	No phone service	DSL	No	No	Month-to-month	29.85	29.85	0
1	Male	34	No	DSL	Yes	No	One year	56.95	1889.50	0
2	Male	2	No	DSL	No	No	Month-to-month	53.85	108.15	1
3	Male	45	No phone service	DSL	Yes	Yes	One year	42.30	1840.75	0
4	Female	2	No	Fiber optic	No	No	Month-to-month	70.70	151.65	1
...	...	...	...	...	...	...	...	...	...	...
7038	Male	24	Yes	DSL	Yes	Yes	One year	84.80	1990.50	0
7039	Female	72	Yes	Fiber optic	Yes	No	One year	103.20	7362.90	0
7040	Female	11	No phone service	DSL	No	No	Month-to-month	29.60	346.45	0
7041	Male	4	Yes	Fiber optic	No	No	Month-to-month	74.40	306.60	1
7042	Male	66	No	Fiber optic	Yes	Yes	Two year	105.65	6844.50	0

7043 rows × 10 columns

In [39]:
```
data3=pd.get_dummies(data2)
data3
```

Out[39]:

	tenure	MonthlyCharges	TotalCharges	Churn	gender_Female	gender_Male	MultipleLines_No	MultipleLines_No phone service	MultipleLines_Yes	Internet
0	1	29.85	29.85	0	1	0	0	1	0	
1	34	56.95	1889.50	0	0	1	1	0	0	
2	2	53.85	108.15	1	0	1	1	0	0	
3	45	42.30	1840.75	0	0	1	0	1	0	
4	2	70.70	151.65	1	1	0	1	0	0	
...	...	...	...	...	...	...	...	...	...	
7038	24	84.80	1990.50	0	0	1	0	0	1	
7039	72	103.20	7362.90	0	1	0	0	0	1	
7040	11	29.60	346.45	0	1	0	0	1	0	
7041	4	74.40	306.60	1	0	1	0	0	1	
7042	66	105.65	6844.50	0	0	1	1	0	0	

7043 rows × 21 columns

In [40]:
```
data3.shape
```

Out[40]:
```
(7043, 21)
```

In [41]:
```
y=data3['Churn']
x=data3.drop('Churn',axis=1)
```

In [42]:
```
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 [43]:

```
from sklearn.linear_model import LogisticRegression
classifier=LogisticRegression()
classifier.fit(x_train,y_train)
```

```
/home/placement/anaconda3/lib/python3.10/site-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWa
rning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-learn.org/stable/modules/pre
processing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression (https://scikit-learn.or
g/stable/modules/linear_model.html#logistic-regression)
  n_iter_i = _check_optimize_result(
```

Out[43]:

```
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 [44]:

```
y_pred=classifier.predict(x_test)
y_pred
```

Out[44]:

```
array([1, 0, 0, ..., 1, 1, 0])
```

In [45]:

```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,y_pred)
```

Out[45]:

```
array([[1513,  184],
       [ 272,  356]])
```

In [46]:

```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

Out[46]:

```
0.8038709677419354
```

In [ ]:

```

```