In [1]:

```
import pandas as pd
import numpy as np
import pickle
import warnings
warnings.filterwarnings("ignore")
```

In [2]:

```
a=pd.read_csv("C:/Users/reshma_koduri/OneDrive/Documents/TelecomCustomerChurn CRT.cs
```

In [3]:

```
a
```

Out[3]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service
1	5575-GNVDE	Male	0	No	No	34	Yes	No
2	3668-QPYBK	Male	0	No	No	2	Yes	No
3	7795-CFOCW	Male	0	No	No	45	No	No phone service
4	9237-HQITU	Female	0	No	No	2	Yes	No
...	...	...	...	...	...	...	...	...
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes
7039	2234-XADUH	Female	0	Yes	Yes	72	Yes	Yes
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phone service
7041	8361-LTMKD	Male	1	Yes	No	4	Yes	Yes
7042	3186-AJIEK	Male	0	No	No	66	Yes	No

7043 rows × 21 columns

In [4]:

```
a.head(20)
```

Out[4]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	I
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	In
1	5575-GNVDE	Male	0	No	No	34	Yes	No	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	
4	9237-HQITU	Female	0	No	No	2	Yes	No	
5	9305-CDSKC	Female	0	No	No	8	Yes	Yes	
6	1452-KIOVK	Male	0	No	Yes	22	Yes	Yes	
7	6713-OKOMC	Female	0	No	No	10	No	No phone service	
8	7892-POOKP	Female	0	Yes	No	28	Yes	Yes	
9	6388-TABGU	Male	0	No	Yes	62	Yes	No	
10	9763-GRSKD	Male	0	Yes	Yes	13	Yes	No	
11	7469-LKBCI	Male	0	No	No	16	Yes	No	
12	8091-TTVAX	Male	0	Yes	No	58	Yes	Yes	
13	0280-XJGEX	Male	0	No	No	49	Yes	Yes	
14	5129-JLPIS	Male	0	No	No	25	Yes	No	
15	3655-SNQYZ	Female	0	Yes	Yes	69	Yes	Yes	
16	8191-XWSZG	Female	0	No	No	52	Yes	No	
17	9959-WOFKT	Male	0	No	Yes	71	Yes	Yes	
18	4190-MFLUW	Female	0	Yes	Yes	10	Yes	No	
19	4183-MYFRB	Female	0	No	No	21	Yes	No	

20 rows × 21 columns

In [5]:

```
a.tail(10)
```

Out[5]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines
7033	9767-FFLEM	Male	0	No	No	38	Yes	No
7034	0639-TSIQW	Female	0	No	No	67	Yes	Yes
7035	8456-QDAVC	Male	0	No	No	19	Yes	No
7036	7750-EYXWZ	Female	0	No	No	12	No	No phone service
7037	2569-WGERO	Female	0	No	No	72	Yes	No
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes
7039	2234-XADUH	Female	0	Yes	Yes	72	Yes	Yes
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phone service
7041	8361-LTMKD	Male	1	Yes	No	4	Yes	Yes
7042	3186-AJIEK	Male	0	No	No	66	Yes	No

10 rows × 21 columns

In [6]:

```
a.shape
```

Out[6]:

```
(7043, 21)
```

In [7]:

```
a.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 [8]:

```
#a["TotalCharges"] = pd.to_numeric(a["TotalCharges"])
#a['TotalCharges'] = a['TotalCharges'].astype(str).astype(int)
#a['TotalCharges1'] = a['TotalCharges'].astype(float)
#a['TotalCharges']=pd.to_numeric(a['TotalCharges'], downcast="float")
a['TC']=pd.to_numeric(a['TotalCharges'], errors='coerce')
```

In [9]:

```
a.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 22 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 
 21  TC                7032 non-null   float64
dtypes: float64(2), int64(2), object(18)
memory usage: 1.2+ MB
```

In [10]:

```
a.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 22 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 
 21  TC                7032 non-null   float64
dtypes: float64(2), int64(2), object(18)
memory usage: 1.2+ MB
```

In [11]:

```
a.describe()
```

Out[11]:

	SeniorCitizen	tenure	MonthlyCharges	TC
count	7043.000000	7043.000000	7043.000000	7032.000000
mean	0.162147	32.371149	64.761692	2283.300441
std	0.368612	24.559481	30.090047	2266.771362
min	0.000000	0.000000	18.250000	18.800000
25%	0.000000	9.000000	35.500000	401.450000
50%	0.000000	29.000000	70.350000	1397.475000
75%	0.000000	55.000000	89.850000	3794.737500
max	1.000000	72.000000	118.750000	8684.800000

In [12]:

```
list(a)
```

Out[12]:

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

In [13]:

```
a.isna().sum()
```

Out[13]:

```
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
TC                  11
dtype: int64
```

In [14]:

```
a['TC'].unique()
```

Out[14]:

```
array([  29.85, 1889.5 ,  108.15, ...,  346.45,  306.6 , 6844.5 ])
```

In [15]:

```
a.fillna(35,inplace=True)
```

In [16]:

```
a['customerID'].unique()
```

Out[16]:

```
array(['7590-VHVEG', '5575-GNVDE', '3668-QPYBK', ..., '4801-JZAZL',
       '8361-LTMKD', '3186-AJIEK'], dtype=object)
```

In [17]:

```
a.groupby(['gender']).count()
```

Out[17]:

	customerID	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	Intern
gender								
Female	3488	3488	3488	3488	3488	3488	3488	
Male	3555	3555	3555	3555	3555	3555	3555	

2 rows × 21 columns

In [18]:

```
b=a.drop(['customerID','TotalCharges','StreamingTV','StreamingMovies','SeniorCitizen
b
```

Out[18]:

	gender	tenure	MultipleLines	InternetService	DeviceProtection	TechSupport	Contract	Mont
0	Female	1	No phone service	DSL	No	No	Month-to-month	
1	Male	34	No	DSL	Yes	No	One year	
2	Male	2	No	DSL	No	No	Month-to-month	
3	Male	45	No phone service	DSL	Yes	Yes	One year	
4	Female	2	No	Fiber optic	No	No	Month-to-month	
...	...	...	...	...	...	...	...	
7038	Male	24	Yes	DSL	Yes	Yes	One year	
7039	Female	72	Yes	Fiber optic	Yes	No	One year	
7040	Female	11	No phone service	DSL	No	No	Month-to-month	
7041	Male	4	Yes	Fiber optic	No	No	Month-to-month	
7042	Male	66	No	Fiber optic	Yes	Yes	Two year	

7043 rows × 10 columns

In [19]:

```
b.shape
```

Out[19]:

```
(7043, 10)
```

In [20]:

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

In [21]:

```
c=pd.get_dummies(b,dtype=int)
c
```

Out[21]:

	tenure	MonthlyCharges	Churn	TC	gender_Female	gender_Male	MultipleLines_No	Mul
0	1	29.85	0	29.85	1	0	0	
1	34	56.95	0	1889.50	0	1	1	
2	2	53.85	1	108.15	0	1	1	
3	45	42.30	0	1840.75	0	1	0	

	tenure	MonthlyCharges	Churn	TC	gender_Female	gender_Male	MultipleLines_No	Mul I
4	2	70.70	1	151.65	1	0	1	
...	...	...	...	...	...	...	...	
7038	24	84.80	0	1990.50	0	1	0	
7039	72	103.20	0	7362.90	1	0	0	
7040	11	29.60	0	346.45	1	0	0	
7041	4	74.40	1	306.60	0	1	0	
7042	66	105.65	0	6844.50	0	1	1	

7043 rows × 21 columns

In [22]:

```
c.shape
```

Out[22]:

```
(7043, 21)
```

In [23]:

```
y=c['Churn']
y
```

Out[23]:

```
0       0
1       0
2       1
3       0
4       1
       ..
7038    0
7039    0
7040    0
7041    1
7042    0
Name: Churn, Length: 7043, dtype: int64
```

In [24]:

```
x=c.drop(['Churn'],axis=1)
x
```

Out[24]:

	tenure	MonthlyCharges	TC	gender_Female	gender_Male	MultipleLines_No	MultipleLine phone se
0	1	29.85	29.85	1	0	0	
1	34	56.95	1889.50	0	1	1	
2	2	53.85	108.15	0	1	1	
3	45	42.30	1840.75	0	1	0	
4	2	70.70	151.65	1	0	1	
...	...	...	...	...	...	...	
7038	24	84.80	1990.50	0	1	0	
7039	72	103.20	7362.90	1	0	0	
7040	11	29.60	346.45	1	0	0	

	tenure	MonthlyCharges	TC	gender_Female	gender_Male	MultipleLines_No	MultipleLine phone se
7041	4	74.40	306.60	0	1	0	
7042	66	105.65	6844.50	0	1	1	

7043 rows × 20 columns

In [25]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30,random_state=56)
from sklearn.neighbors import KNeighborsRegressor
```

In [26]:

```
x_train.head(5)
```

Out[26]:

	tenure	MonthlyCharges	TC	gender_Female	gender_Male	MultipleLines_No	MultipleLine phone se
168	58	45.30	2651.20	0	1	0	
4908	28	106.15	3152.50	1	0	0	
2063	51	76.40	3966.30	0	1	1	
4283	4	70.20	280.35	0	1	1	
4856	31	91.15	2995.45	1	0	0	

In [27]:

```
y_train.head(5)
```

Out[27]:

```
168     1
4908    1
2063    0
4283    0
4856    1
Name: Churn, dtype: int64
```

In [28]:

```
y_test.head(5)
```

Out[28]:

```
6046    0
6494    0
2702    0
1532    0
3999    1
Name: Churn, dtype: int64
```

In [29]:

```
#from sklearn.linear_model import LogisticRegression
#knn = KNeighborsRegressor(n_neighbors=1)
#knn.fit(x_train,y_train)
```

In [30]:

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

Out[30]:

```
LogisticRegression()
```

In [31]:

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

Out[31]:

```
array([1, 0, 0, ..., 0, 0, 0], dtype=int64)
```

In [32]:

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

Out[32]:

```
array([[1403,  151],
       [ 271,  288]], dtype=int64)
```

In [33]:

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

Out[33]:

```
0.8002839564600095
```

In [ ]: