telco-customer-churn

November 4, 2024

[214]:
```
#Python Project On Customer Churn
import seaborn as sns
import pandas as pd
from matplotlib import pyplot as plt
```

[216]:
```
churn=pd.read_csv("TCA.csv")
churn.head(10)
```

[216]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	\
0	7590-VHVEG	Female	0	Yes	No	1	No	
1	5575-GNVDE	Male	0	No	No	34	Yes	
2	3668-QPYBK	Male	0	No	No	2	Yes	
3	7795-CFOCW	Male	0	No	No	45	No	
4	9237-HQITU	Female	0	No	No	2	Yes	
5	9305-CDSKC	Female	0	No	No	8	Yes	
6	1452-KIOVK	Male	0	No	Yes	22	Yes	
7	6713-OKOMC	Female	0	No	No	10	No	
8	7892-POOKP	Female	0	Yes	No	28	Yes	
9	6388-TABGU	Male	0	No	Yes	62	Yes	

	MultipleLines	InternetService	OnlineSecurity	…	DeviceProtection	\
0	No phone service	DSL	No	…	No	
1	No	DSL	Yes	…	Yes	
2	No	DSL	Yes	…	No	
3	No phone service	DSL	Yes	…	Yes	
4	No	Fiber optic	No	…	No	
5	Yes	Fiber optic	No	…	Yes	
6	Yes	Fiber optic	No	…	No	
7	No phone service	DSL	Yes	…	No	
8	Yes	Fiber optic	No	…	Yes	
9	No	DSL	Yes	…	No	

	TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling	\
0	No	No	No	Month-to-month	Yes	
1	No	No	No	One year	No	
2	No	No	No	Month-to-month	Yes	
3	Yes	No	No	One year	No	

```
4          No          No              No  Month-to-month              Yes
5          No         Yes             Yes  Month-to-month              Yes
6          No         Yes              No  Month-to-month              Yes
7          No          No              No  Month-to-month               No
8         Yes         Yes             Yes  Month-to-month              Yes
9          No          No              No        One year               No

               PaymentMethod MonthlyCharges  TotalCharges Churn
0           Electronic check          29.85         29.85    No
1               Mailed check          56.95        1889.5    No
2               Mailed check          53.85        108.15   Yes
3  Bank transfer (automatic)          42.30       1840.75    No
4           Electronic check          70.70        151.65   Yes
5           Electronic check          99.65         820.5   Yes
6    Credit card (automatic)          89.10        1949.4    No
7               Mailed check          29.75         301.9    No
8           Electronic check         104.80       3046.05   Yes
9  Bank transfer (automatic)          56.15       3487.95    No

[10 rows x 21 columns]
```

[217]:
```
churn.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
```

[220]:
```
churn.isnull().sum()
```

[220]:
```
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
```

[222]:
```
#Replacing The Blank Values " " to 0
churn["TotalCharges"]=churn["TotalCharges"].replace(" ", "0")
```

[224]:
```
#changing the datatype From int To Float Of Total Charges Coloumn
churn["TotalCharges"]=churn["TotalCharges"].astype(float)
```

[226]:
```
churn.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   float64
 20  Churn             7043 non-null   object
dtypes: float64(2), int64(2), object(17)
memory usage: 1.1+ MB
```

[228]:
```
churn.describe()
```

[228]:

	SeniorCitizen	tenure	MonthlyCharges	TotalCharges
count	7043.000000	7043.000000	7043.000000	7043.000000
mean	0.162147	32.371149	64.761692	2279.734304
std	0.368612	24.559481	30.090047	2266.794470
min	0.000000	0.000000	18.250000	0.000000
25%	0.000000	9.000000	35.500000	398.550000
50%	0.000000	29.000000	70.350000	1394.550000
75%	0.000000	55.000000	89.850000	3786.600000
max	1.000000	72.000000	118.750000	8684.800000

[230]:
```
#to check the duplicates in dataset
```

[232]:
```
churn.duplicated().sum()
```

[232]: 0

[234]:
```
#to check the duplicates in customerId column
```

[236]:
```
churn["customerID"].duplicated().sum()
```

[236]: 0

[238]:
```
#Converting Senior Citizen Values From 0 to Yes/No Means If Senior Citizen Is␣
 ↪Present Than Yes Else No
```

[240]:
```
def conv(value):
    if value==1:
        return "Yes"
    else:
        return "No"

churn["SeniorCitizen"]=churn["SeniorCitizen"].apply(conv)
```

[242]:
```
#Lets See The Output
churn.head()
```

[242]:

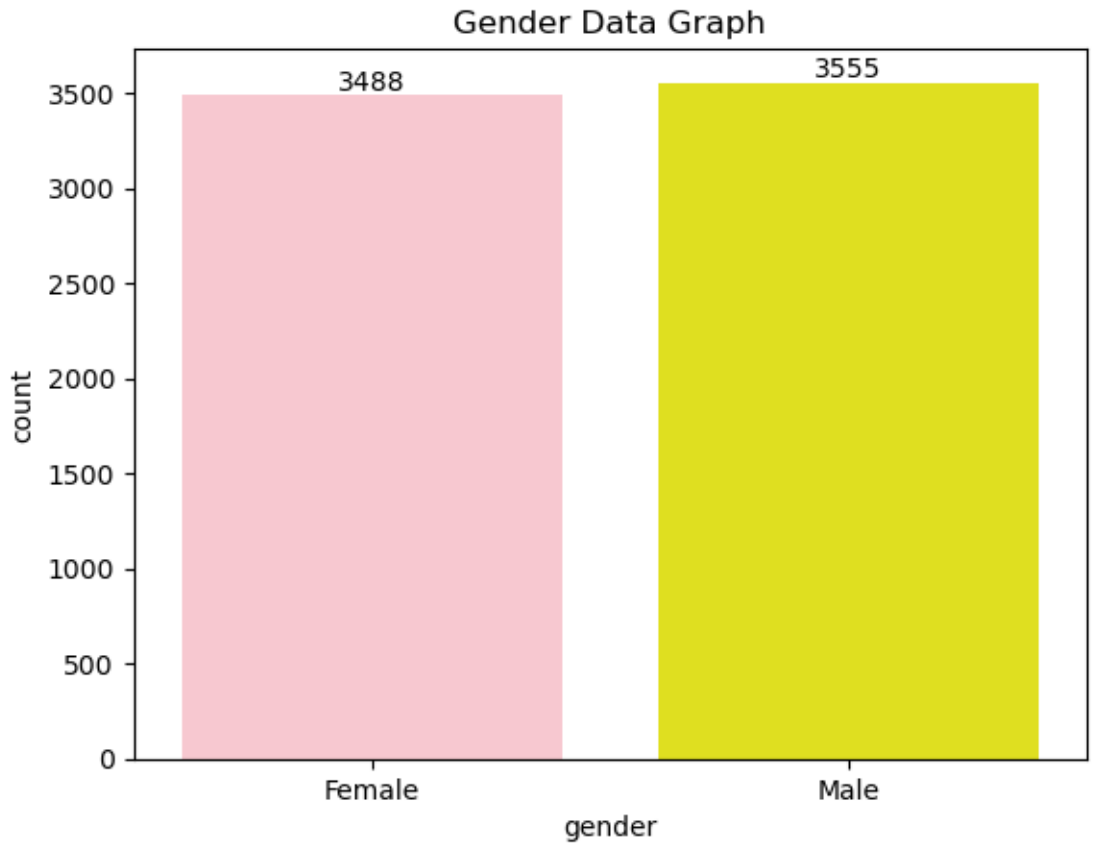
	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	…	DeviceProtection	TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling	PaymentMethod	MonthlyCharges	TotalCharges	Churn
0	7590-VHVEG	Female	No	Yes	No	1	No	…	No	No	No	No	Month-to-month	Yes	Electronic check	29.85	29.85	No
1	5575-GNVDE	Male	No	No	No	34	Yes	…	Yes	No	No	No	One year	No	Mailed check	56.95	1889.50	No
2	3668-QPYBK	Male	No	No	No	2	Yes	…	No	No	No	No	Month-to-month	Yes	Mailed check	53.85	108.15	Yes
3	7795-CFOCW	Male	No	No	No	45	No	…	Yes	Yes	No	No	One year	No	Bank transfer (automatic)	42.30	1840.75	No
4	9237-HQITU	Female	No	No	No	2	Yes	…	No	No	No	No	Month-to-month	Yes	Electronic check	70.70	151.65	Yes

[5 rows x 21 columns]

[244]:
```
#Now lets See The "Gender" Values In Terms Of Graph
```

[246]:
```
plt.title("Gender Data Graph")
cp=sns.countplot(x="gender",data=churn,hue="gender",palette=["pink","yellow"])
cp.bar_label(cp.containers[0])
cp.bar_label(cp.containers[1])
```

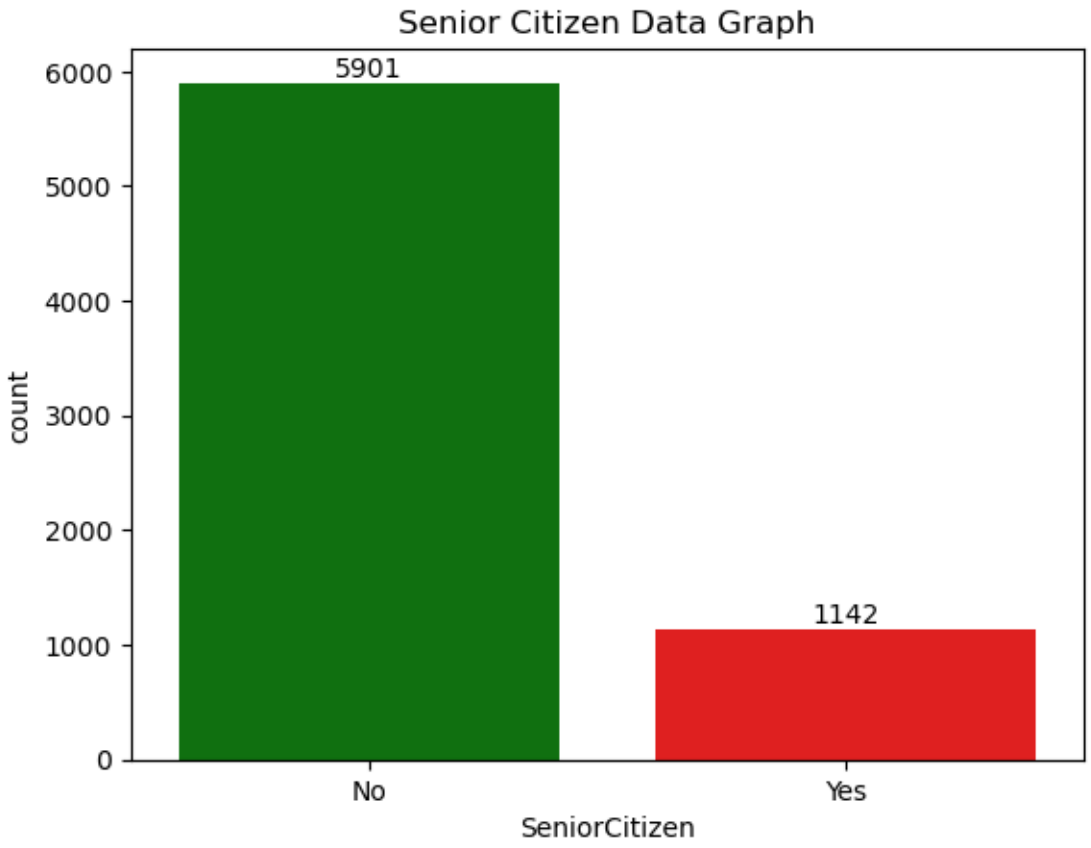
```
plt.show()
```



[247]:
```
#We Can See That Male Subscribers Are More Than Female Subscribers
```

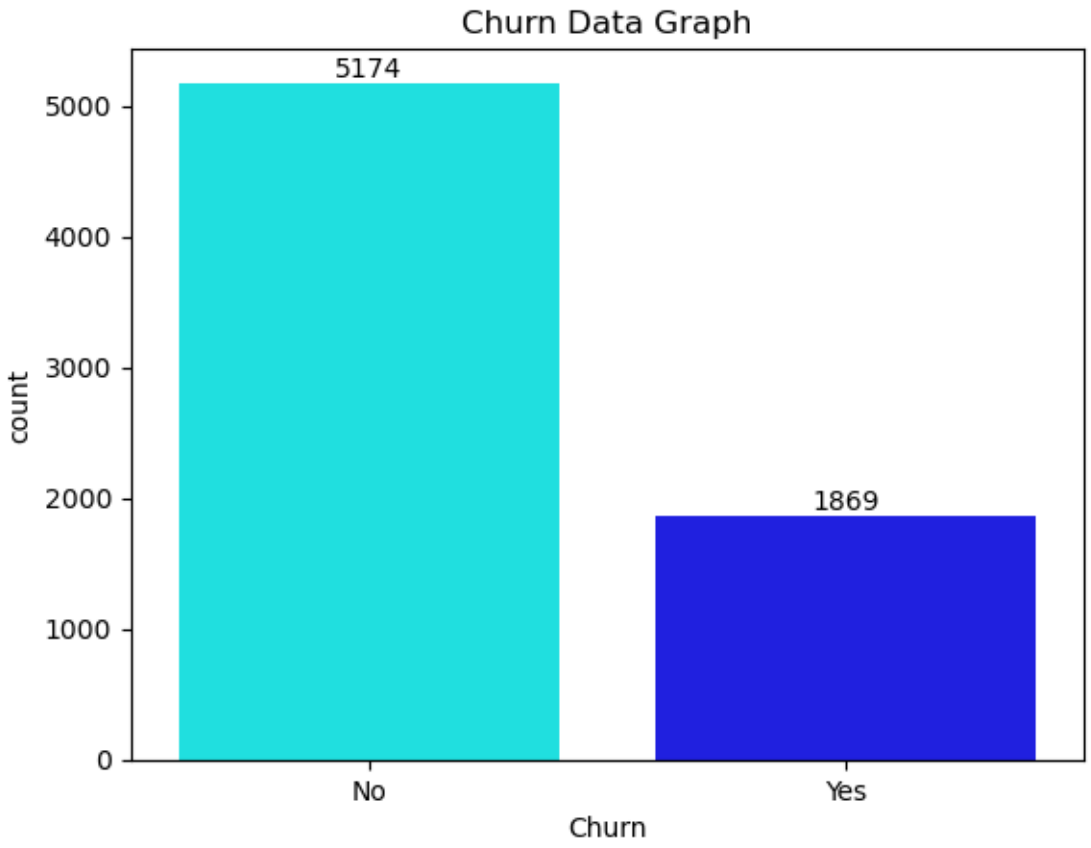
[250]:
```
#Lets See How Many Senior Citizen Frequency Are Their
```

[252]:
```
plt.title("Senior Citizen Data Graph")
cp=sns.
 ↪countplot(x="SeniorCitizen",data=churn,hue="SeniorCitizen",palette=["green","red"])
cp.bar_label(cp.containers[0])
cp.bar_label(cp.containers[1])
plt.show()
```



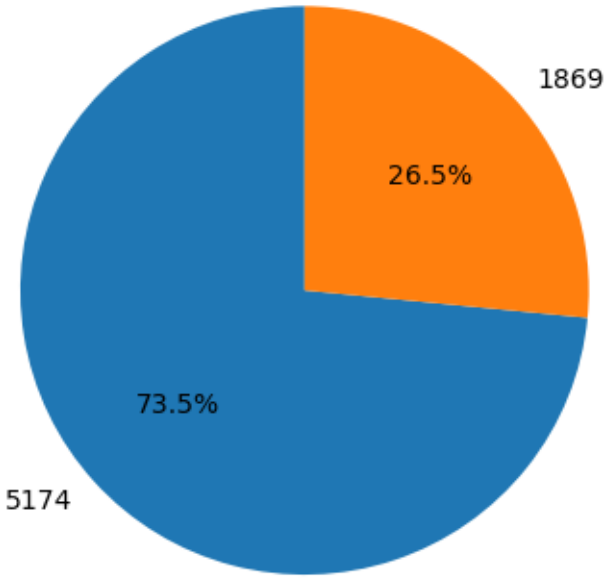
[253]:
```
#Lets See The Churn Subscriber Data
```

[256]:
```
plt.title("Churn Data Graph")
cp=sns.countplot(x="Churn",data=churn,hue="Churn",palette=["cyan","blue"])
cp.bar_label(cp.containers[0])
cp.bar_label(cp.containers[1])
plt.show()
```



[257]:
```
#Now Lets See The Churn Value In Percentage Or We Say In Pie
```

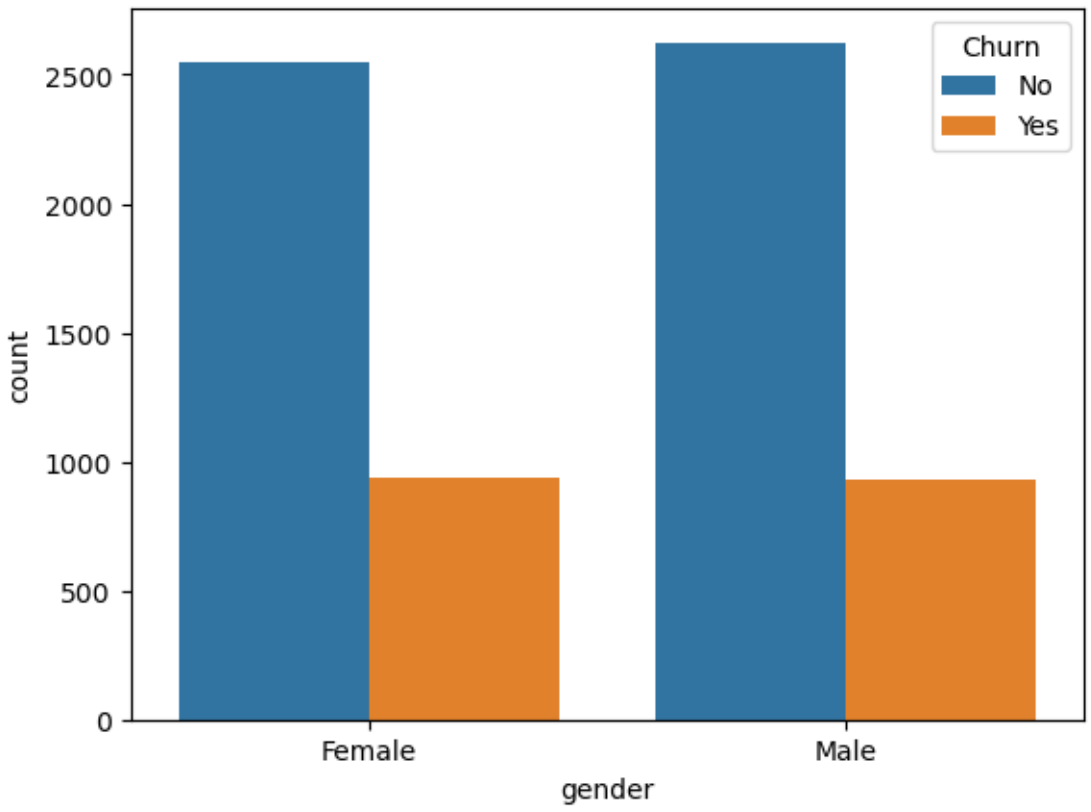
[296]:
```
gb=churn.groupby("Churn").agg({"Churn":"count"})
plt.pie(gb["Churn"],labels=gb["Churn"],autopct="%0.1f%%",startangle=90)
plt.show()
```



[262]:
```
#We Can See By Pie Chart That 73.5% People Npt Churn As A Subscriber Compared␣
↪To Person Who Churn As Subscriber
```

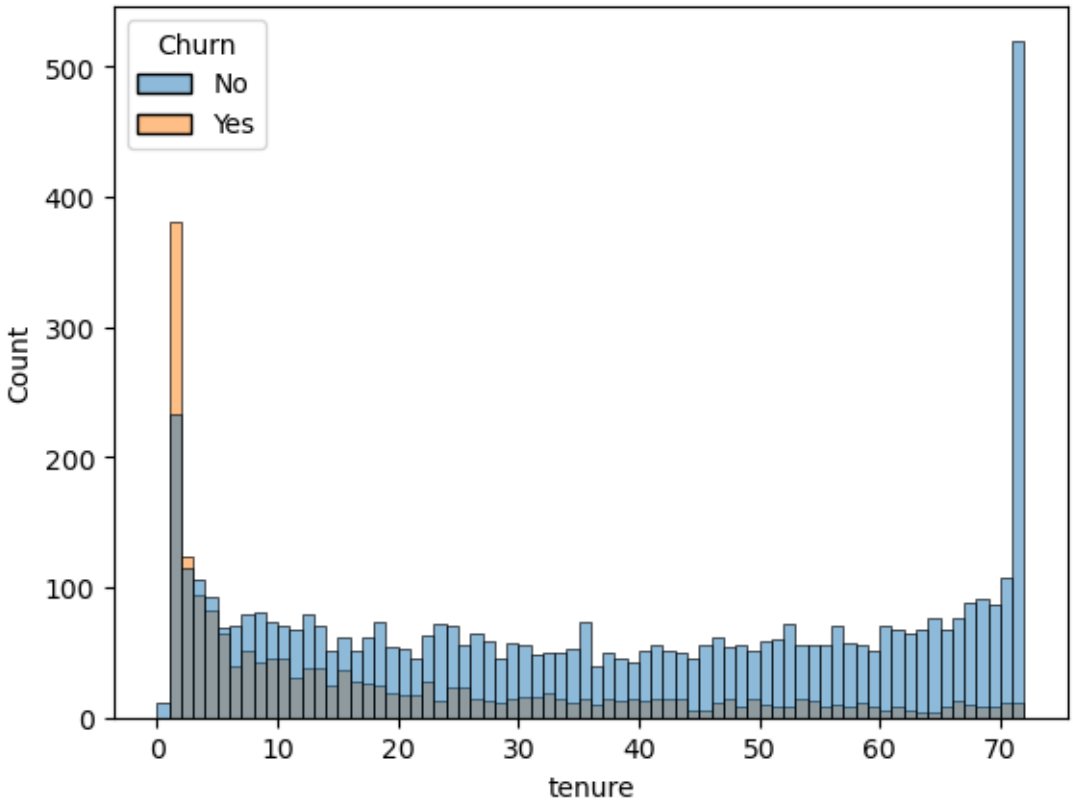
[264]:
```
#To Know The Churn By Gender As A Subscriber
```

[266]:
```
sns.countplot(x="gender", data=churn,hue="Churn")
plt.show()
```



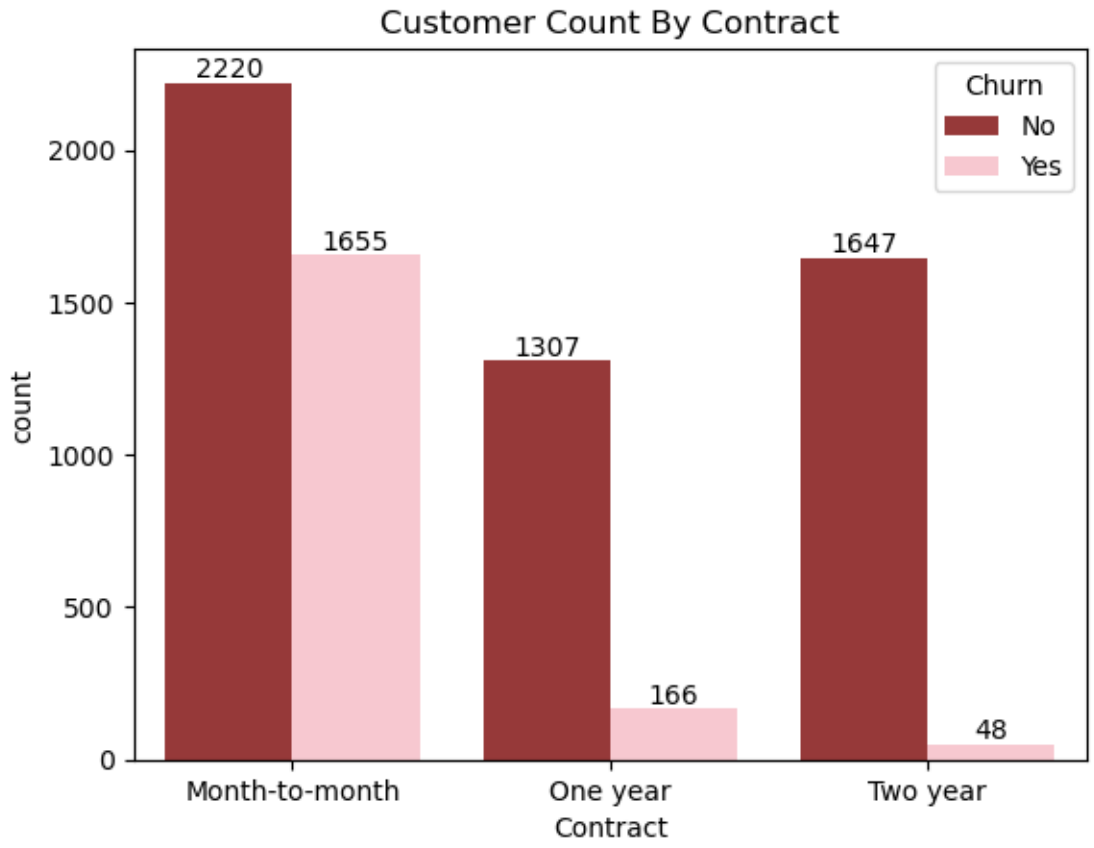
[267]:
```
#lets see the tenure using histogram
```

[270]:
```
sns.histplot(x="tenure",data=churn,bins=72,hue="Churn")
plt.show()
```



[271]:
```
#We Can See The Histogram Of Tenure On The Basis Of Churn
```

[272]:
```
plt.title("Customer Count By Contract")
cp=sns.countplot(x="Contract",data=churn,hue="Churn",palette=["brown","pink"])
cp.bar_label(cp.containers[0])
cp.bar_label(cp.containers[1])
plt.show()
```

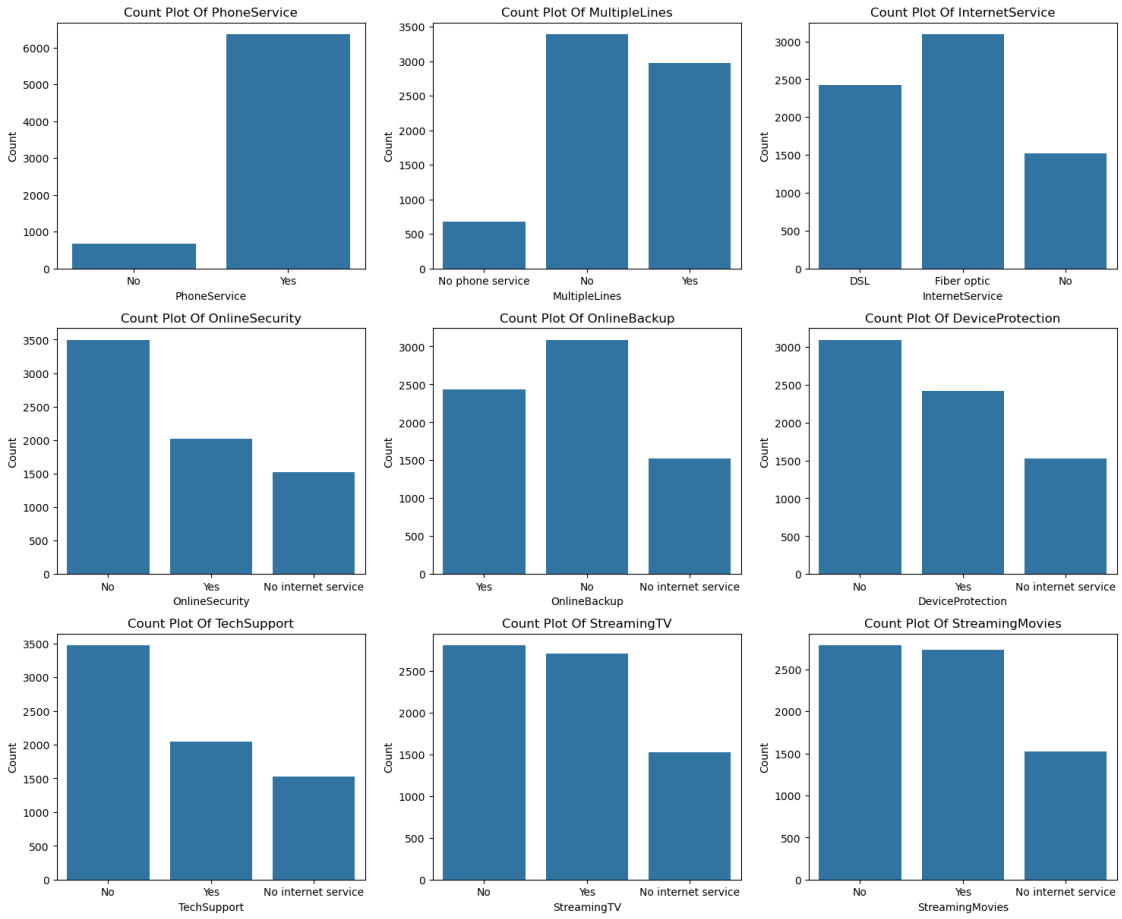


[273]:
```
#Here We Can See Customer Count On The Basis Of Churn
```

[276]:
```
#To See The Countplot Of All Remaining Services On The Basis Of Churn
```

[277]:
```
columns=["PhoneService","MultipleLines","InternetService","OnlineSecurity","OnlineBackup","Dev
n_cols=3
n_rows=(len(columns)+n_cols-1)
fig,axes=plt.subplots(n_rows,n_cols,figsize=(15,n_rows*4))
axes=axes.flatten()
for i, col in enumerate(columns):
    sns.countplot(x=col,data=churn,ax=axes[i])
    axes[i].set_title(f"Count Plot Of {col}")
    axes[i].set_xlabel(col)
    axes[i].set_ylabel("Count")

for j in range (i+1, len(axes)):
    fig.delaxes(axes[j])
plt.tight_layout()
plt.show()
```



[281]:
```
#Here Is The Graphs Countplot
```

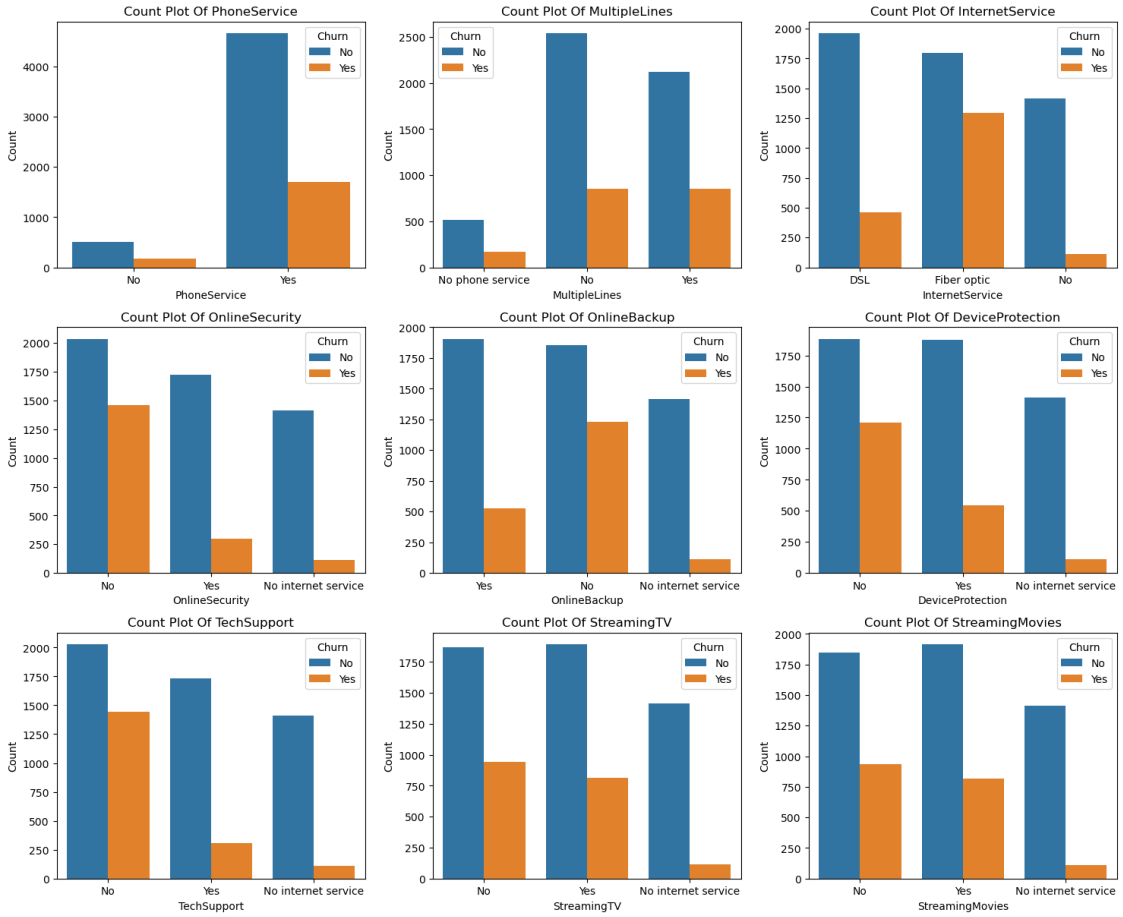
[282]:
```
#Countplots on the basis comparison using hue
```

[283]:
```
columns=["PhoneService","MultipleLines","InternetService","OnlineSecurity","OnlineBackup","Dev
n_cols=3
n_rows=(len(columns)+n_cols-1)
fig,axes=plt.subplots(n_rows,n_cols,figsize=(15,n_rows*4))
axes=axes.flatten()
for i, col in enumerate(columns):
    sns.countplot(x=col,data=churn,ax=axes[i],hue=churn["Churn"])
    axes[i].set_title(f"Count Plot Of {col}")

    axes[i].set_xlabel(col)
    axes[i].set_ylabel("Count")

for j in range (i+1, len(axes)):
    fig.delaxes(axes[j])
```

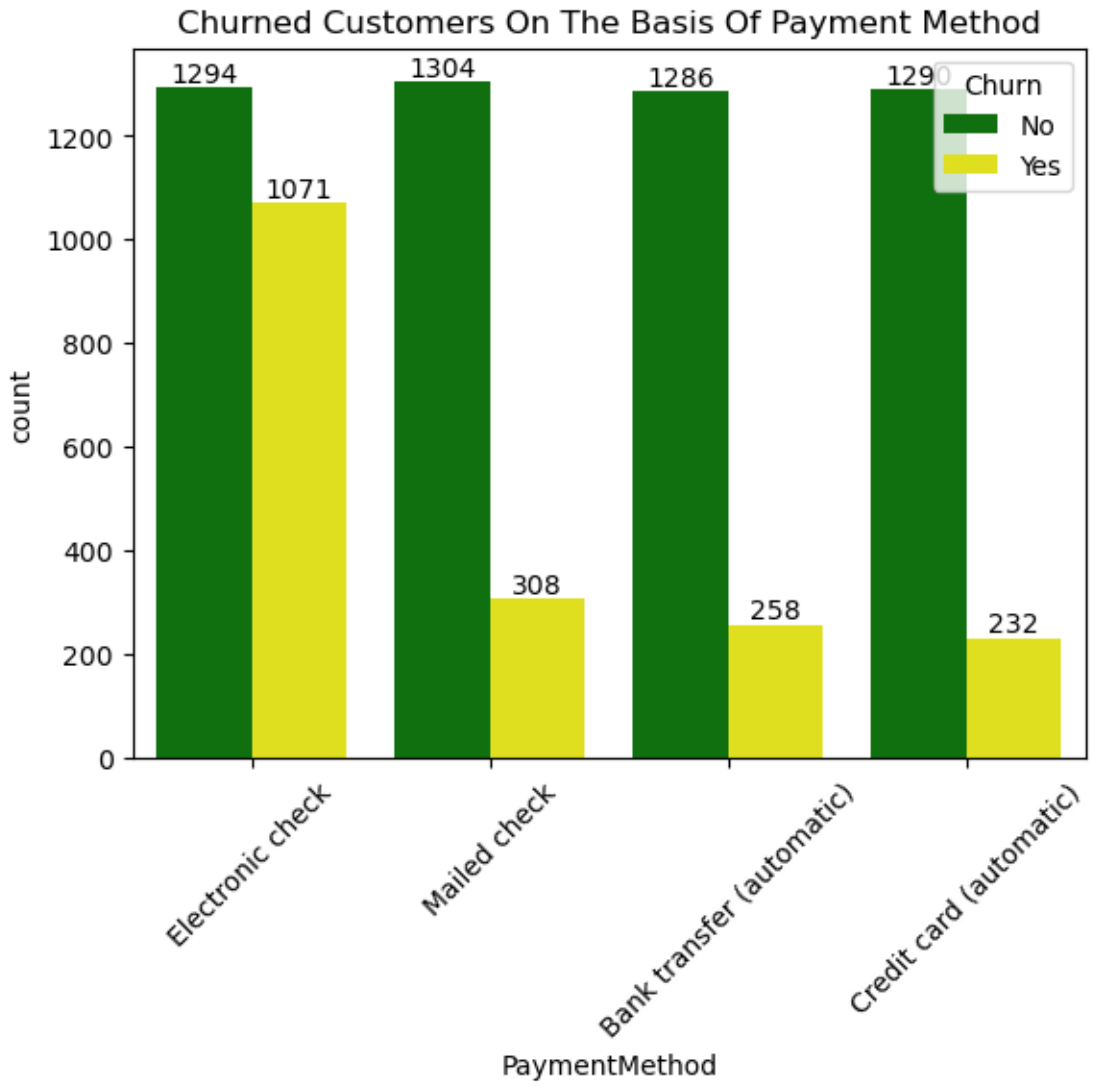
```
plt.tight_layout()
plt.show()
```



[284]:
```
#Here is the comparison graph table
```

[285]:
```
#At Last Lets See The Churned Customer On The Basis Of Payment Methods
```

[286]:
```
cp=sns.countplot(x="PaymentMethod",␣
  ↪data=churn,hue="Churn",palette=["green","yellow"])
plt.xticks(rotation=45)
plt.title("Churned Customers On The Basis Of Payment Method")
cp.bar_label(cp.containers[0])
cp.bar_label(cp.containers[1])
plt.show()
```



[287]:
```
#Here The Final Comparison Table On The Basis Of Payemnt Method
#That Tells That Customers Are Churn When They Using Electic Check As A Payment␣
 ↪Method!!!
```