jxj2addwb

February 26, 2025

[6]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv("C:/Users/rekha/Downloads/WA_Fn-UseC_-Telco-Customer-Churn.
 ↪csv")
df
```

[6]:

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

	PhoneService	MultipleLines	InternetService	OnlineSecurity	…	\
0	No	No phone service	DSL	No	…	
1	Yes	No	DSL	Yes	…	
2	Yes	No	DSL	Yes	…	
3	No	No phone service	DSL	Yes	…	
4	Yes	No	Fiber optic	No	…	
…	…	…	…	…	…	
7038	Yes	Yes	DSL	Yes	…	
7039	Yes	Yes	Fiber optic	No	…	
7040	No	No phone service	DSL	Yes	…	
7041	Yes	Yes	Fiber optic	No	…	
7042	Yes	No	Fiber optic	Yes	…	

	DeviceProtection	TechSupport	StreamingTV	StreamingMovies	Contract	\
0	No	No	No	No	Month-to-month	

```
1                 Yes          No          No              No        One year
2                  No          No          No              No  Month-to-month
3                 Yes         Yes          No              No        One year
4                  No          No          No              No  Month-to-month
…                 …           …           …               …              …
7038              Yes         Yes         Yes             Yes        One year
7039              Yes          No         Yes             Yes        One year
7040               No          No          No              No  Month-to-month
7041               No          No          No              No  Month-to-month
7042              Yes         Yes         Yes             Yes        Two year

     PaperlessBilling              PaymentMethod MonthlyCharges  TotalCharges  \
0                 Yes           Electronic check          29.85         29.85
1                  No               Mailed check          56.95        1889.5
2                 Yes               Mailed check          53.85        108.15
3                  No  Bank transfer (automatic)          42.30       1840.75
4                 Yes           Electronic check          70.70        151.65
…                 …                          …             …           …
7038              Yes               Mailed check          84.80        1990.5
7039              Yes    Credit card (automatic)         103.20        7362.9
7040              Yes           Electronic check          29.60        346.45
7041              Yes               Mailed check          74.40         306.6
7042              Yes  Bank transfer (automatic)         105.65        6844.5

     Churn
0       No
1       No
2      Yes
3       No
4      Yes
…    …
7038    No
7039    No
7040    No
7041   Yes
7042    No

[7043 rows x 21 columns]
```

[8]:
```
df.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
```

[10]:
```
df.head(3)
```

[10]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	…	DeviceProtection	TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling	PaymentMethod	MonthlyCharges	TotalCharges	Churn
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	…	No	No	No	No	Month-to-month	Yes	Electronic check	29.85	29.85	No
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	…	Yes	No	No	No	One year	No	Mailed check	56.95	1889.5	No
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	…	No	No	No	No	Month-to-month	Yes	Mailed check	53.85	108.15	Yes

[3 rows x 21 columns]

[12]:
```
df['TotalCharges'] = df['TotalCharges'].replace(" " ,"0")
df['TotalCharges'] = df['TotalCharges'].astype("float")
```

[14]:
```
df.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
```

[16]:
```
def conv(value):
    if value == 1:
        return "yes"
    else :
        return "no"

df['SeniorCitizen'] = df['SeniorCitizen'].apply(conv)
```

[18]:
```
df.head(4)
```

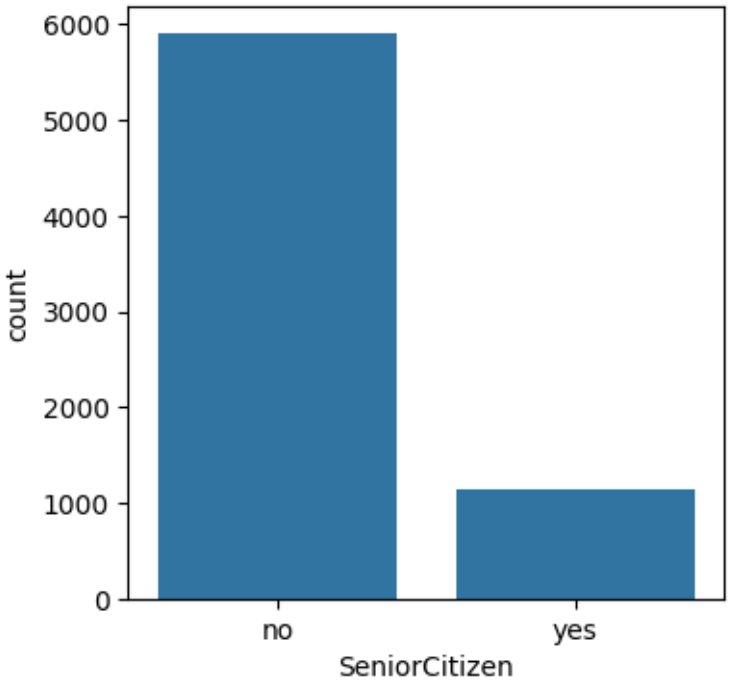
[18]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	\
0	7590-VHVEG	Female	no	Yes	No	1	No	
1	5575-GNVDE	Male	no	No	No	34	Yes	

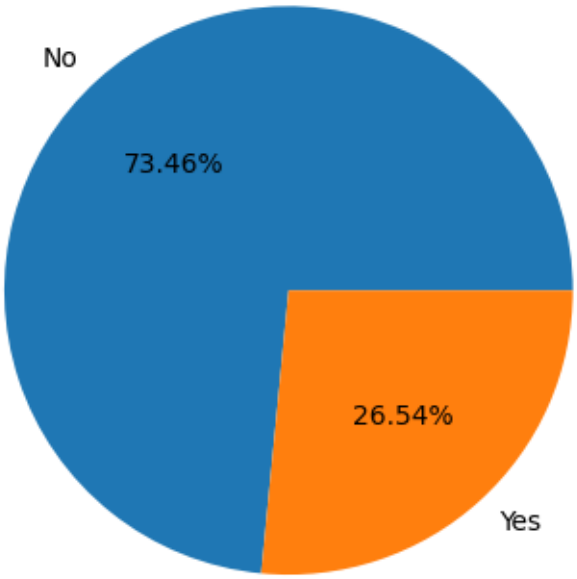
```
2  3668-QPYBK    Male            no      No         No       2          Yes
3  7795-CFOCW    Male            no      No         No      45           No

      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

  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

               PaymentMethod MonthlyCharges  TotalCharges  Churn
0           Electronic check          29.85         29.85     No
1               Mailed check          56.95       1889.50     No
2               Mailed check          53.85        108.15    Yes
3  Bank transfer (automatic)          42.30       1840.75     No

[4 rows x 21 columns]
```

[20]:
```
plt.figure(figsize = (4,4))
ax = sns.countplot(x = "SeniorCitizen" ,data =df)
ax.bar_labels(ax.containers[0])
ax.title("churn of customer data")
plt.show()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[20], line 3
      1 plt.figure(figsize = (4,4))
      2 ax = sns.countplot(x = "SeniorCitizen" ,data =df)
----> 3 ax.bar_labels(ax.containers[0])
      4 ax.title("churn of customer data")
      5 plt.show()

AttributeError: 'Axes' object has no attribute 'bar_labels'
```



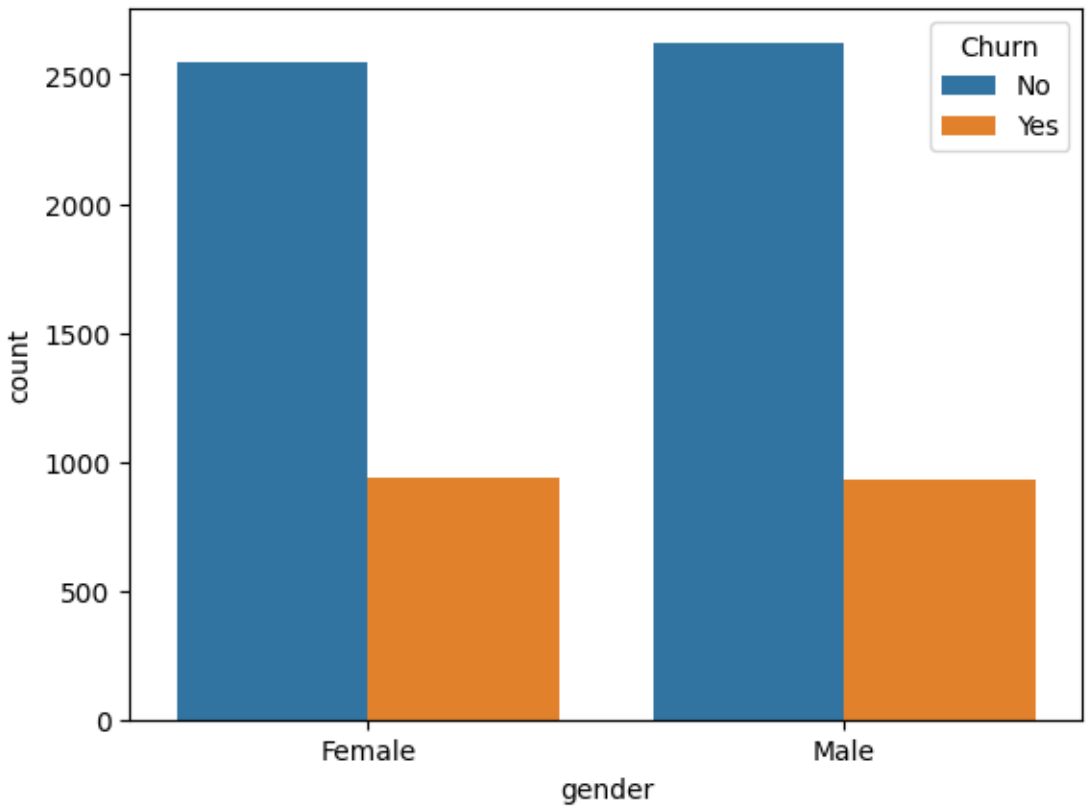
[46]:
```
gb = df.groupby("Churn").agg({'Churn' : "count"})
plt.pie(gb['Churn'] , labels = gb.index , autopct = "%1.2f%%")
plt.show()
gb
```



[46]:

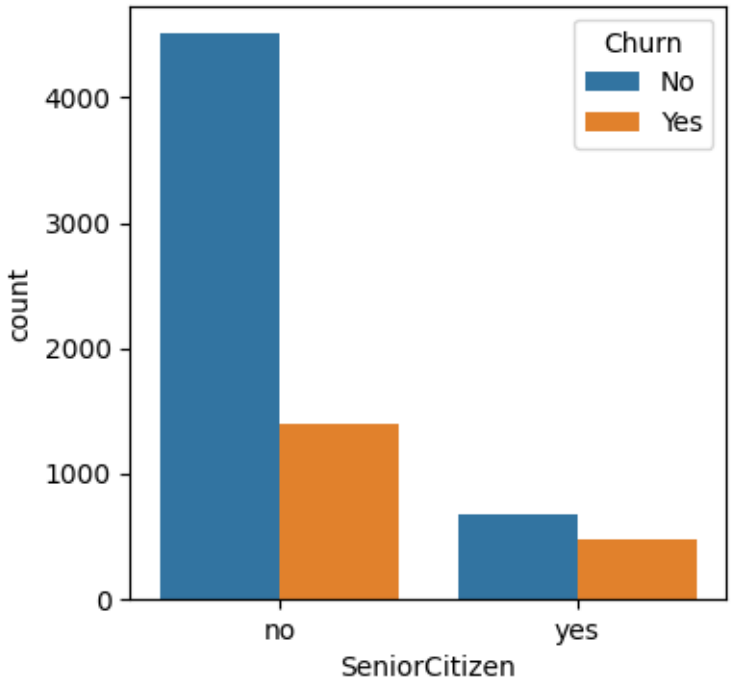
Churn	Churn
No	5174
Yes	1869

[54]:
```
sns.countplot( x ="gender" , data = df, hue = "Churn")
plt.show()
```

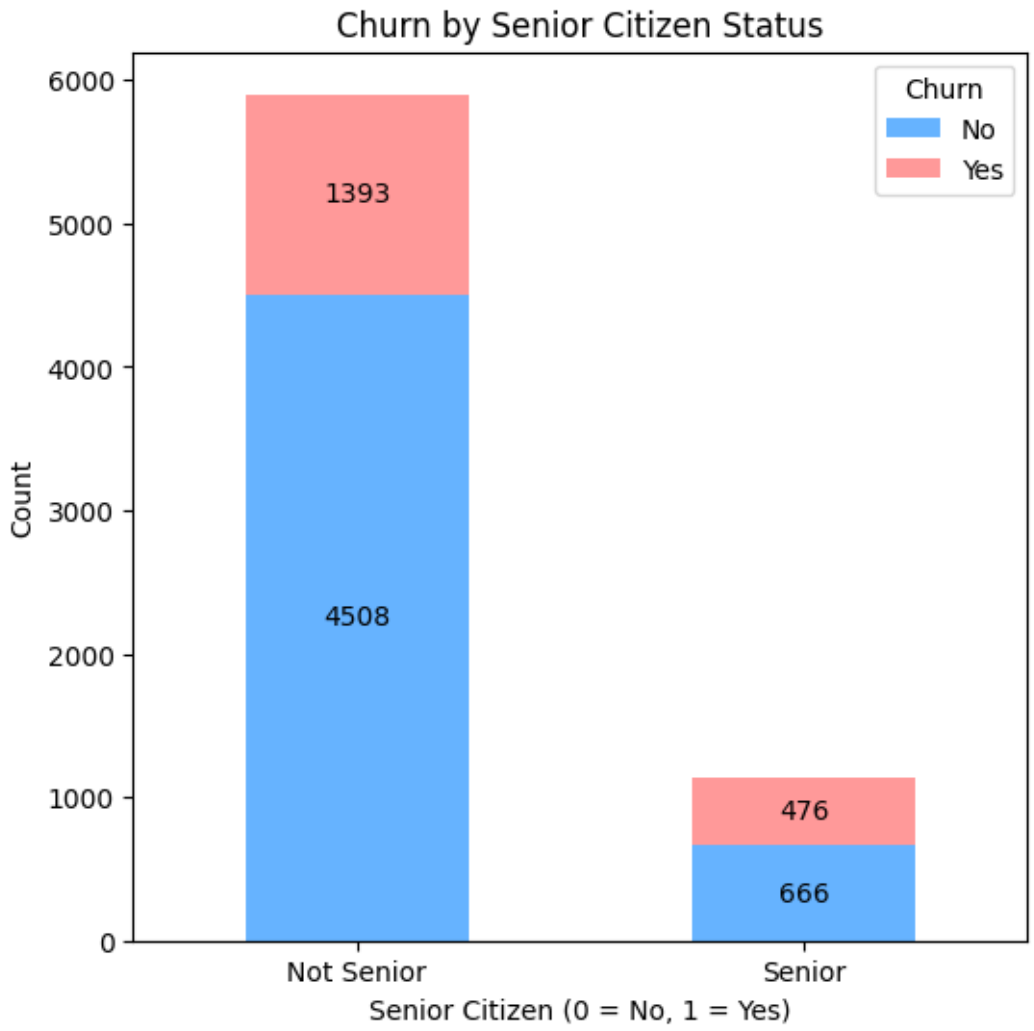


[56]:
```
plt.figure(figsize = (4,4))
ax = sns.countplot(x = "SeniorCitizen" ,data =df , hue="Churn")
ax.title("churn by SeniorCitizen")
plt.show()
```

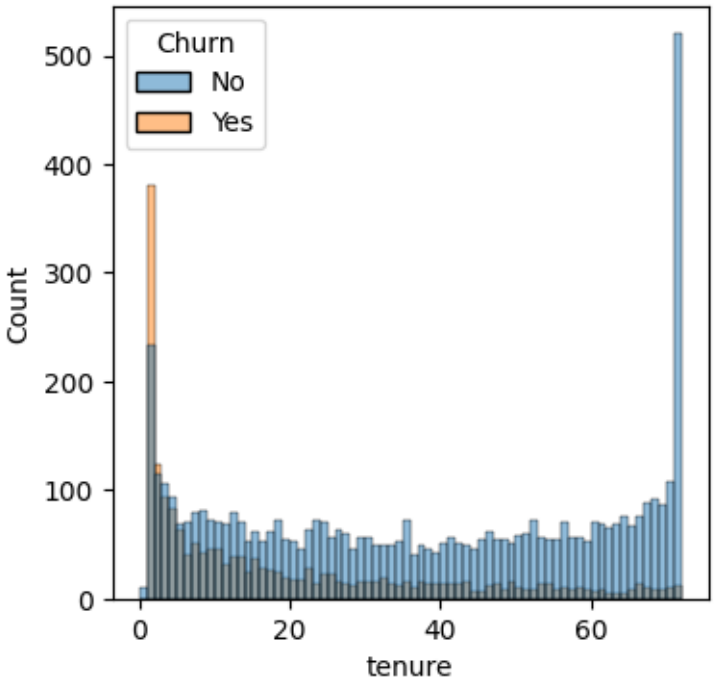
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[56], line 3
      1 plt.figure(figsize = (4,4))
      2 ax = sns.countplot(x = "SeniorCitizen" ,data =df , hue="Churn")
----> 3 ax.title("churn by SeniorCitizen")
      4 plt.show()

TypeError: 'Text' object is not callable
```



[58]:
```
# Creating a pivot table for stacked bar chart
senior_churn_counts = pd.crosstab(df['SeniorCitizen'], df['Churn'])

# Plotting the stacked bar chart
ax = senior_churn_counts.plot(kind='bar', stacked=True, figsize=(6, 6),␣
 ↪color=['#66b3ff', '#ff9999'])

# Adding labels
for container in ax.containers:
    ax.bar_label(container, fmt='%d', label_type='center')  # Adds labels␣
 ↪inside bars

# Formatting the chart
plt.title("Churn by Senior Citizen Status")
plt.xlabel("Senior Citizen (0 = No, 1 = Yes)")
plt.ylabel("Count")
plt.xticks(ticks=[0, 1], labels=["Not Senior", "Senior"], rotation=0)
plt.legend(title="Churn", labels=["No", "Yes"])
plt.show()
```

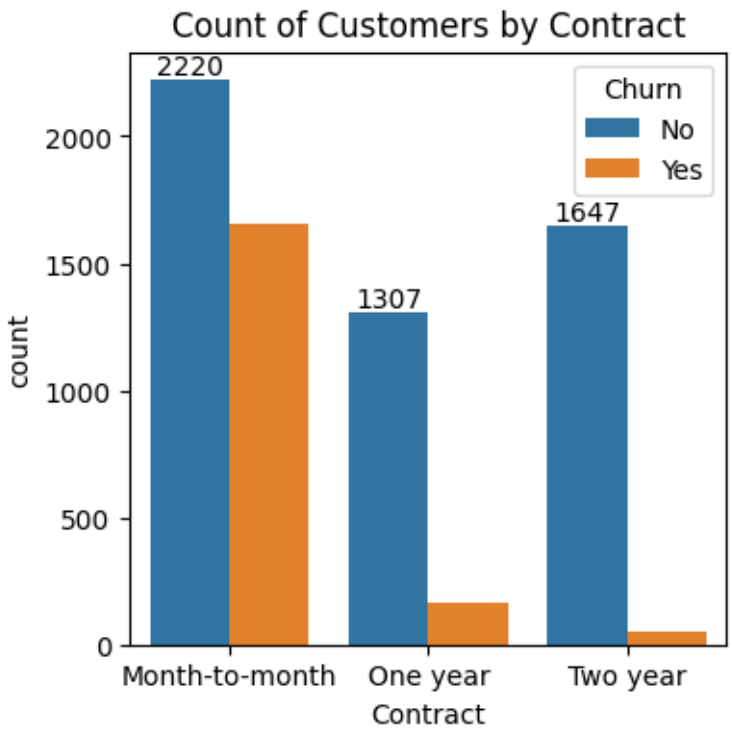


[66]:
```
plt.figure(figsize = (4,4))
sns.histplot(x = "tenure" , data=df , bins=72, hue = "Churn")
plt.show()
```



people who have used our services for a long time have stayed and people who have used our sevices

[69]:
```
plt.figure(figsize = (4,4))
ax = sns.countplot(x = "Contract", data = df, hue = "Churn")
ax.bar_label(ax.containers[0])
plt.title("Count of Customers by Contract")
plt.show()
```



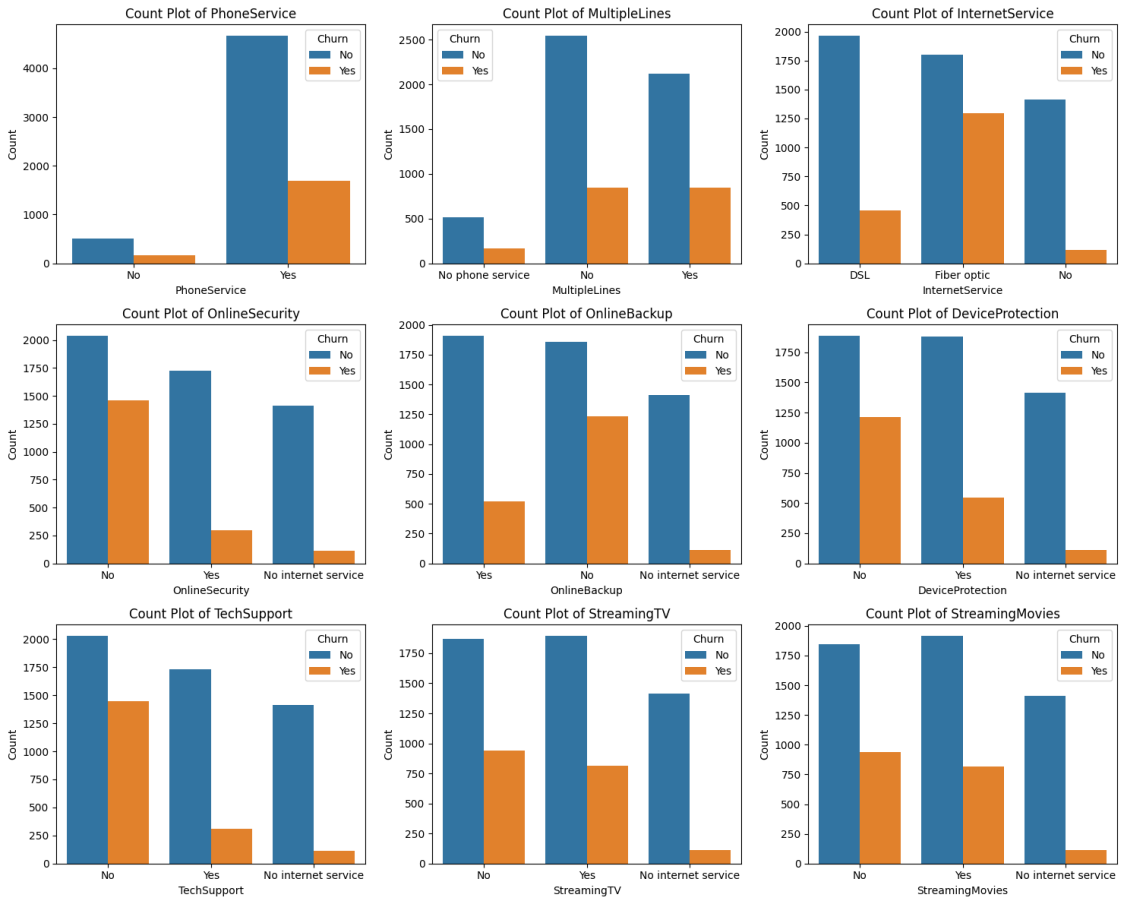
#people who have month to month contract are likely to churn then from those who have 1 or 2 years or contract.

[72]:
```
df.columns.values
```

[72]:
```
array(['customerID', 'gender', 'SeniorCitizen', 'Partner', 'Dependents',
       'tenure', 'PhoneService', 'MultipleLines', 'InternetService',
       'OnlineSecurity', 'OnlineBackup', 'DeviceProtection',
       'TechSupport', 'StreamingTV', 'StreamingMovies', 'Contract',
       'PaperlessBilling', 'PaymentMethod', 'MonthlyCharges',
       'TotalCharges', 'Churn'], dtype=object)
```

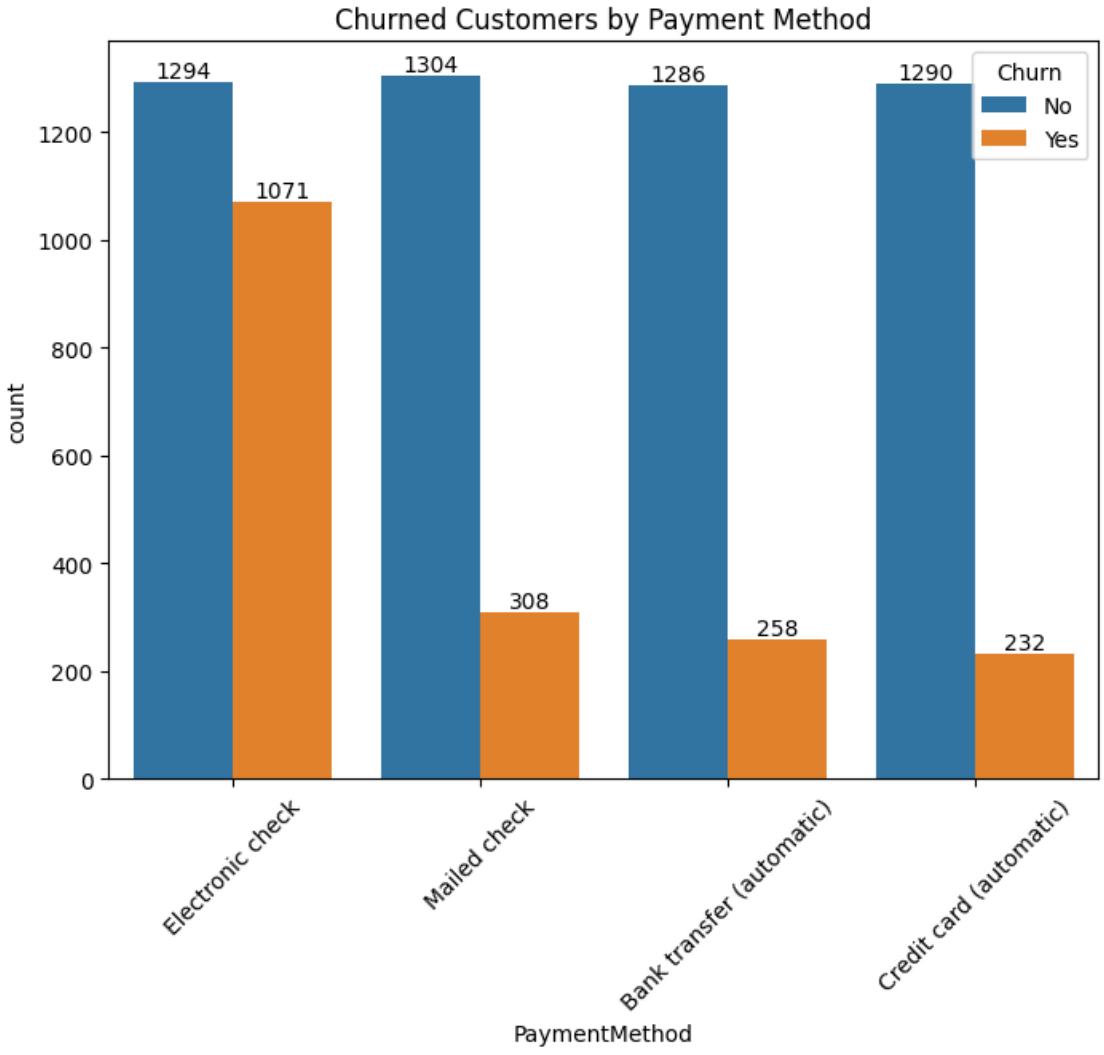
[74]:
```
columns = ['PhoneService', 'MultipleLines', 'InternetService', 'OnlineSecurity',
           'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingTV',␣
 ↪'StreamingMovies']

# Number of columns for the subplot grid (you can change this)
n_cols = 3
n_rows = (len(columns) + n_cols - 1) // n_cols  # Calculate number of rows␣
 ↪needed

# Create subplots
```

```
fig, axes = plt.subplots(n_rows, n_cols, figsize=(15, n_rows * 4))  # Adjust␣
 ↪figsize as needed

# Flatten the axes array for easy iteration (handles both 1D and 2D arrays)
axes = axes.flatten()

# Iterate over columns and plot count plots
for i, col in enumerate(columns):
    sns.countplot(x=col, data=df, ax=axes[i], hue = df["Churn"])
    axes[i].set_title(f'Count Plot of {col}')
    axes[i].set_xlabel(col)
    axes[i].set_ylabel('Count')

# Remove empty subplots (if any)
for j in range(i + 1, len(axes)):
    fig.delaxes(axes[j])

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



The majority of customers who do not churn tend to have services like PhoneService, InternetService (particularly DSL), and OnlineSecurity enabled. For services like OnlineBackup, TechSupport, and StreamingTV, churn rates are noticeably higher when these services are not used or are unavailable.

[81]:
```
plt.figure(figsize = (8,6))
ax = sns.countplot(x = "PaymentMethod", data = df, hue = "Churn")
ax.bar_label(ax.containers[0])
ax.bar_label(ax.containers[1])
plt.title("Churned Customers by Payment Method")
plt.xticks(rotation = 45)
plt.show()
```



[ ]: