telco-customer-churn-analysis

November 2, 2024

[1]:
```
pip install seaborn
```

```
Collecting seaborn
  Downloading seaborn-0.13.2-py3-none-any.whl.metadata (5.4 kB)
Requirement already satisfied: numpy!=1.24.0,>=1.20 in
c:\users\az682\appdata\local\programs\python\python311\lib\site-packages (from
seaborn) (1.25.2)
Requirement already satisfied: pandas>=1.2 in
c:\users\az682\appdata\local\programs\python\python311\lib\site-packages (from
seaborn) (2.2.3)
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in
c:\users\az682\appdata\local\programs\python\python311\lib\site-packages (from
seaborn) (3.7.2)
Requirement already satisfied: contourpy>=1.0.1 in
c:\users\az682\appdata\local\programs\python\python311\lib\site-packages (from
matplotlib!=3.6.1,>=3.4->seaborn) (1.1.0)
Requirement already satisfied: cycler>=0.10 in
c:\users\az682\appdata\local\programs\python\python311\lib\site-packages (from
matplotlib!=3.6.1,>=3.4->seaborn) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in
c:\users\az682\appdata\local\programs\python\python311\lib\site-packages (from
matplotlib!=3.6.1,>=3.4->seaborn) (4.42.1)
Requirement already satisfied: kiwisolver>=1.0.1 in
c:\users\az682\appdata\local\programs\python\python311\lib\site-packages (from
matplotlib!=3.6.1,>=3.4->seaborn) (1.4.5)
Requirement already satisfied: packaging>=20.0 in
c:\users\az682\appdata\local\programs\python\python311\lib\site-packages (from
matplotlib!=3.6.1,>=3.4->seaborn) (23.1)
Requirement already satisfied: pillow>=6.2.0 in
c:\users\az682\appdata\local\programs\python\python311\lib\site-packages (from
matplotlib!=3.6.1,>=3.4->seaborn) (10.0.0)
Requirement already satisfied: pyparsing<3.1,>=2.3.1 in
c:\users\az682\appdata\local\programs\python\python311\lib\site-packages (from
matplotlib!=3.6.1,>=3.4->seaborn) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in
c:\users\az682\appdata\local\programs\python\python311\lib\site-packages (from
matplotlib!=3.6.1,>=3.4->seaborn) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in
```

```
c:\users\az682\appdata\local\programs\python\python311\lib\site-packages (from
pandas>=1.2->seaborn) (2023.3)
Requirement already satisfied: tzdata>=2022.7 in
c:\users\az682\appdata\local\programs\python\python311\lib\site-packages (from
pandas>=1.2->seaborn) (2023.3)
Requirement already satisfied: six>=1.5 in
c:\users\az682\appdata\local\programs\python\python311\lib\site-packages (from
python-dateutil>=2.7->matplotlib!=3.6.1,>=3.4->seaborn) (1.16.0)
Downloading seaborn-0.13.2-py3-none-any.whl (294 kB)
Installing collected packages: seaborn
Successfully installed seaborn-0.13.2
Note: you may need to restart the kernel to use updated packages.


[notice] A new release of pip is available: 24.2 -> 24.3.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

[2]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

Creating a dataframe

[3]:
```
df=pd.read_csv("Customer Churn.csv")
df
```

[3]:

	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]
```

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

[4]:
```
   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

      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

  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

                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 rows x 21 columns]
```

[5]:
```
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
```

Replace the empty value of totalcharge with 0 as tanure is 0

[3]:
```
df["TotalCharges"]=df["TotalCharges"].replace(' ','0') #replace empty values␣
 ↪with 0
df["TotalCharges"]=df["TotalCharges"].astype("float")  #change the datatype of␣
 ↪the colume TotalCharge
```

[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   float64
```

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

[12]:
```
df.isnull().sum()  #Give the sum of the null values in each column
```

[12]:
```
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
```

[13]:
```
df.describe()
```

[13]:

	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

[16]:
```
df.duplicated() #check the dublicated value in each row
```

[16]:
```
0       False
1       False
2       False
3       False
```

```
4       False
        …
7038    False
7039    False
7040    False
7041    False
7042    False
Length: 7043, dtype: bool
```

[19]:
```
df["customerID"].duplicated().sum() #sum of the duplicated values in customerID column
```

[19]:
```
0
```

To convert values of 0 and 1 in SeniorCitizen column into yes and no

[5]:
```
def conv(value):
    if value == 1:
        return "Yes"
    else:
        return "No"


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

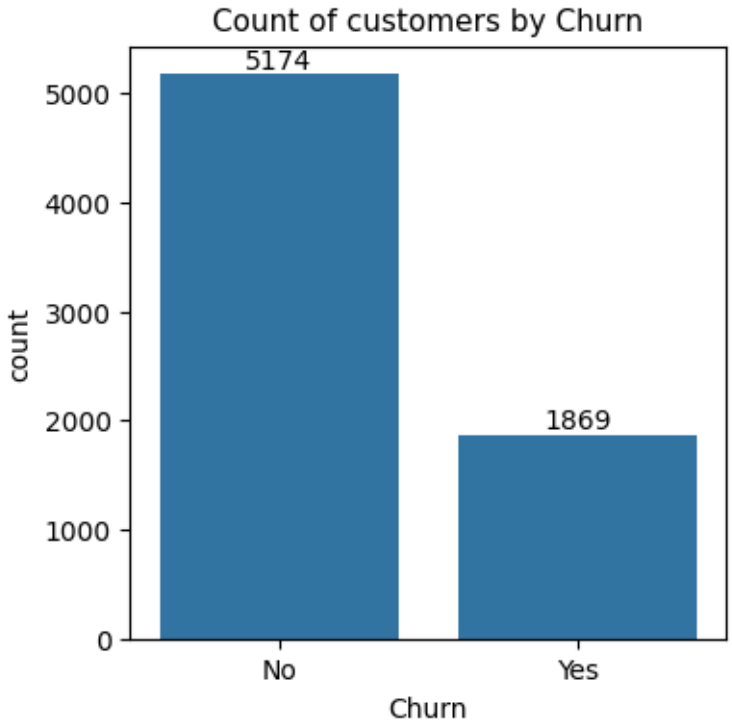
[21]:
```
df.head()
```

[21]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	…	DeviceProtection	TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling
0	7590-VHVEG	Female	No	Yes	No	1	No	…	No	No	No	No	Month-to-month	Yes
1	5575-GNVDE	Male	No	No	No	34	Yes	…	Yes	No	No	No	One year	No
2	3668-QPYBK	Male	No	No	No	2	Yes	…	No	No	No	No	Month-to-month	Yes
3	7795-CFOCW	Male	No	No	No	45	No	…	Yes	Yes	No	No	One year	No
4	9237-HQITU	Female	No	No	No	2	Yes	…	No	No	No	No	Month-to-month	Yes

```
              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          Electronic check          70.70        151.65    Yes

[5 rows x 21 columns]
```

[8]:
```
plt.figure(figsize = (4,4))
ax=sns.countplot(x="Churn", data=df)
ax.bar_label(ax.containers[0]) #to add labels above the bars
plt.title("Count of customers by Churn", fontsize=11)
plt.show()
```

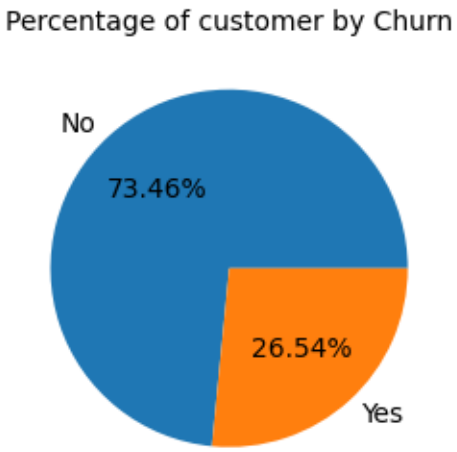


[28]:
```
#group data by chrun value count
gp = df.groupby("Churn").agg({"Churn":"count"})
gp
```

[28]:
```
       Churn
Churn
No      5174
```

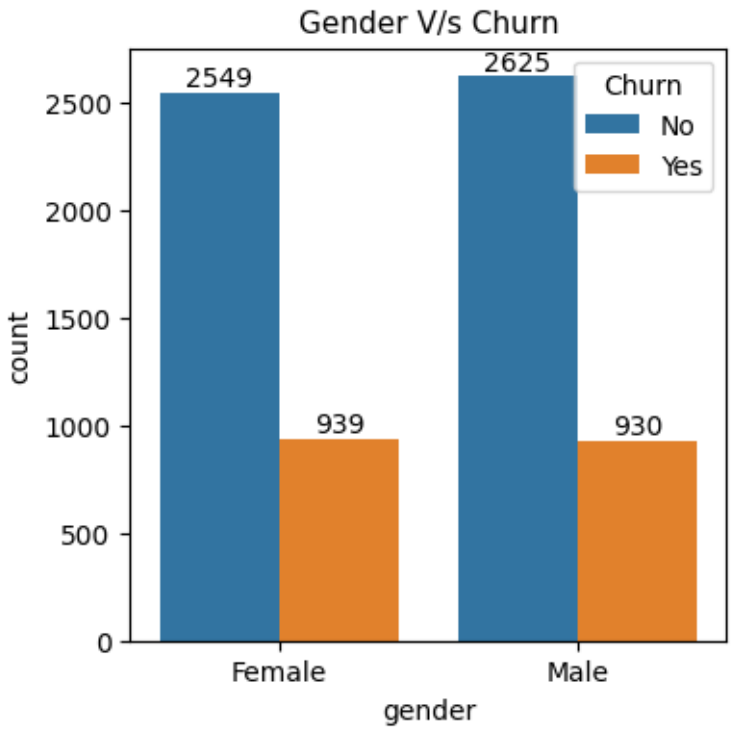
```
Yes     1869
```

[40]:
```
# to make pie chart using group by
plt.figure(figsize = (3,4)) #to change to size fo the graph
plt.pie(gp["Churn"], labels=gp.index, autopct = "%1.2f%%") #labels for (yes,␣
 ↪no) autopct for(percentage)
plt.title("Percentage of customer by Churn", fontsize=10)
plt.show()
```



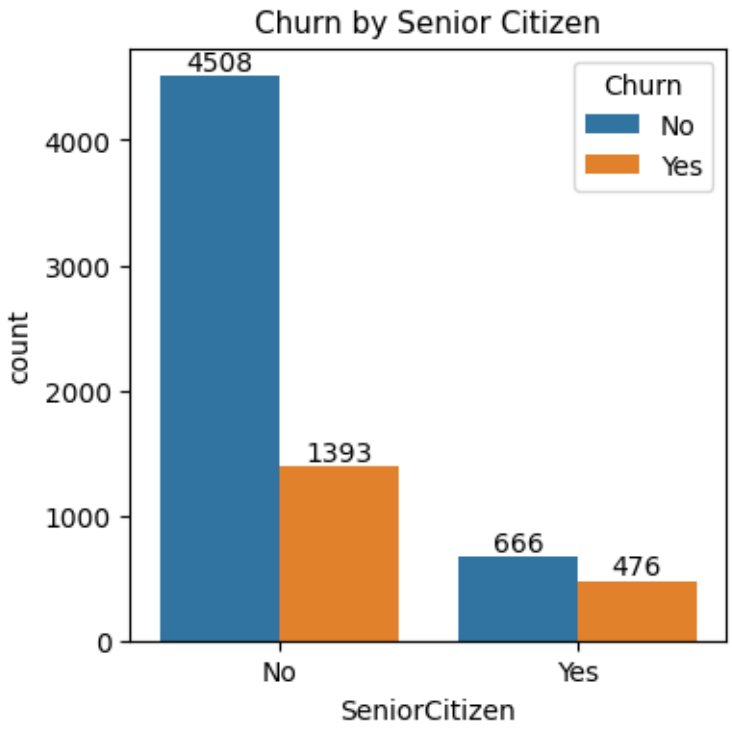
average churn % = 26.54%

[45]:
```
plt.figure(figsize=(4, 4))
ax = sns.countplot(x="gender", data=df, hue="Churn")

# Loop over each container and add labels
for container in ax.containers:
    ax.bar_label(container)

plt.title("Gender V/s Churn", fontsize=11)
plt.show()
```



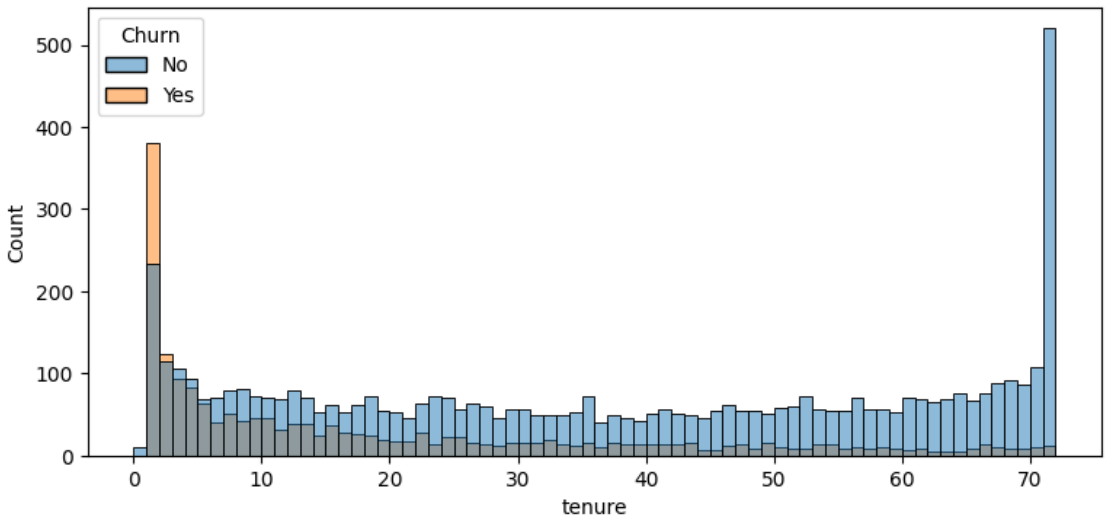
From the above graph, it is clear that gender have not a huge impact on churn.

[50]:
```
plt.figure(figsize = (4,4))
ax=sns.countplot(x = "SeniorCitizen", data=df, hue = "Churn")
for container in ax.containers:
    ax.bar_label(container)
plt.title("Churn by Senior Citizen", fontsize=11)
plt.show()
```



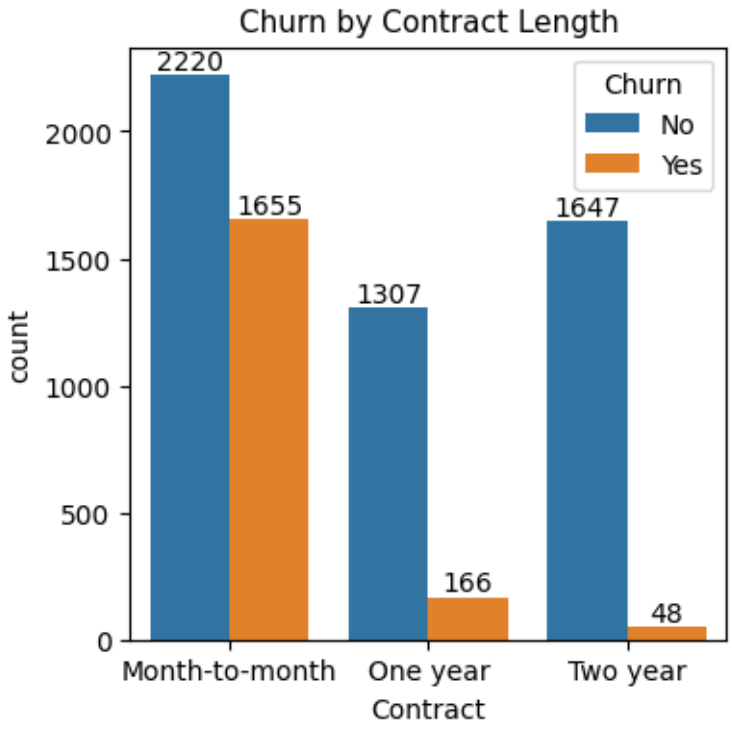
From the above graph, it is clear that older people have churned more.
Churn % of Customers who are Senior Citizen = 41.68.
Churn % of Customers who are not Senior Citizen = 23.61

[53]:
```
plt.figure(figsize = (9,4))
sns.histplot(x = "tenure", data = df, hue = "Churn", bins = 72)
plt.show()
```



From the above graph, it is clear that more people are churning in the initial months

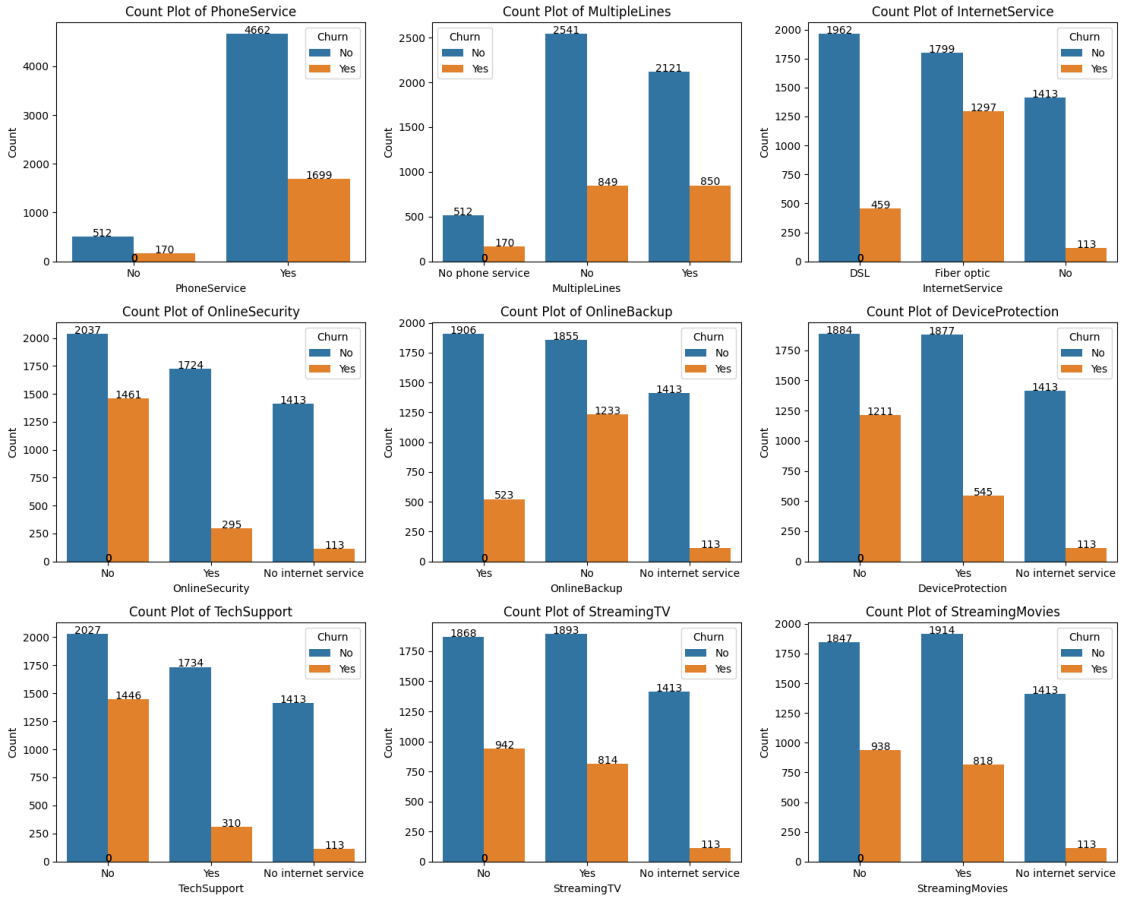
[10]:
```
plt.figure(figsize = (4,4))
ax=sns.countplot(x = "Contract", data=df, hue = "Churn")
for container in ax.containers:
    ax.bar_label(container)
plt.title("Churn by Contract Length", fontsize=11)
plt.show()
```



From the above graph, it is clear that people who have month to month contract more likely to churn

[33]:
```
columns = ['PhoneService', 'MultipleLines', 'InternetService', 'OnlineSecurity',
           'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingTV',␣
 ↪'StreamingMovies']

n_cols = 3
n_rows = (len(columns) + n_cols - 1) // n_cols

# Create subplots
fig, axes = plt.subplots(n_rows, n_cols, figsize=(15, n_rows * 4))
```

```
axes = axes.flatten()

# Iterate over columns and plot count plots
for i, col in enumerate(columns):
    plot = 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')

    # Add values on top of each bar
    for p in plot.patches:
        # Get the height of the bar and use it to set the position for the text
        height = p.get_height()
        plot.text(
            p.get_x() + p.get_width() / 2,  # x position of the label
            height + 1,                     # y position of the label, slightly␣
 ↪above the bar
            f'{int(height)}',               # label text (converted to integer)
            ha="center"                     # center alignment
        )

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

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



People who are using services like Multiple lines, phone services and fiber optic are more likely to churn and most people who are not using the services like online security, online backup, device protection, tech support are more likely to churn and services like steaming tv and movies have similar chances of churning

[5]:
```
df.head()
```

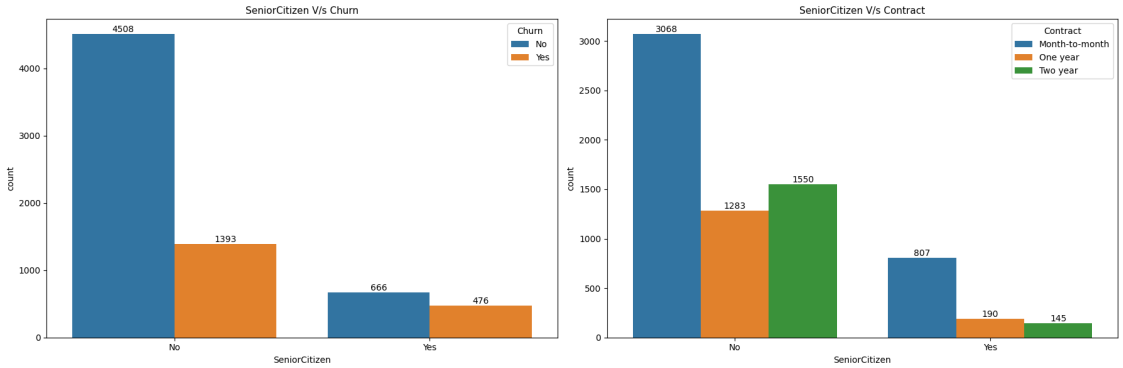
[5]:

	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	
4	9237-HQITU	Female	No	No	No	2	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	

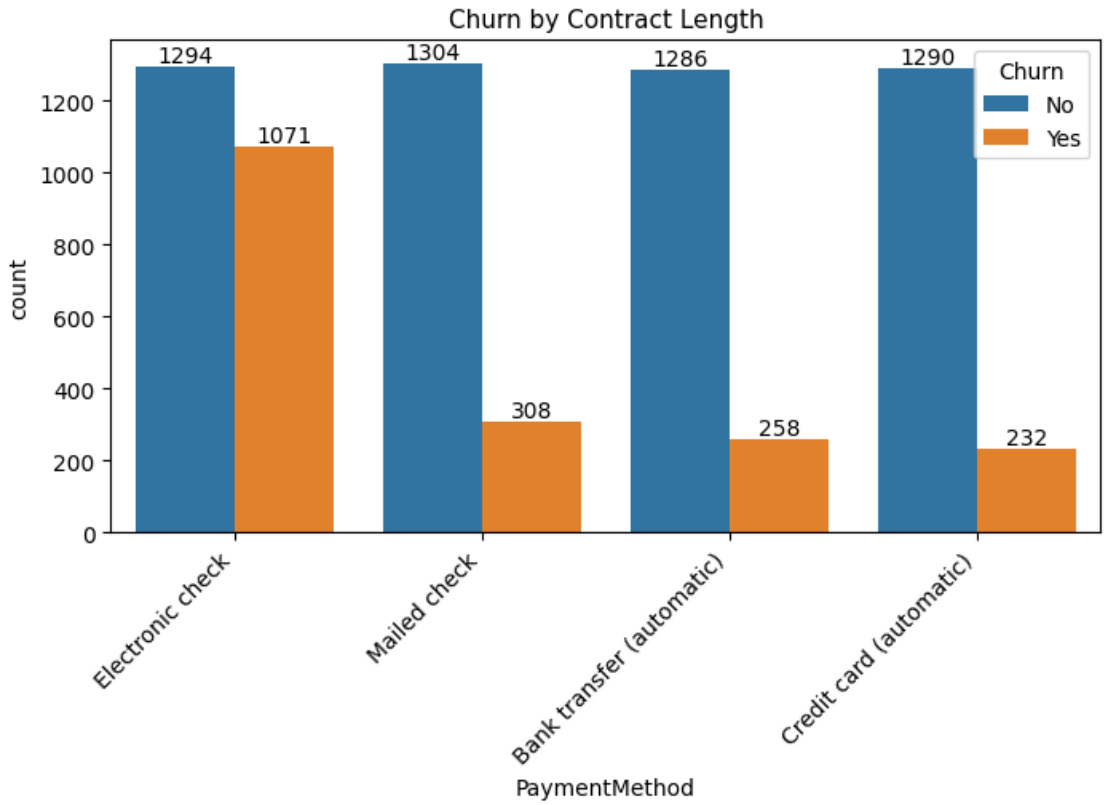
```
4                No     Fiber optic             No  …               No

  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

               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           Electronic check          70.70        151.65   Yes

[5 rows x 21 columns]
```

[26]:
```python
# Create a figure with 1 row and 3 columns, setting a larger figure size
fig, axes = plt.subplots(1, 2, figsize=(18, 6))  # Adjust the figsize as needed


ax1 = sns.countplot(x="SeniorCitizen", data=df, hue="Churn", ax=axes[0])
for container in ax1.containers:
    ax1.bar_label(container)
ax1.set_title("SeniorCitizen V/s Churn", fontsize=11)


ax2 = sns.countplot(x="SeniorCitizen", data=df, hue="Contract", ax=axes[1])
for container in ax2.containers:
    ax2.bar_label(container)
ax2.set_title("SeniorCitizen V/s Contract", fontsize=11)



# Adjust layout for proper spacing
plt.tight_layout()

# Show the plots
plt.show()
```



From the above plots, it is clear that senior citizens have more month-to-month contract hence they are churning more i.e.
-month-to-month contract % in senior citizen = $(807/(807+190+145))100$ = *70.07%*
*-churn % in senior citizen = (476/(476+666))*100 = 41.17% which is higher than average churn % which is 26.54%

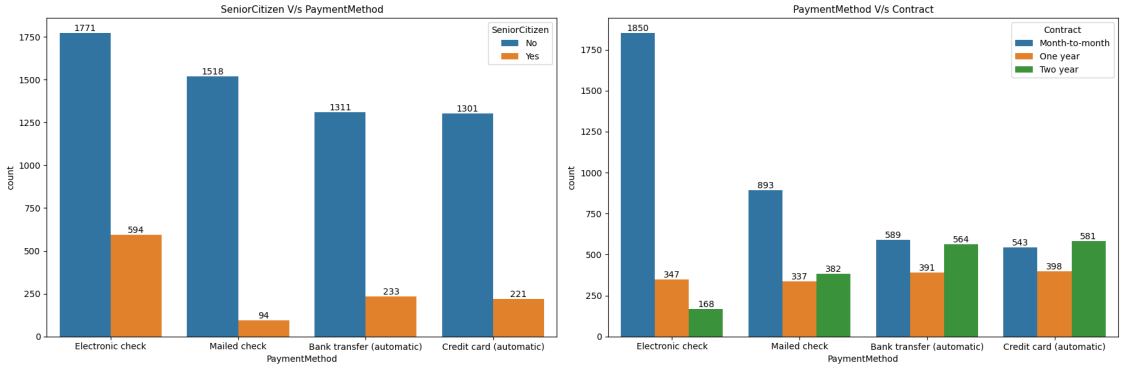
[15]:
```
plt.figure(figsize = (8,4))
ax=sns.countplot(x = "PaymentMethod", data=df, hue = "Churn")
for container in ax.containers:
    ax.bar_label(container)

plt.title("Churn by Contract Length", fontsize=11)

plt.xticks(rotation=45, ha='right')
plt.show()
```



From the above plot, people using electronic check as payment method have churned more

[32]:
```
# Create a figure with 1 row and 3 columns, setting a larger figure size
fig, axes = plt.subplots(1,2, figsize=(18, 6))  # Adjust the figsize as needed

# Plot 1: Churn by Contract Length for Senior Citizens
ax1 = sns.countplot(x="PaymentMethod", data=df, hue="SeniorCitizen", ax=axes[0])
for container in ax1.containers:
    ax1.bar_label(container)
ax1.set_title("SeniorCitizen V/s PaymentMethod", fontsize=11)

# Plot 2: Churn by Contract Length for Senior Citizens with different hue
ax2 = sns.countplot(x="PaymentMethod", data=df, hue="Contract", ax=axes[1])
for container in ax2.containers:
    ax2.bar_label(container)
ax2.set_title("PaymentMethod V/s Contract", fontsize=11)


# Adjust layout for proper spacing
plt.tight_layout()

# Show the plots
```

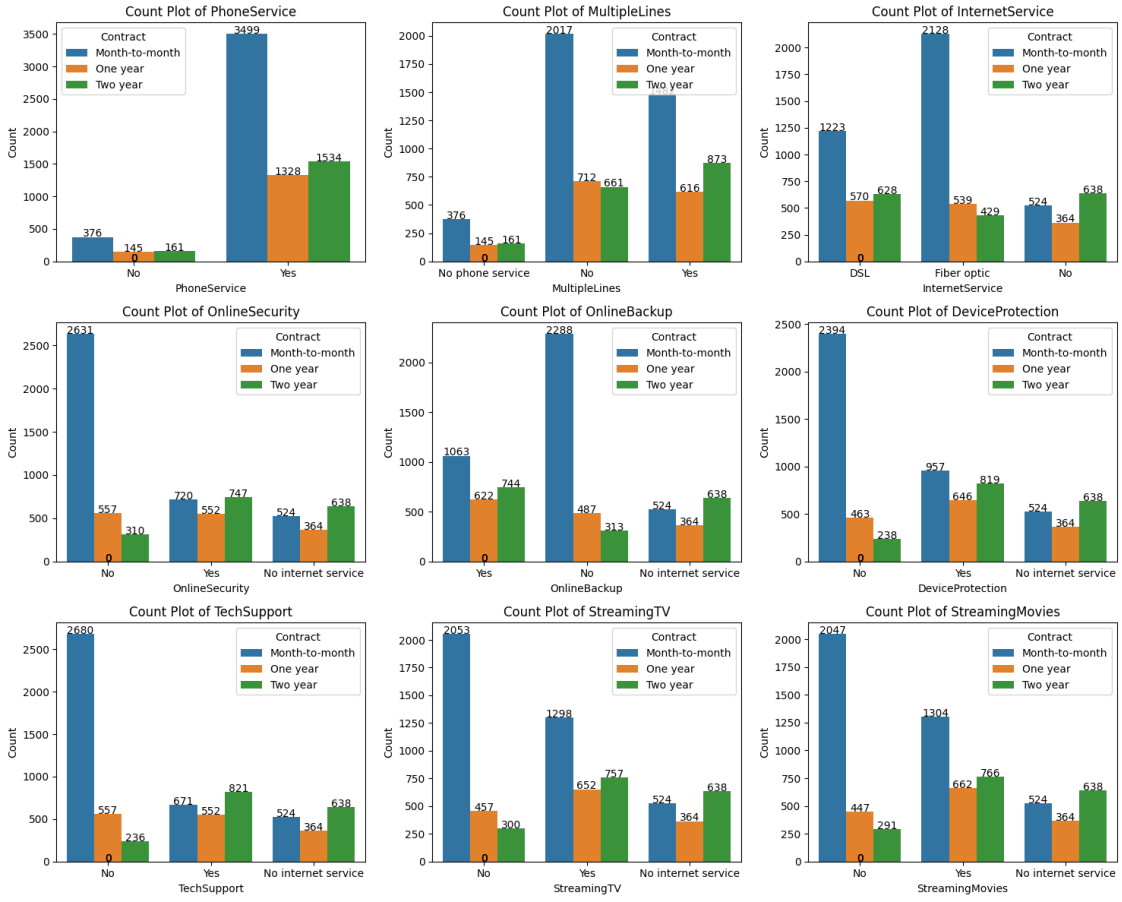
```
plt.show()
```



From the above plots , it is clear that majority or electronic check payments are month-to-month basics hence more likely to churn

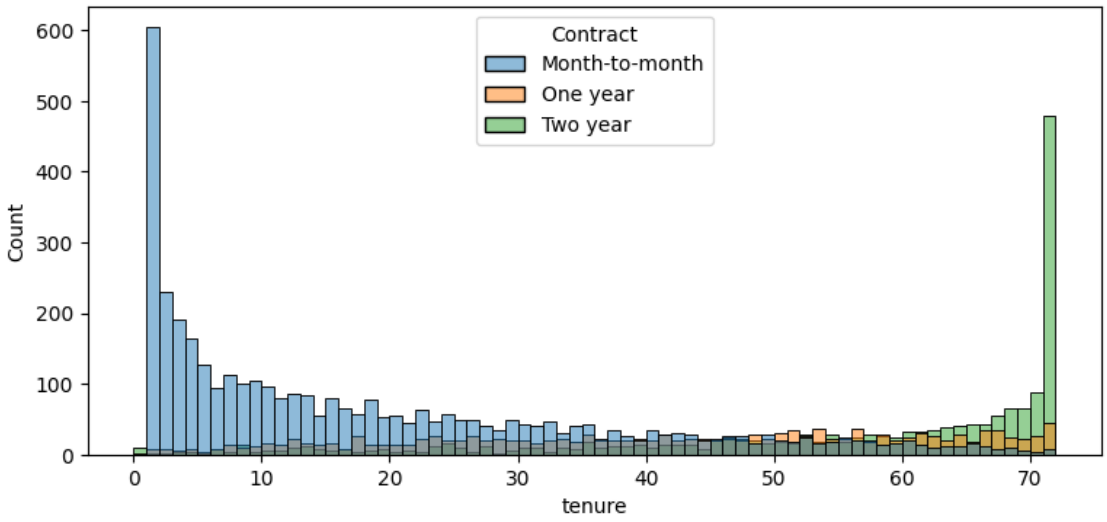
[48]:
```
df.head()



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

n_cols = 3
n_rows = (len(columns) + n_cols - 1) // n_cols

# Create subplots
fig, axes = plt.subplots(n_rows, n_cols, figsize=(15, n_rows * 4))
axes = axes.flatten()

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

    # Add values on top of each bar
    for p in plot.patches:
        # Get the height of the bar and use it to set the position for the text
        height = p.get_height()
        plot.text(
            p.get_x() + p.get_width() / 2,  # x position of the label
```

```
            height + 1,                    # y position of the label, slightly␣
 ↪above the bar
            f'{int(height)}',              # label text (converted to integer)
            ha="center"                    # center alignment
        )

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

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



From above plots , we concluded that people using services more on month-to month contract basics hence services have no issue but contract length have issue

[45]:
```
plt.figure(figsize = (9,4))
sns.histplot(x = "tenure", data = df, hue = "Contract", bins = 72)
plt.show()
```



From the above plots the customers who stayed for longer period (>60 months) have taken one or two year subscription more but in the initial months more customers have taken more month-to-months contracts and hence churn rate are higher in starting months

[ ]: