TELECOM CHURN ANALYSIS USING PYTHON

BY- VINAY JESPAL

Overview

The project aims to analyze customer churn in a telecommunications company and develop predictive models to identify at-risk customers. The ultimate goal is to provide actionable insights and recommendations to reduce churn and improve customer retention.

1. Understanding The Data

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:
```
df= pd.read_csv('Customer churn.csv')
```

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

[3]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupport	St
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	...	No	No	
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	...	Yes	No	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	...	No	No	
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes	...	Yes	Yes	
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	...	No	No	

5 rows × 21 columns

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

[4]:

	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

```
[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
```

2. Cleaning the Data

Replacing the blanks with 0 as tenure is 0 and no total charges were recorded.

[11]:
```
df["TotalCharges"]=df["TotalCharges"].replace(" ","0")
df["TotalCharges"]=df["TotalCharges"].astype("float")
```

[12]:
```
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)
```

Check for missing values in each column

[6]:
```
df.isnull().sum()
```

[6]:
```
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
```

Checking for Duplicate values in the data.

[8]:
```
print(df["customerID"].duplicated().sum())
```

```
0
```

Converting the value of Senior citizen from 1 and 0 to Yes/No for better readability and implementation on graphs.

[10]:
```
# Creating a function that converts the values 0 and 1 to Yes/No
def conv(value):
    if value==1:
        return "Yes"
    else:
        return "No"
```

[11]:
```
df["SeniorCitizen"]=df["SeniorCitizen"].apply(conv)
```

[12]:
```
df.head()
```

[12]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupport	S
0	7590-VHVEG	Female	No	Yes	No	1	No	No phone service	DSL	No	...	No	No	
1	5575-GNVDE	Male	No	No	No	34	Yes	No	DSL	Yes	...	Yes	No	
2	3668-QPYBK	Male	No	No	No	2	Yes	No	DSL	Yes	...	No	No	
3	7795-CFOCW	Male	No	No	No	45	No	No phone service	DSL	Yes	...	Yes	Yes	
4	9237-HQITU	Female	No	No	No	2	Yes	No	Fiber optic	No	...	No	No	

3. Exploratory Data Analysis(EDA)

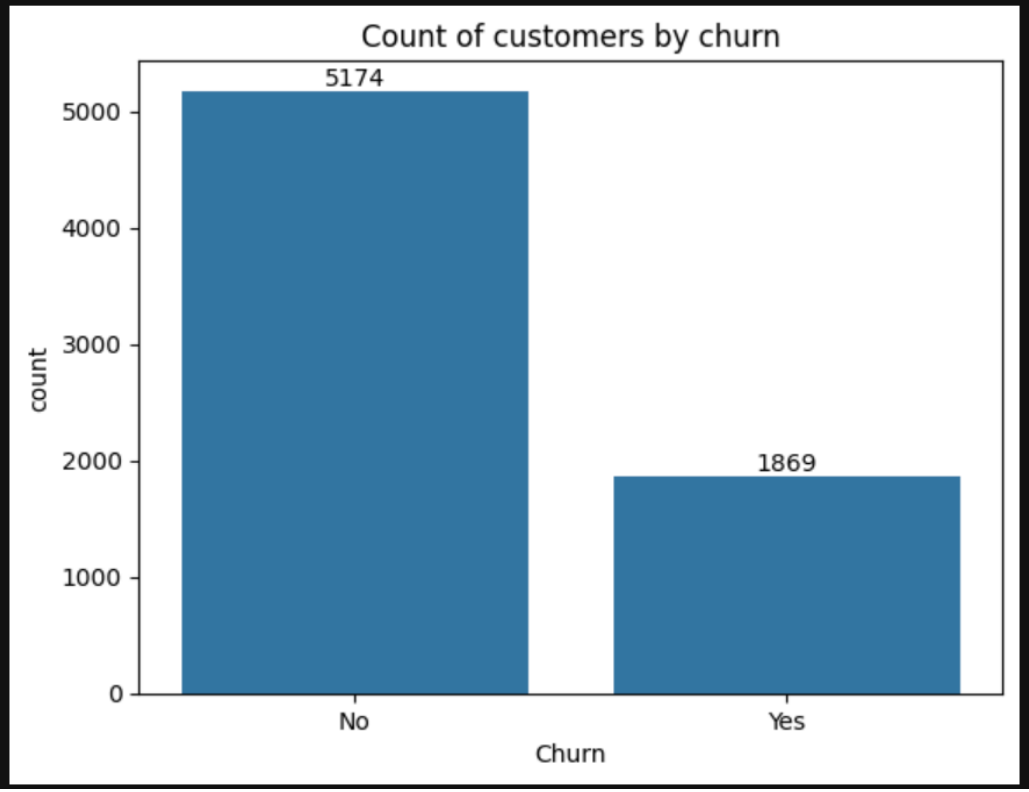
[14]:
```
# Mode of Each column
print(df.mode().iloc[0])
```

```
customerID                0002-ORFBO
gender                          Male
SeniorCitizen                     No
Partner                           No
Dependents                        No
tenure                           1.0
PhoneService                     Yes
MultipleLines                     No
InternetService          Fiber optic
OnlineSecurity                    No
OnlineBackup                      No
DeviceProtection                  No
TechSupport                       No
StreamingTV                       No
StreamingMovies                   No
Contract              Month-to-month
PaperlessBilling                 Yes
PaymentMethod       Electronic check
MonthlyCharges                 20.05
TotalCharges
Churn                             No
Name: 0, dtype: object
```

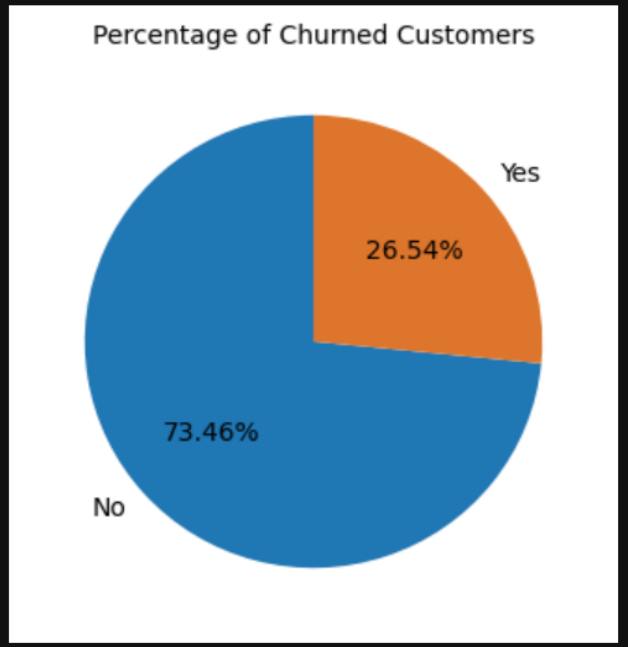
Create Visualizations for Numerical Columns

i. Countplot and Pie Chart (Count of Customers and their percentage by Churn)

[4]:
```
ax = sns.countplot(x="Churn", data=df)

ax.bar_label(ax.containers[0])
plt.title('Count of customers by churn')
plt.show()
```



[5]:
```
plt.figure(figsize=(4,4))
gb= df['Churn'].value_counts()
plt.pie(gb, labels=gb.index , autopct= "%1.2f%%", startangle=90, colors=['#1f77b4','#dd762b'])
plt.title('Percentage of Churned Customers', fontsize=10)
plt.show()
print(gb.to_string())
```

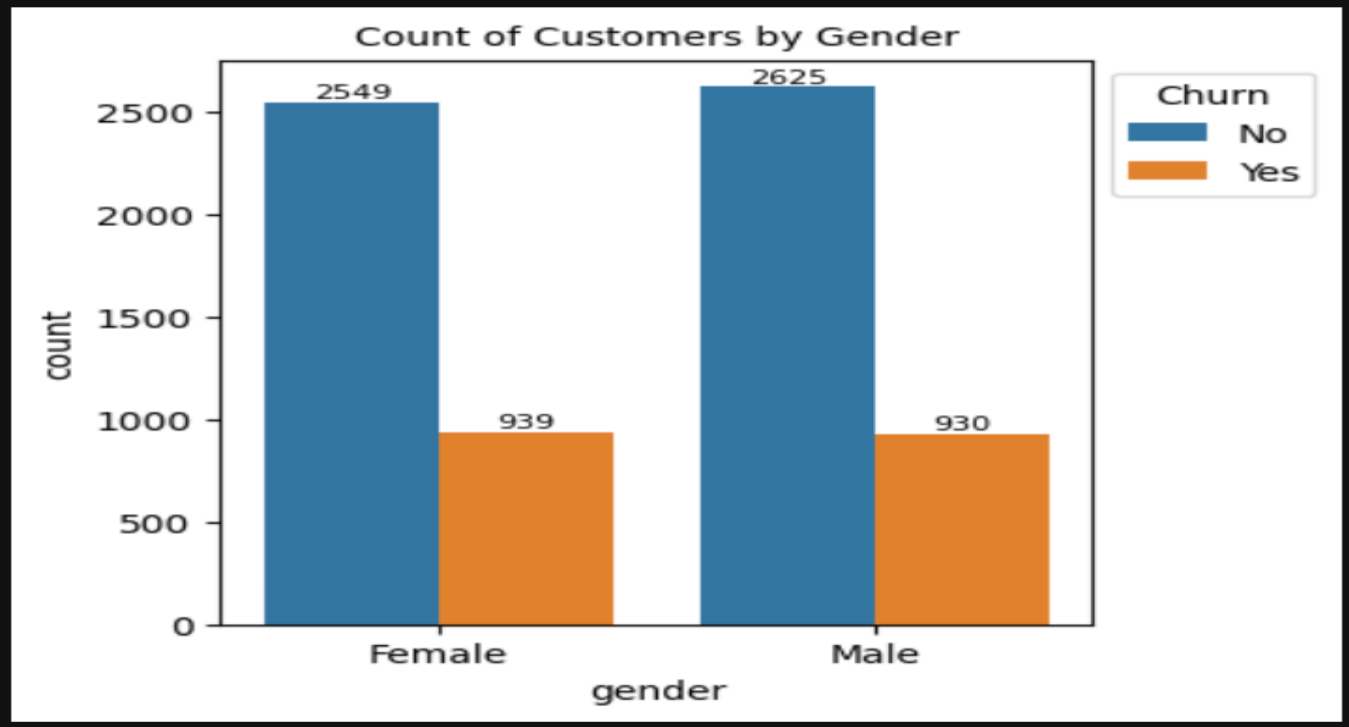



```
Churn
No     5174
Yes    1869
```

ii. Countplot (Churn of Customer by Gender)

[64]:
```
plt.figure(figsize=(4,4))
count_gen= sns.countplot(x="gender",data=df, hue="Churn")
# for loop for mentioning data labels on all the columns.
for container in count_gen.containers:
    count_gen.bar_label(container, fontsize=8)
plt.title('Count of Customers by Gender', fontsize=10)
plt.legend(title="Churn", bbox_to_anchor=(1,1))
plt.show
```

[64]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```

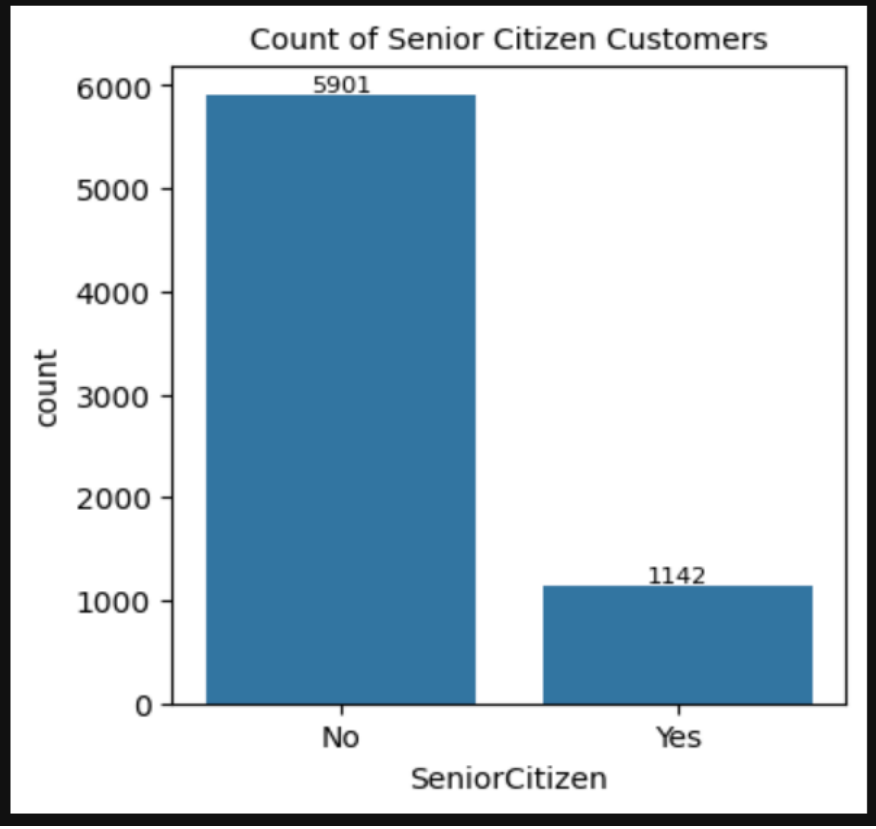


#Gender appears to have no significant effect on churn, as the numbers for both genders are quite similar.

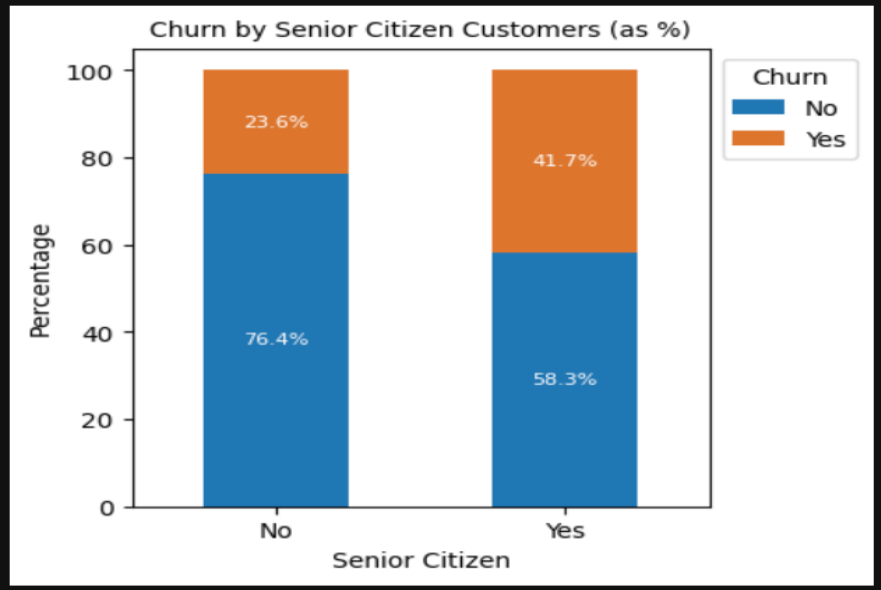
iii. Countplot and Stacked Bar chart (Count of SeniorCitizen customers with Percentage of Churn)

[59]:
```
plt.figure(figsize=(4,4))
count_src=sns.countplot(x="SeniorCitizen",data=df)
count_src.bar_label(count_src.containers[0], fontsize=8)
plt.title('Count of Senior Citizen Customers', fontsize=10)
plt.show
```

[59]: <function matplotlib.pyplot.show(close=None, block=None)>



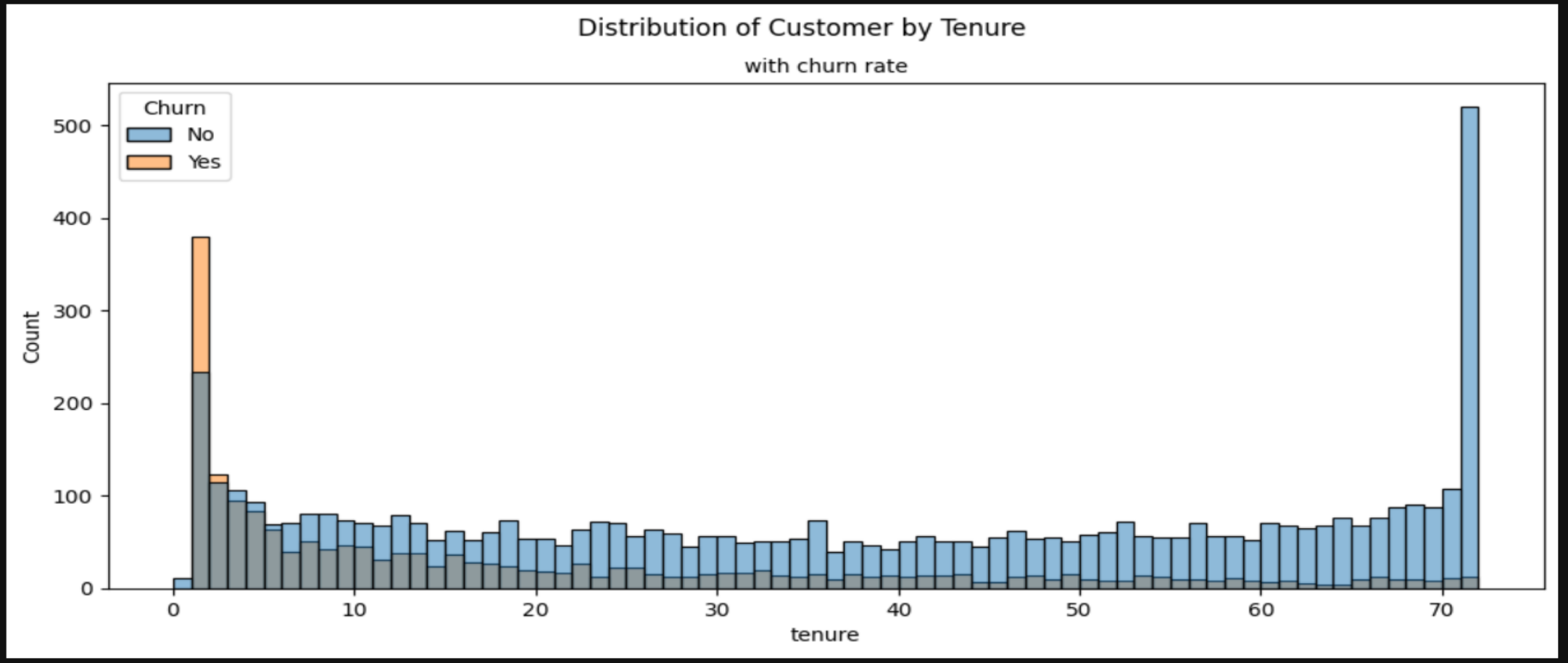
[18]:
```
# Calculate the percentage values
df_grouped = df.groupby("SeniorCitizen")["Churn"].value_counts(normalize=True).unstack() * 100

# Plot the stacked bar chart
count_src1 = df_grouped.plot(kind="bar", stacked=True, figsize=(4, 4), color=['#1f77b4','#dd762b'])

# Add percentage labels
for container in count_src1.containers:
    count_src1.bar_label(container, fmt="%.1f%%", label_type="center", fontsize=8, color='white')

plt.title("Churn by Senior Citizen Customers (as %)", fontsize=10)
plt.ylabel("Percentage")
plt.xlabel("Senior Citizen")
plt.xticks(rotation=0)
plt.legend(title="Churn", bbox_to_anchor=(1,1))
plt.show()
```



#Comparatively a greater percentage of customers from senior citizen category have churned out.

iv. A Histogram (Distribution of Churn of Customer by Tenure)

[21]:
```
plt.figure(figsize=(12,5))
sns.histplot(x='tenure', data= df, edgecolor='black', bins=72, hue='Churn')
plt.suptitle("Distribution of Customer by Tenure", fontsize=12)
plt.title("with churn rate", fontsize=10)
plt.show()
```

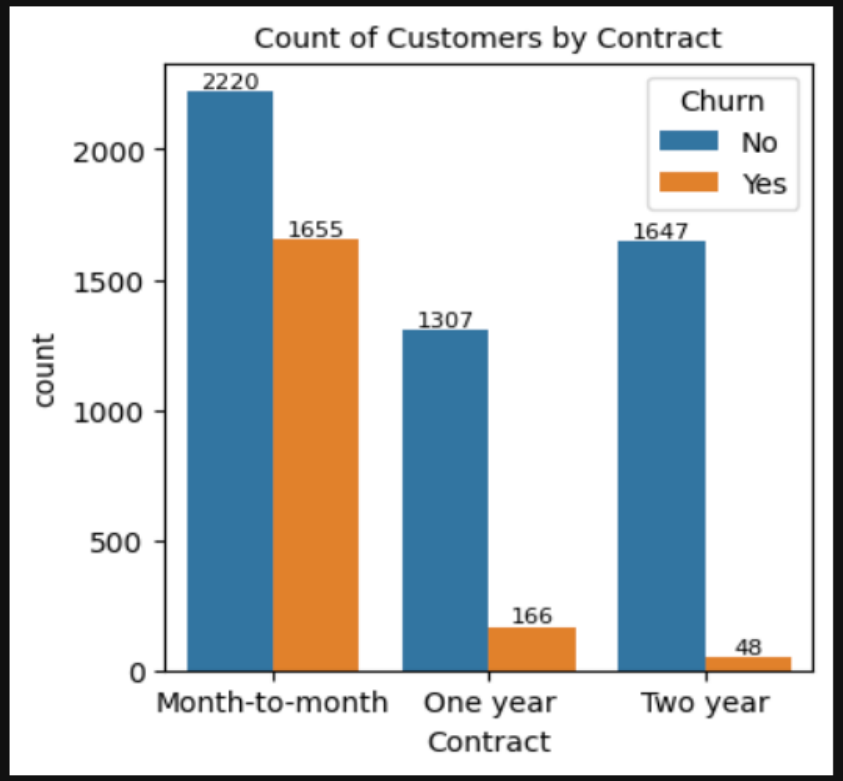


#People who have used our services for a long period of timee have stayed and people who have used for our services for less time(2 to 3 months) have churned.

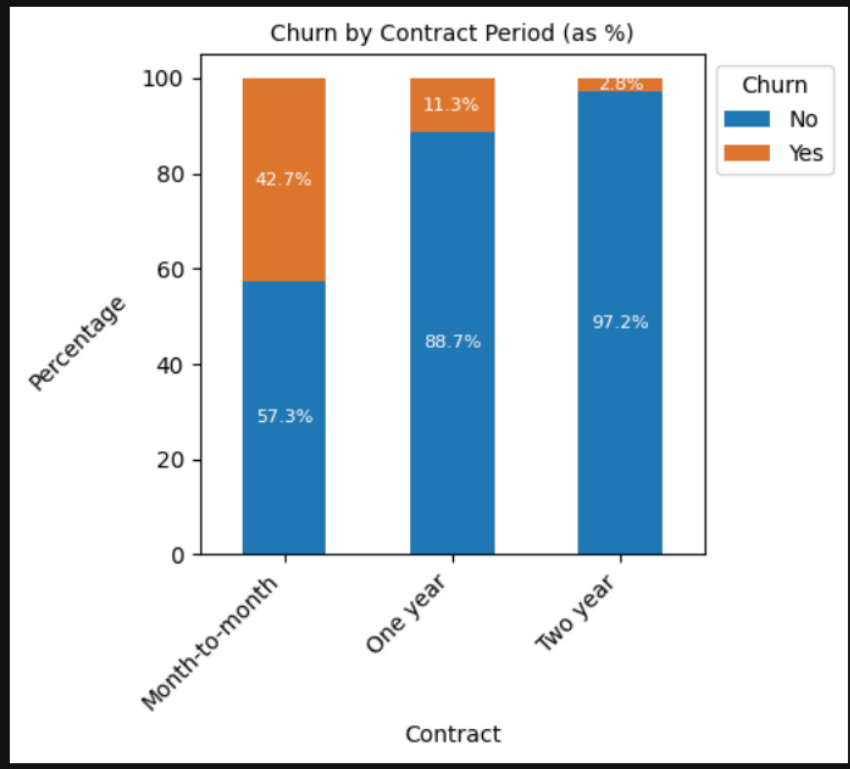
v. Counplot and Stacked Bar Chart (Churn of Customer by Contract Period)

[76]:
```
plt.figure(figsize=(4,4))
count_cont= sns.countplot(x="Contract",data=df, hue="Churn")
# for loop for mentioning data labels on all the columns.
for container in count_cont.containers:
    count_cont.bar_label(container, fontsize=8)
plt.title('Count of Customers by Contract', fontsize=10)
plt.legend(title="Churn", bbox_to_anchor=(1,1))
plt.show
```

[76]: <function matplotlib.pyplot.show(close=None, block=None)>



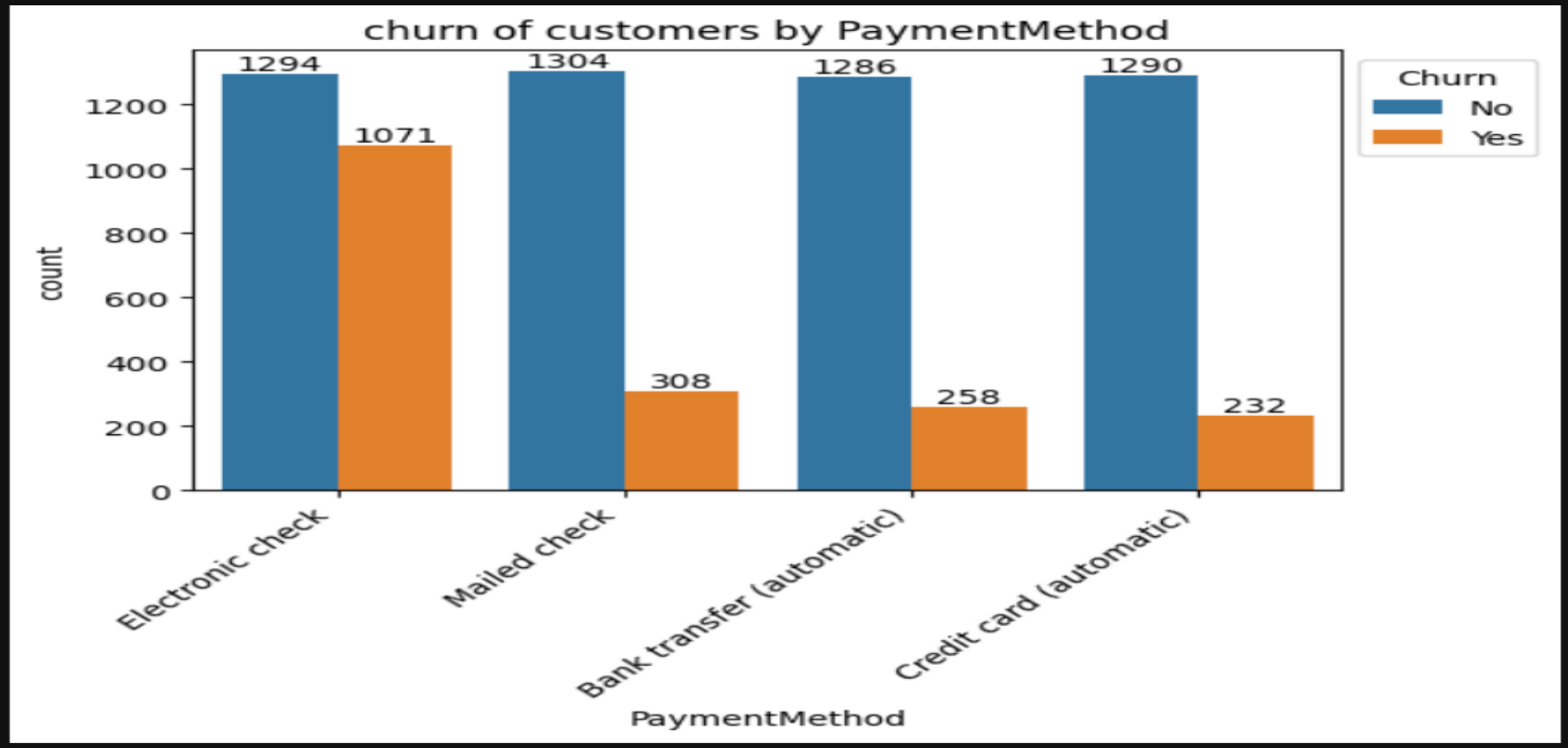
[105]:
```
df_grouped = df.groupby("Contract")["Churn"].value_counts(normalize=True).unstack() * 100
count_cont1 = df_grouped.plot(kind="bar", stacked=True, figsize=(4, 4), color=['#1f77b4','#dd762b'])
for container in count_cont1.containers:
    count_cont1.bar_label(container, fmt="%.1f%%", label_type="center", fontsize=8, color='white')

plt.title("Churn by Contract Period (as %)", fontsize=10)
plt.ylabel("Percentage",rotation=45, ha='right')
plt.xlabel("Contract")
plt.xticks(rotation=45, ha='right')
plt.legend(title="Churn", bbox_to_anchor=(1,1))
plt.show()
```



#Customers who have a Month-to-Month contract are likely to churn than from the customers with One or Two year contract.

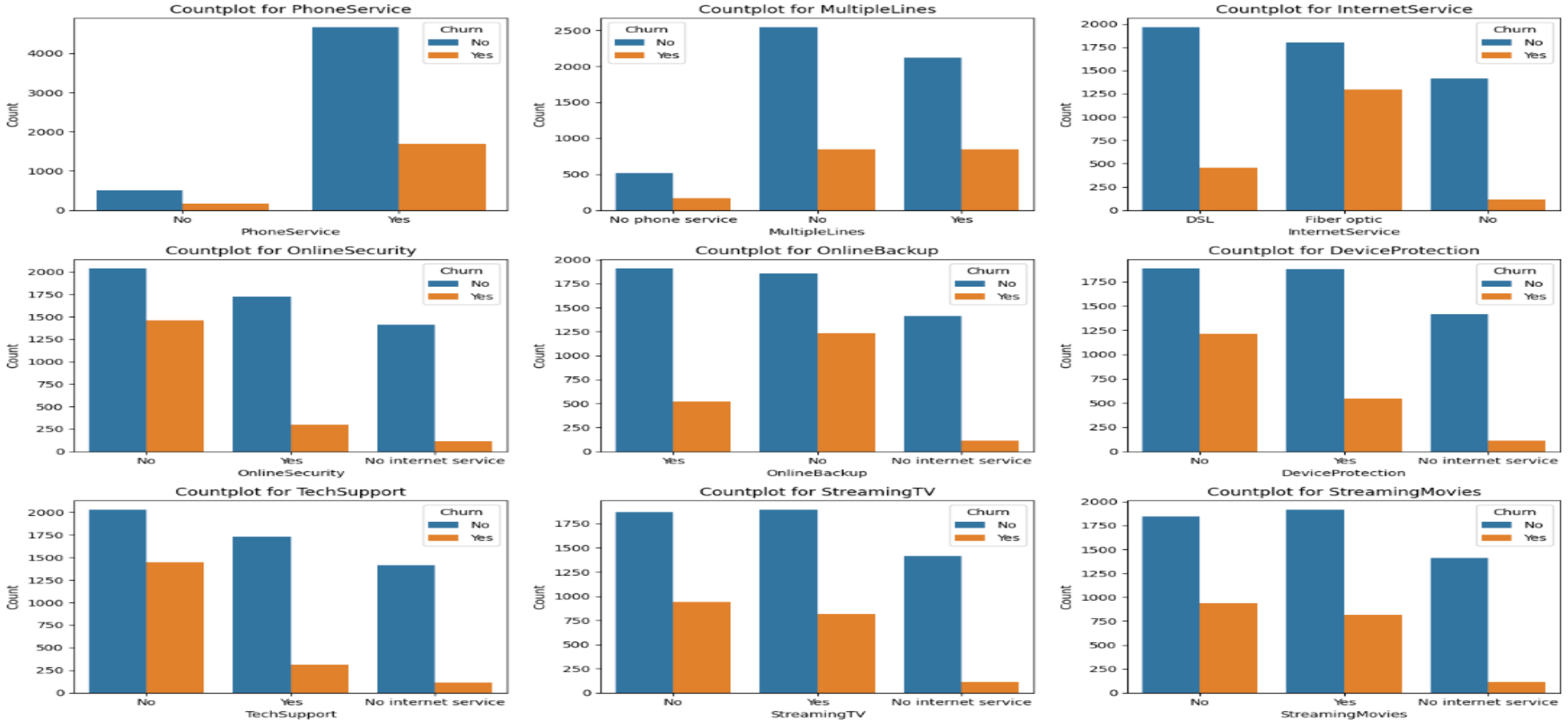
vi. Countplot (Churn of Customer by Paymentmethod)

[22]:
```
plt.figure(figsize=(6,4))
pay_m = sns.countplot(x="PaymentMethod", data=df, hue='Churn')
pay_m.bar_label(pay_m.containers[0])
pay_m.bar_label(pay_m.containers[1])
plt.xticks(rotation=45, ha='right')
plt.title('churn of customers by PaymentMethod')
plt.legend(title="Churn", bbox_to_anchor=(1,1))
plt.show()
```



#Customer is more likely to churn when the payment method is electronic check while curn rate is less in other payemnt methods.

vii. SubPlot (Churn of Customer by various other factors)



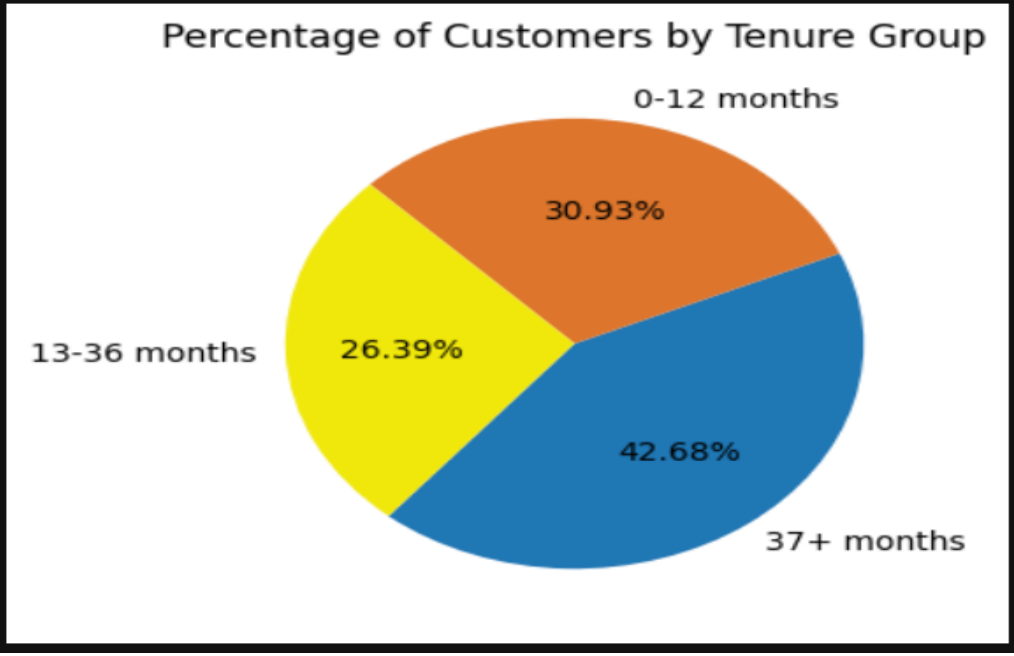
#Customers are most likely to churn when they use Fiber optic InternetService, lack OnlineSecurity and TechSupport, or do not subscribe to StreamingTV. These services appear to be critical drivers of churn behavior, indicating areas where focused retention strategies can have the most impact.

4. Customer Segmentation Visualization

[35]:
```
bins= [0,12,36,df['tenure'].max()]
labels=['0-12 months','13-36 months','37+ months']
df['tenure group']=pd.cut(df['tenure'],bins=bins, labels=labels)

Tenure_count=df['tenure group'].value_counts()
print(Tenure_count.to_string())
```

```
tenure group
37+ months      3001
0-12 months     2175
13-36 months    1856
```

[63]:
```
plt.figure(figsize=(4,4))
plt.pie(Tenure_count, labels=Tenure_count.index , autopct= "%1.2f%%", startangle=230, colors=['#1f77b4','#dd762b','#f0e80c'])
plt.title('Percentage of Customers by Tenure Group')
plt.show()
```

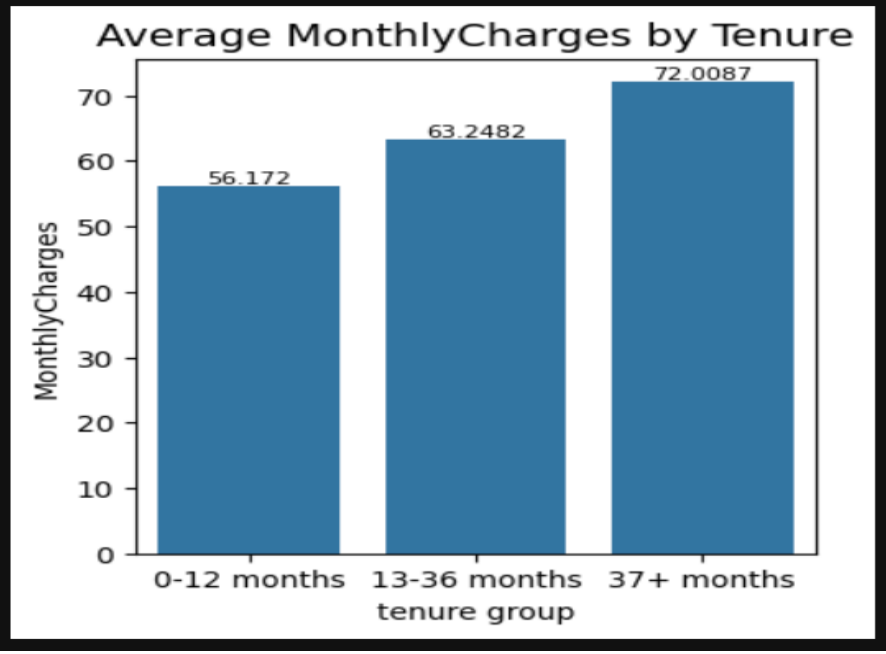


[50]:
```
Tenure_average_charges=df.groupby('tenure group')['MonthlyCharges'].mean().reset_index()

print(Tenure_count.to_string())
print(Tenure_average_charges)
```

```
tenure group
37+ months       3001
0-12 months      2175
13-36 months     1856
   tenure group  MonthlyCharges
0   0-12 months       56.172023
1  13-36 months       63.248195
2    37+ months       72.008730
```

```
C:\Users\Vinay\AppData\Local\Temp\ipykernel_14884\3423064416.py:1: FutureWarning: The default of observed=False is deprecated and will be changed to Tru
e in a future version of pandas. Pass observed=False to retain current behavior or observed=True to adopt the future default and silence this warning.
  Tenure_average_charges=df.groupby('tenure group')['MonthlyCharges'].mean().reset_index()
```

[62]:
```
plt.figure(figsize=(4,4))
tg=sns.barplot(x='tenure group',y='MonthlyCharges', data=Tenure_average_charges)
tg.bar_label(tg.containers[0], fontsize=8)
plt.title('Average MonthlyCharges by Tenure', fontsize=14)
plt.show()
```



Insights

1. **Overall Churn Rate**

    - **26.5%** of customers have churned, highlighting an indicating retention challenge.

2. **Demographic Insights**

    - **Senior Citizens** have a **41% churn rate**, nearly double the **20%** churn among younger customers.

3. **Service-Related Insights**

    - Customers with **Fiber Optic Internet** churn at **42%**, in contrast to just 15% for DSL users.
    - Lack of additional services (such as **Online Security, Tech Support, or Streaming Services**) increases churn probability.

4. **Contract Type & Payment Methods**

    - **Month-to-month contract holders** have the highest churn rate at **45%**, while customers with **two-year contracts** churn only at **8%**. Customers using **Electronic Check payments** have a **38% churn rate**, compared to **16%** for those using credit cards.

Recommendations to Reduce Churn

1. **Loyalty and Retention Programs:**
    - Offer discounts or loyalty rewards for customers opting for long-term contracts.
    - Provide personalized offers to senior citizens and at-risk segments.
2. **Improve Customer Experience:**
    - Identify and address pain points related to fiber optic internet service.
    - Enhance customer support services to improve problem resolution and reduce frustration.
3. **Flexible Billing and Payment Options:**
    - Promote credit cards and automatic bank transfers to reduce electronic check churn rates.
    - Provide incentives for switching to annual or two-year plans.
4. **Proactive Engagement Strategies:**
    - Develop predictive models to identify at-risk customers and reach out proactively with retention offers.
    - Increase engagement with new customers in the first few months to build loyalty.

THANK YOU