```
In [2]: pip install pandas
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pandas in c:\programdata\anaconda3\lib\site-packages (1.4.2)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\programdata\anaconda3\lib\site-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\programdata\anaconda3\lib\site-packages (from pandas) (2021.3)
Requirement already satisfied: numpy>=1.18.5 in c:\programdata\anaconda3\lib\site-packages (from pandas) (1.21.5)
Requirement already satisfied: six>=1.5 in c:\programdata\anaconda3\lib\site-packages (from python-dateutil>=2.8.1->pandas) (1.1
6.0)
Note: you may need to restart the kernel to use updated packages.
```

```
In [3]: pip install matplotlib
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: matplotlib in c:\programdata\anaconda3\lib\site-packages (3.5.1)
Requirement already satisfied: packaging>=20.0 in c:\programdata\anaconda3\lib\site-packages (from matplotlib) (21.3)
Requirement already satisfied: numpy>=1.17 in c:\programdata\anaconda3\lib\site-packages (from matplotlib) (1.21.5)
Requirement already satisfied: python-dateutil>=2.7 in c:\programdata\anaconda3\lib\site-packages (from matplotlib) (2.8.2)
Requirement already satisfied: fonttools>=4.22.0 in c:\programdata\anaconda3\lib\site-packages (from matplotlib) (4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\programdata\anaconda3\lib\site-packages (from matplotlib) (1.3.2)
Requirement already satisfied: pyparsing>=2.2.1 in c:\programdata\anaconda3\lib\site-packages (from matplotlib) (3.0.4)
Requirement already satisfied: pillow>=6.2.0 in c:\programdata\anaconda3\lib\site-packages (from matplotlib) (9.0.1)
Requirement already satisfied: cycler>=0.10 in c:\programdata\anaconda3\lib\site-packages (from matplotlib) (0.11.0)
Requirement already satisfied: six>=1.5 in c:\programdata\anaconda3\lib\site-packages (from python-dateutil>=2.7->matplotlib)
(1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

```
In [4]: pip install seaborn
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: seaborn in c:\programdata\anaconda3\lib\site-packages (0.11.2)
Requirement already satisfied: scipy>=1.0 in c:\programdata\anaconda3\lib\site-packages (from seaborn) (1.7.3)
Requirement already satisfied: matplotlib>=2.2 in c:\programdata\anaconda3\lib\site-packages (from seaborn) (3.5.1)
Requirement already satisfied: numpy>=1.15 in c:\programdata\anaconda3\lib\site-packages (from seaborn) (1.21.5)
Requirement already satisfied: pandas>=0.23 in c:\programdata\anaconda3\lib\site-packages (from seaborn) (1.4.2)
Requirement already satisfied: pillow>=6.2.0 in c:\programdata\anaconda3\lib\site-packages (from matplotlib>=2.2->seaborn) (9.0.
1)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\programdata\anaconda3\lib\site-packages (from matplotlib>=2.2->seaborn)
(1.3.2)
Requirement already satisfied: fonttools>=4.22.0 in c:\programdata\anaconda3\lib\site-packages (from matplotlib>=2.2->seaborn)
(4.25.0)
Requirement already satisfied: python-dateutil>=2.7 in c:\programdata\anaconda3\lib\site-packages (from matplotlib>=2.2->seabor
n) (2.8.2)
Requirement already satisfied: pyparsing>=2.2.1 in c:\programdata\anaconda3\lib\site-packages (from matplotlib>=2.2->seaborn)
(3.0.4)
Requirement already satisfied: packaging>=20.0 in c:\programdata\anaconda3\lib\site-packages (from matplotlib>=2.2->seaborn) (2
1.3)
Requirement already satisfied: cycler>=0.10 in c:\programdata\anaconda3\lib\site-packages (from matplotlib>=2.2->seaborn) (0.11.
0)
Requirement already satisfied: pytz>=2020.1 in c:\programdata\anaconda3\lib\site-packages (from pandas>=0.23->seaborn) (2021.3)
Requirement already satisfied: six>=1.5 in c:\programdata\anaconda3\lib\site-packages (from python-dateutil>=2.7->matplotlib>=2.
2->seaborn) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

In [5]:

```
pip install numpy
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: numpy in c:\programdata\anaconda3\lib\site-packages (1.21.5)
Note: you may need to restart the kernel to use updated packages.
```

In [9]:

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv('Customer Churn.csv')
df.head()
```

Out[9]:

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

5 rows × 21 columns

In [10]:

```
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
```

replacing blanks with 0 as tenure is 0 and no total charges are recorded

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

In [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)
memory usage: 1.1+ MB
```

In [13]:
```
df.isnull().sum().sum()
```

Out[13]:
```
0
```

In [14]:
```
df.describe()
```

Out[14]:

	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

In [15]:
```
df["customerID"].duplicated().sum()
```

Out[15]:
```
0
```

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

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

converted 0 and 1 values of senior citizen to "yes / no" to make it easier to understand

In [17]:
```
df.head(30)
```

Out[17]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection
0	7590-VHVEG	Female	no	Yes	No	1	No	No phone service	DSL	No	...	No
1	5575-GNVDE	Male	no	No	No	34	Yes	No	DSL	Yes	...	Yes
2	3668-QPYBK	Male	no	No	No	2	Yes	No	DSL	Yes	...	No
3	7795-CFOCW	Male	no	No	No	45	No	No phone service	DSL	Yes	...	Yes
4	9237-HQITU	Female	no	No	No	2	Yes	No	Fiber optic	No	...	No
5	9305-CDSKC	Female	no	No	No	8	Yes	Yes	Fiber optic	No	...	Yes
6	1452-KIOVK	Male	no	No	Yes	22	Yes	Yes	Fiber optic	No	...	No
7	6713-OKOMC	Female	no	No	No	10	No	No phone service	DSL	Yes	...	No
8	7892-POOKP	Female	no	Yes	No	28	Yes	Yes	Fiber optic	No	...	Yes
9	6388-TABGU	Male	no	No	Yes	62	Yes	No	DSL	Yes	...	No
10	9763-GRSKD	Male	no	Yes	Yes	13	Yes	No	DSL	Yes	...	No
11	7469-LKBCI	Male	no	No	No	16	Yes	No	No	No internet service	...	No internet service

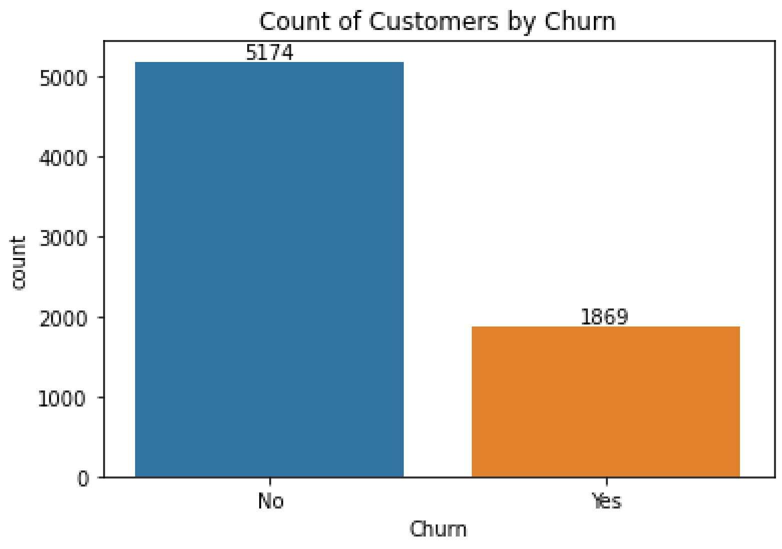
	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection
12	8091-TTVAX	Male	no	Yes	No	58	Yes	Yes	Fiber optic	No	...	Yes
13	0280-XJGEX	Male	no	No	No	49	Yes	Yes	Fiber optic	No	...	Yes
14	5129-JLPIS	Male	no	No	No	25	Yes	No	Fiber optic	Yes	...	Yes
15	3655-SNQYZ	Female	no	Yes	Yes	69	Yes	Yes	Fiber optic	Yes	...	Yes
16	8191-XWSZG	Female	no	No	No	52	Yes	No	No	No internet service	...	No internet service
17	9959-WOFKT	Male	no	No	Yes	71	Yes	Yes	Fiber optic	Yes	...	Yes
18	4190-MFLUW	Female	no	Yes	Yes	10	Yes	No	DSL	No	...	Yes
19	4183-MYFRB	Female	no	No	No	21	Yes	No	Fiber optic	No	...	Yes
20	8779-QRDMV	Male	yes	No	No	1	No	No phone service	DSL	No	...	Yes
21	1680-VDCWW	Male	no	Yes	No	12	Yes	No	No	No internet service	...	No internet service
22	1066-JKSGK	Male	no	No	No	1	Yes	No	No	No internet service	...	No internet service
23	3638-WEABW	Female	no	Yes	No	58	Yes	Yes	DSL	No	...	No
24	6322-HRPFA	Male	no	Yes	Yes	49	Yes	No	DSL	Yes	...	No

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection
25	6865-JZNKO	Female	no	No	No	30	Yes	No	DSL	Yes	...	No
26	6467-CHFZW	Male	no	Yes	Yes	47	Yes	Yes	Fiber optic	No	...	No
27	8665-UTDHZ	Male	no	Yes	Yes	1	No	No phone service	DSL	No	...	No
28	5248-YGIJN	Male	no	Yes	No	72	Yes	Yes	DSL	Yes	...	Yes
29	8773-HHUOZ	Female	no	No	Yes	17	Yes	No	DSL	No	...	No

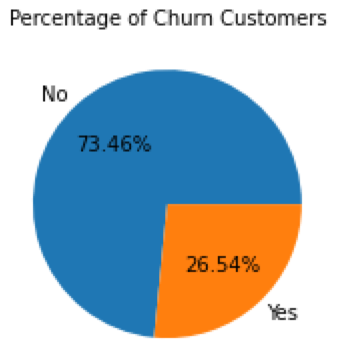
30 rows × 21 columns

In [18]:
```
ax = sns.countplot(x = 'Churn', data = df)

ax.bar_label(ax.containers[0])
plt.title("Count of Customers by Churn")
plt.show
```

Out[18]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```

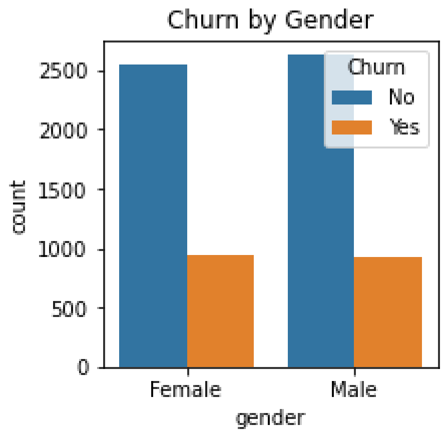


In [19]:
```
plt.figure(figsize = (3,4))
gb = df.groupby("Churn").agg({'Churn':"count"})
plt.pie(gb['Churn'], labels = gb.index, autopct = "%1.2f%%")
plt.title("Percentage of Churn Customers", fontsize = 10)
plt.show()
```

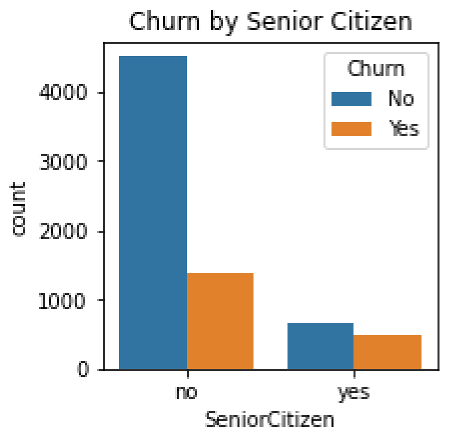


from the given pie chart we can conclude that 26.54% of our customers have churned out. #now let's exploer the reason behind it.

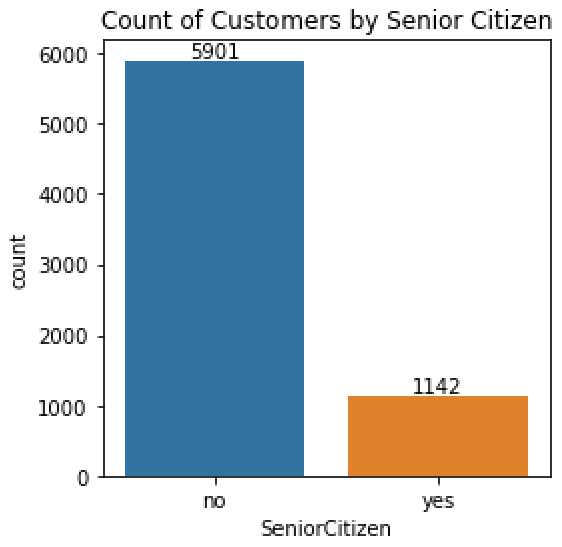
In [20]:
```
plt.figure(figsize = (3,3))
sns.countplot(x = "gender", data = df, hue = "Churn")
plt.title("Churn by Gender")
plt.show()
```



In [21]:
```
plt.figure(figsize = (3,3))
sns.countplot(x = "SeniorCitizen", data = df, hue = "Churn")
plt.title("Churn by Senior Citizen")
plt.show()
```

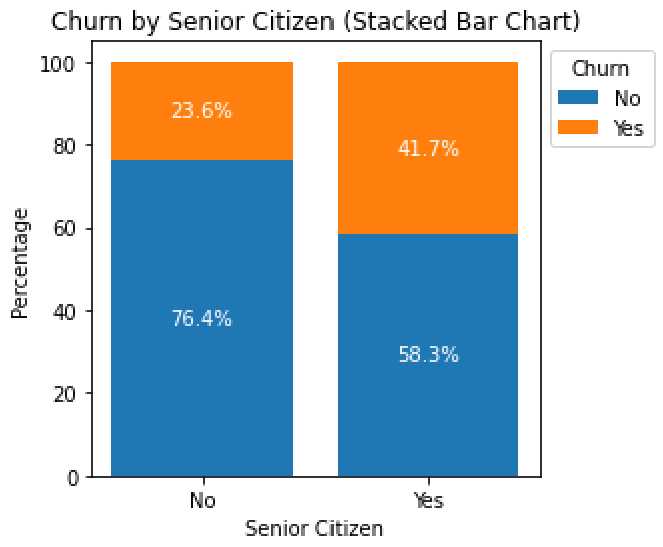


In [22]:
```python
plt.figure(figsize = (4,4))
ax = sns.countplot(x = "SeniorCitizen", data = df)
ax.bar_label(ax.containers[0])
plt.title("Count of Customers by Senior Citizen")
plt.show()
```



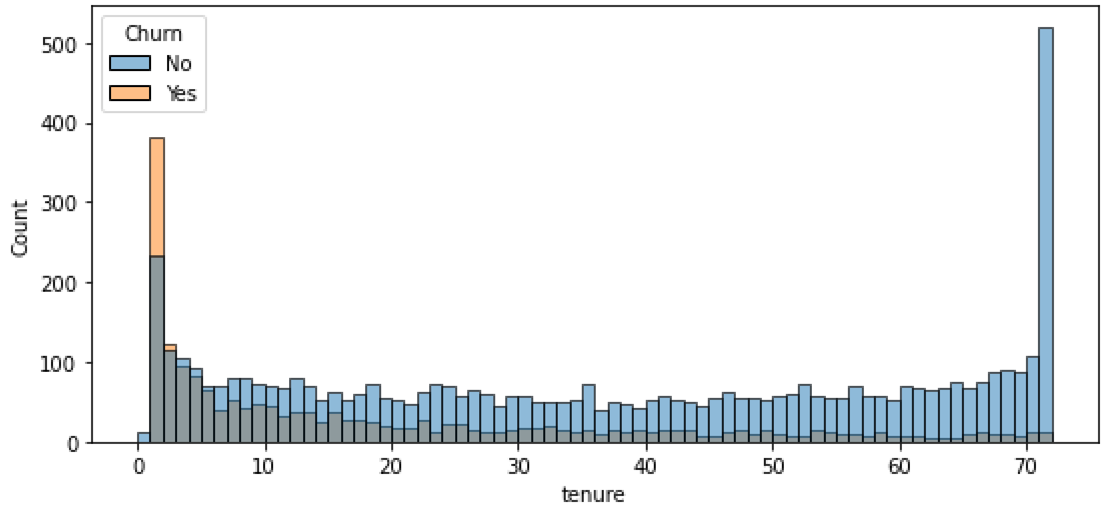
In [23]:
```python
# Calculate percentages
count_data = df.groupby(['SeniorCitizen', 'Churn']).size().reset_index(name='Count')
total_per_senior = count_data.groupby('SeniorCitizen')['Count'].transform('sum')
count_data['Percentage'] = (count_data['Count'] / total_per_senior) * 100
```

```python
# Pivot for stacking
pivot_data = count_data.pivot(index='SeniorCitizen', columns='Churn', values='Percentage').fillna(0)

# Plot stacked bar chart
fig, ax = plt.subplots(figsize=(4, 4))
bars = []
bottoms = [0] * len(pivot_data)
labels = pivot_data.columns.tolist()

for label in labels:
    bars.append(ax.bar(
        pivot_data.index,
        pivot_data[label],
        bottom=bottoms,
        label=label
    ))
    bottoms += pivot_data[label]

# Annotate bars
for bar, label in zip(bars, labels):
    for rect in bar:
        height = rect.get_height()
        if height > 0:
            ax.text(
                rect.get_x() + rect.get_width() / 2,
                rect.get_y() + height / 2,
                f'{height:.1f}%',
                ha='center', va='center', fontsize=10, color='white'
            )

# Customizing the plot
ax.set_xticks(pivot_data.index)
ax.set_xticklabels(['No', 'Yes'])  # Assuming 0 = No, 1 = Yes
ax.set_xlabel('Senior Citizen')
ax.set_ylabel('Percentage')
ax.set_title('Churn by Senior Citizen (Stacked Bar Chart)')
ax.legend(title='Churn', bbox_to_anchor = (1.0,1.0))

plt.show()
```



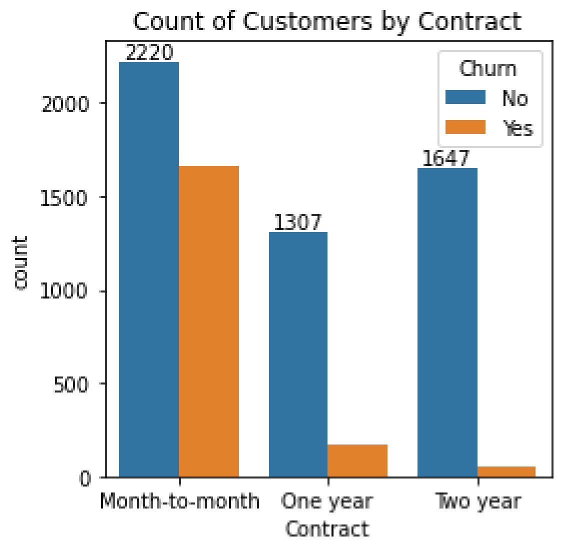
comparatively a generated percentage of people in senior citizen category have churned

In [24]:
```
plt.figure(figsize = (9,4))
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 services #1 or 2 months have churned

In [25]:
```
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()
```



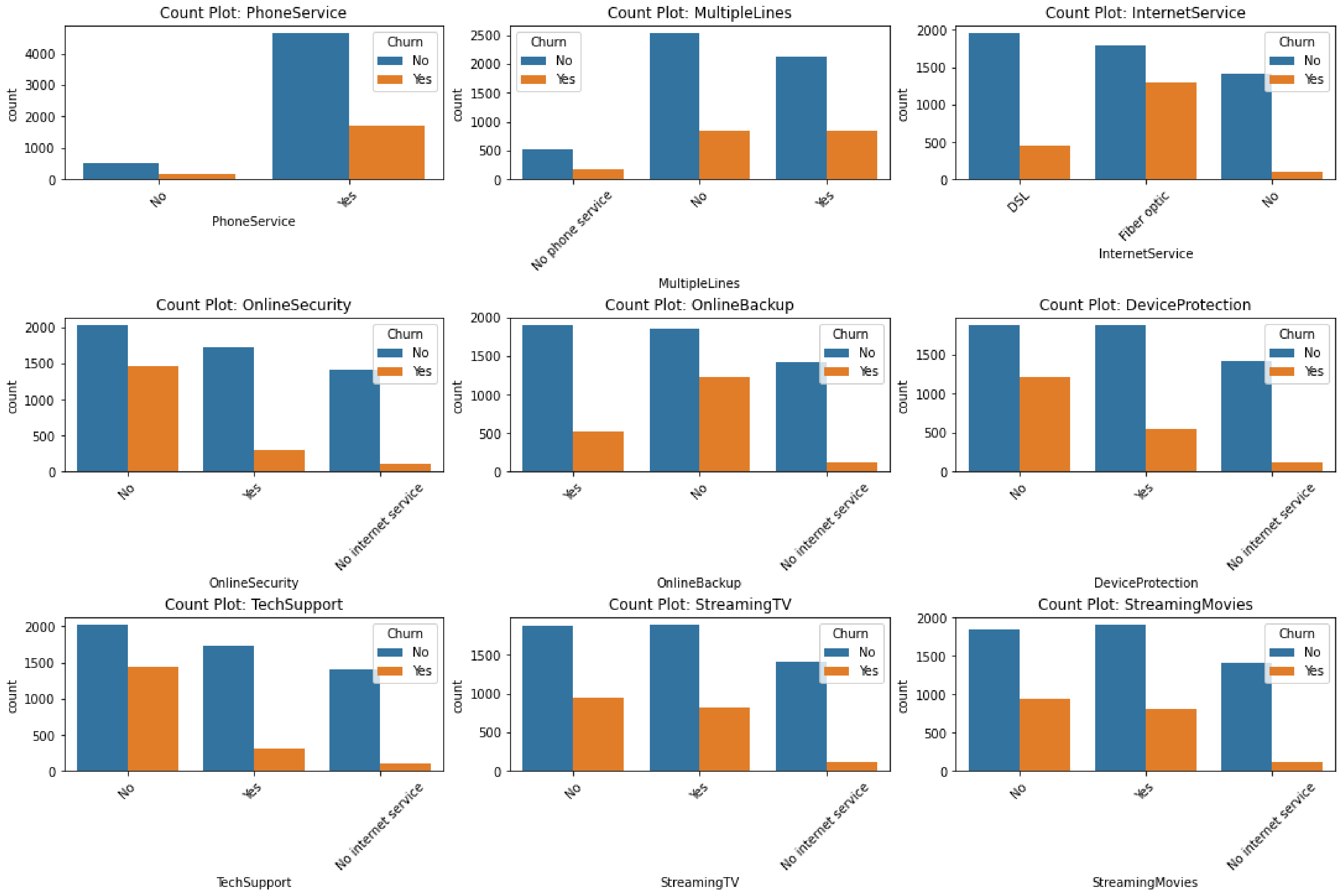
people who have month to month contract are likely to chrn then from those who have 1 or 2 years of contract

In [26]:
```
df.columns.values
```

Out[26]:
```
array(['customerID', 'gender', 'SeniorCitizen', 'Partner', 'Dependents',
       'tenure', 'PhoneService', 'MultipleLines', 'InternetService',
       'OnlineSecurity', 'OnlineBackup', 'DeviceProtection',
       'TechSupport', 'StreamingTV', 'StreamingMovies', 'Contract',
       'PaperlessBilling', 'PaymentMethod', 'MonthlyCharges',
       'TotalCharges', 'Churn'], dtype=object)
```

In [28]:
```
# List of columns to plot
columns = [
    'PhoneService', 'MultipleLines', 'InternetService',
    'OnlineSecurity', 'OnlineBackup', 'DeviceProtection',
    'TechSupport', 'StreamingTV', 'StreamingMovies'
]

# Number of rows and columns for subplots
n_cols = 3
n_rows = (len(columns) + n_cols - 1) // n_cols
```

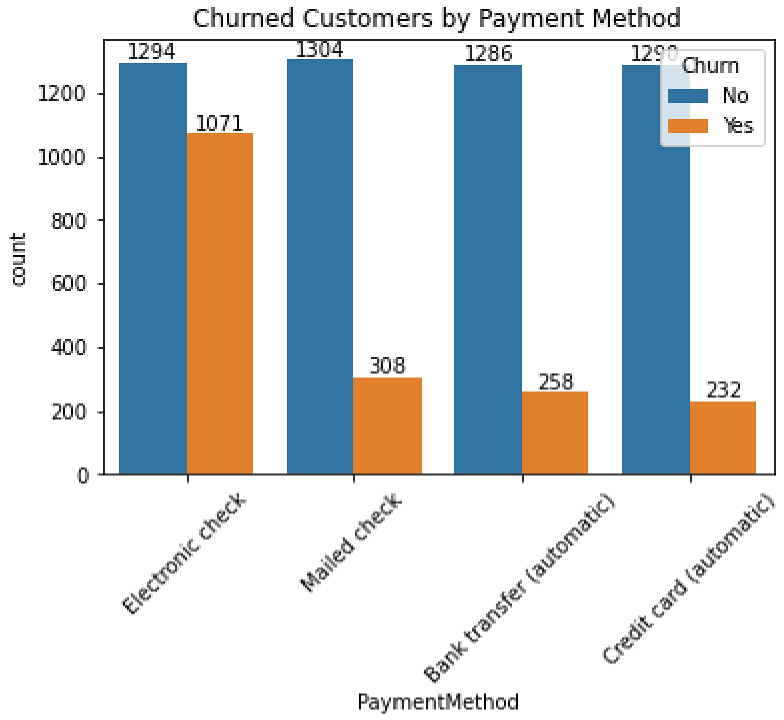
```python
# Create subplots
fig, axes = plt.subplots(n_rows, n_cols, figsize=(15, 10), constrained_layout=True)

# Flatten axes for easy iteration
axes = axes.flatten()

# Plot each count plot
for i, column in enumerate(columns):
    sns.countplot(x=column, data=df, ax=axes[i], hue = df["Churn"])
    axes[i].set_title(f"Count Plot: {column}")
    axes[i].tick_params(axis='x', rotation=45)

# Hide any extra subplots if columns don't fill the grid
for j in range(len(columns), len(axes)):
    axes[j].axis('off')

plt.show()
```



The majority of customers have active PhoneService and InternetService, while a smaller proportion of users utilize services

like MultipleLines or Fiber optic. Services such as OnlineSecurity, OnlineBackup, and DeviceProtection show a significant portion of customers opting for "No" compared to "Yes". StreamingTV and StreamingMovies exhibit a more balanced distribution, with a slightly higher preference towards customers not using these services. In many categories, the Churn rate is notably higher for customers who do not use the respective services, indicating a potential correlation.

In [32]:
```
plt.figure(figsize = (6,4))
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()
```



customer is likely to churn when they are using electronic check as a payment method

```
In [ ]:
```