

```
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()
```

	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-CF0CW	Male	0	No	No	45
4	9237-HQITU	Female	0	No	No	2

	MultipleLines	InternetService	OnlineSecurity	...
0	No phone service	DSL	No	...
1	No	DSL	Yes	...
2	No	DSL	Yes	...
3	No phone service	DSL	Yes	...
4	No	Fiber optic	No	...

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

	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]

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 in "TotalCharges" column as tenure is 0 and no total charges are recorded and changing the datatype of "TotalCharges" column to float

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

```
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
```

```
df.isnull().sum()
```

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

df.describe()

```

	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

```

df["customerID"].duplicated().sum()

np.int64(0)

```

converted 0 and 1 value of "SeniorCitizen" column to Yes and No

```

def conv(value):
    if value == 1:
        return "Yes"
    else:
        return "No"

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

df.head()

```

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure
0	7590-VHVEG	Female	No	Yes	No	1
1	5575-GNVDE	Male	No	No	No	34
2	3668-QPYBK	Male	No	No	No	2
3	7795-CF0CW	Male	No	No	No	45
4	9237-HQITU	Female	No	No	No	2

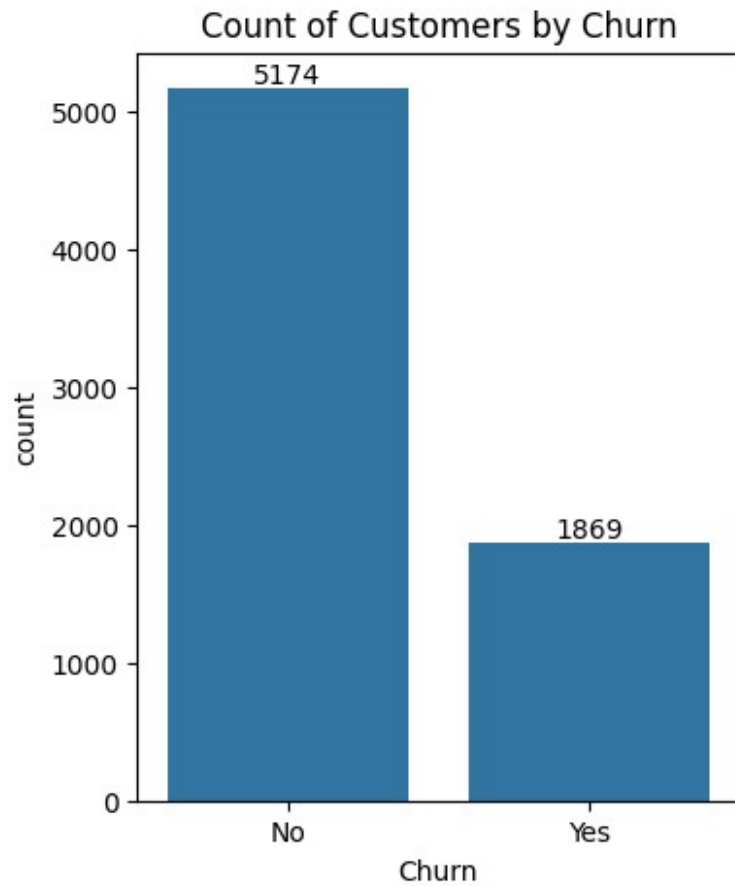
	MultipleLines	InternetService	OnlineSecurity	...
DeviceProtection \				
0	No phone service	DSL	No	...
No				
1	No	DSL	Yes	...
Yes				
2	No	DSL	Yes	...
No				
3	No phone service	DSL	Yes	...
Yes				
4	No	Fiber optic	No	...
No				

	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

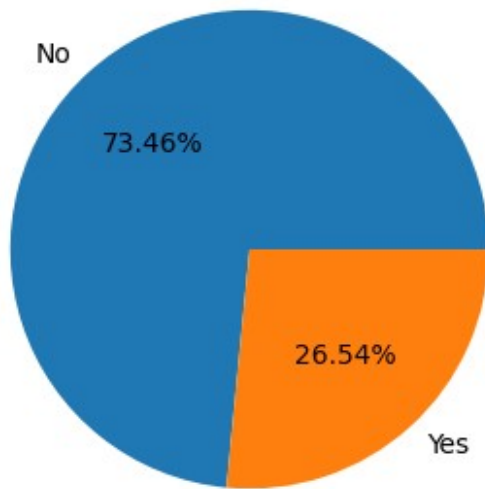
[5 rows x 21 columns]

```
plt.figure(figsize = (4,5))
ax = sns.countplot(x = 'Churn', data = df)
ax.bar_label(ax.containers[0])
plt.title("Count of Customers by Churn")
plt.show()
```



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

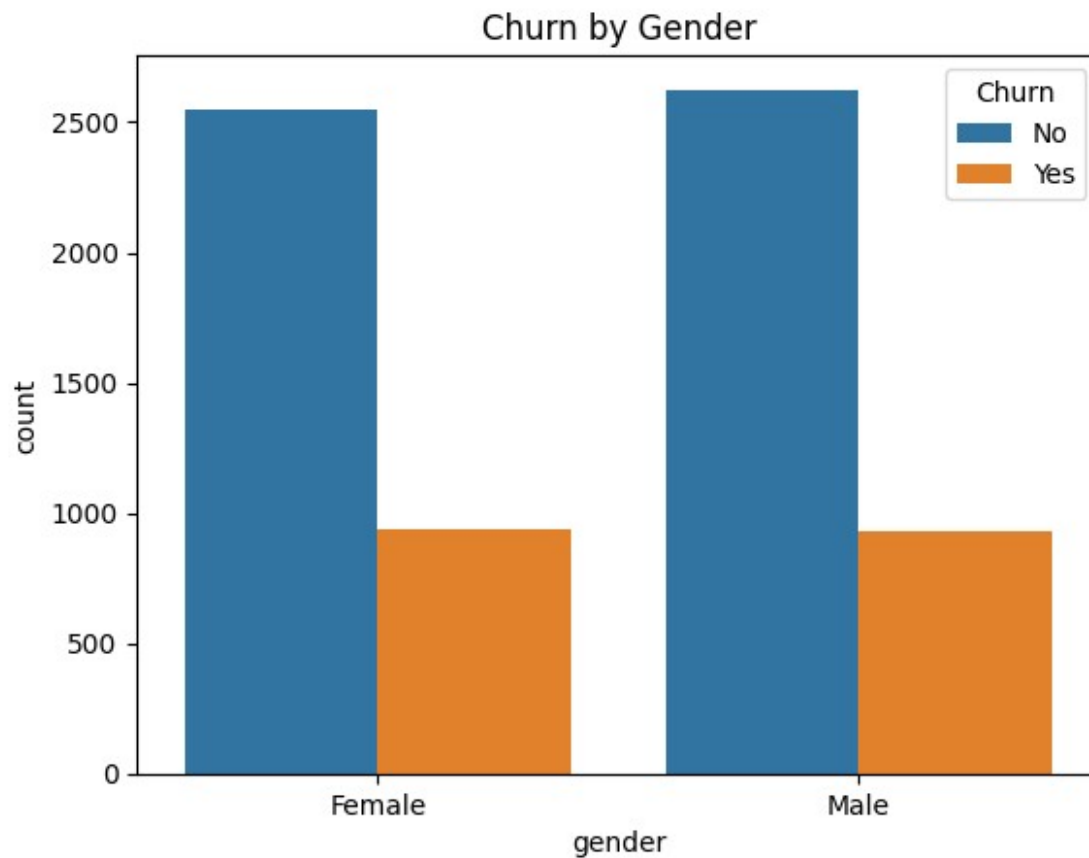
Percentage of Churned Customers



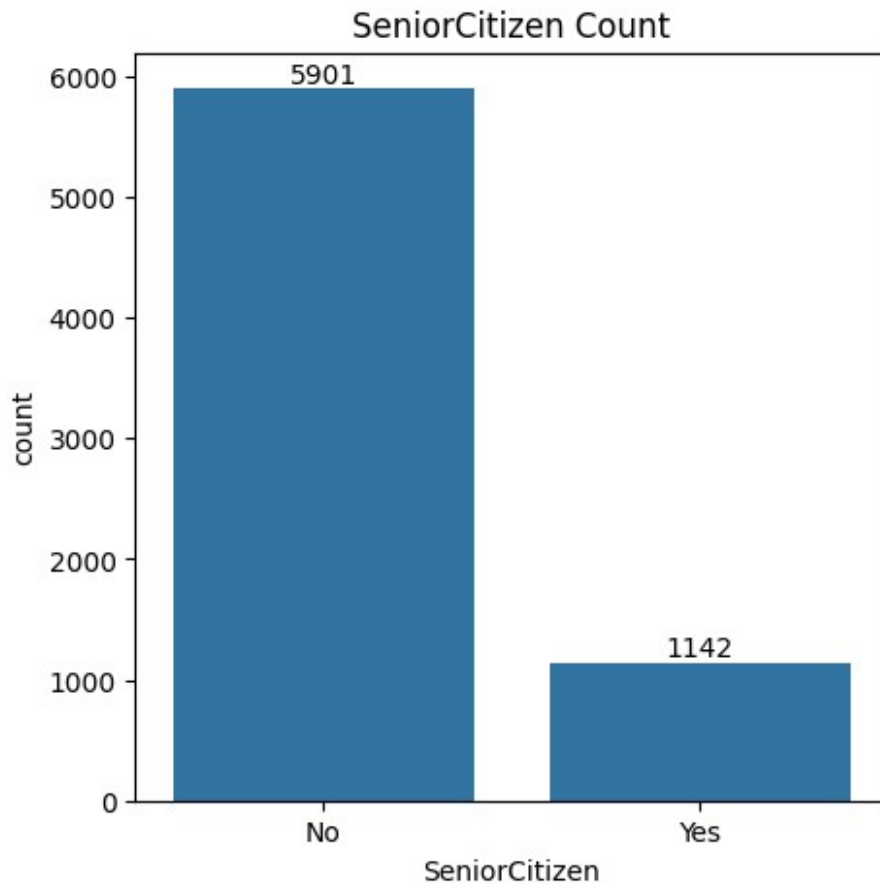
From the given piechart we can conclude that 26.54% of our customer have churned out

Now lets explore the reason behind it

```
sns.countplot(x = 'gender', data = df, hue = "Churn")  
plt.title("Churn by Gender")  
plt.show()
```



```
plt.figure(figsize = (5,5))  
ax = sns.countplot(x = 'SeniorCitizen', data = df)  
plt.title("SeniorCitizen Count")  
ax.bar_label(ax.containers[0])  
plt.show()
```

```
count_data = pd.crosstab(df['SeniorCitizen'], df['Churn'])

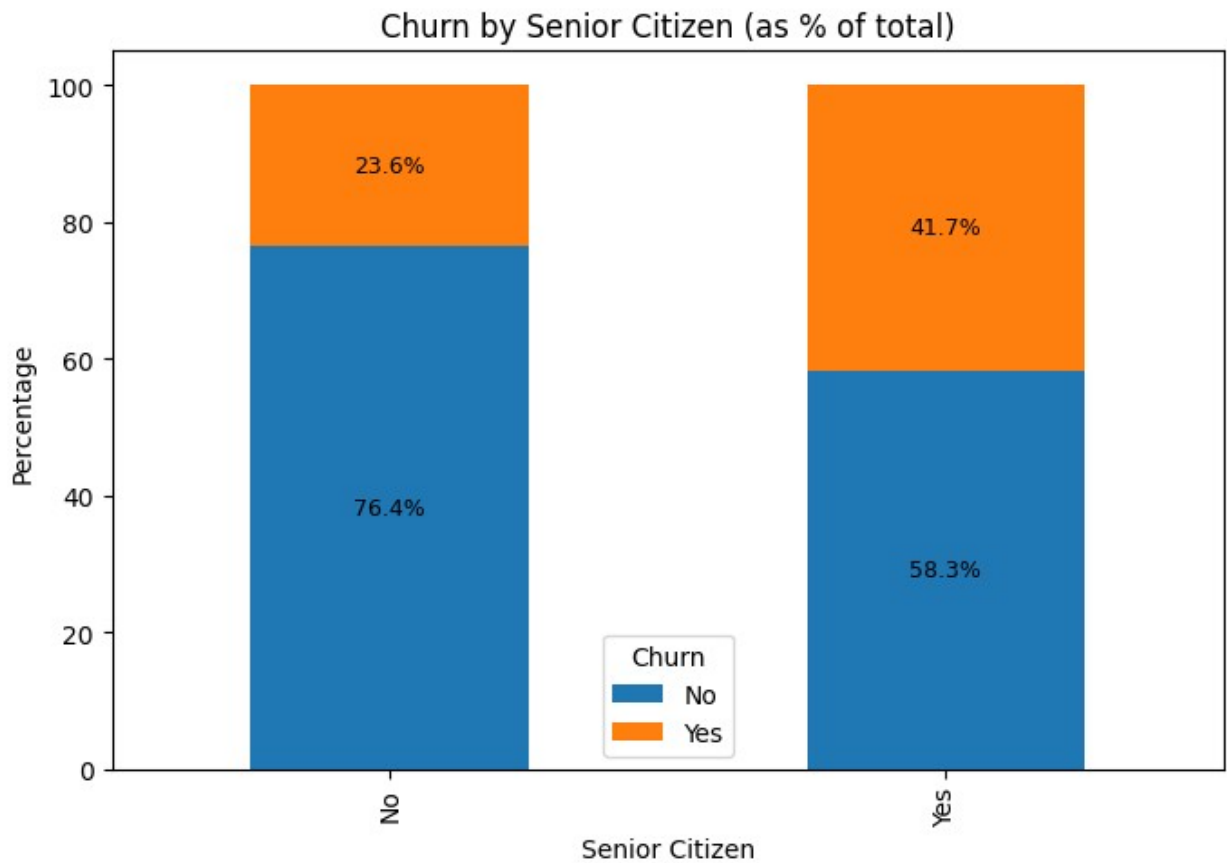
# Convert to percentages
percentage_data = count_data.div(count_data.sum(axis=1), axis=0) * 100

# Plot stacked bar with default matplotlib colors
ax = percentage_data.plot(kind='bar', stacked=True, figsize=(7, 5))

# Add % labels on bars
for idx, row in enumerate(percentage_data.values):
    y_offset = 0
    for pct in row:
        if pct > 0:
            ax.text(idx, y_offset + pct / 2, f'{pct:.1f}%',
                    ha='center', va='center', fontsize=9)
            y_offset += pct

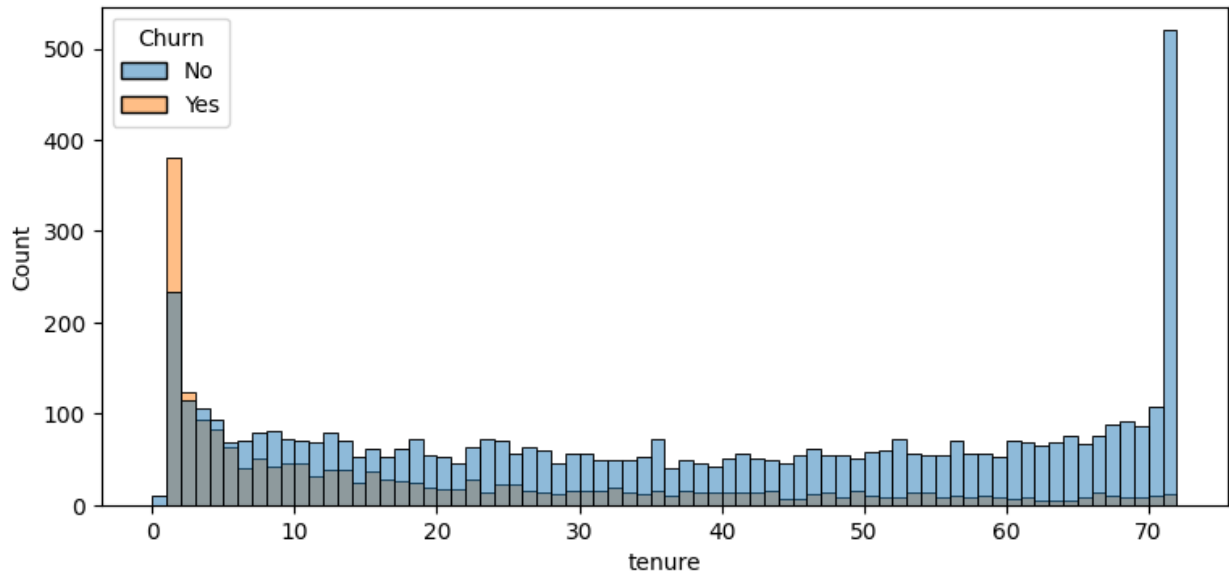
# Final formatting
plt.title("Churn by Senior Citizen (as % of total)")
plt.xlabel("Senior Citizen")
plt.ylabel("Percentage")
plt.legend(title="Churn")
```

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



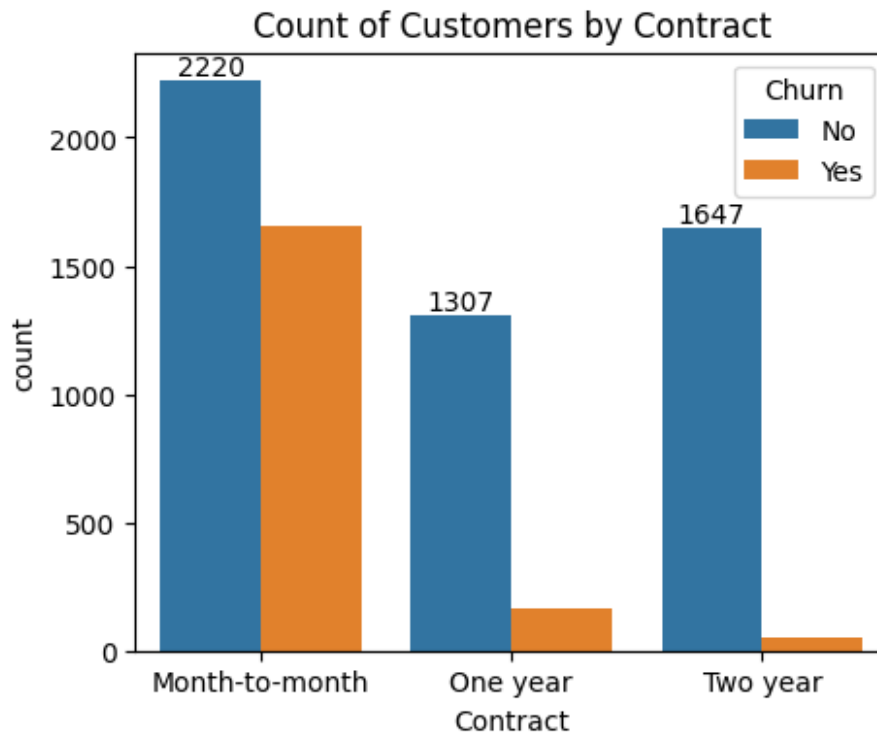
Comparatively a greater percentage of people have churned from senior citizen category

```
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 for 1 or 2 months have churned

```
plt.figure(figsize = (5,4))
ax = sns.countplot(x = 'Contract', data = df, hue = "Churn")
plt.title("Count of Customers by Contract")
ax.bar_label(ax.containers[0])
plt.show()
```



People who have 'month-to-month' contract are likely to churn than those who have 'one year contract' or 'two year contract'

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

cols = ['PhoneService', 'MultipleLines', 'InternetService',
        'OnlineSecurity', 'OnlineBackup', 'DeviceProtection',
        'TechSupport', 'StreamingTV', 'StreamingMovies']

rows, cols_per_row = 3, 3
fig, axes = plt.subplots(rows, cols_per_row, figsize=(16, 16))
axes = axes.flatten()

for i, col in enumerate(cols):
    sns.countplot(x=col, data=df, ax=axes[i], hue = "Churn") # No
```

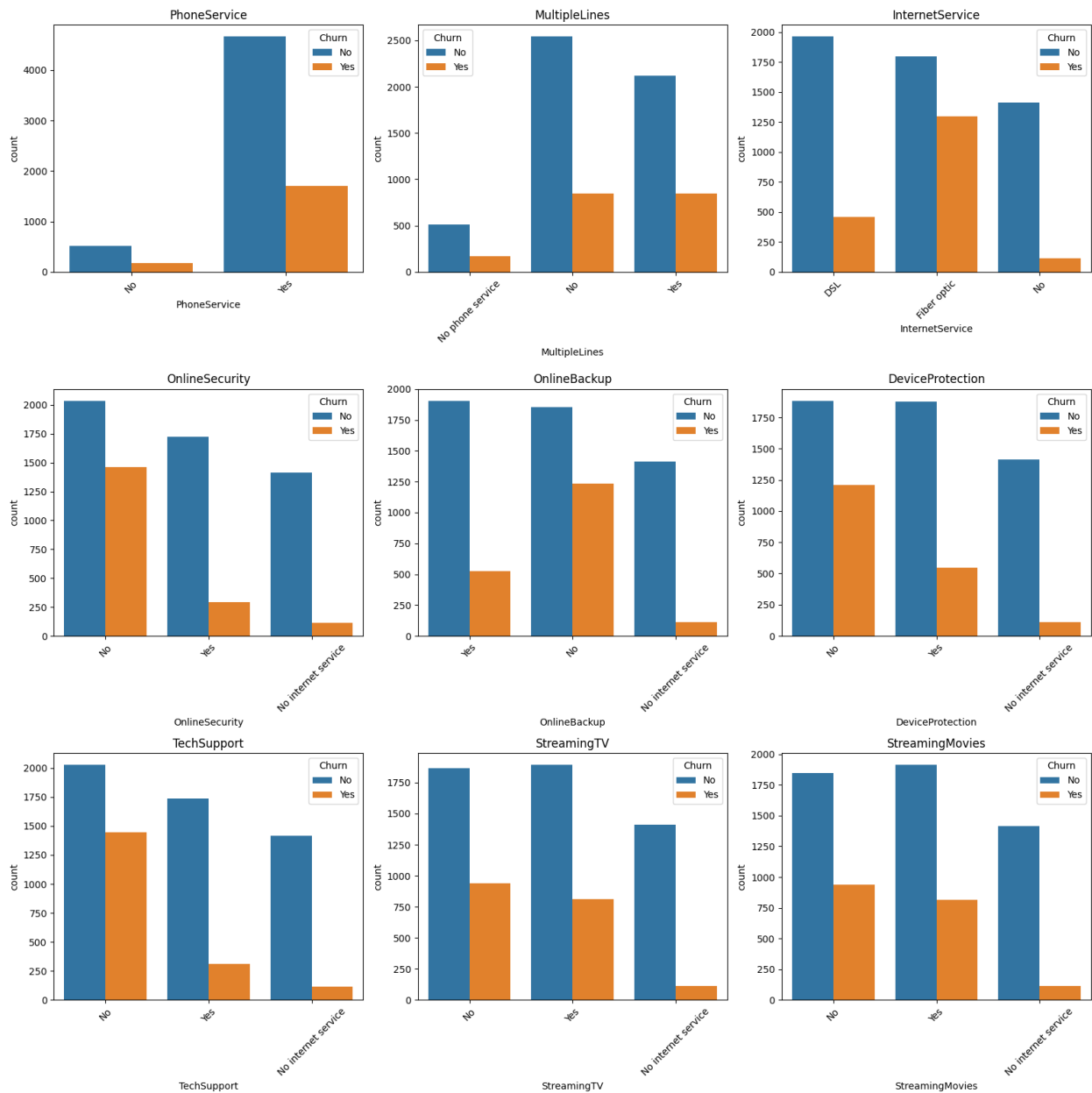
```

palette, uses default colors
axes[i].set_title(col)
axes[i].tick_params(axis='x', rotation=45)

# Remove unused axes if any
for j in range(len(cols), len(axes)):
    fig.delaxes(axes[j])

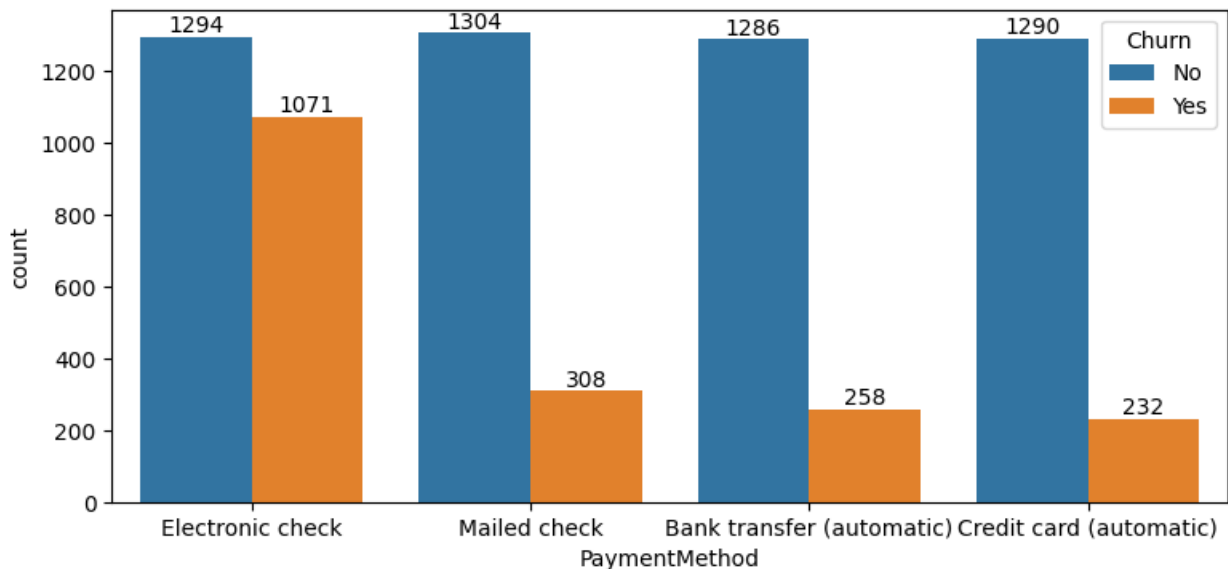
plt.tight_layout()
plt.show()

```



Customers without internet-related services (like No InternetService, No OnlineSecurity, etc.) show significantly lower churn. Services like OnlineSecurity, TechSupport, and StreamingTV have higher churn among users who opted for them, suggesting potential dissatisfaction or other influencing factors.

```
plt.figure(figsize = (9,4))
ax = sns.countplot(x = 'PaymentMethod', data = df, hue = "Churn")
ax.bar_label(ax.containers[0])
ax.bar_label(ax.containers[1])
plt.show()
```



Customer is likely to churn if he is using 'electronic check' as a payment method