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

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
df = pd.read_csv('/content/drive/MyDrive/Data /Customer Churn.csv', encoding='unicode_escape')
```

```
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
```

```
df.head(5)
```

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

5 rows × 21 columns

```
#Total charge convert object to float.
df['TotalCharges'] = df["TotalCharges"].replace(" ","0") # replace nulll value in to 0
df['TotalCharges'] = df["TotalCharges"].astype("float")
```

```
df.info() # after converting total chart in float
```

```
<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()
```

```
                  0
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.isnull().sum().sum() # Check null values in all data.
```

```
0
```

```
df.describe() # Descrive the data
```

	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.duplicated().sum() # check duplicate values
```

```
0
```

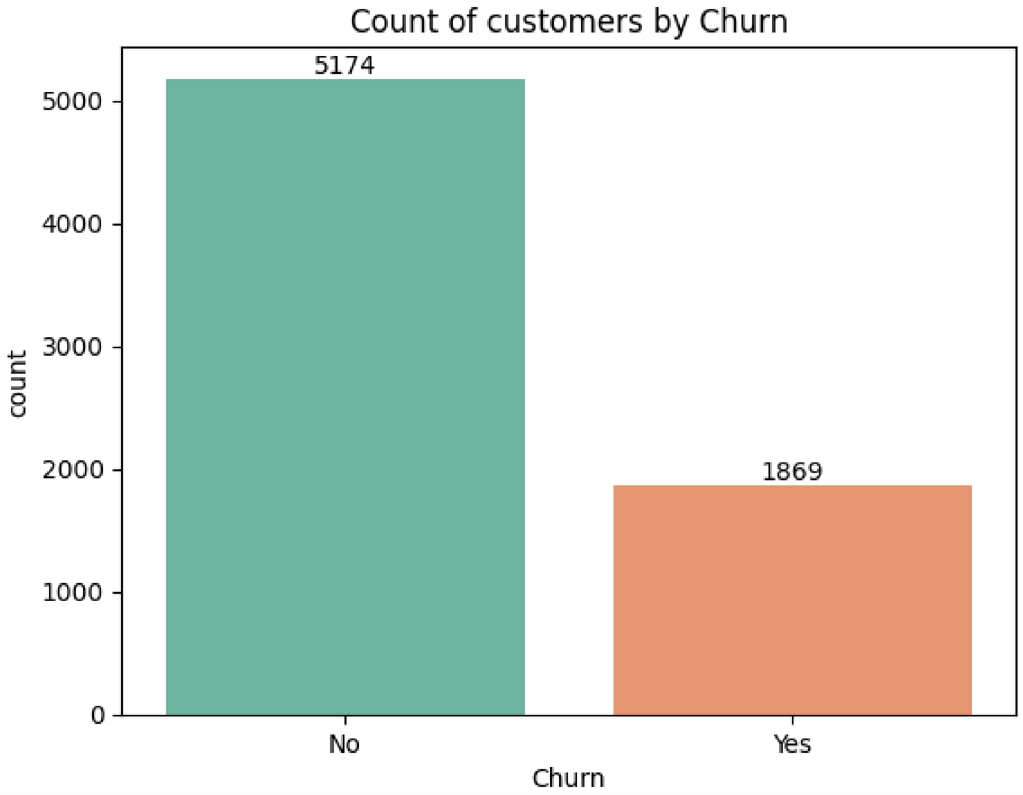
```
df["customerID"].duplicated().sum() # Check duplicated value behalf of custumerID
```

```
0
```

```
# define function to SeniorCitizen column values convert 0,1 to yes or no
def conv(value):
  if value == 1:
    return "yes"
  else:
    return "no"
df['SeniorCitizen'] = df['SeniorCitizen'].apply(conv) # Apply function in Seniorcitizen column
```

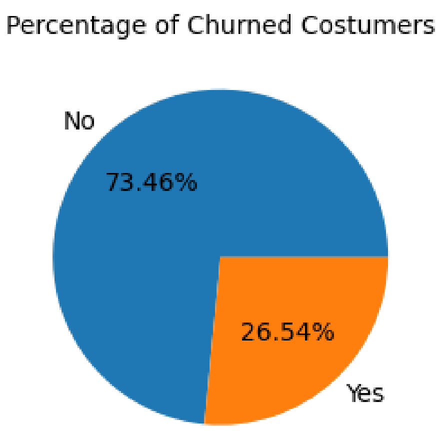
```
ax = sns.countplot(x='Churn', data=df, palette='Set2', hue = 'Churn')
ax.bar_label(ax.containers[0])
ax.bar_label(ax.containers[1])
```

```
plt.title("Count of customers by Churn")
plt.show()
```



Count Churn out Costomers by using Countplot ther are 1869 costumer Churn out.

```
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 Churned Costumers", fontsize =10)
plt.show()
```

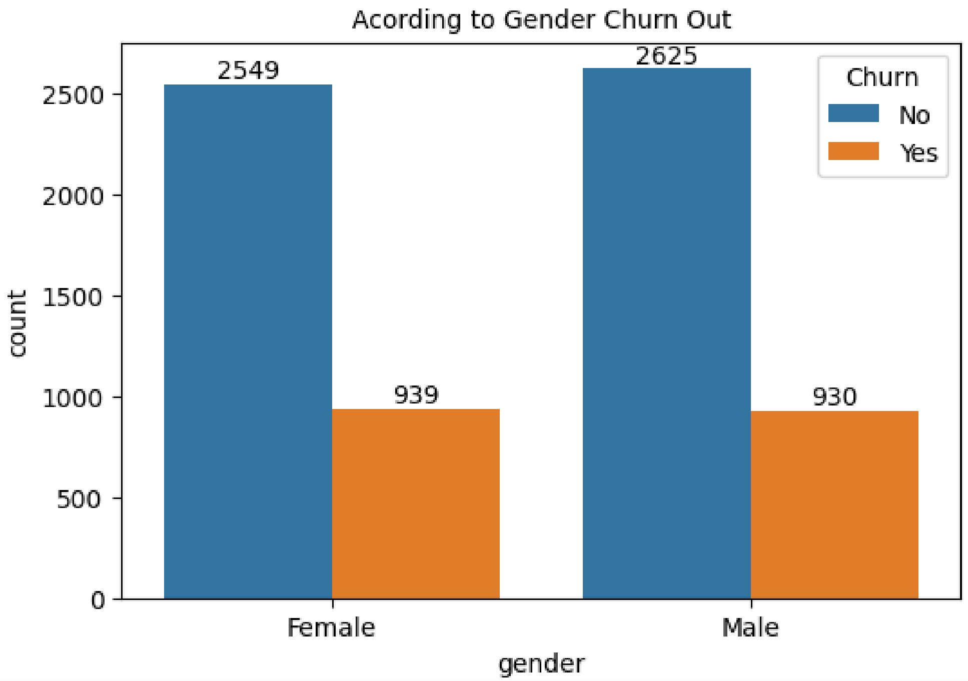


The given pie chart we can cnclud 26.54% (1869) costumer have churned out

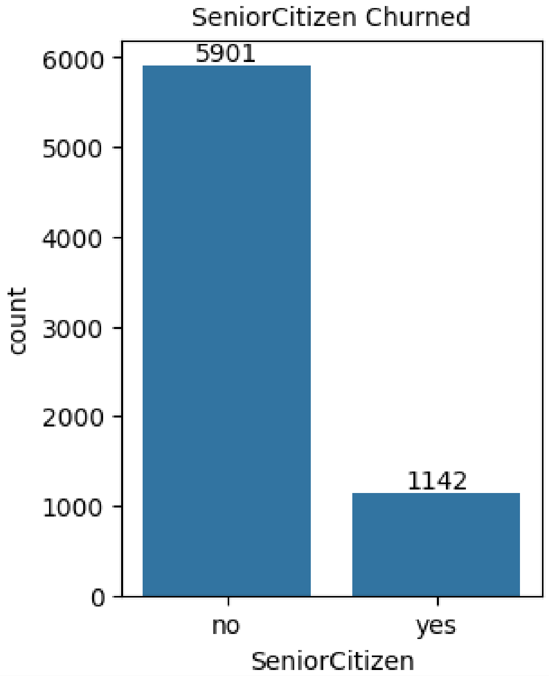
```
df.columns # Show all Columns
```

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

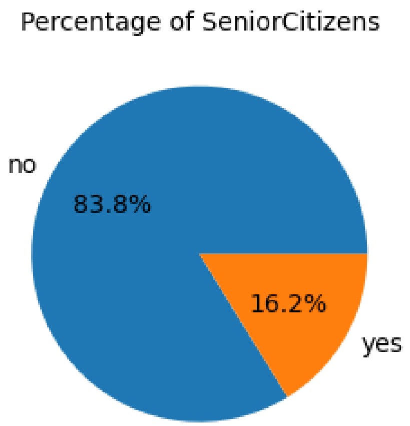
```
plt.figure(figsize=(6,4))
ax = sns.countplot(x= 'gender', data = df, hue = 'Churn')
ax.bar_label(ax.containers[0])
ax.bar_label(ax.containers[1])
plt.title("Acording to Gender Churn Out", fontsize = 10)
plt.show()
```



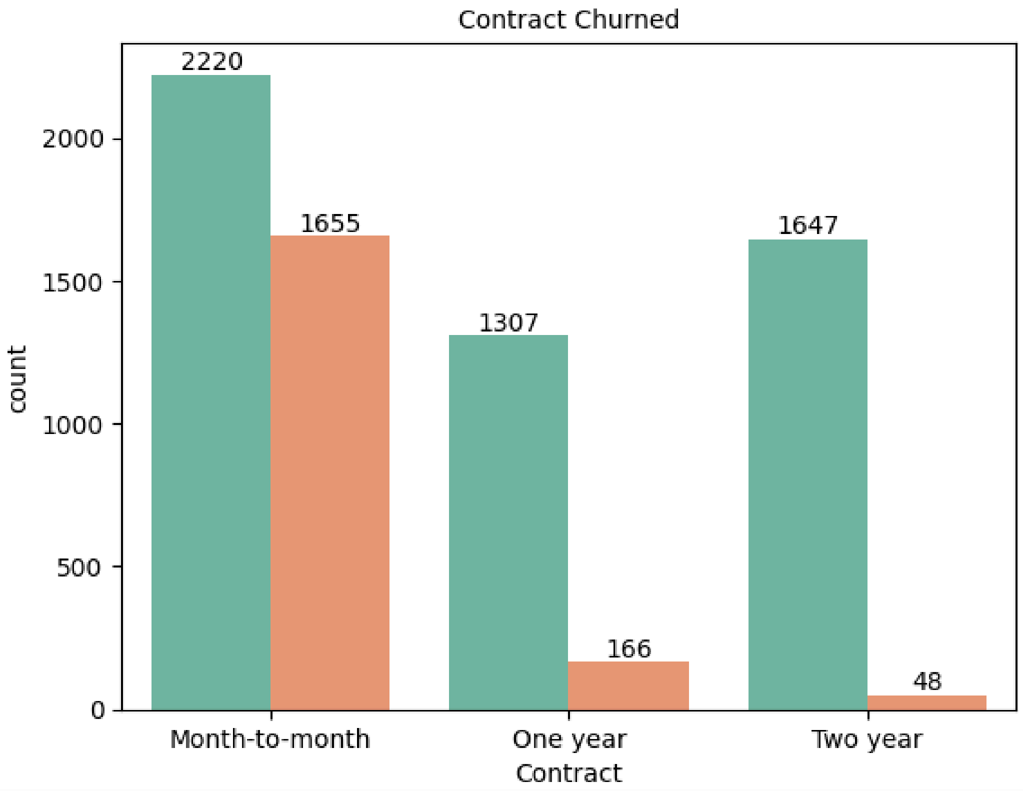
```
# Count plot for seniorcitizen churned out
plt.figure(figsize=(3,4))
ax = sns.countplot(x= 'SeniorCitizen', data = df)
ax.bar_label(ax.containers[0])
plt.title("SeniorCitizen Churned", fontsize = 10)
plt.show()
```



```
# Pie Chart for seniorcitizen churned out
plt.figure(figsize=(3,4))
gb = df.groupby("SeniorCitizen").agg({'SeniorCitizen':'count'})
plt.pie(gb["SeniorCitizen"], labels=gb.index, autopct = '%1.1f%%')
plt.title("Percentage of SeniorCitizens", fontsize =10)
plt.show()
```



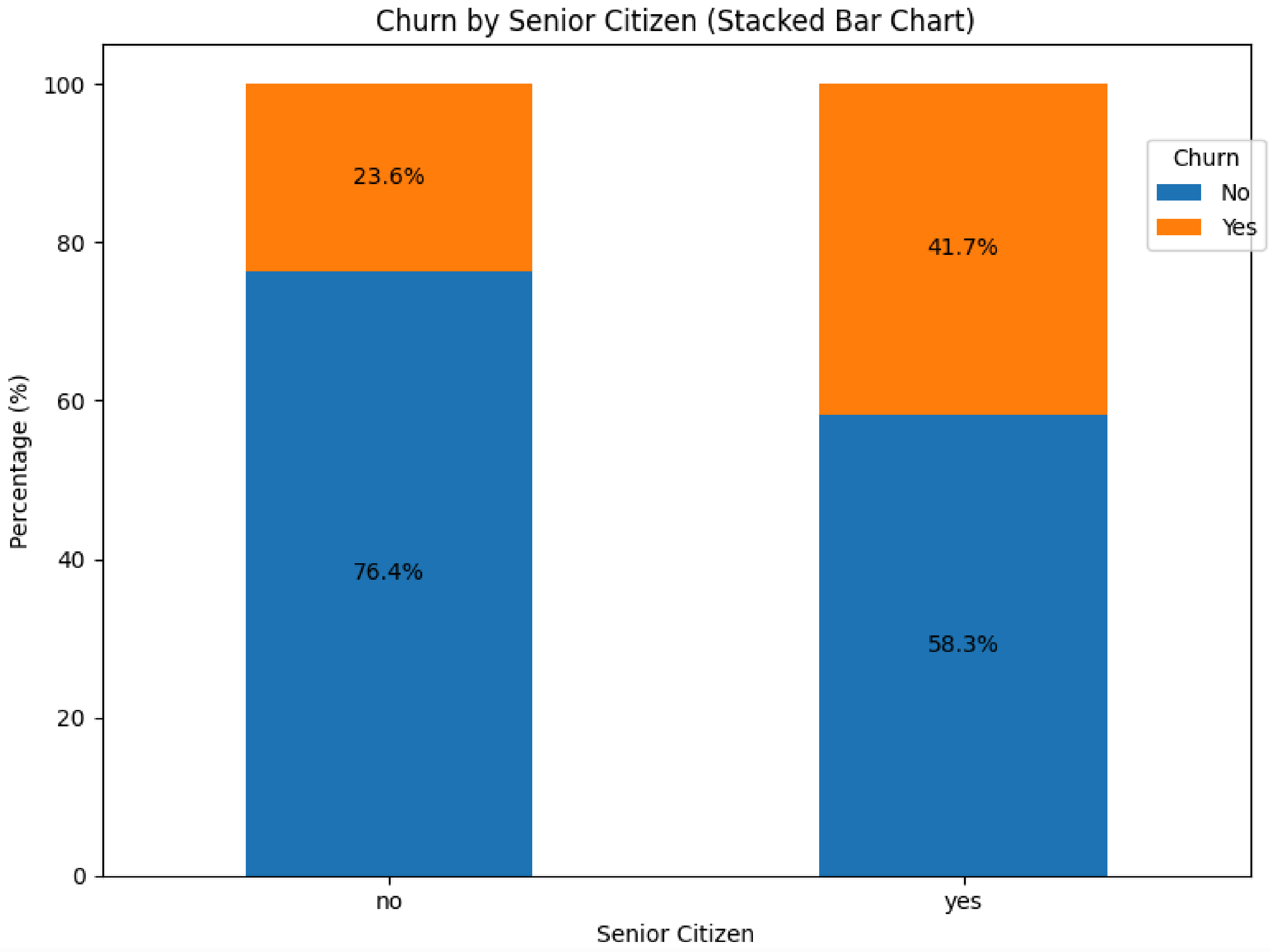
```
# Churned out by contrect basis.
ax= sns.countplot(x = 'Contract', data =df, palette='Set2',legend=False, hue='Churn')
ax.bar_label(ax.containers[0])
ax.bar_label(ax.containers[1])
plt.title("Contract Churned", fontsize = 10)
plt.show()
```



Uslly short term contract based people churn out early stage and who take longer duration contract they connect with us.

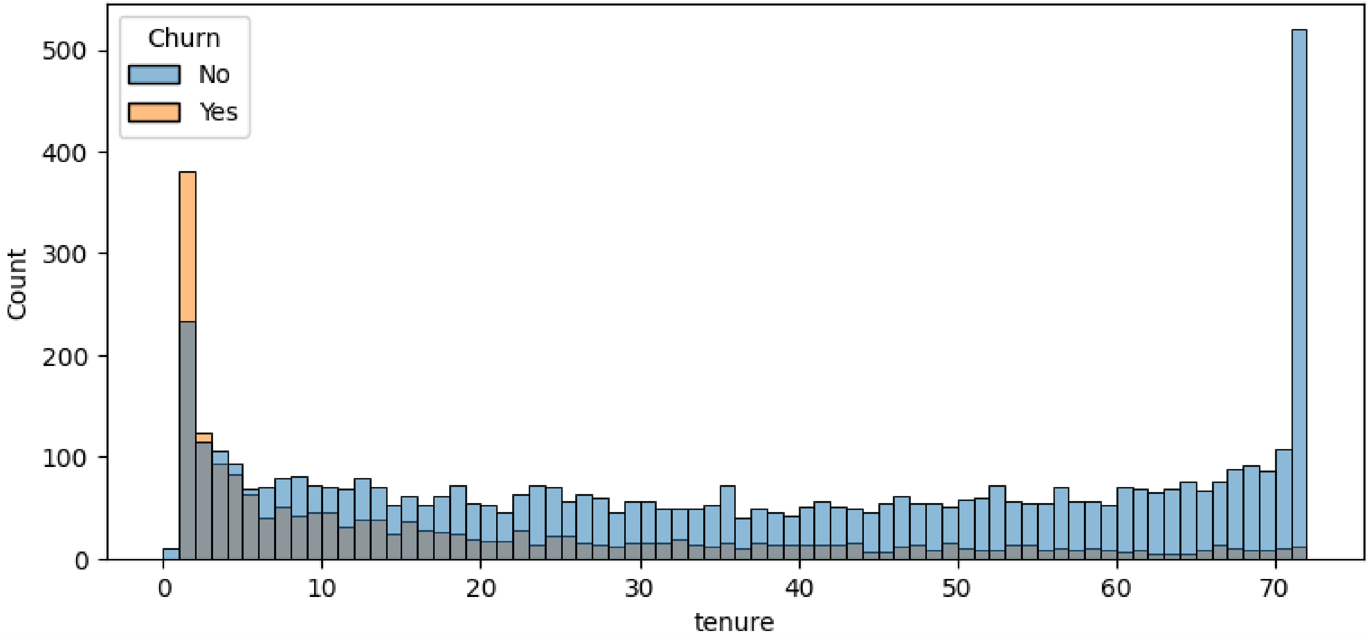
```
# Calculate total churn counts grouped by 'SeniorCitizen' and 'churn', with percentage normalization
total_counts = df.groupby('SeniorCitizen')['Churn'].value_counts(normalize=True).unstack() * 100

# Plot
fig, ax = plt.subplots(figsize=(8,6))  # Adjust figsize for better visualization

# Plot the stacked bar chart
total_counts.plot(kind='bar', stacked=True, ax=ax, color=['#1f77b4', '#ff7f0e'])

# Add percentage labels on the bars
for p in ax.patches:
    width, height = p.get_width(), p.get_height()
    x, y = p.get_xy()
    ax.text(x + width / 2, y + height / 2, f'{height:.1f}%', ha='center', va='center', fontsize=10)
```

```
# Title and labels
plt.title('Churn by Senior Citizen (Stacked Bar Chart)')
plt.xlabel('Senior Citizen')
plt.ylabel('Percentage (%)')

# Rotate x-axis labels if needed
plt.xticks(rotation=0)

# Customize legend location
plt.legend(title='Churn', bbox_to_anchor=(0.9, 0.9))

# Show the plot
plt.tight_layout()  # Ensure layout fits well
plt.show()
```



Above Stacked bar chart we show Churn out Senior Citizen acording to data

```
# Who have used sevices used services use short time and churned.
plt.figure(figsize=(9,4))
sns.histplot(x= 'tenure', data = df, bins = 72, hue = 'Churn')
plt.show()
```



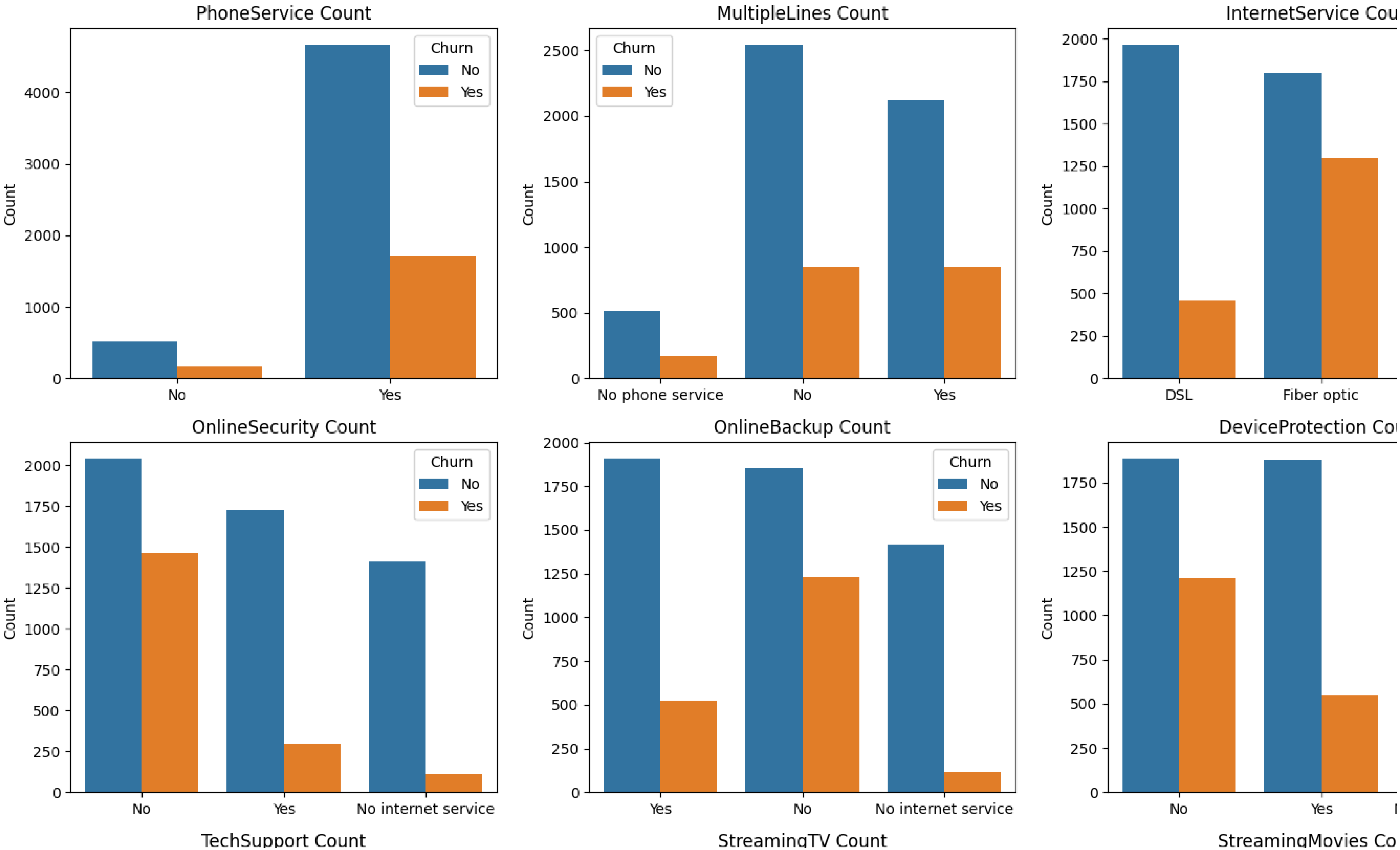
People who have used our service for a long time 72 months and people who have used our services one or two month. This chart connect to contrat chart which people tak yearly plane who stay with long terms.

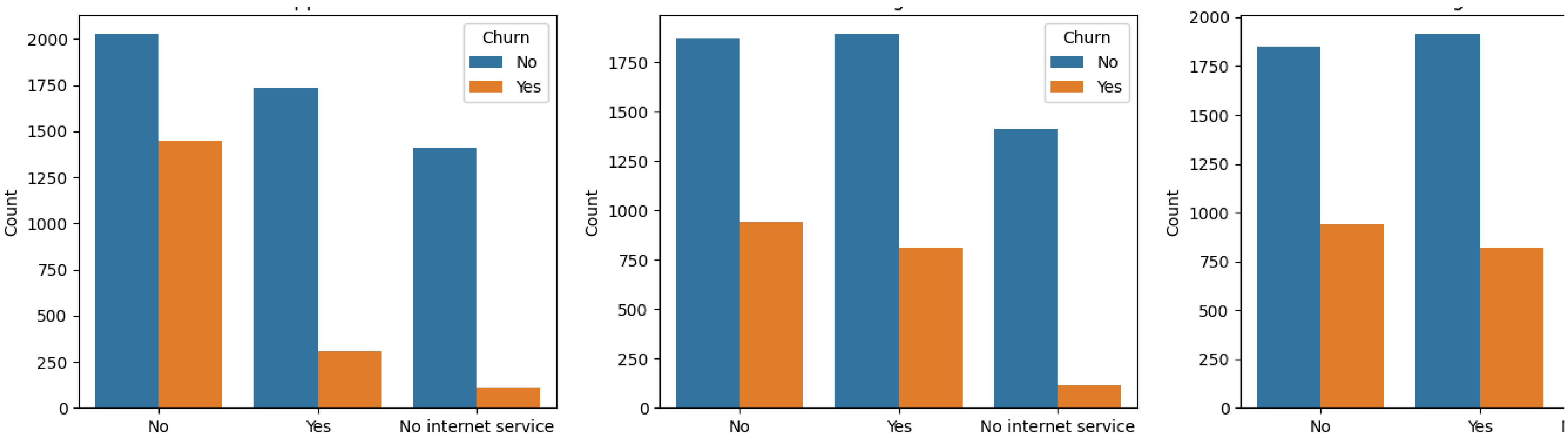
```
df.columns.values # Check the column values for making subplot.
```

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

```
# List of columns to create countplots for
columns = ['PhoneService', 'MultipleLines', 'InternetService', 'OnlineSecurity', 'OnlineBackup',
           'DeviceProtection', 'TechSupport', 'StreamingTV', 'StreamingMovies']

# Create subplots with 3 rows and 3 columns (adjust rows and cols if needed)
fig, axes = plt.subplots(nrows=3, ncols=3, figsize=(15, 12))

# Flatten the axes array for easier iteration
```

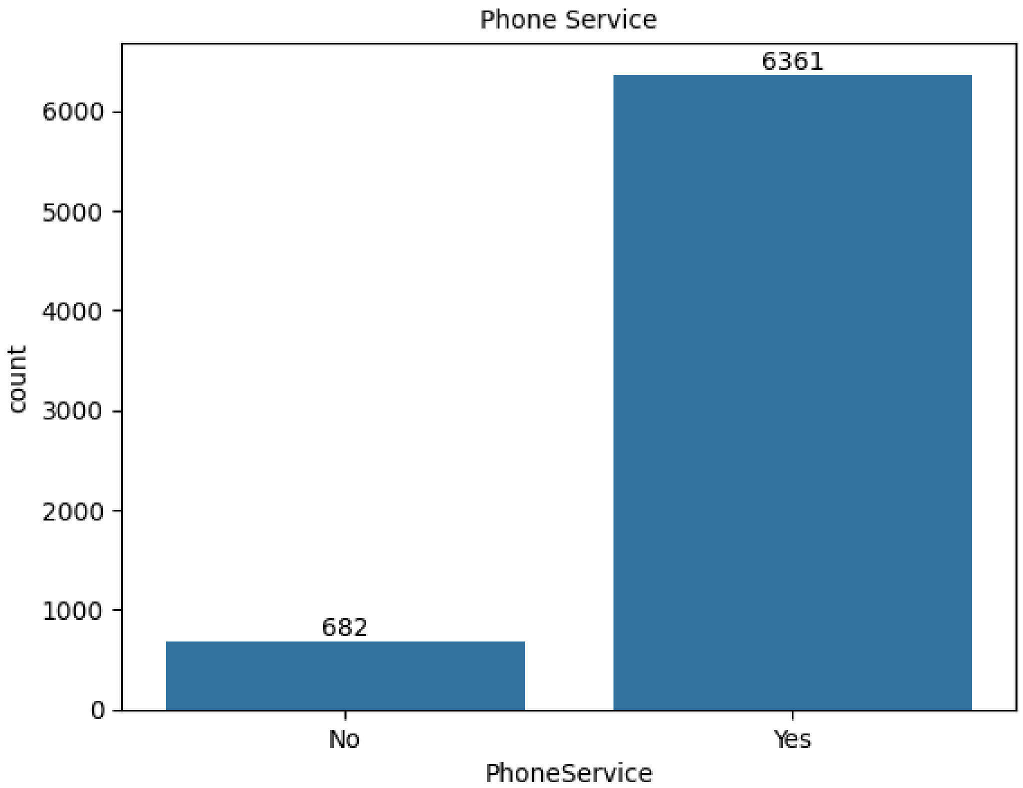
```
axes = axes.flatten()

# Loop through the columns and corresponding axes to create countplots
for i, col in enumerate(columns):
    sns.countplot(x=col, data=df, ax=axes[i], hue = df['Churn'])
    axes[i].set_title(f'{col} Count')
    axes[i].set_xlabel('')
    axes[i].set_ylabel('Count')

# Adjust layout for better visualization
plt.tight_layout()

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


PhoneService Count
Count
0
1000
2000
3000
4000
No
Yes
Churn
No
Yes
MultipleLines Count
Count
0
500
1000
1500
2000
2500
No phone service
No
Yes
Churn
No
Yes
InternetService Cou
Count
0
250
500
750
1000
1250
1500
1750
2000
DSL
Fiber optic
OnlineSecurity Count
Count
0
250
500
750
1000
1250
1500
1750
2000
No
Yes
No internet service
Churn
No
Yes
OnlineBackup Count
Count
0
250
500
750
1000
1250
1500
1750
2000
Yes
No
No internet service
Churn
No
Yes
DeviceProtection Co
Count
0
250
500
750
1000
1250
1500
1750
No
Yes
TechSupport Count
StreamingTV Count
StreamingMovies Co



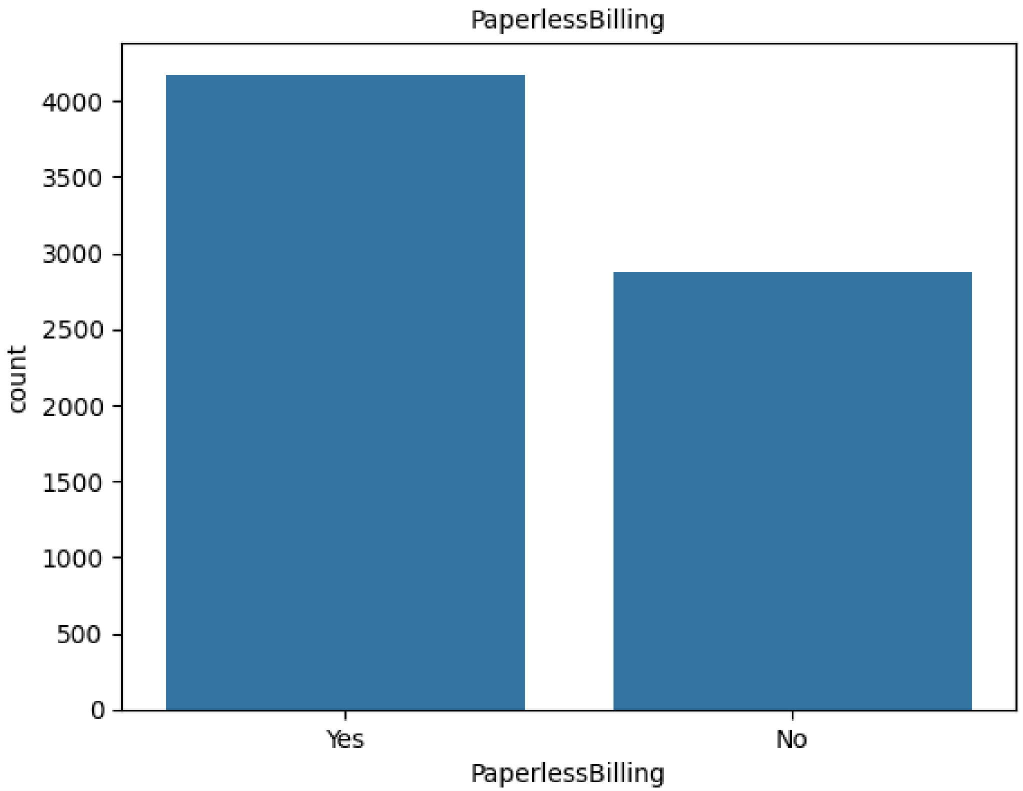
In this chart indicates that customers with no internet-related add-ons (like online security, backup, device protection, or tech support) show higher churn rates. Additionally, customers with fiber optic internet or multiple lines are more likely to churn compared to those using DSL or a single line. Basic services like phone service generally have lower churn rates.

```
# Find people who using phone services.
ax= sns.countplot(x = 'PhoneService', data = df)
ax.bar_label(ax.containers[0])
plt.title('Phone Service', fontsize = 10)
plt.show()
```



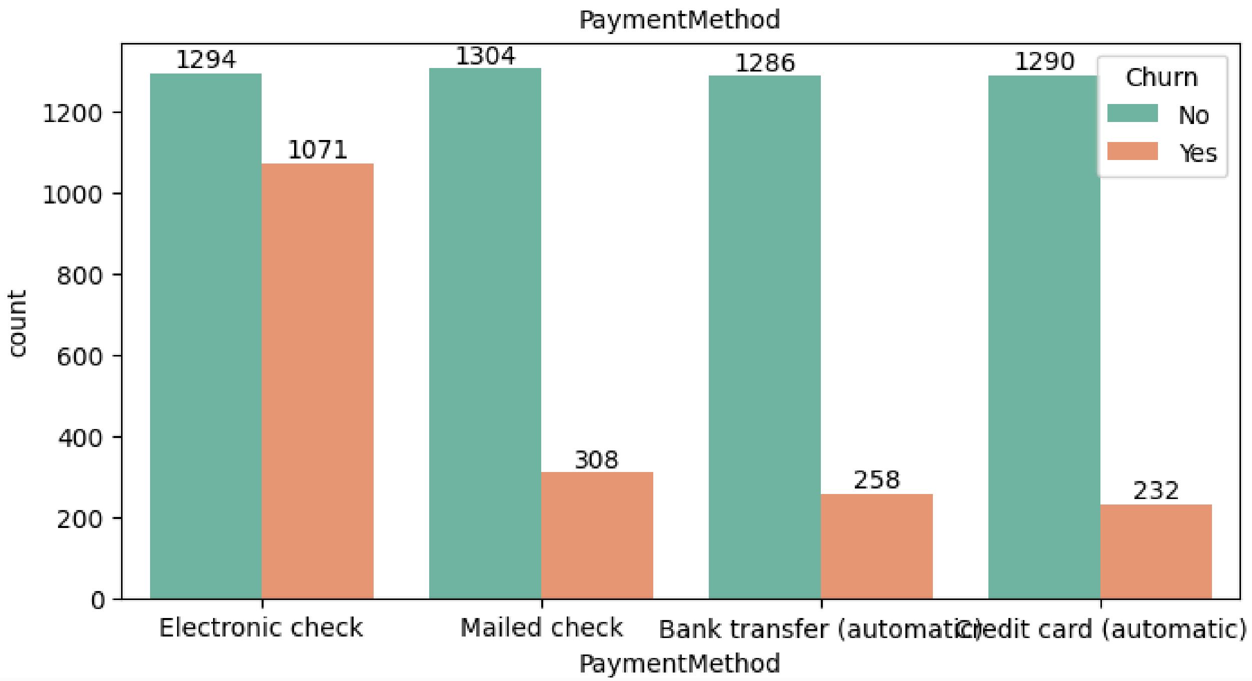
Here above graph we show clearly most of the people using PhoneServices

```
# PaperlessBilling
sns.countplot(x = 'PaperlessBilling', data = df)
plt.title('PaperlessBilling', fontsize = 10)
plt.show()
```



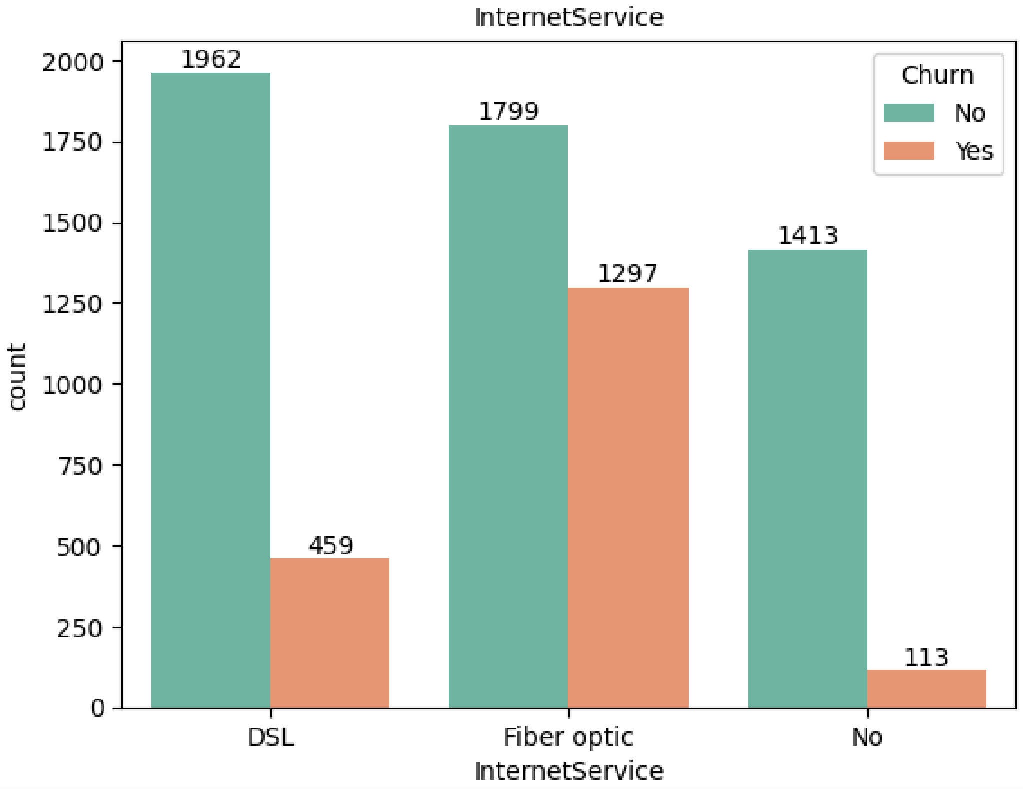
In this Chart showing people who using peperless Billing.

```
# PaymentMethod
plt.figure(figsize=(8,4))
ax = sns.countplot(x = 'PaymentMethod', data = df, palette = 'Set2', hue='Churn')
ax.bar_label(ax.containers[0])
ax.bar_label(ax.containers[1])
plt.title('PaymentMethod', fontsize = 10)
plt.show()
```



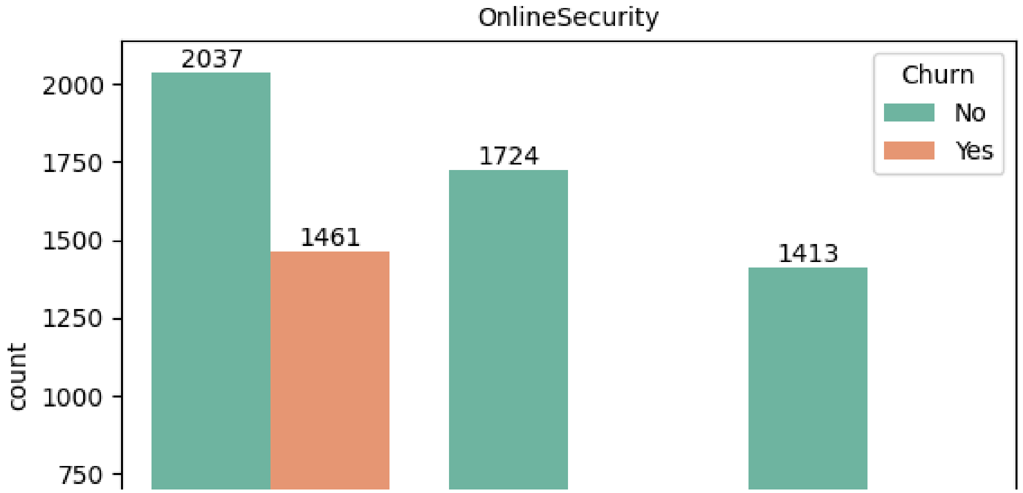
Above Graph showing Payment Method it's very importent chart for us Churn out less, where Automatic payment method used

```
# InternetService
ax= sns.countplot(x = 'InternetService', data = df, palette = 'Set2', hue='Churn')
ax.bar_label(ax.containers[0])
ax.bar_label(ax.containers[1])
plt.title('InternetService', fontsize = 10)
plt.show()
```



Most of the people who churn out they used Fiber optic Internet Service use.

```
# OnlineSecurity
ax = sns.countplot(x = 'OnlineSecurity', data = df, palette='Set2', hue='Churn')
ax.bar_label(ax.containers[0])
ax.bar_label(ax.containers[1])
plt.title('OnlineSecurity', fontsize = 10)
plt.show()
```



Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.