customer-churn-analysis

June 25, 2025

Customer Churn Analysis

```
[81]: 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(3)
```

```
[81]:    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

            MultipleLines InternetService OnlineSecurity  … DeviceProtection  \
      0  No phone service             DSL             No  …               No
      1                No             DSL            Yes  …              Yes
      2                No             DSL            Yes  …               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

            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 rows x 21 columns]
```

```
[82]: df.shape  #checking the shape of the dataset
```

```
[82]: (7043, 21)
```

```
[83]: df.isnull().sum() #checking the null values in the dataset
```

[83]:
```
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
```

Replacing Blank Values with 0

[84]:
```
df["TotalCharges"]=df["TotalCharges"].replace(" ","0")
df["TotalCharges"]=df["TotalCharges"].astype ("float")  #Change data type of␣
 ↪TotalCharges (object to float)
```

[85]:
```
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
```

[86]:
```
df.describe()
```

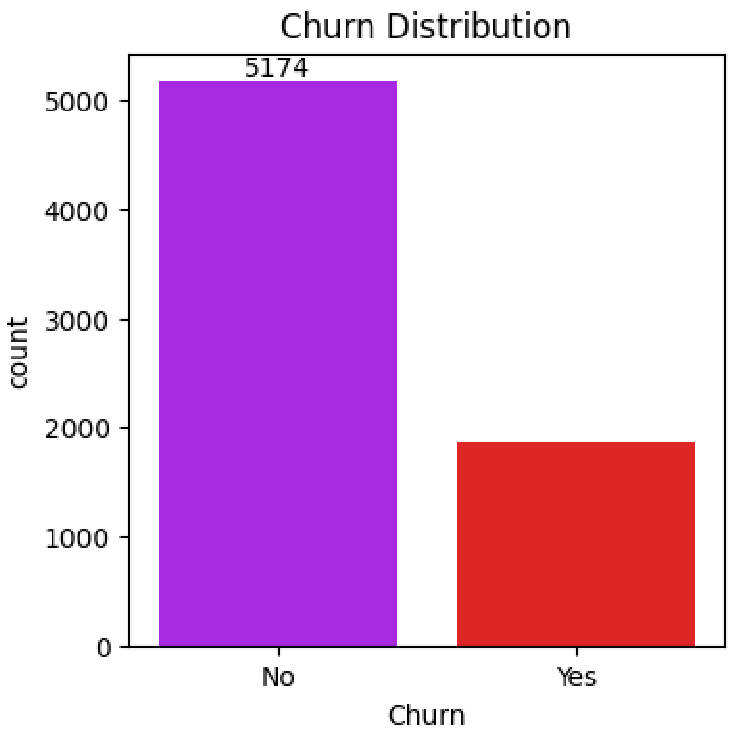
[86]:
```
       SeniorCitizen       tenure  MonthlyCharges  TotalCharges
count    7043.000000  7043.000000     7043.000000   7043.000000
mean        0.162147    32.371149       64.761692   2279.734304
std         0.368612    24.559481       30.090047   2266.794470
min         0.000000     0.000000       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
```

[87]:
```
df.duplicated().sum() # checking duplicate values present in data sets
```

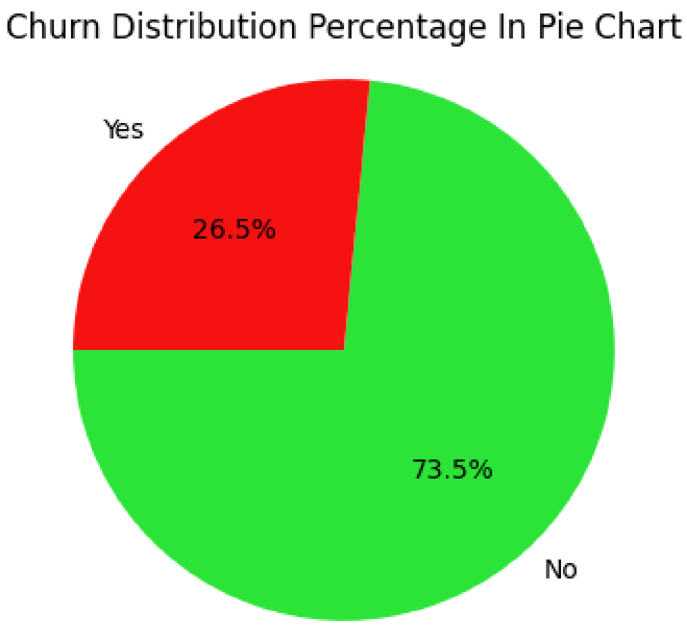
[87]:
```
np.int64(0)
```

[88]:
```
def conv(value):
    if value == 1:
        return "Yes"
    else:
        return "No"
    df["SeniorCitizen"]=df["SeniorCitizen"].apply(conv)
```

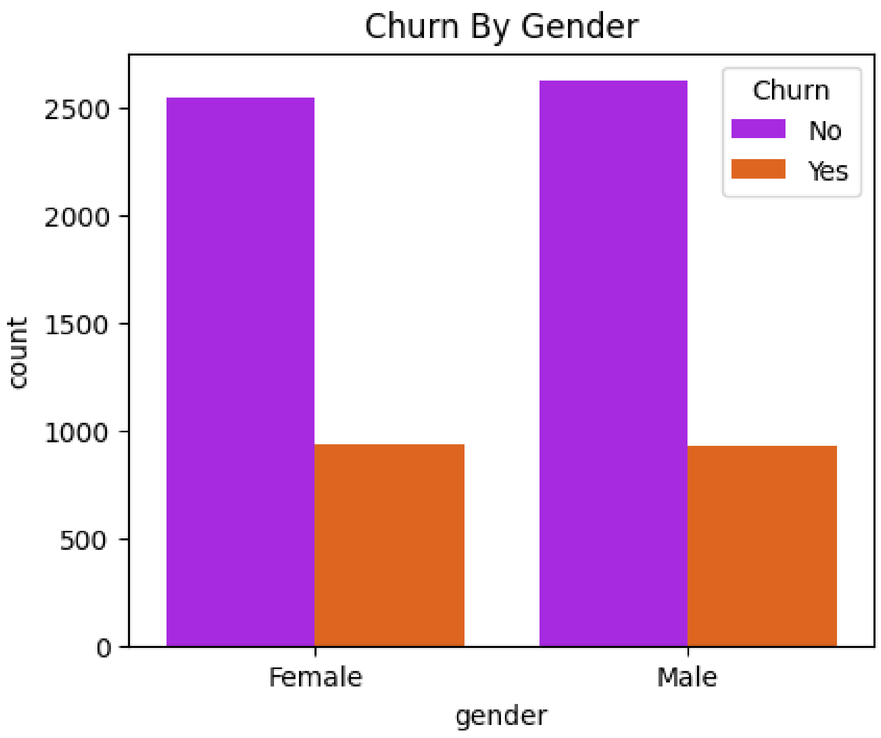
[89]:
```
plt.figure(figsize=(4, 4))
ax=sns.countplot(x='Churn', data=df,hue='Churn', palette=["#b20cff","#fc0707"])
ax.bar_label(ax.containers[0])
plt.title('Churn Distribution')
plt.show()
```



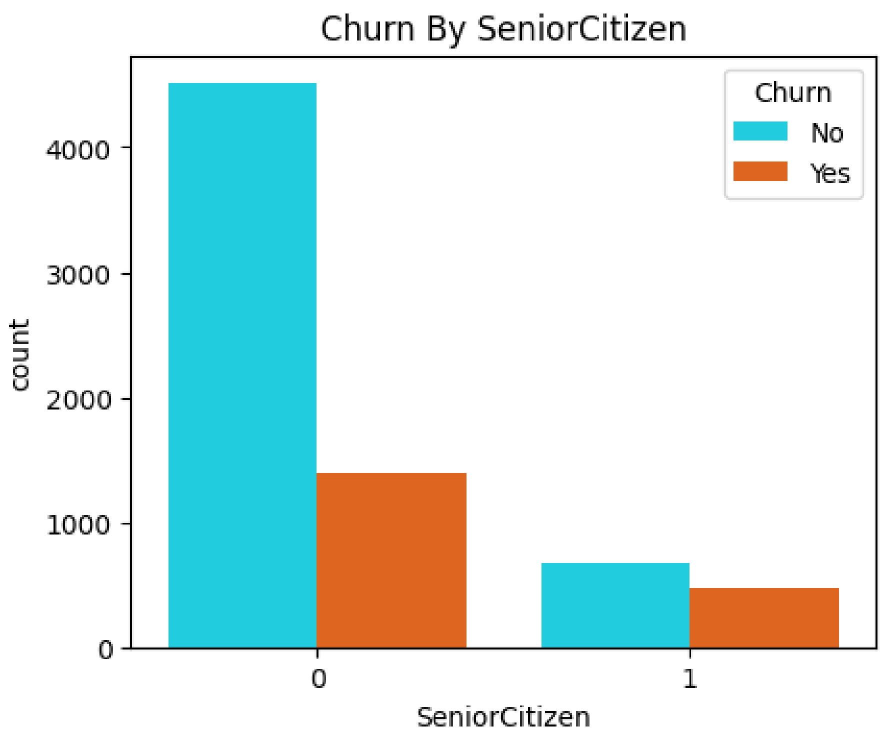
```
[90]: plt.figure(figsize=(4, 4))
      gb=df.groupby('Churn').agg({"Churn":"count"})
      plt.pie(gb["Churn"],labels=gb.index,autopct='%1.1f%%', startangle=180,␣
       ↪colors=["#2be437","#f71212"])
      plt.title('Churn Distribution Percentage In Pie Chart')
      plt.axis('equal')  # Equal aspect ratio ensures that pie chart is a circle.
      plt.show()
```



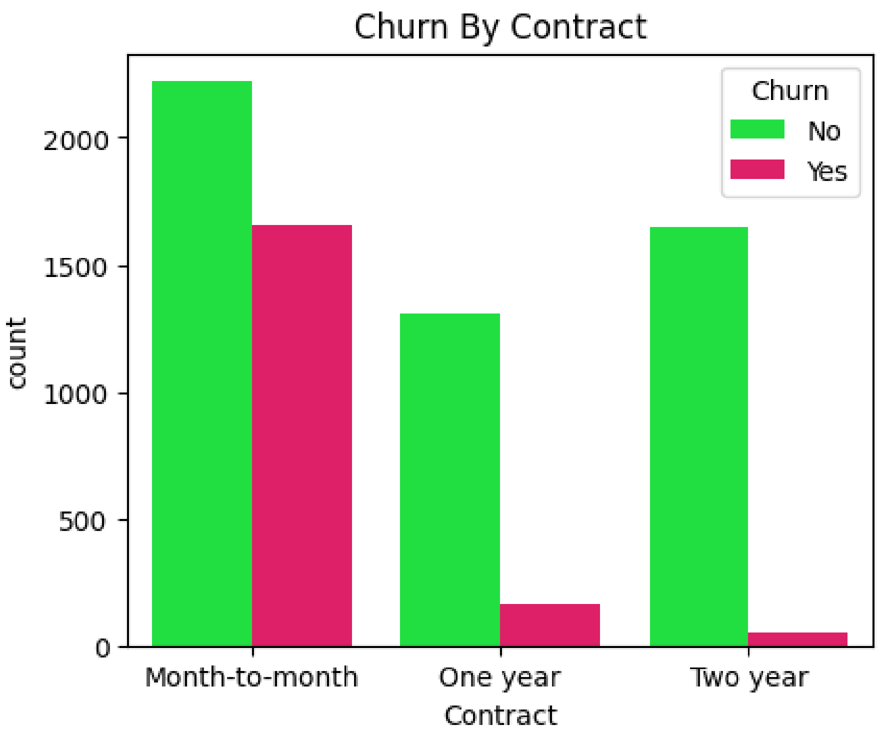
[91]:
```
plt.figure(figsize=(5,4))
sns. countplot(x="gender", data=df, hue="Churn", palette=["#b20cff","#fc5c00"])
plt.title("Churn By Gender")
plt.show()
```



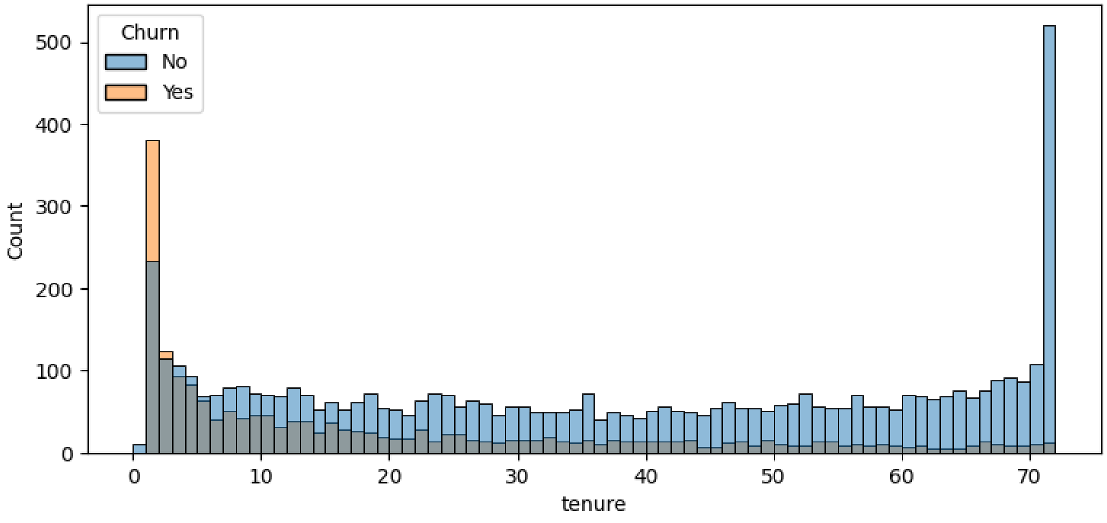
[92]:
```
plt.figure(figsize=(5,4))
sns. countplot(x="SeniorCitizen", data=df, hue="Churn",␣
 ↪palette=["#02e5fe","#fc5c00"])
plt.title("Churn By SeniorCitizen")
plt.show()
```



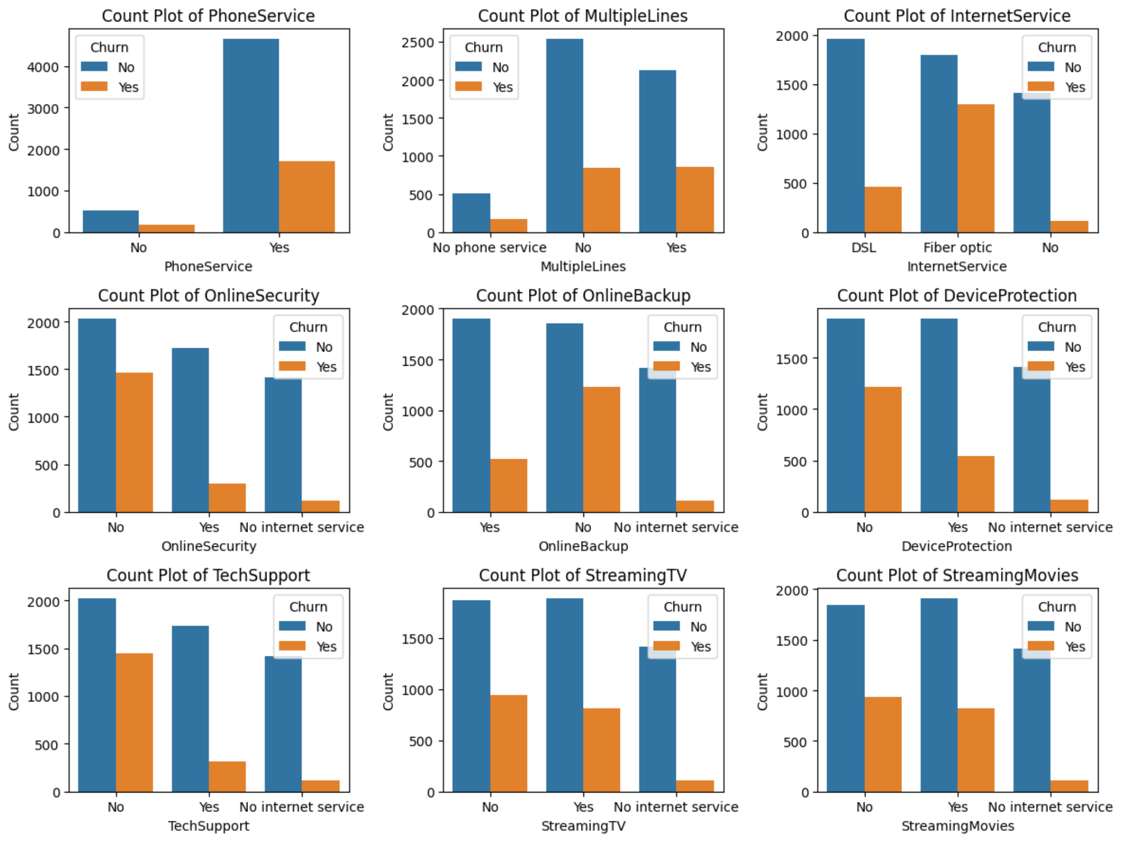
[93]:
```
plt.figure(figsize=(5,4))
sns. countplot(x="Contract", data=df, hue="Churn",␣
 ↪palette=["#02fe2c","#fc0061"])
plt.title("Churn By Contract")
plt.show()
```



[94]:
```
plt.figure(figsize = (9,4))
sns.histplot(x = "tenure", data = df, bins = 72, hue = "Churn")
plt.show()
```



[104]:
```
columns = ['PhoneService', 'MultipleLines', 'InternetService', 'OnlineSecurity',
           'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingTV',␣
↪'StreamingMovies']

# Number of columns for the subplot grid (you can change this)
n_cols = 3
n_rows = (len(columns) + n_cols - 1) // n_cols  # Calculate number of rows␣
↪needed

# Create subplots
fig, axes = plt.subplots(n_rows, n_cols, figsize=(12, n_rows * 3))  # Adjust␣
↪figsize as needed

# Flatten the axes array for easy iteration (handles both 1D and 2D arrays)
axes = axes.flatten()

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

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

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



#The majority of customers who do not churn tend to have services like PhoneService, InternetService (particularly DSL), and OnlineSecurity enabled. For services like OnlineBackup, TechSupport, and StreamingTV, churn rates are noticeably higher when these services are not used or are unavailable.

```
[ ]:
```