Case study on Customer Retention dataset, EDA and Feature selection

Rohit Bahadur Bista | AP21110010941

February 22, 2024

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

[2]:
```
df = pd.read_csv("C:/123/SRMAP/Semester 6/Applied Data Science/Customer␣
↪Retention Dataset based work/WA_Fn-UseC_-Telco-Customer-Churn.csv")
```

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

[3]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSupport	StreamingTV	StreamingMovies	Contract	PaperlessBilling	PaymentMethod	MonthlyCharges	TotalCharges	Churn
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	...	No	No	No	No	Month-to-month	Yes	Electronic check	29.85	29.85	No
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	...	Yes	No	No	No	One year	No	Mailed check	56.95	1889.5	No
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	...	No	No	No	No	Month-to-month	Yes	Mailed check	53.85	108.15	Yes
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes	...	Yes	Yes	No	No	One year	No	Bank transfer (automatic)	42.30	1840.75	No
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	...	No	No	No	No	Month-to-month	Yes				

```
4           Electronic check          70.70        151.65   Yes

[5 rows x 21 columns]
```

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

[4]:
```
      customerID  gender  SeniorCitizen Partner Dependents  tenure  \
7038  6840-RESVB    Male              0     Yes        Yes      24
7039  2234-XADUH  Female              0     Yes        Yes      72
7040  4801-JZAZL  Female              0     Yes        Yes      11
7041  8361-LTMKD    Male              1     Yes         No       4
7042  3186-AJIEK    Male              0      No         No      66

     PhoneService     MultipleLines InternetService OnlineSecurity  ...  \
7038          Yes               Yes             DSL            Yes  ...
7039          Yes               Yes     Fiber optic             No  ...
7040           No  No phone service             DSL            Yes  ...
7041          Yes               Yes     Fiber optic             No  ...
7042          Yes                No     Fiber optic            Yes  ...

     DeviceProtection TechSupport StreamingTV StreamingMovies        Contract  \
7038              Yes         Yes         Yes              Yes        One year
7039              Yes          No         Yes              Yes        One year
7040               No          No          No               No  Month-to-month
7041               No          No          No               No  Month-to-month
7042              Yes         Yes         Yes              Yes        Two year

     PaperlessBilling              PaymentMethod MonthlyCharges  TotalCharges  \
7038              Yes               Mailed check          84.80        1990.5
7039              Yes    Credit card (automatic)         103.20        7362.9
7040              Yes           Electronic check          29.60        346.45
7041              Yes               Mailed check          74.40         306.6
7042              Yes  Bank transfer (automatic)         105.65        6844.5

     Churn
7038    No
7039    No
7040    No
7041   Yes
7042    No

[5 rows x 21 columns]
```

[5]:
```
print(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
```

[6]:
```
df.shape
```

[6]:
```
(7043, 21)
```

[7]:
```
duplicate_rows_df = df[df.duplicated()]
print("number of duplicate rows: ", duplicate_rows_df.shape)
```

```
number of duplicate rows:  (0, 21)
```

[8]:
```
# Data Cleaning
df['TotalCharges'] = pd.to_numeric(df['TotalCharges'], errors='coerce')  #␣
 ↪Convert TotalCharges to numeric
df.dropna(inplace=True)
```

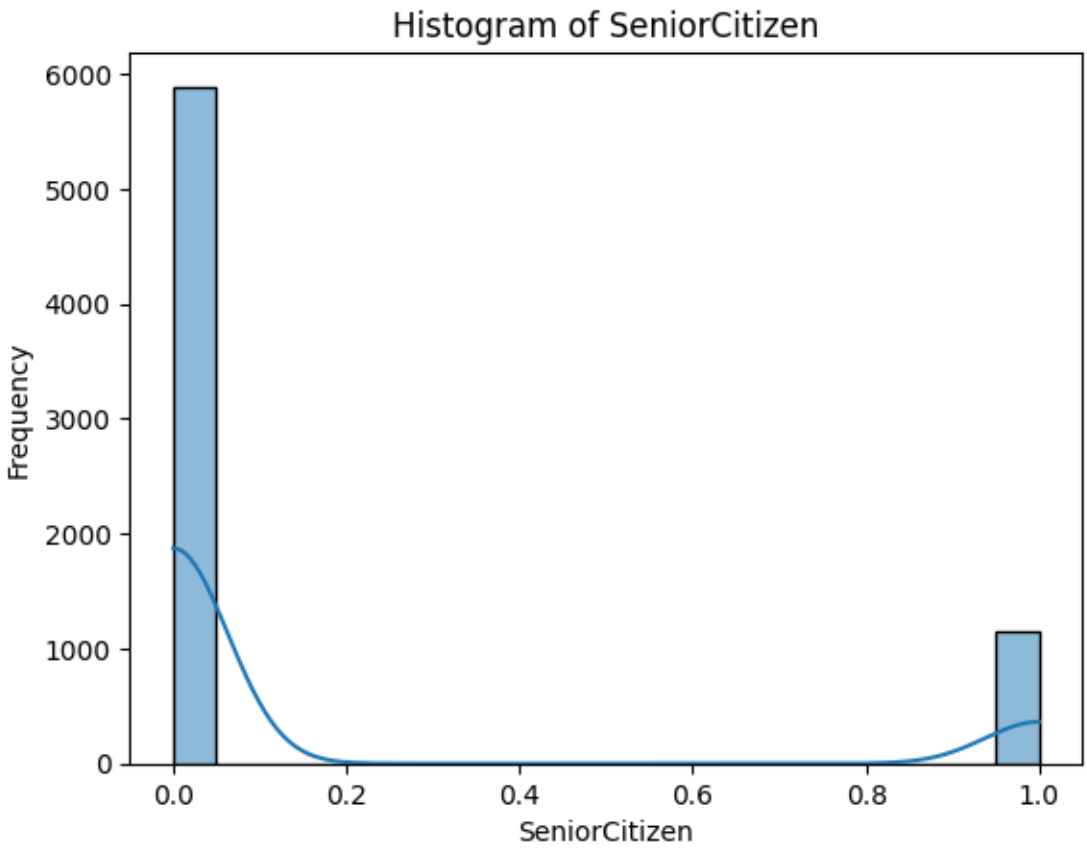
[9]:
```
summary_stats = df.describe()
print(summary_stats)
```

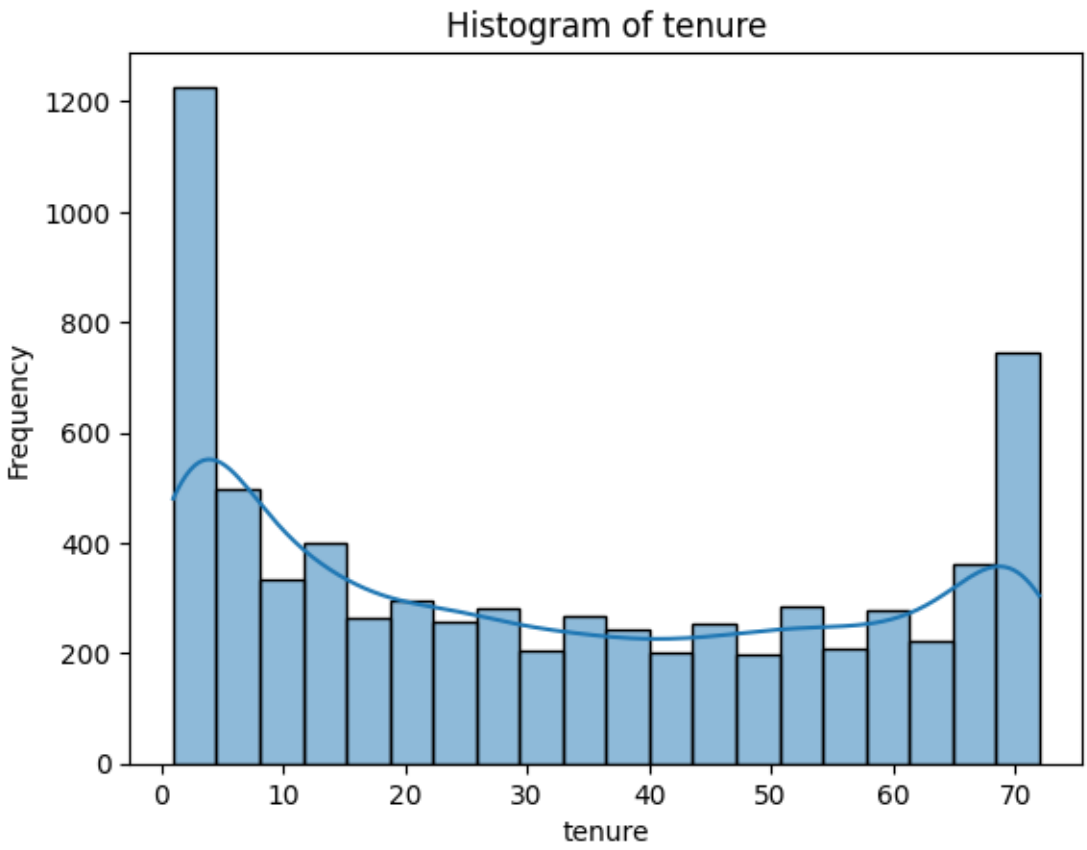
```
       SeniorCitizen       tenure  MonthlyCharges  TotalCharges
count    7032.000000  7032.000000     7032.000000   7032.000000
mean        0.162400    32.421786       64.798208   2283.300441
std         0.368844    24.545260       30.085974   2266.771362
min         0.000000     1.000000       18.250000     18.800000
25%         0.000000     9.000000       35.587500    401.450000
50%         0.000000    29.000000       70.350000   1397.475000
75%         0.000000    55.000000       89.862500   3794.737500
max         1.000000    72.000000      118.750000   8684.800000
```

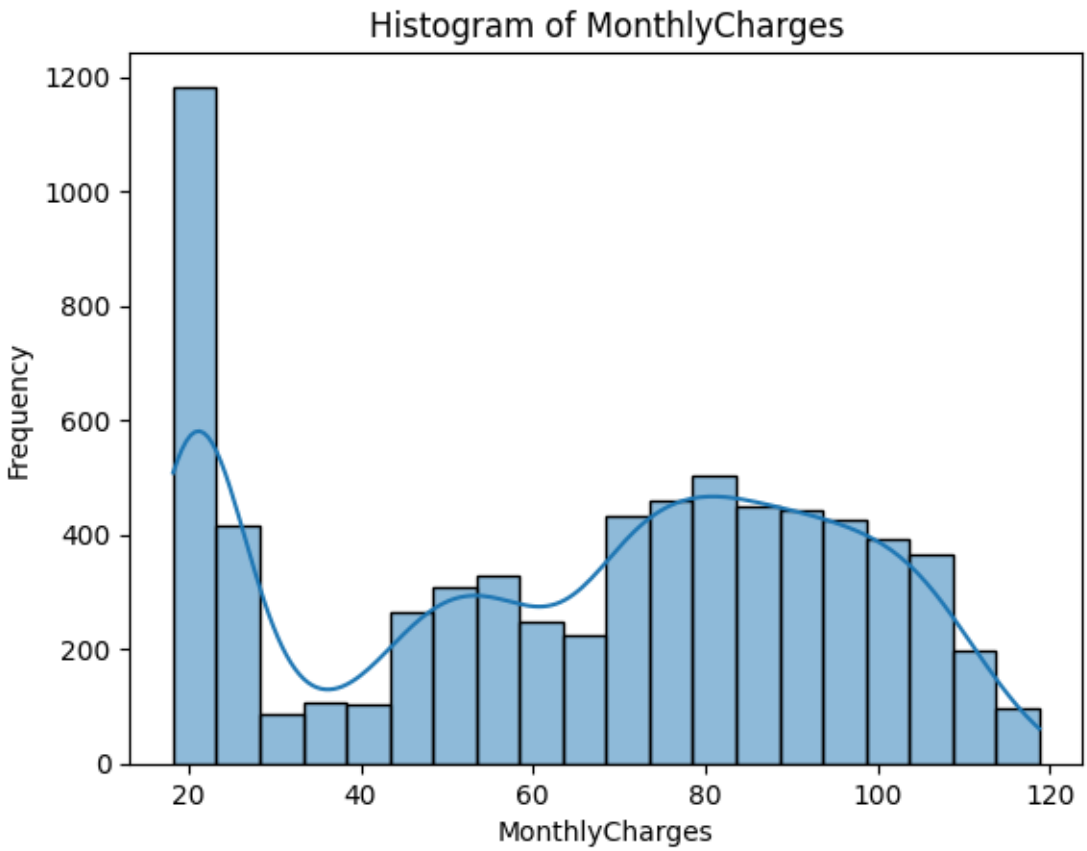
[10]:
```
print(df.dtypes)
```

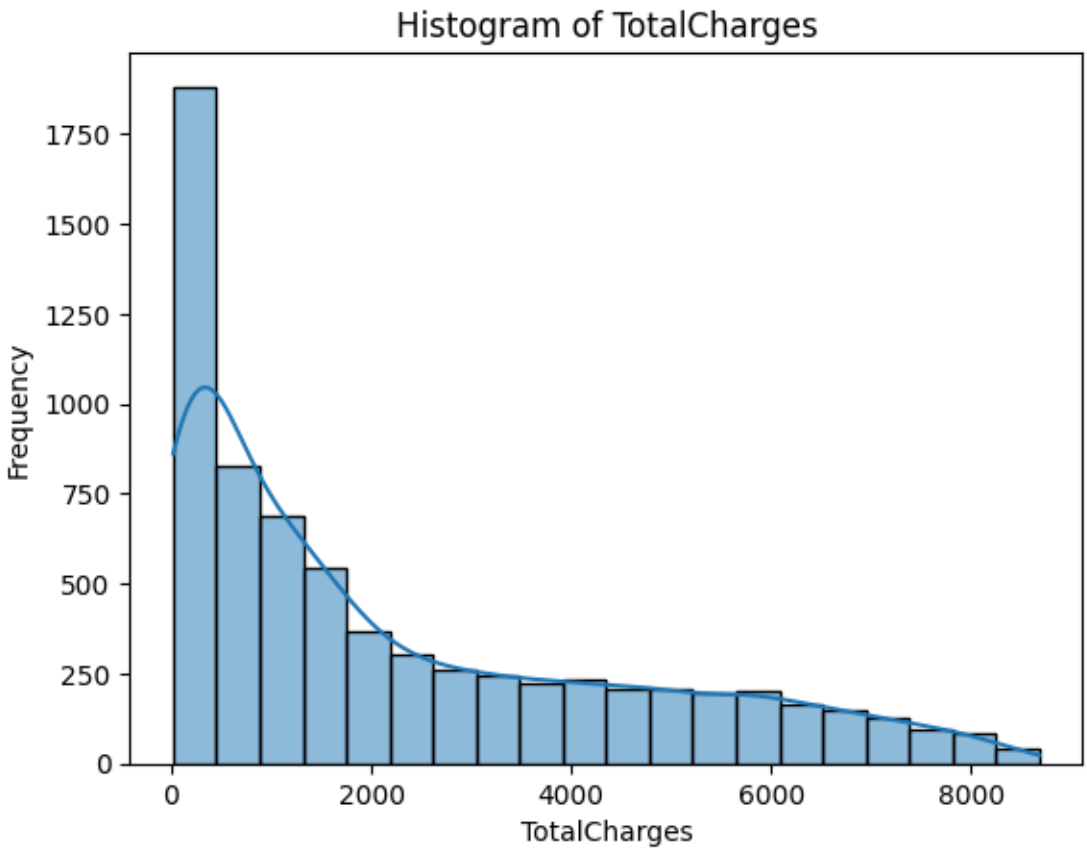
```
customerID           object
gender               object
SeniorCitizen         int64
Partner              object
Dependents           object
tenure                int64
PhoneService         object
MultipleLines        object
InternetService      object
OnlineSecurity       object
OnlineBackup         object
DeviceProtection     object
TechSupport          object
StreamingTV          object
StreamingMovies      object
Contract             object
PaperlessBilling     object
PaymentMethod        object
MonthlyCharges      float64
TotalCharges        float64
Churn                object
dtype: object
```

[11]:
```python
# Data Visualization
# Histogram of numerical variables
numerical_cols = df.select_dtypes(include=['int64', 'float64']).columns
for col in numerical_cols:
    plt.figure()
    sns.histplot(df[col], bins=20, kde=True)
    plt.title(f'Histogram of {col}')
    plt.xlabel(col)
    plt.ylabel('Frequency')
    plt.show()
```

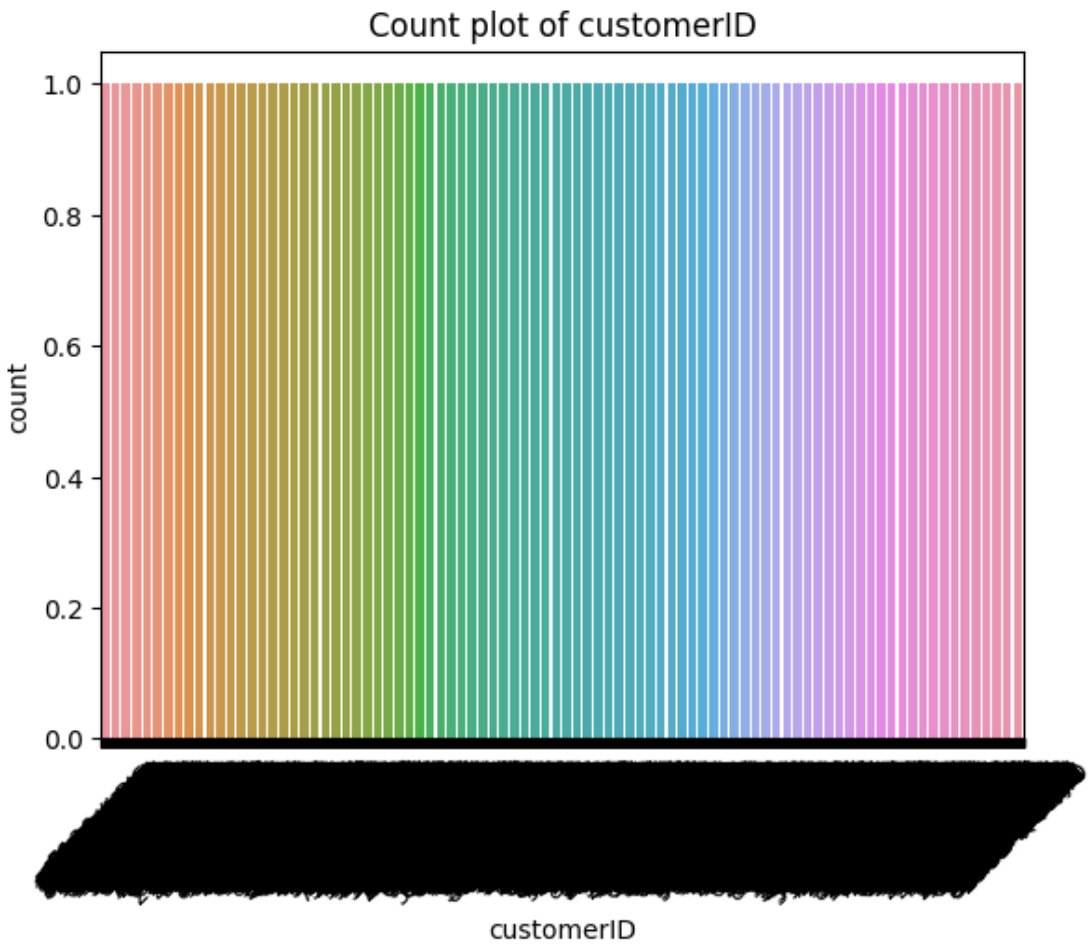

Histogram of SeniorCitizen
Frequency
SeniorCitizen
0
1000
2000
3000
4000
5000
6000
0.0
0.2
0.4
0.6
0.8
1.0

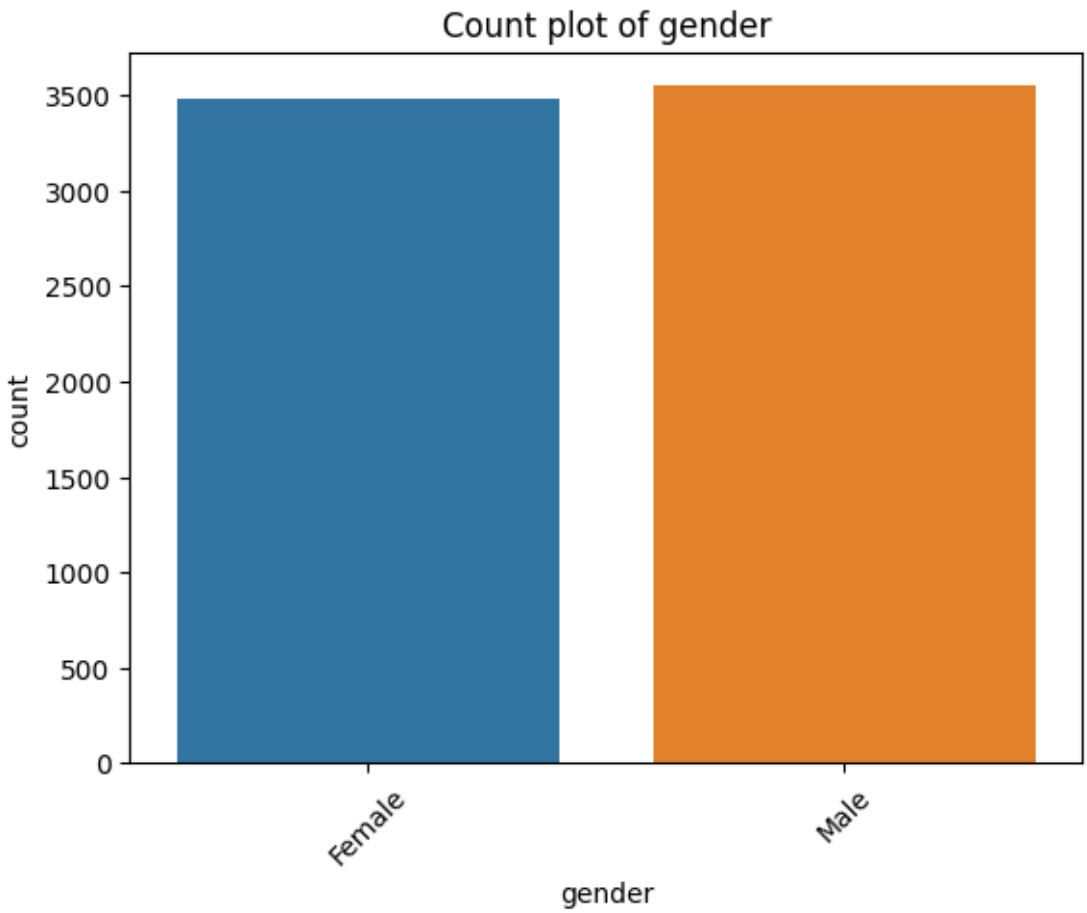

Histogram of tenure
Frequency
tenure
0
200
400
600
800
1000
1200
0
10
20
30
40
50
60
70

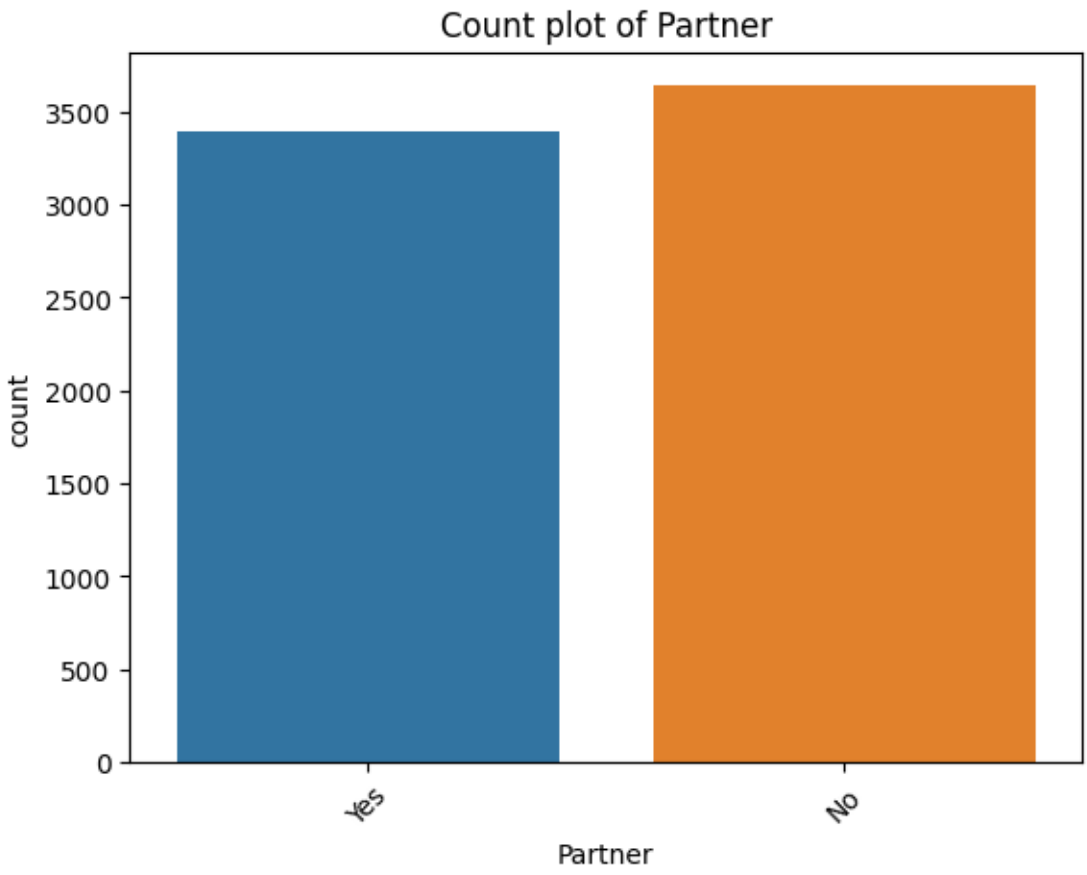

Histogram of MonthlyCharges
Frequency
MonthlyCharges
0
200
400
600
800
1000
1200
20
40
60
80
100
120

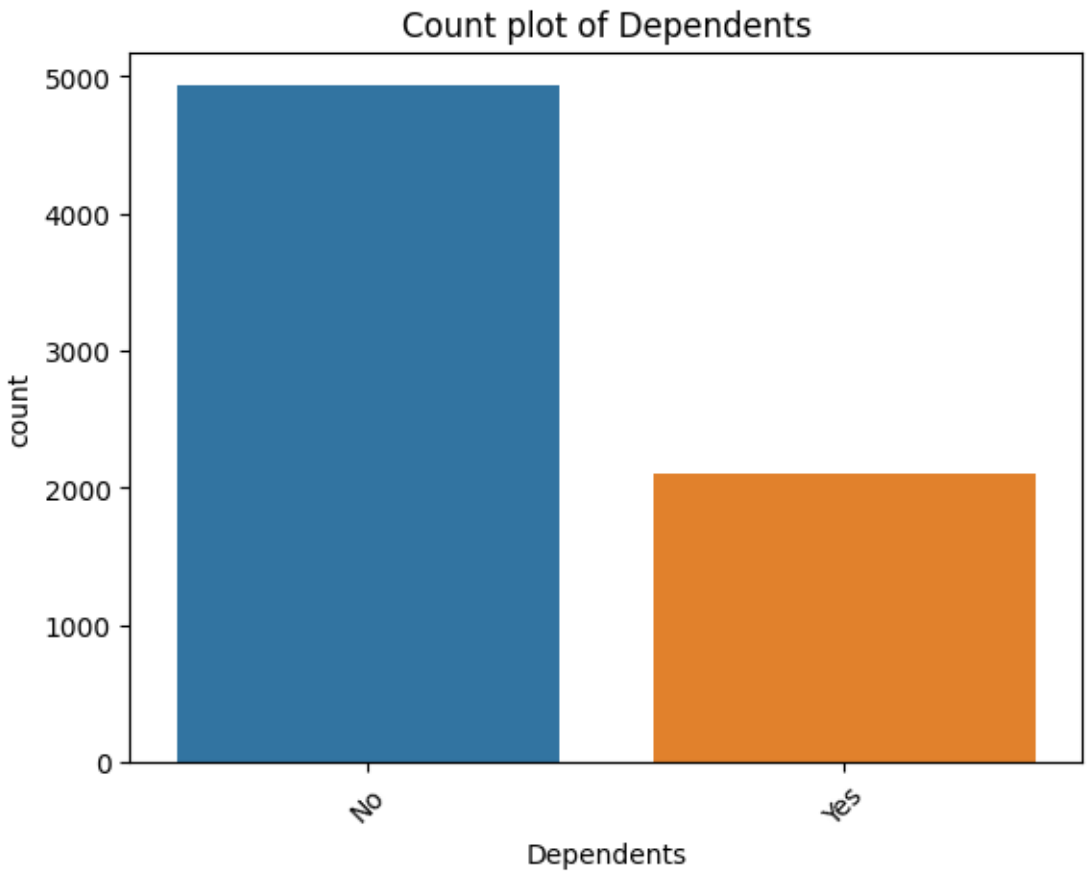


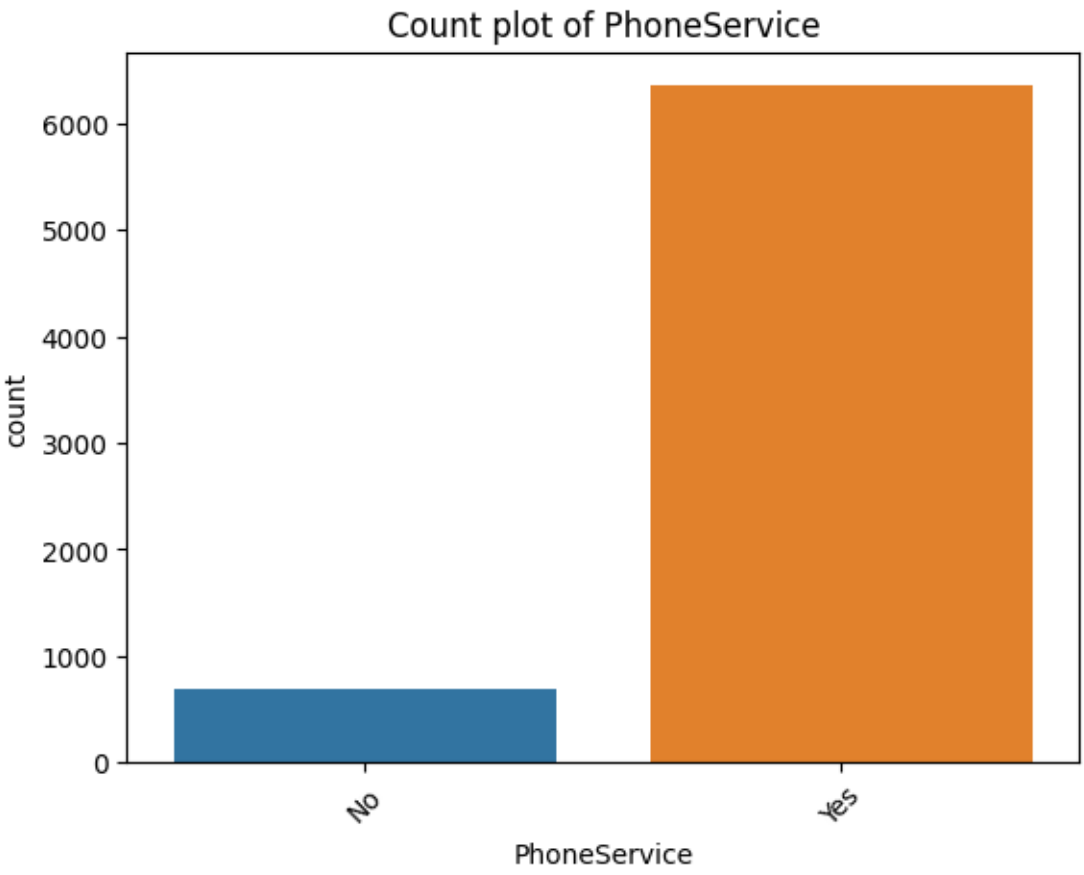
[12]:
```
# Count plot of categorical variables
categorical_cols = df.select_dtypes(include=['object']).columns
for col in categorical_cols:
    plt.figure()
    sns.countplot(data=df, x=col)
    plt.title(f'Count plot of {col}')
    plt.xticks(rotation=45)
    plt.show()
```

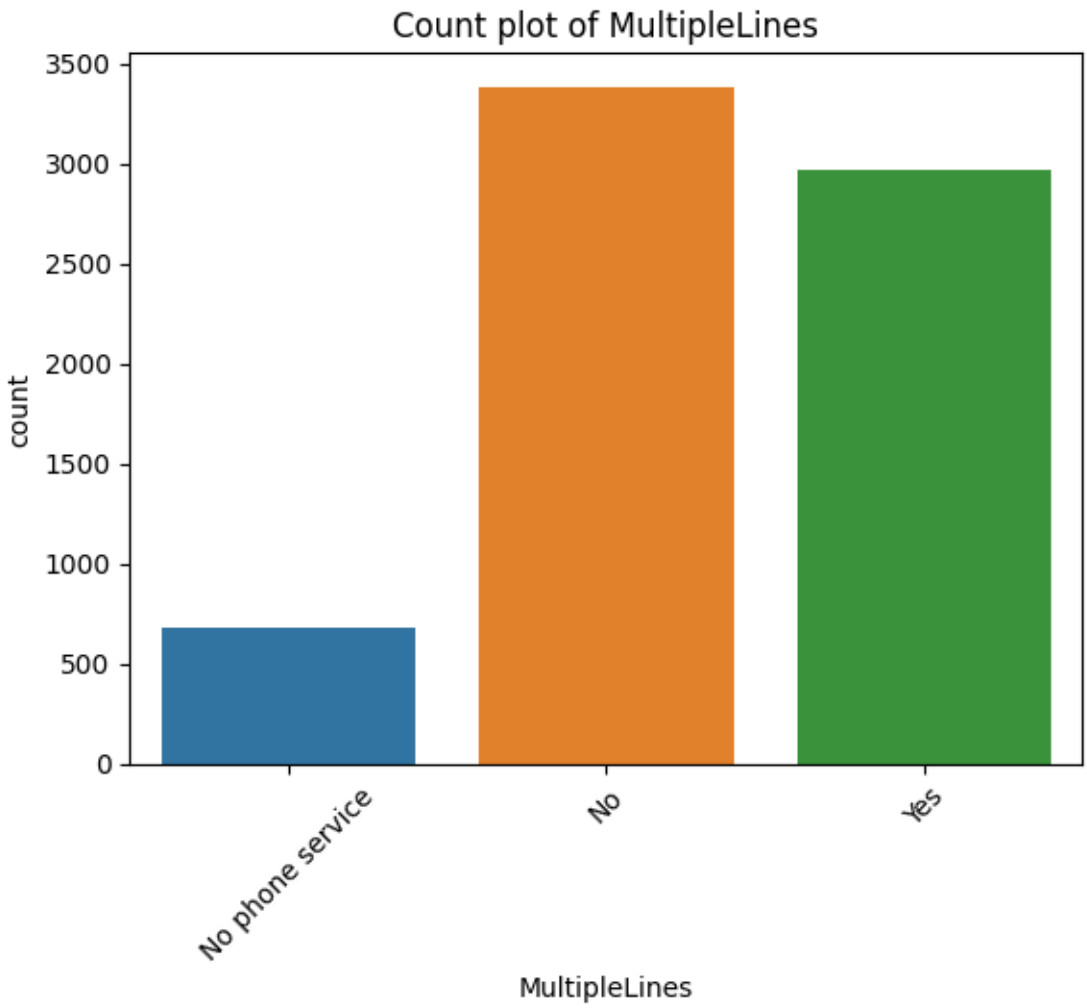

Count plot of customerID
1.0
0.8
0.6
0.4
0.2
0.0
count
customerID

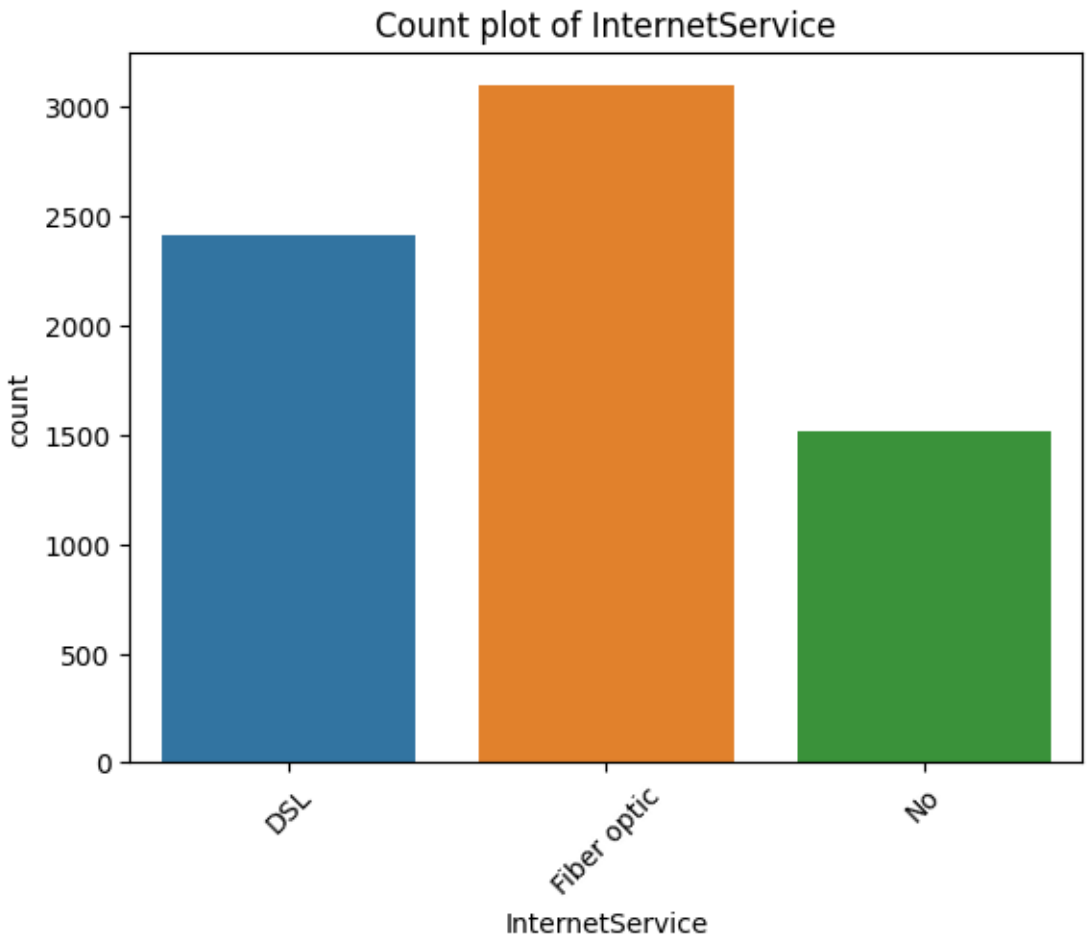

Count plot of gender
count
3500
3000
2500
2000
1500
1000
500
0
Female
Male
gender

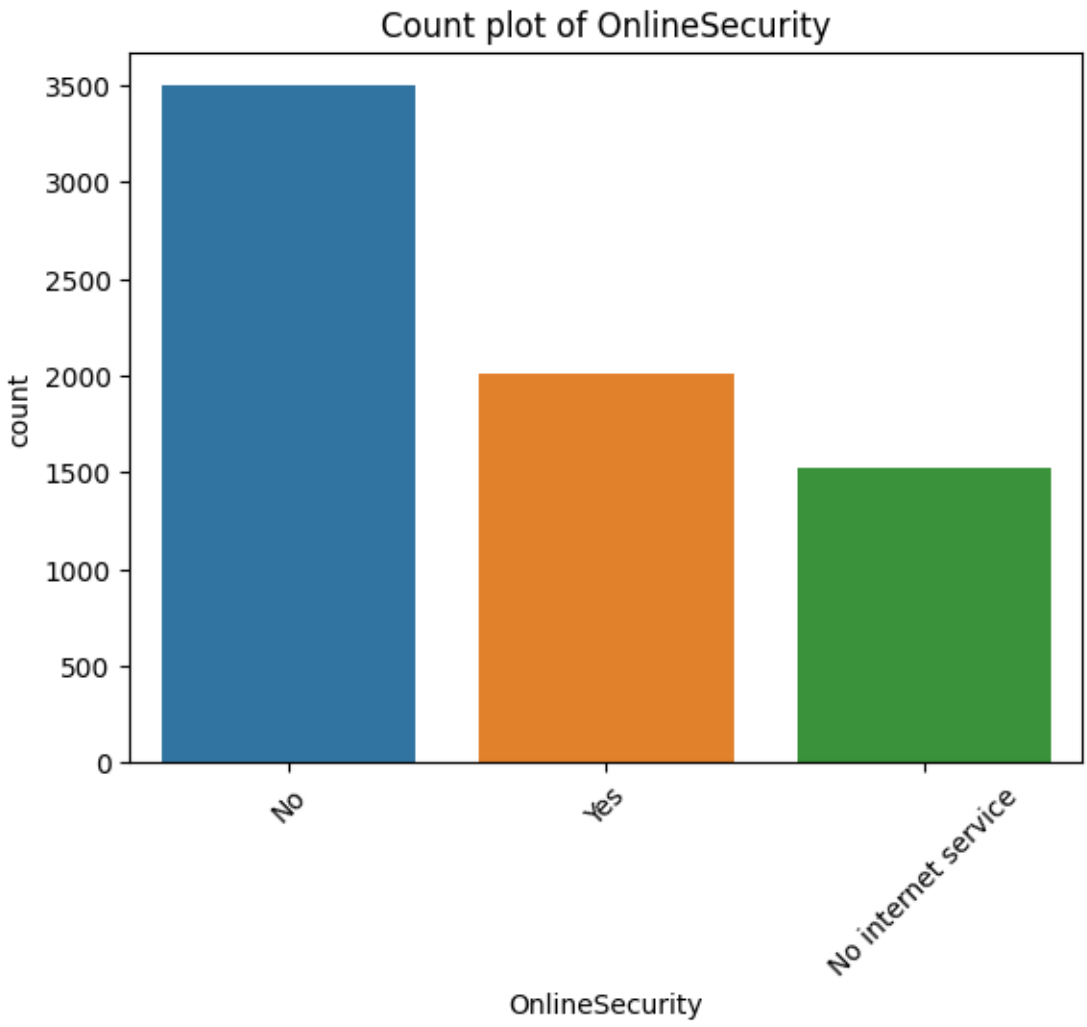

Count plot of Partner
count
3500
3000
2500
2000
1500
1000
500
0
Yes
No
Partner

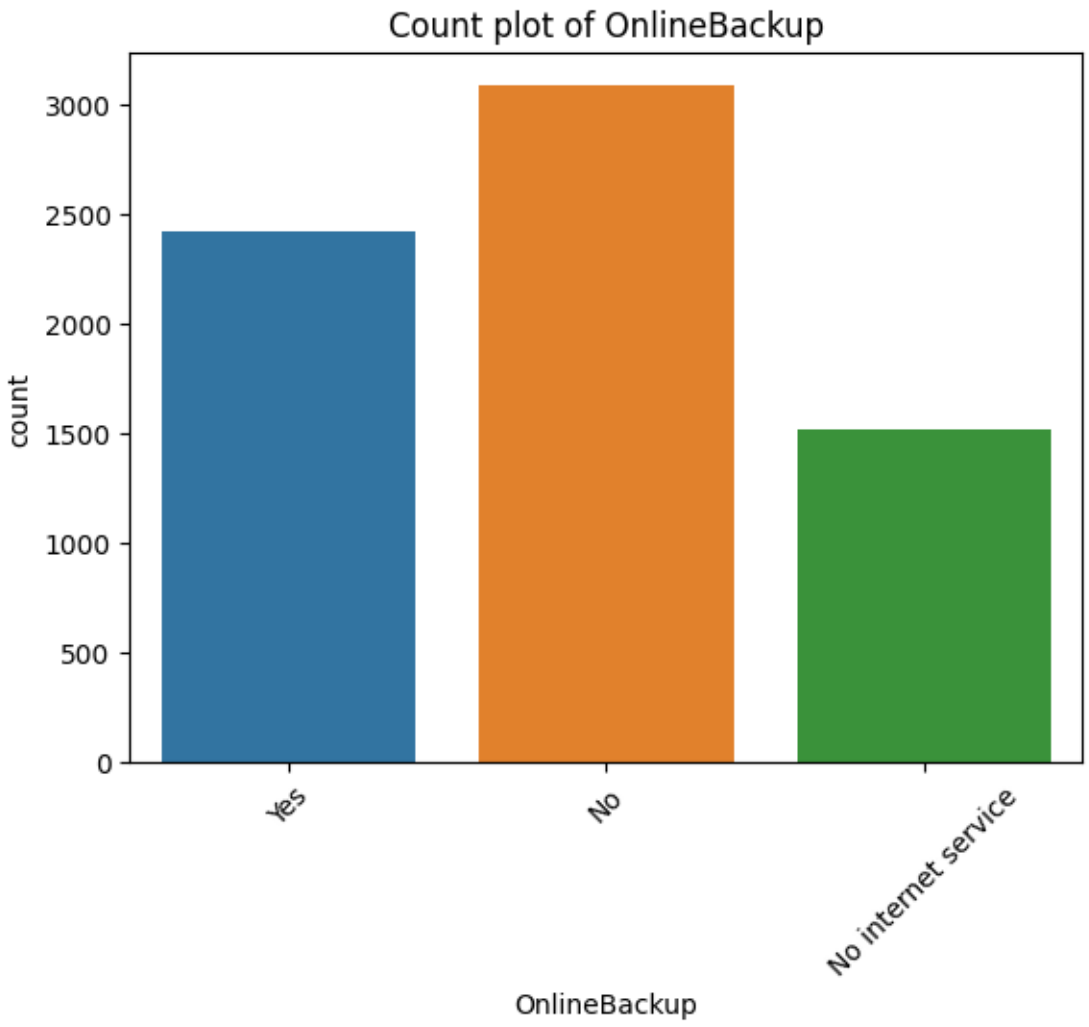

Count plot of Dependents
5000
4000
3000
2000
1000
0
count
No
Yes
Dependents

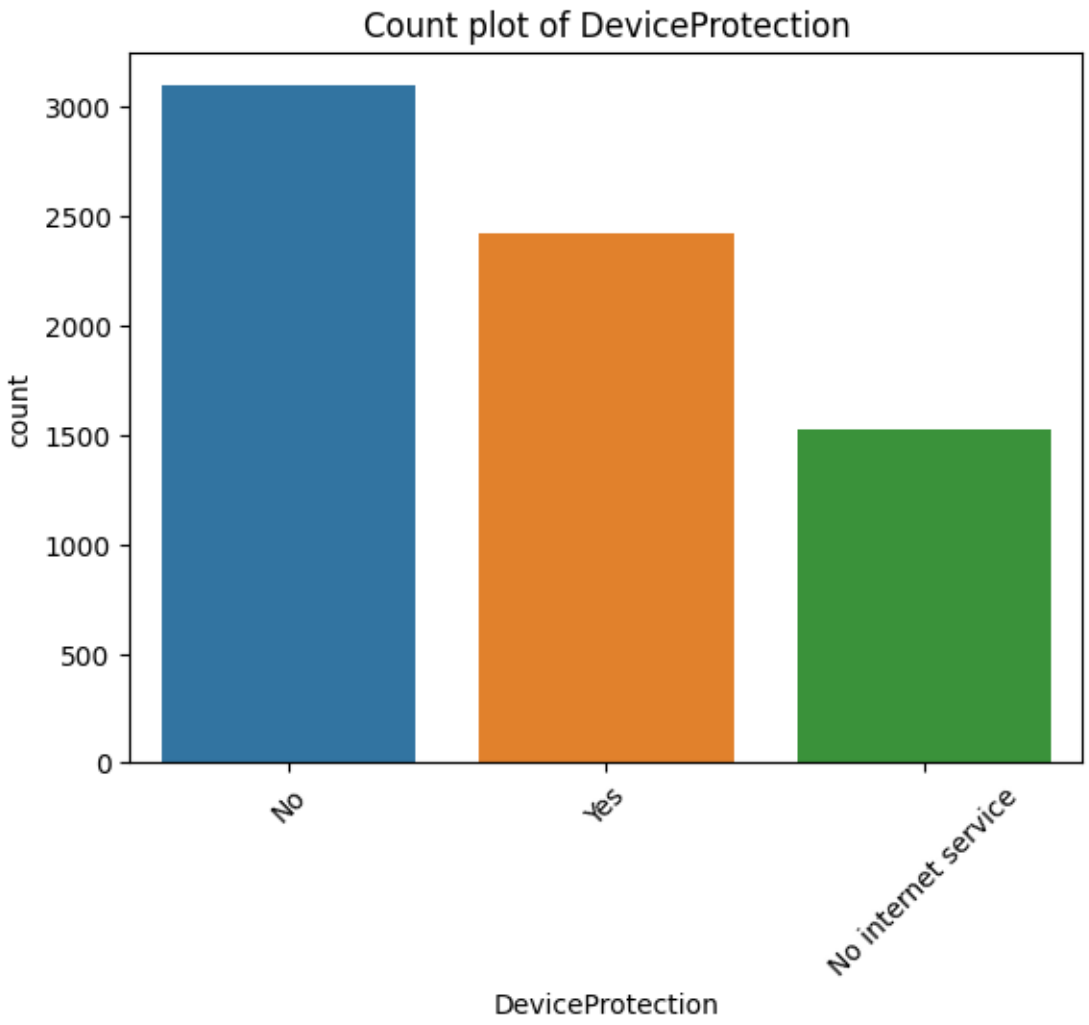

Count plot of PhoneService
6000
5000
4000
3000
2000
1000
0
count
No
Yes
PhoneService

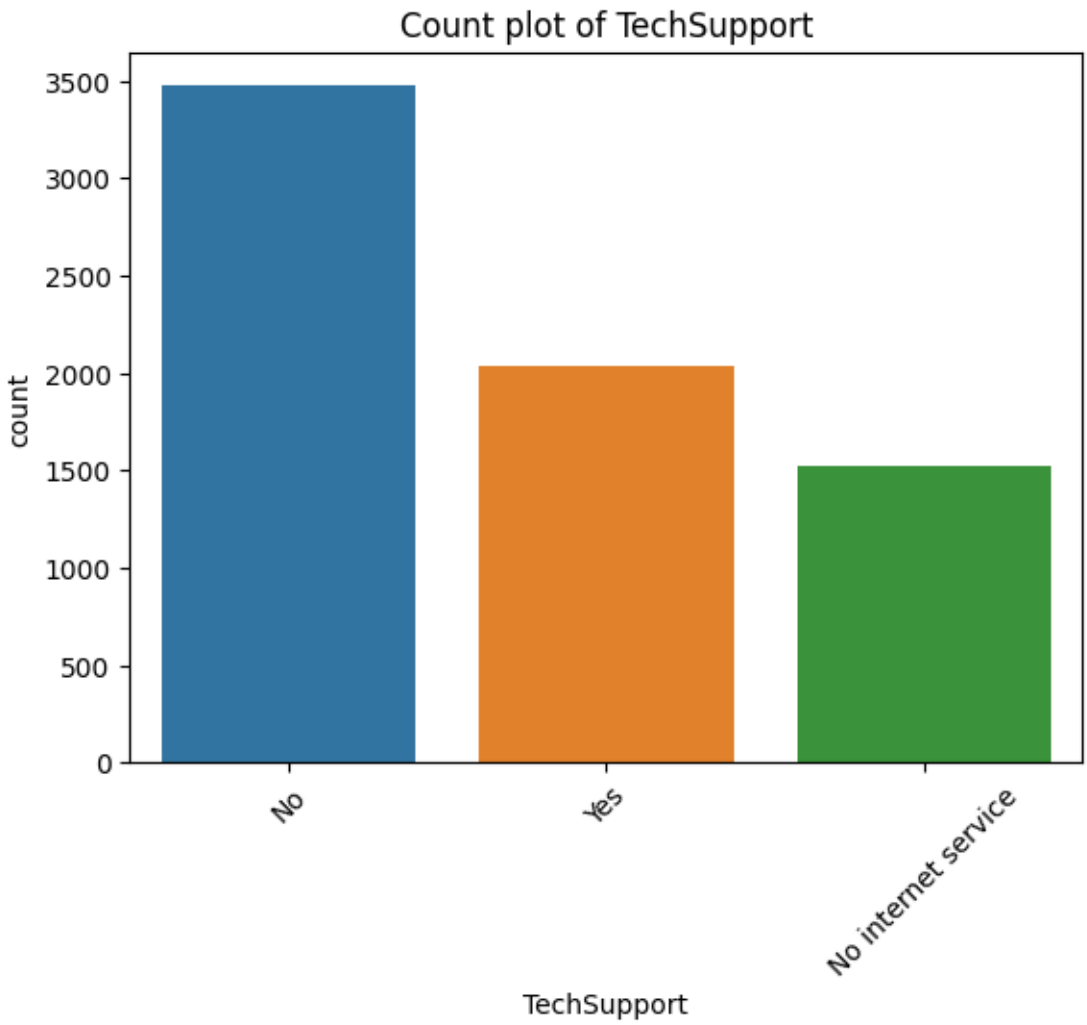

Count plot of MultipleLines
3500
3000
2500
2000
1500
1000
500
0
count
No phone service
No
Yes
MultipleLines

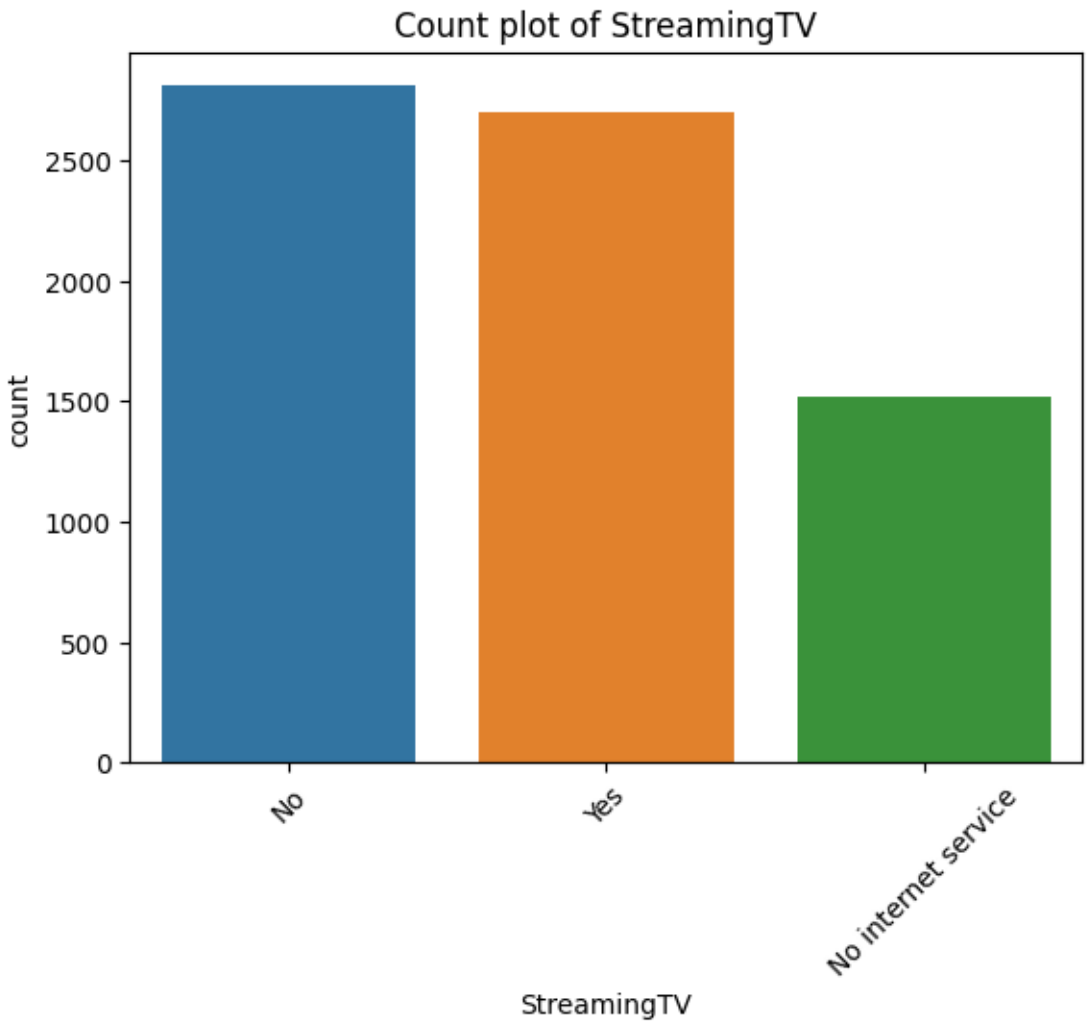

Count plot of InternetService
count
3000
2500
2000
1500
1000
500
0
DSL
Fiber optic
No
InternetService

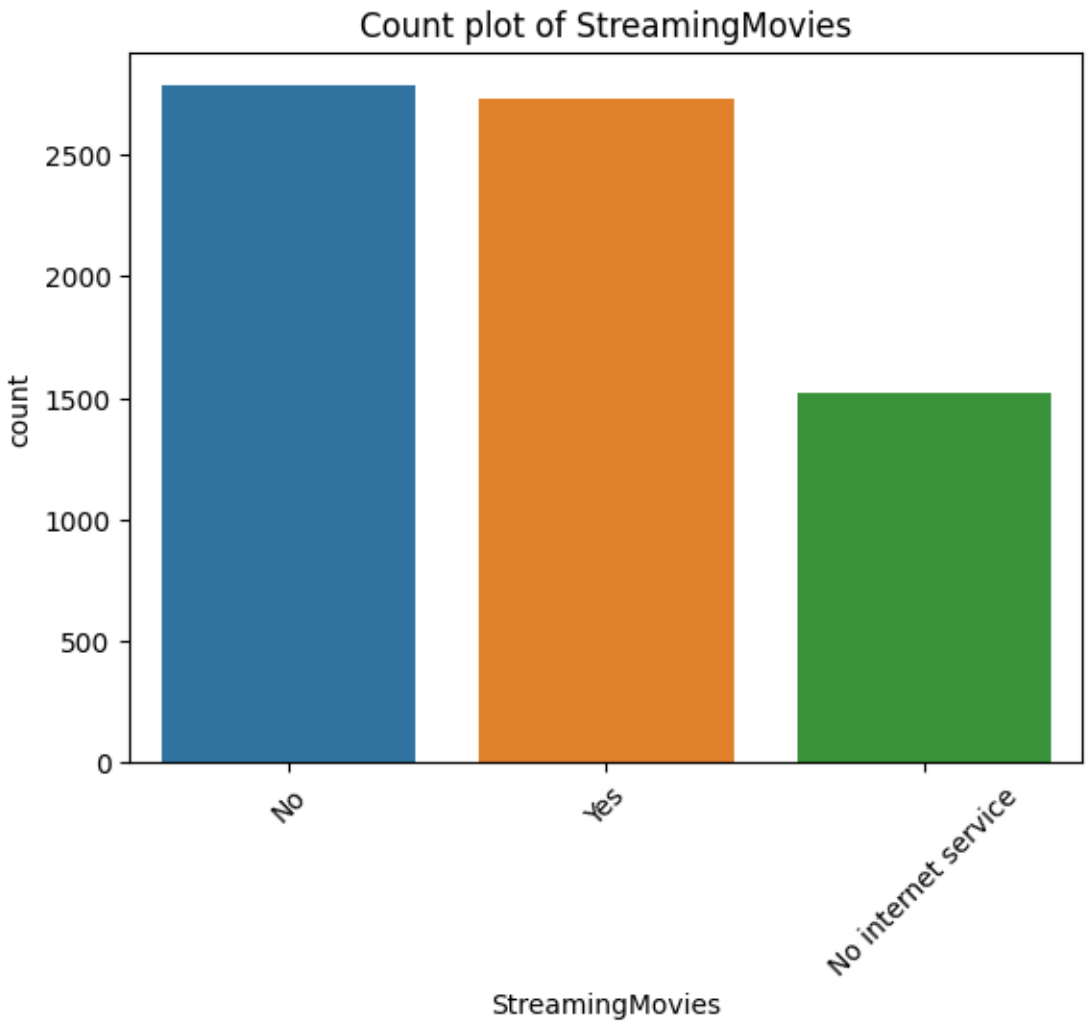

Count plot of OnlineSecurity
count
3500
3000
2500
2000
1500
1000
500
0
No
Yes
No internet service
OnlineSecurity

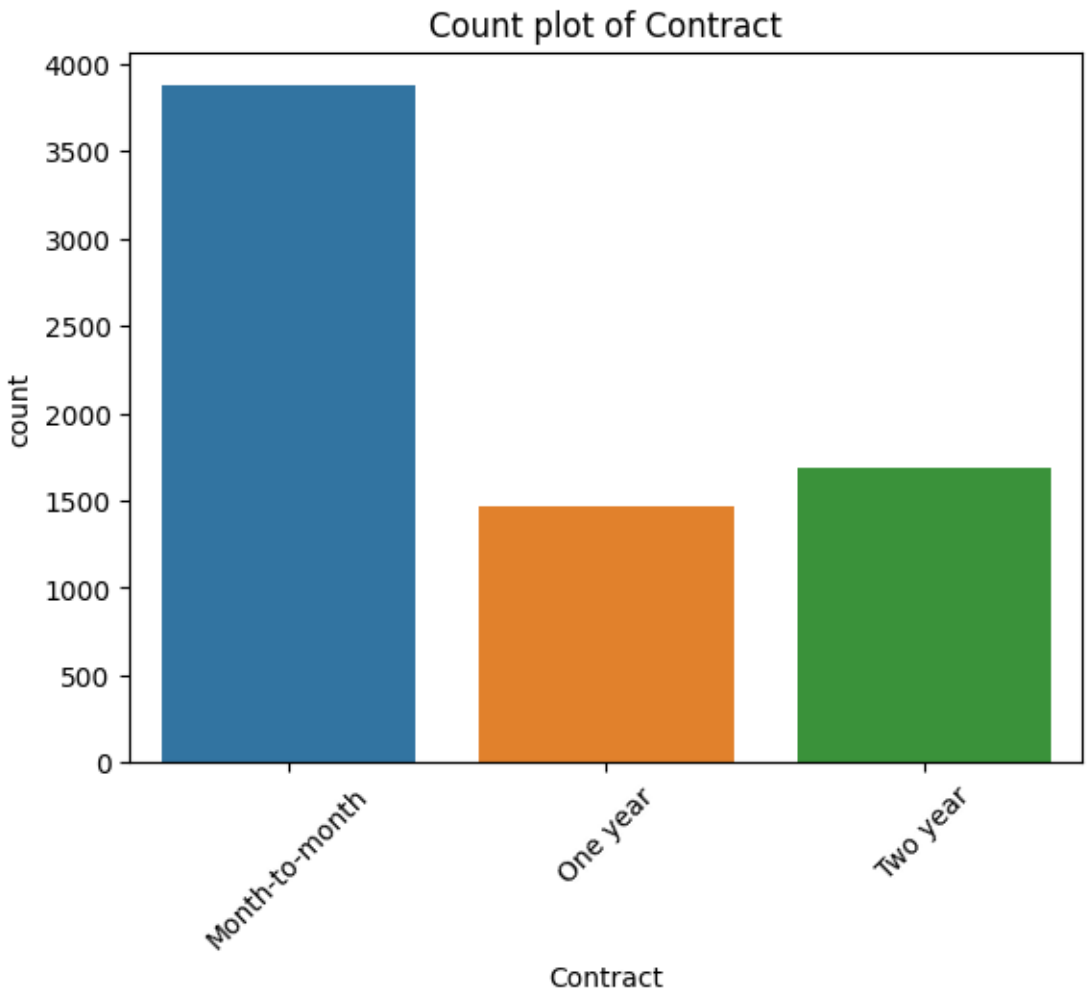

Count plot of OnlineBackup
count
3000
2500
2000
1500
1000
500
0
Yes
No
No internet service
OnlineBackup

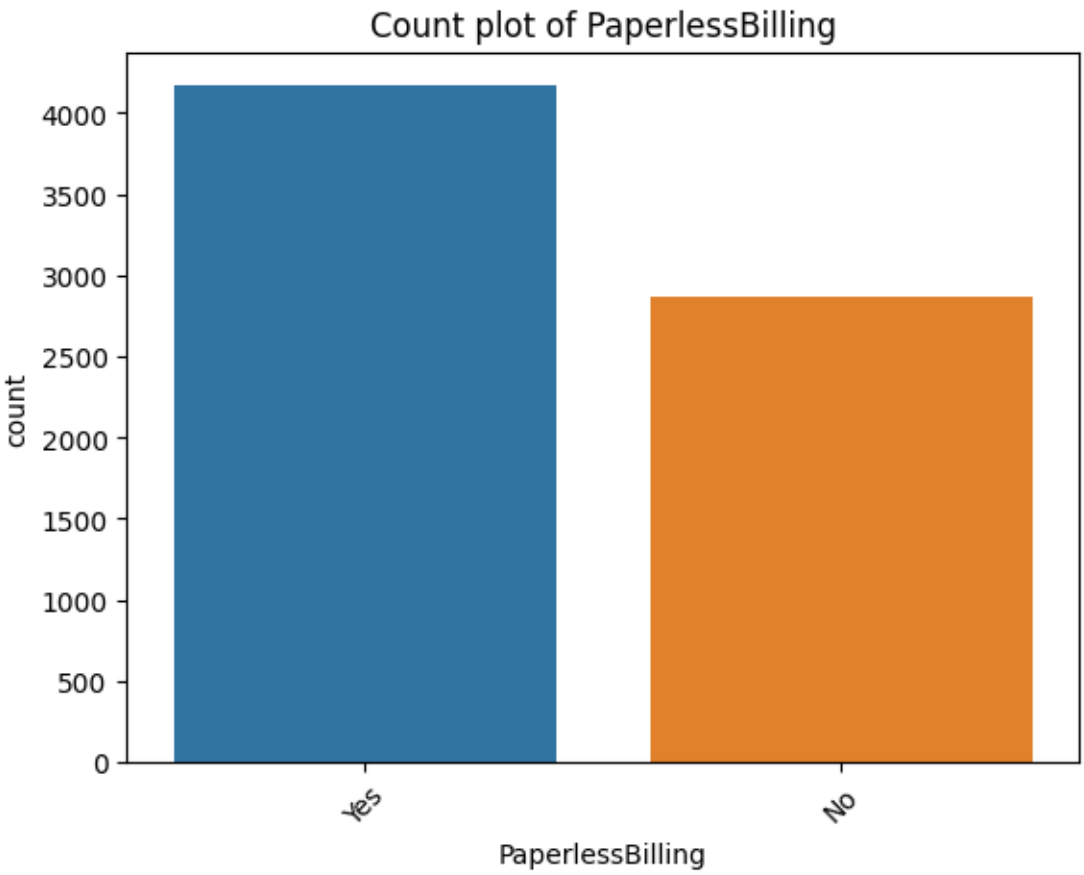

Count plot of DeviceProtection
3000
2500
2000
1500
1000
500
0
count
No
Yes
No internet service
DeviceProtection

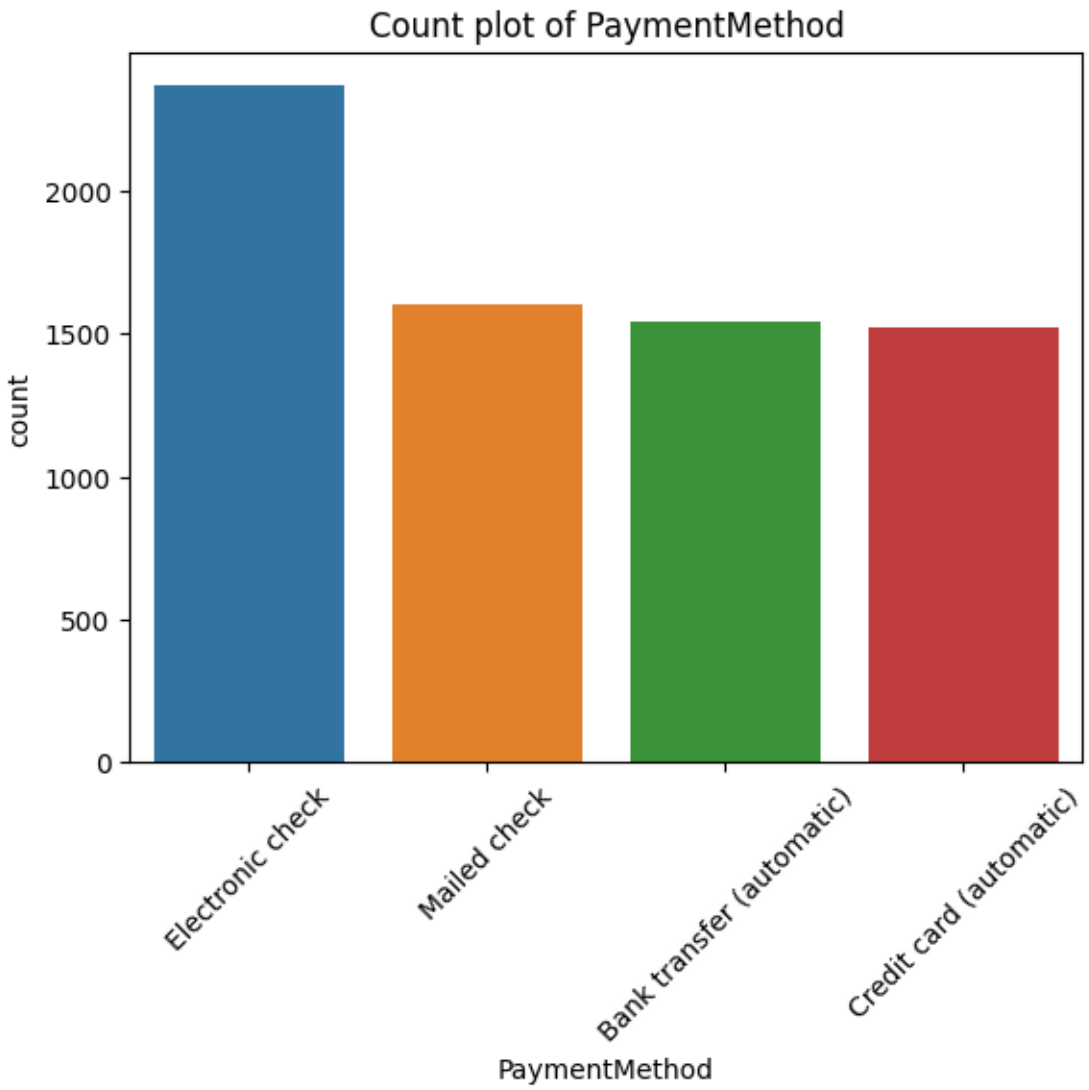

Count plot of TechSupport
count
3500
3000
2500
2000
1500
1000
500
0
No
Yes
No internet service
TechSupport

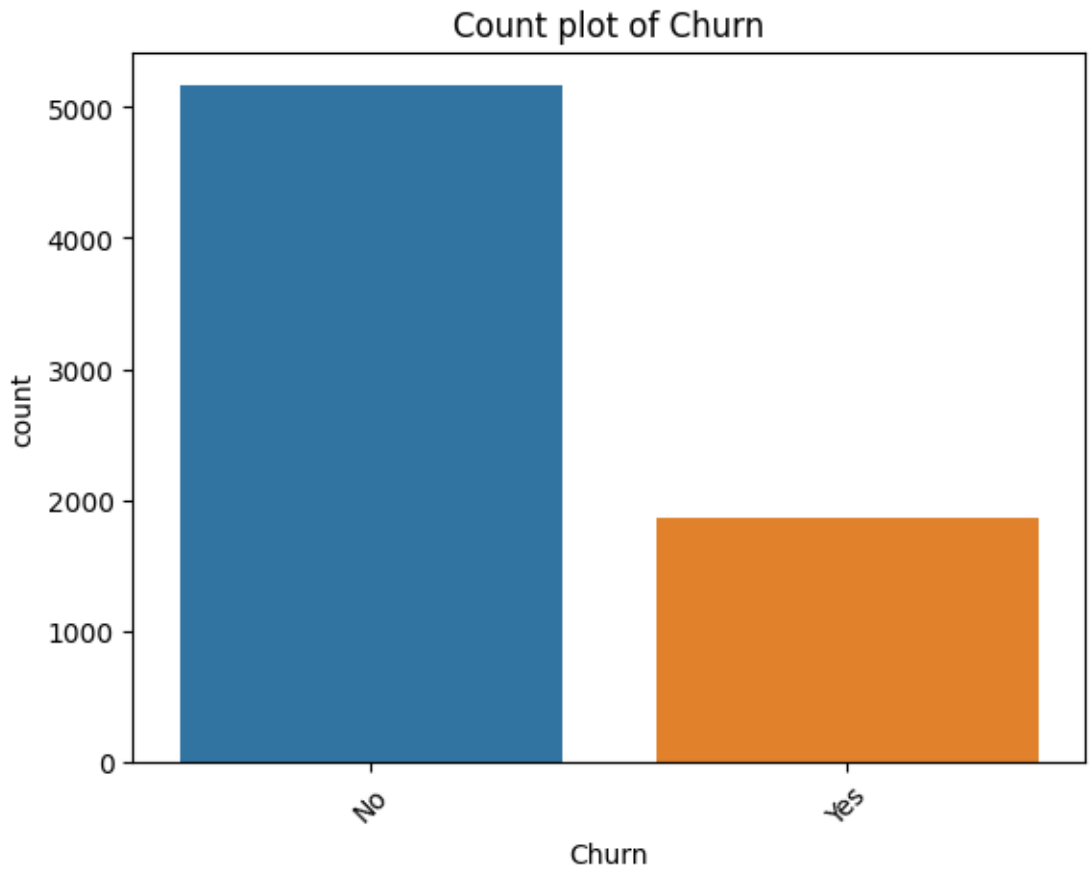

Count plot of StreamingTV
count
2500
2000
1500
1000
500
0
No
Yes
No internet service
StreamingTV


Count plot of StreamingMovies
count
2500
2000
1500
1000
500
0
No
Yes
No internet service
StreamingMovies

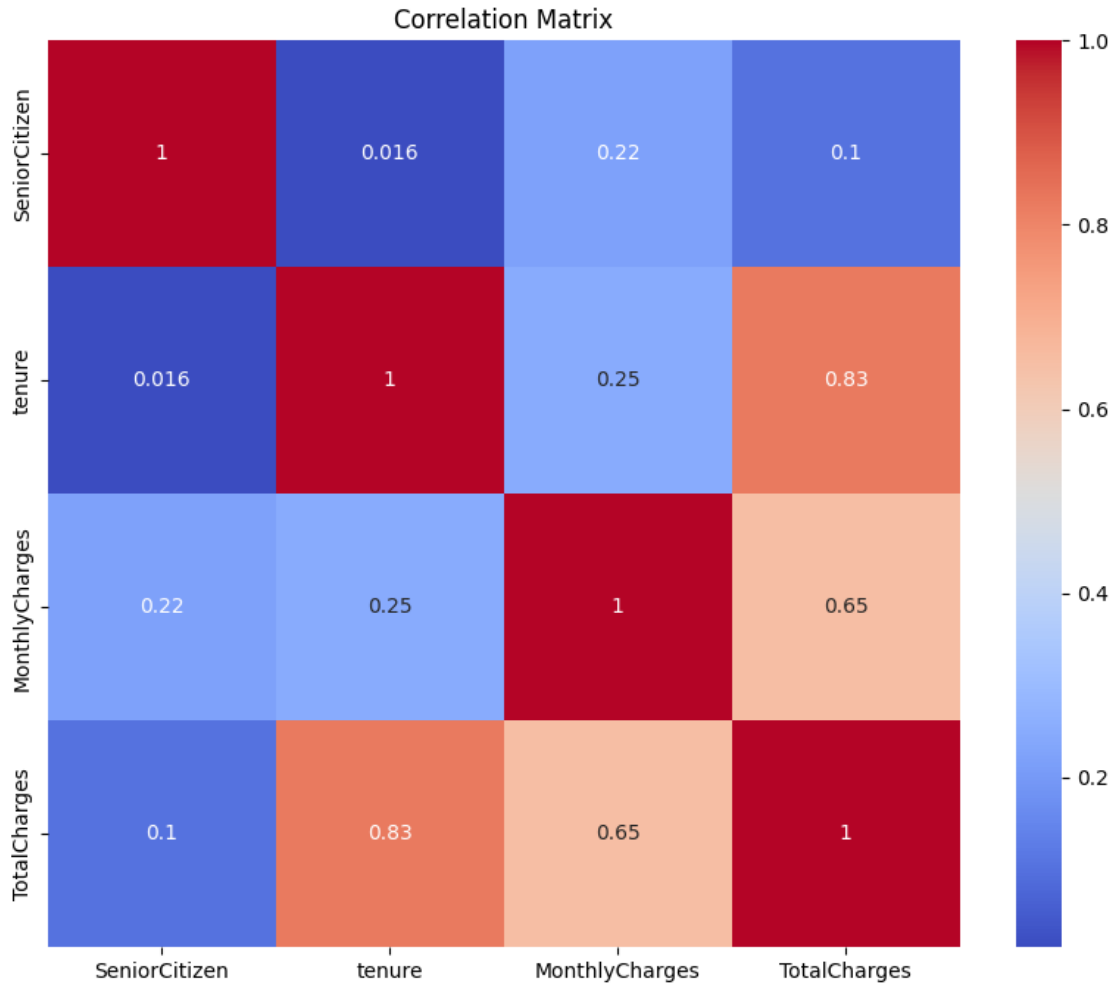

Count plot of Contract
4000
3500
3000
2500
2000
1500
1000
500
0
count
Month-to-month
One year
Two year
Contract


Count plot of PaperlessBilling
count
4000
3500
3000
2500
2000
1500
1000
500
0
Yes
No
PaperlessBilling


Count plot of PaymentMethod
count
2000
1500
1000
500
0
Electronic check
Mailed check
Bank transfer (automatic)
Credit card (automatic)
PaymentMethod



[17]:
```
# Correlation Analysis
columns_of_interest = ['SeniorCitizen', 'tenure', 'MonthlyCharges',␣
 ↪'TotalCharges']
df1 = df[columns_of_interest]
correlation_matrix = df1.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```



[14]:
```
# Perform one-hot encoding for categorical variables
df_without_customerID = df.drop('customerID', axis=1)
df_encoded = pd.get_dummies(df_without_customerID)
```

[15]:
```
df_encoded.head()
```

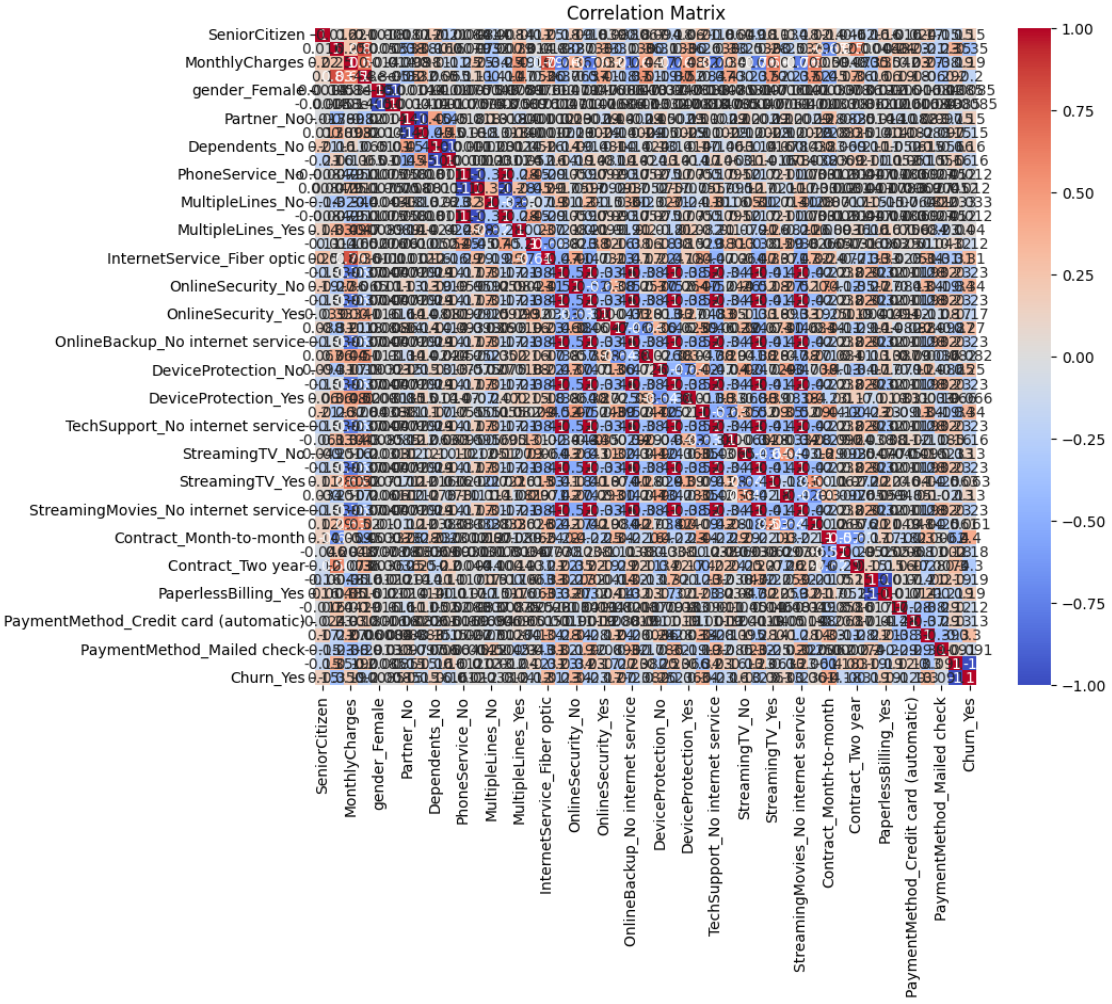
[15]:

	SeniorCitizen	tenure	MonthlyCharges	TotalCharges	gender_Female	\
0	0	1	29.85	29.85	True	\
1	0	34	56.95	1889.50	False	
2	0	2	53.85	108.15	False	
3	0	45	42.30	1840.75	False	
4	0	2	70.70	151.65	True	

	gender_Male	Partner_No	Partner_Yes	Dependents_No	Dependents_Yes	...	
0	False	False	True	True	False	...	\
1	True	True	False	True	False	...	

```
2         True        True        False           True           False  ...
3         True        True        False           True           False  ...
4        False        True        False           True           False  ...

   Contract_One year  Contract_Two year  PaperlessBilling_No
0              False              False                False  \
1               True              False                 True
2              False              False                False
3               True              False                 True
4              False              False                False

   PaperlessBilling_Yes  PaymentMethod_Bank transfer (automatic)
0                  True                                   False  \
1                 False                                   False
2                  True                                   False
3                 False                                    True
4                  True                                   False

   PaymentMethod_Credit card (automatic)  PaymentMethod_Electronic check
0                                  False                           True  \
1                                  False                          False
2                                  False                          False
3                                  False                          False
4                                  False                           True

   PaymentMethod_Mailed check  Churn_No  Churn_Yes
0                       False      True      False
1                        True      True      False
2                        True     False       True
3                       False      True      False
4                       False     False       True

[5 rows x 47 columns]
```

[16]:
```python
# Correlation Analysis
correlation_matrix = df_encoded.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```



0.0.1 Feature Selection Using RFE and RandomForestClassifier

[18]:
```
from sklearn.feature_selection import RFE
from sklearn.ensemble import RandomForestClassifier
```

[22]:
```
X = df_encoded.drop(['Churn_No', 'Churn_Yes'], axis=1)
y = df['Churn']
```

[23]:
```
model = RandomForestClassifier()
rfe = RFE(model, n_features_to_select=4)
rfe = rfe.fit(X, y)
```

[24]:
```
print("Selected Features:")
for i in range(len(rfe.support_)):
    if rfe.support_[i]:
        print(X.columns[i])
```

```
Selected Features:
tenure
MonthlyCharges
TotalCharges
Contract_Month-to-month
```

[26]:
```
print("\nFeature Ranking:")
for i in range(len(rfe.ranking_)):
    print("Feature %d: %s, Rank: %d" % (i+1, X.columns[i], rfe.ranking_[i]))
```

```
Feature Ranking:
Feature 1: SeniorCitizen, Rank: 12
Feature 2: tenure, Rank: 1
Feature 3: MonthlyCharges, Rank: 1
Feature 4: TotalCharges, Rank: 1
Feature 5: gender_Female, Rank: 17
Feature 6: gender_Male, Rank: 6
Feature 7: Partner_No, Rank: 9
Feature 8: Partner_Yes, Rank: 21
Feature 9: Dependents_No, Rank: 13
Feature 10: Dependents_Yes, Rank: 24
Feature 11: PhoneService_No, Rank: 36
Feature 12: PhoneService_Yes, Rank: 33
Feature 13: MultipleLines_No, Rank: 11
Feature 14: MultipleLines_No phone service, Rank: 37
Feature 15: MultipleLines_Yes, Rank: 20
Feature 16: InternetService_DSL, Rank: 32
Feature 17: InternetService_Fiber optic, Rank: 4
Feature 18: InternetService_No, Rank: 40
Feature 19: OnlineSecurity_No, Rank: 2
Feature 20: OnlineSecurity_No internet service, Rank: 42
Feature 21: OnlineSecurity_Yes, Rank: 31
Feature 22: OnlineBackup_No, Rank: 7
Feature 23: OnlineBackup_No internet service, Rank: 38
Feature 24: OnlineBackup_Yes, Rank: 23
Feature 25: DeviceProtection_No, Rank: 10
Feature 26: DeviceProtection_No internet service, Rank: 35
Feature 27: DeviceProtection_Yes, Rank: 26
Feature 28: TechSupport_No, Rank: 3
Feature 29: TechSupport_No internet service, Rank: 34
Feature 30: TechSupport_Yes, Rank: 30
Feature 31: StreamingTV_No, Rank: 29
Feature 32: StreamingTV_No internet service, Rank: 41
Feature 33: StreamingTV_Yes, Rank: 15
Feature 34: StreamingMovies_No, Rank: 25
Feature 35: StreamingMovies_No internet service, Rank: 39
Feature 36: StreamingMovies_Yes, Rank: 14
```

```
Feature 37: Contract_Month-to-month, Rank: 1
Feature 38: Contract_One year, Rank: 27
Feature 39: Contract_Two year, Rank: 16
Feature 40: PaperlessBilling_No, Rank: 8
Feature 41: PaperlessBilling_Yes, Rank: 19
Feature 42: PaymentMethod_Bank transfer (automatic), Rank: 18
Feature 43: PaymentMethod_Credit card (automatic), Rank: 22
Feature 44: PaymentMethod_Electronic check, Rank: 5
Feature 45: PaymentMethod_Mailed check, Rank: 28
```

0.0.2 Feature Selection using mRMR Algorithm

[27]:
```
from mrmr import mrmr_classif
```

[28]:
```
selected_features = mrmr_classif(X, y, 5) # Selecting 5 features
print(selected_features)
```

```
100%|
| 5/5 [00:00<00:00, 25.73it/s]
```

```
['Contract_Month-to-month', 'PaymentMethod_Mailed check', 'OnlineSecurity_No',
'tenure', 'InternetService_DSL']
```

[29]:
```
print(X[selected_features])
```

```
      Contract_Month-to-month  PaymentMethod_Mailed check  OnlineSecurity_No
0                        True                       False               True  \
1                       False                        True              False
2                        True                        True              False
3                       False                       False              False
4                        True                       False               True
...                       ...                         ...                ...
7038                    False                        True              False
7039                    False                       False               True
7040                     True                       False              False
7041                     True                        True               True
7042                    False                       False              False

      tenure  InternetService_DSL
0          1                 True
1         34                 True
2          2                 True
3         45                 True
4          2                False
...      ...                  ...
7038      24                 True
7039      72                False
7040      11                 True
```

```
7041       4                False
7042      66                False

[7032 rows x 5 columns]
```

[ ]:

In conclusion, we can see that after performing EDA and Feature selection using RFE with Random Forest Classifier and mRMR feature selection algorithm that few features are highly correlated to the target output that in this dataset is Churn. Through RFE with Random Forest Classifier we can see in terms of Ranking that tenure, Monthly Charges, TotalCharges and few others are highly correlated with the output data. mRMR(minimum Redundancy Maximum Relevance) Algorithm is another advanced algorithm which is used to select features. We put the limit of features to be 5 and we can see that tenure, few others are given by it which was also given by the RFE making our selected features sure of giving good accuracy if we were to just put few of those top selected features as input.