Customer Churn Analysis

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

In [2]:
```
df = pd.read_csv(r"C:\Users\USER\Downloads\Customer-Churn.csv")
df
```

Out[2]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service
1	5575-GNVDE	Male	0	No	No	34	Yes	No
2	3668-QPYBK	Male	0	No	No	2	Yes	No
3	7795-CFOCW	Male	0	No	No	45	No	No phone service
4	9237-HQITU	Female	0	No	No	2	Yes	No
...	...	...	...	...	...	...	...	...
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes
7039	2234-XADUH	Female	0	Yes	Yes	72	Yes	Yes
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phone service
7041	8361-LTMKD	Male	1	Yes	No	4	Yes	Yes
7042	3186-AJIEK	Male	0	No	No	66	Yes	No

7043 rows × 21 columns

In [3]:
```
df.drop_duplicates()
```

Out[3]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service
1	5575-GNVDE	Male	0	No	No	34	Yes	No
2	3668-QPYBK	Male	0	No	No	2	Yes	No
3	7795-CFOCW	Male	0	No	No	45	No	No phone service
4	9237-HQITU	Female	0	No	No	2	Yes	No
...	...	...	...	...	...	...	...	...
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes
7039	2234-XADUH	Female	0	Yes	Yes	72	Yes	Yes
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phone service
7041	8361-LTMKD	Male	1	Yes	No	4	Yes	Yes
7042	3186-AJIEK	Male	0	No	No	66	Yes	No

7043 rows × 21 columns

In [4]:
```
"""
There are no duplicates from the data
"""
```

Out[4]:
```
'\nThere are no duplicates from the data\n'
```

In [5]:
```
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
```

In [6]:
```
pd.set_option('display.max.column', 22)
df
```

Out[6]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service
1	5575-GNVDE	Male	0	No	No	34	Yes	No
2	3668-QPYBK	Male	0	No	No	2	Yes	No
3	7795-CFOCW	Male	0	No	No	45	No	No phone service
4	9237-HQITU	Female	0	No	No	2	Yes	No
...	...	...	...	...	...	...	...	...
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes
7039	2234-XADUH	Female	0	Yes	Yes	72	Yes	Yes
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phone service
7041	8361-LTMKD	Male	1	Yes	No	4	Yes	Yes
7042	3186-AJIEK	Male	0	No	No	66	Yes	No

7043 rows × 21 columns

In [7]:
```
"""
From the data set, the values of TotalCharges are numeric but its data type from the d
Hence the need to change its data type to float
"""
```

Out[7]:
```
'\nFrom the data set, the values of TotalCharges are numeric but its data type from t
he data info is object\nHence the need to change its data type to float\n'
```

In [8]:
```
df['TotalCharges'] = pd.to_numeric(df['TotalCharges'], errors = 'coerce')
df['TotalCharges']
```

Out[8]:
```
0         29.85
1       1889.50
2        108.15
3       1840.75
4        151.65
         ...
7038    1990.50
7039    7362.90
7040     346.45
7041     306.60
7042    6844.50
Name: TotalCharges, Length: 7043, dtype: float64
```

In [9]:
```
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      7032 non-null   float64
 20  Churn             7043 non-null   object 
dtypes: float64(2), int64(2), object(17)
memory usage: 1.1+ MB
```

In [10]:
```
df.isnull().sum()
```

Out[10]:
```
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        11
Churn                0
dtype: int64
```

In [11]:
```
"""
Changing the data type of TotalCharges has indicated there were 11 null values in that
"""
```

Out[11]:
```
'\nChanging the data type of TotalCharges has indicated there were 11 null values in
that column\n'
```

In [70]:
```
df.describe()
```

Out[70]:

	SeniorCitizen	tenure	MonthlyCharges	TotalCharges
count	7043.000000	7043.000000	7043.000000	7032.000000
mean	0.162147	32.371149	64.761692	2283.300441
std	0.368612	24.559481	30.090047	2266.771362
min	0.000000	0.000000	18.250000	18.800000
25%	0.000000	9.000000	35.500000	401.450000
50%	0.000000	29.000000	70.350000	1397.475000
75%	0.000000	55.000000	89.850000	3794.737500
max	1.000000	72.000000	118.750000	8684.800000

In [12]:
```
"""
Analysis on impact on some attritubes on Churn
"""
```

Out[12]:
```
'\nAnalysis on impact on some attritubes on Churn\n'
```

In [13]:
```
df.columns
```

Out[13]:
```
Index(['customerID', 'gender', 'SeniorCitizen', 'Partner', 'Dependents',
       'tenure', 'PhoneService', 'MultipleLines', 'InternetService',
       'OnlineSecurity', 'OnlineBackup', 'DeviceProtection', 'TechSupport',
       'StreamingTV', 'StreamingMovies', 'Contract', 'PaperlessBilling',
       'PaymentMethod', 'MonthlyCharges', 'TotalCharges', 'Churn'],
      dtype='object')
```

In [14]:
```
"""
Impact of gender on Churn
"""
```

Out[14]:
```
'\nImpact of gender on Churn\n'
```

In [16]:
```
df.groupby('gender')['customerID'].count()
```

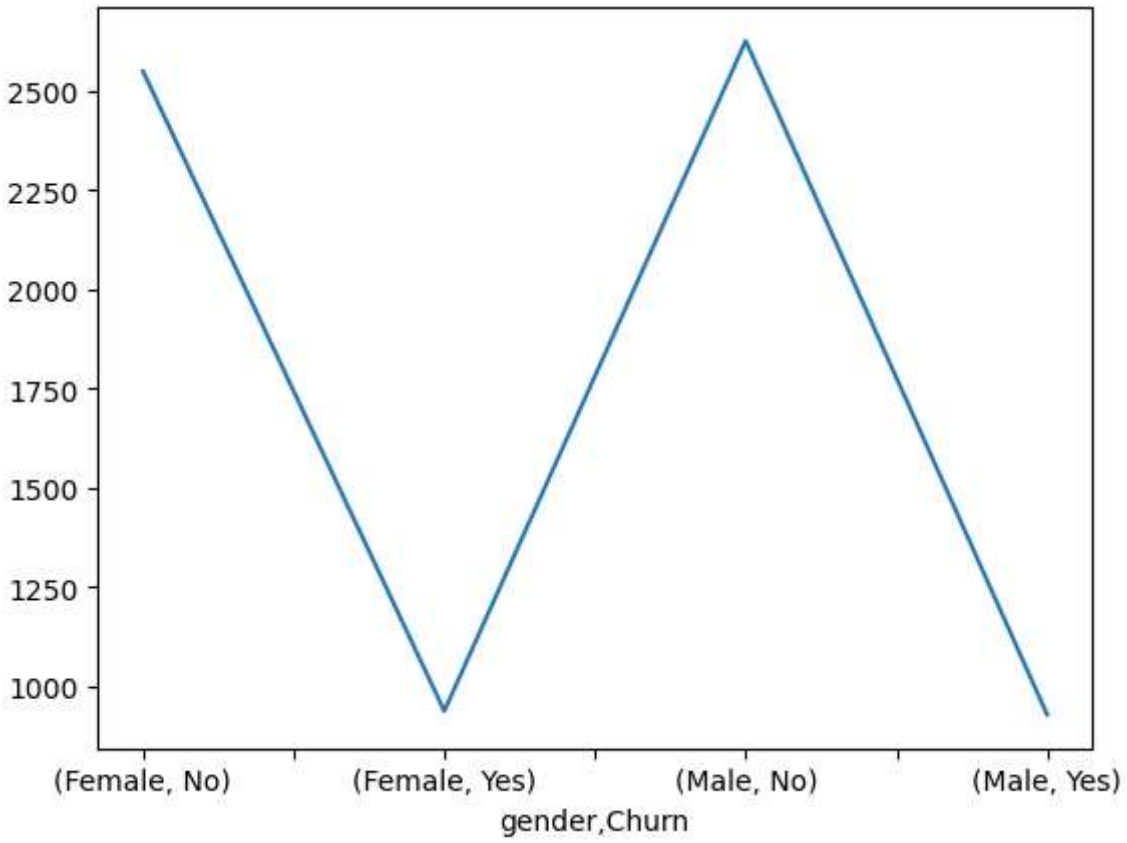
Out[16]:
```
gender
Female    3488
Male      3555
Name: customerID, dtype: int64
```

In [17]:
```
df.groupby(['gender', 'Churn'])['customerID'].count()
```

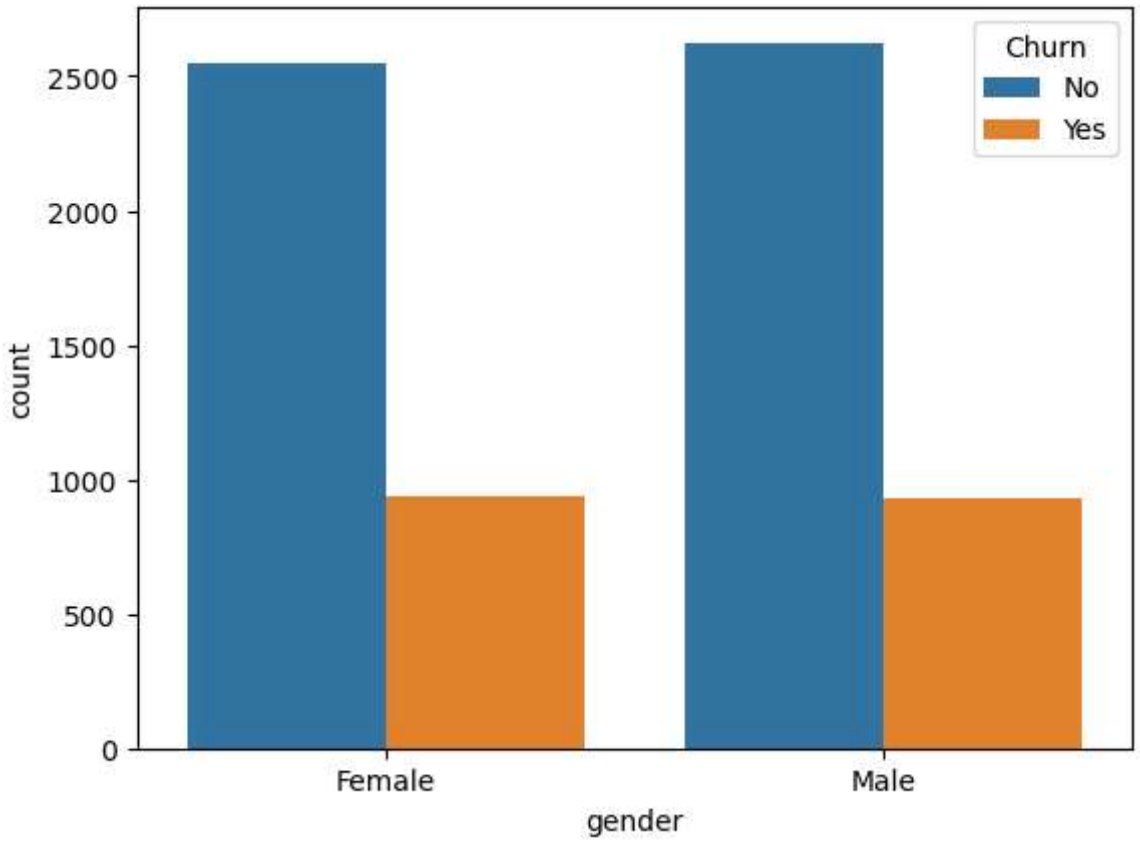
Out[17]:
```
gender  Churn
Female  No       2549
        Yes       939
Male    No       2625
        Yes       930
Name: customerID, dtype: int64
```

In [19]:
```
df.groupby(['gender', 'Churn'])['customerID'].count().plot()
```

Out[19]:
```
<Axes: xlabel='gender,Churn'>
```



In [18]:
```
sns.countplot(x = 'gender', data = df, hue = 'Churn')
plt.show()
```



In [ ]:
```
"""
Impact of Senior Citizen on Churn
"""
```

In [20]:
```
df.groupby('SeniorCitizen')['customerID'].count()
```

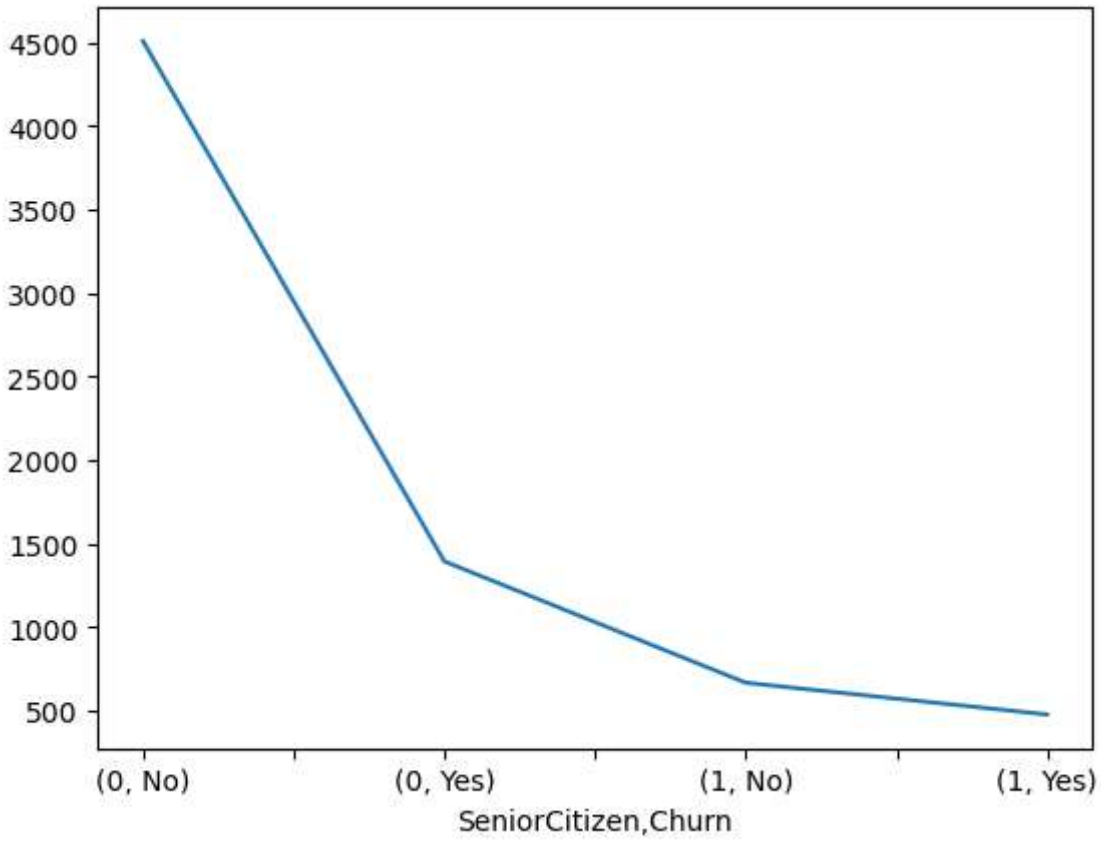
Out[20]:
```
SeniorCitizen
0    5901
1    1142
Name: customerID, dtype: int64
```

In [21]:
```
df.groupby(['SeniorCitizen', 'Churn'])['customerID'].count()
```

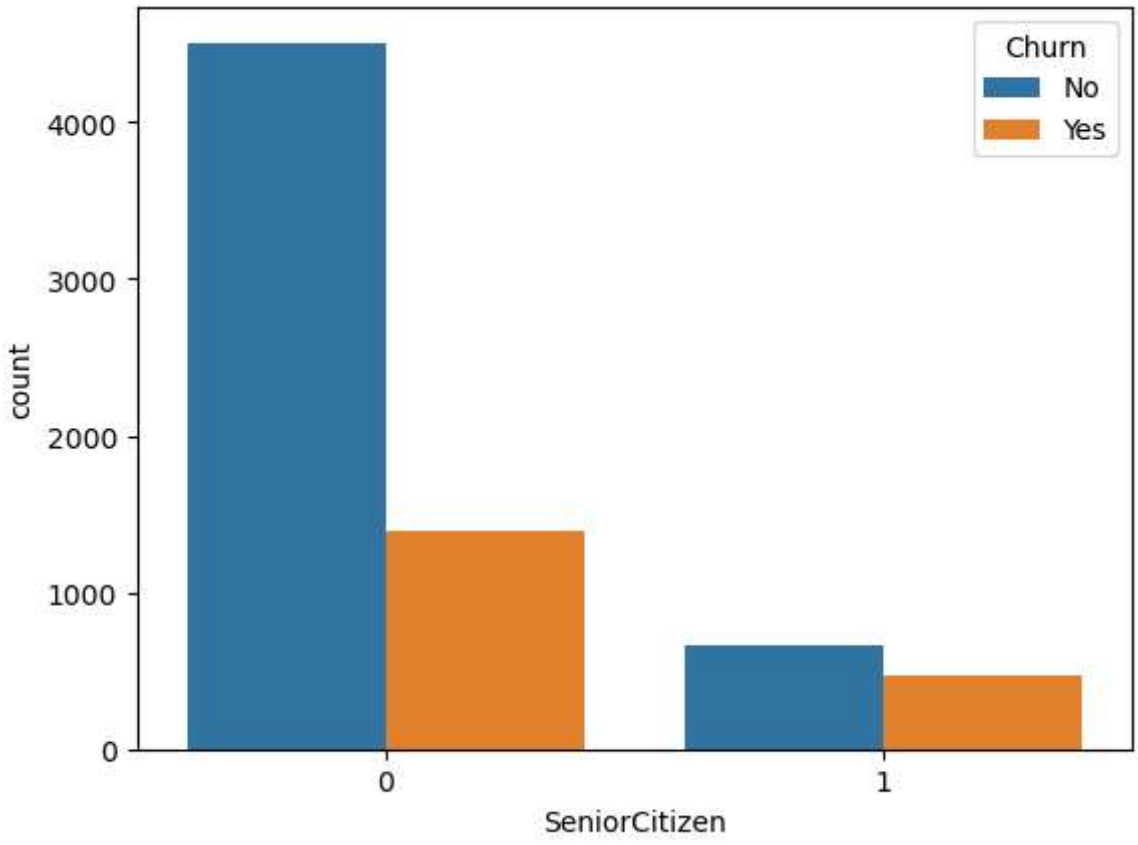
Out[21]:
```
SeniorCitizen  Churn
0              No       4508
               Yes      1393
1              No        666
               Yes       476
Name: customerID, dtype: int64
```

In [22]:
```
df.groupby(['SeniorCitizen', 'Churn'])['customerID'].count().plot()
```

Out[22]:
```
<Axes: xlabel='SeniorCitizen,Churn'>
```



In [23]:
```
sns.countplot(x = 'SeniorCitizen', data = df, hue = 'Churn')
plt.show()
```



In [24]:
```
"""
From the analysis made, those who are senior citizen had higher rate of churned servic
```

```
"""
```

Out[24]:
```
'\nFrom the analysis made, those who are senior citizen had higher rate of churned se
rvices relative to those who are not\n'
```

In [25]:
```
"""
Impact of InternetService on Churn
"""
```

Out[25]:
```
'\nImpact of InternetService on Churn\n'
```

In [26]:
```
df.groupby('InternetService')['customerID'].count()
```

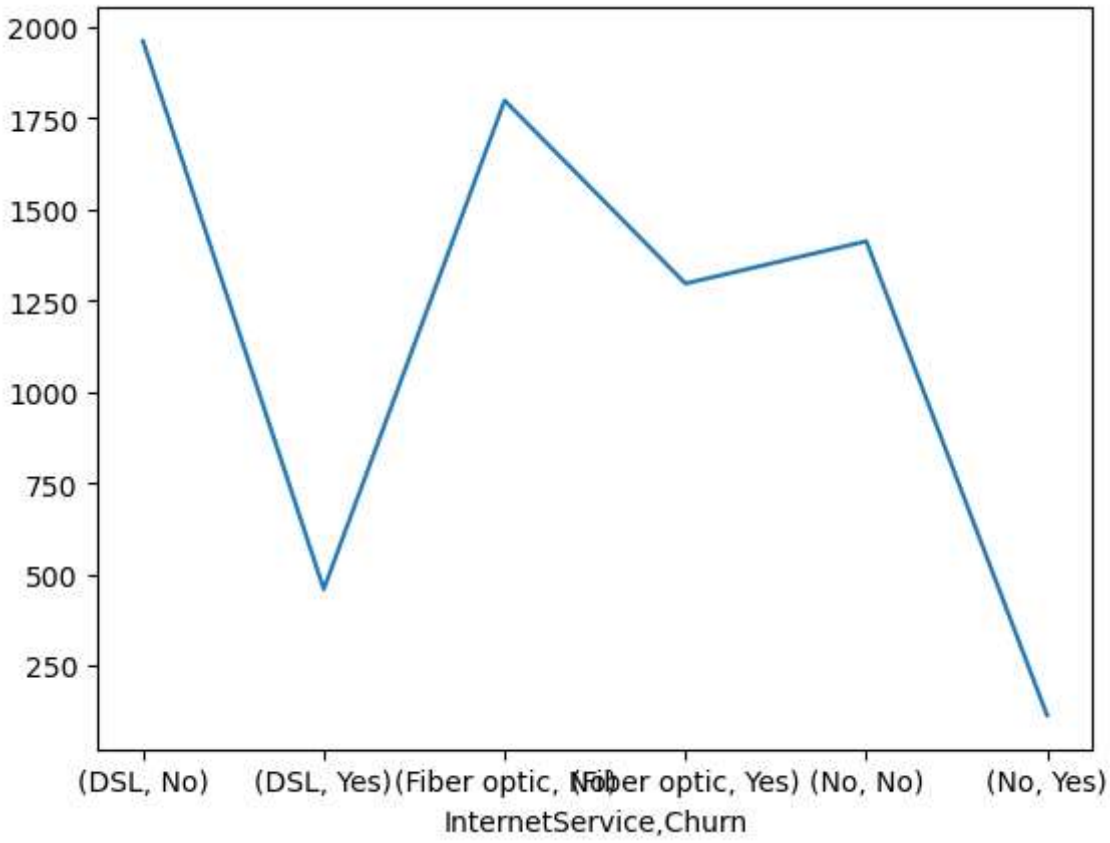
Out[26]:
```
InternetService
DSL            2421
Fiber optic    3096
No             1526
Name: customerID, dtype: int64
```

In [27]:
```
df.groupby(['InternetService', 'Churn'])['customerID'].count()
```

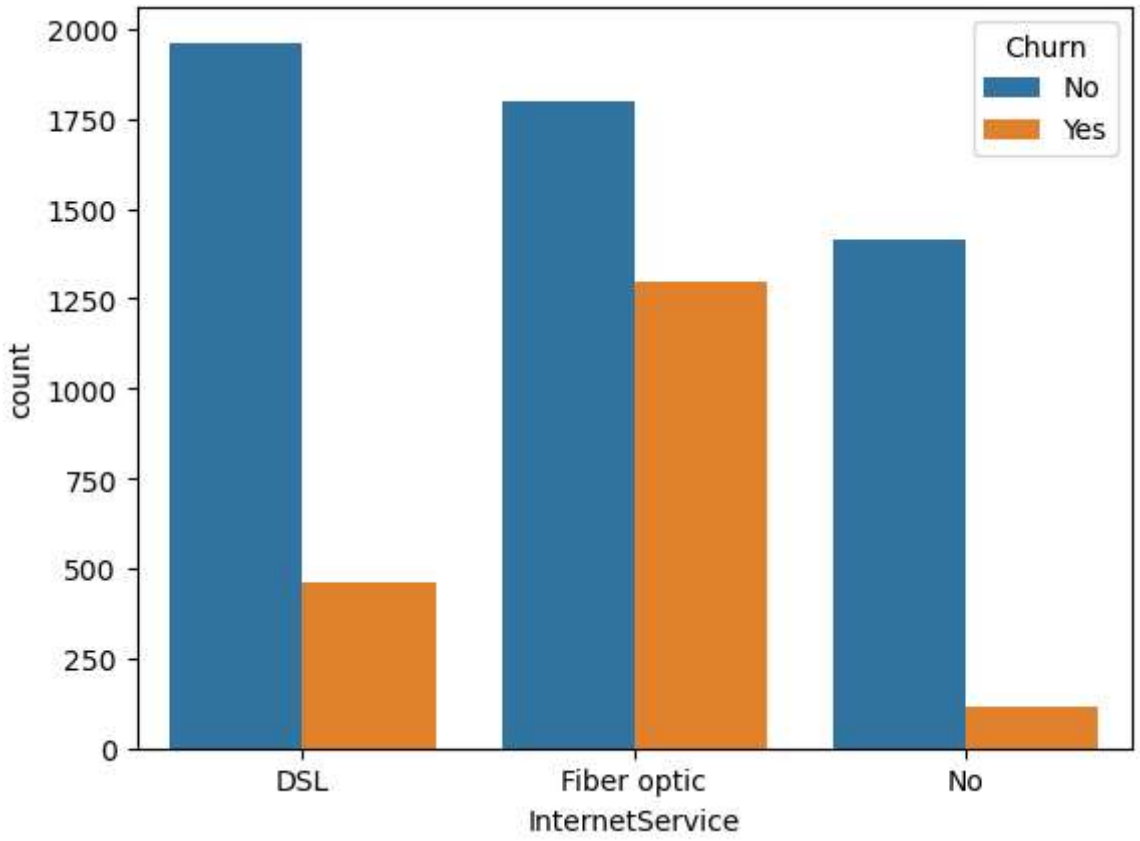
Out[27]:
```
InternetService  Churn
DSL              No       1962
                 Yes       459
Fiber optic      No       1799
                 Yes      1297
No               No       1413
                 Yes       113
Name: customerID, dtype: int64
```

In [28]:
```
df.groupby(['InternetService', 'Churn'])['customerID'].count().plot()
```

Out[28]:
```
<Axes: xlabel='InternetService,Churn'>
```



In [29]:
```
sns.countplot(x = 'InternetService', data = df, hue = 'Churn')
plt.show()
```



In [30]:
```
"""
From the analysis on internet service, fiber optic had higher numbers and rate of chur
services
Hence, services on fiber optic needs to be improved to keep its customers
"""
```

Out[30]:
```
'\nFrom the analysis on internet service, fiber optic had higher numbers and rate of
churned services compared to the other\nservices\nHence, services on fiber optic need
s to be improved to keep its customers\n'
```

In [31]:
```
"""
Impact of online security on churn
"""
```

Out[31]:
```
'\nImpact of online security on churn\n'
```

In [32]:
```
df.groupby('OnlineSecurity')['customerID'].count()
```

Out[32]:
```
OnlineSecurity
No                     3498
No internet service    1526
Yes                    2019
Name: customerID, dtype: int64
```

In [33]:
```
"""
Projection shown that a lot of customers do not have online security services
"""
```

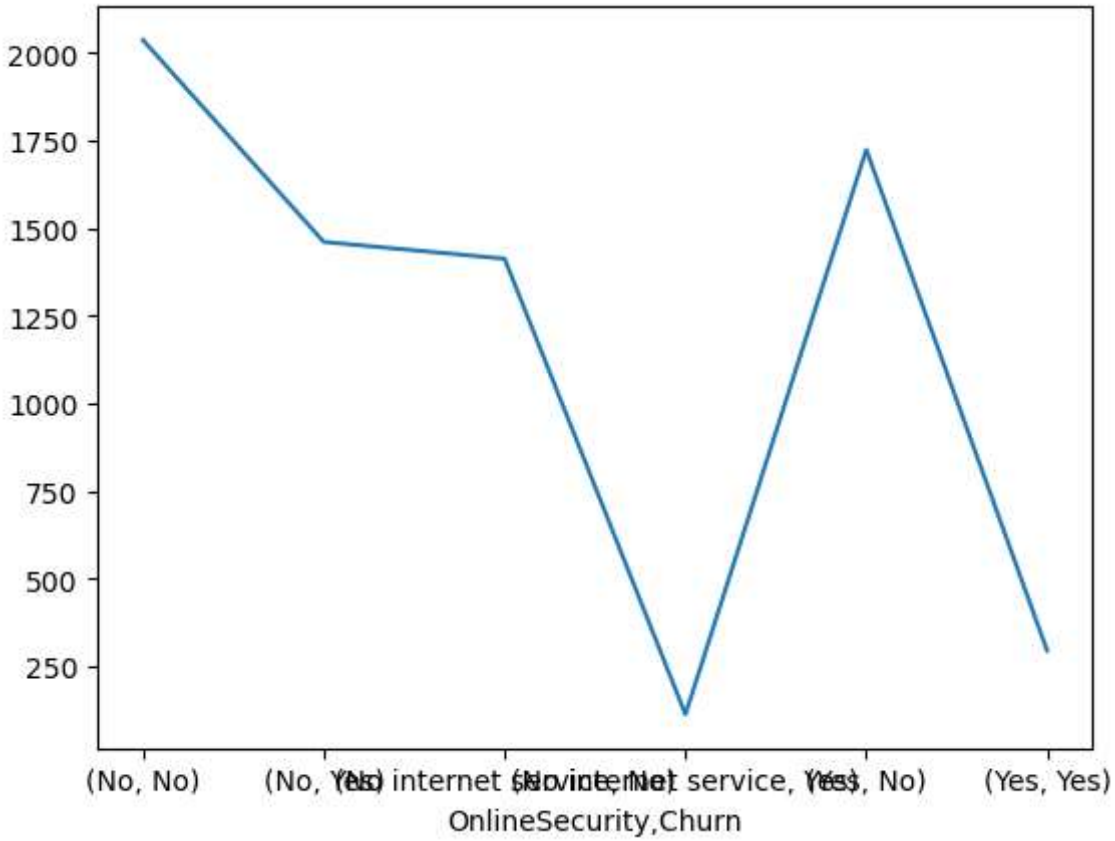
Out[33]: '\nProjection shown that a lot of customers do not have online security services\n'

In [34]:
```
df.groupby(['OnlineSecurity', 'Churn'])['customerID'].count()
```

Out[34]:
```
OnlineSecurity       Churn
No                   No       2037
                     Yes      1461
No internet service  No       1413
                     Yes       113
Yes                  No       1724
                     Yes       295
Name: customerID, dtype: int64
```

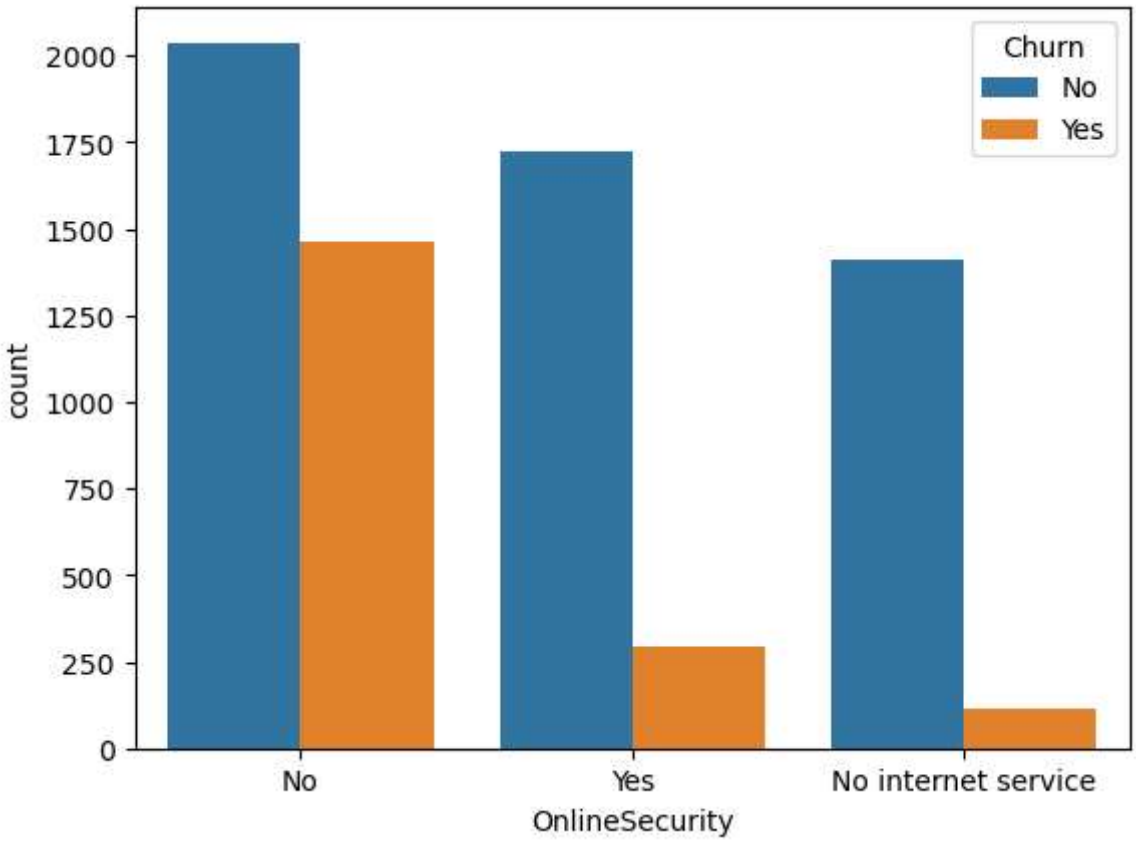
In [35]:
```
df.groupby(['OnlineSecurity', 'Churn'])['customerID'].count().plot()
```

Out[35]: <Axes: xlabel='OnlineSecurity,Churn'>



In [36]:
```
sns.countplot(x = 'OnlineSecurity', data = df, hue = 'Churn')
```

Out[36]: <Axes: xlabel='OnlineSecurity', ylabel='count'>



In [37]:
```
"""
The projection shows that most of the churned services were from customers with no onl
In order to keep customers, the teleco has to work on providing online security to its
"""
```

Out[37]:
```
'\nThe projection shows that most of the churned services were from customers with no
online security\nIn order to keep customers, the teleco has to work on providing onli
ne security to its customers\n'
```

In [38]:
```
"""
Impact on technical Support on churn
"""
```

Out[38]:
```
'\nImpact on technical Support on churn\n'
```

In [39]:
```
df.groupby('TechSupport')['customerID'].count()
```

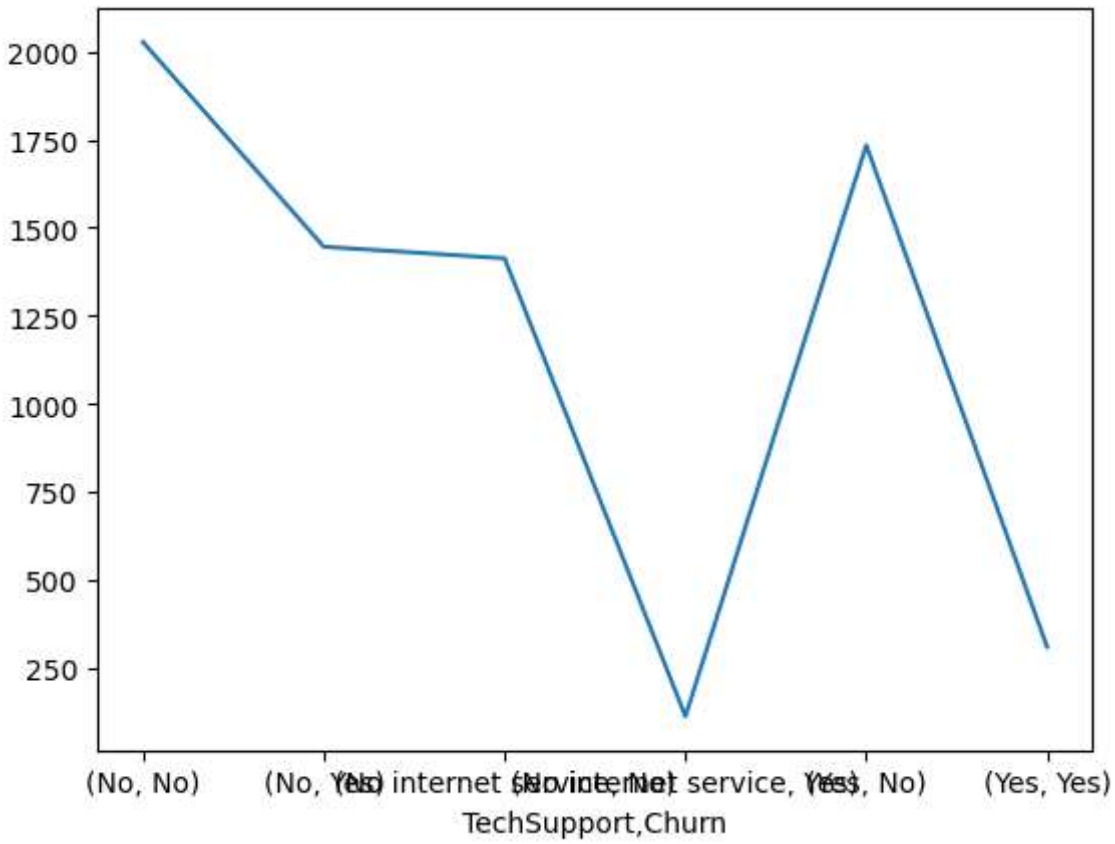
Out[39]:
```
TechSupport
No                     3473
No internet service    1526
Yes                    2044
Name: customerID, dtype: int64
```

In [40]:
```
df.groupby(['TechSupport', 'Churn'])['customerID'].count()
```

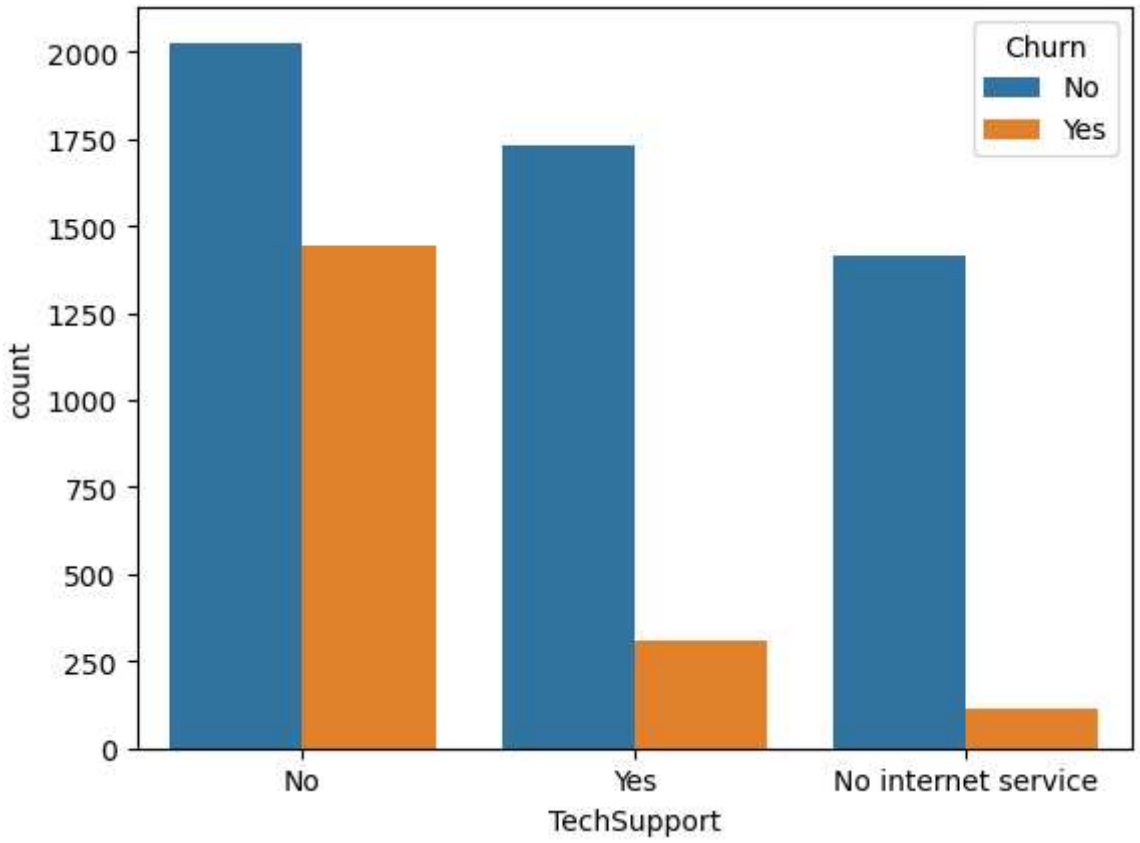
Out[40]:
```
TechSupport          Churn
No                   No       2027
                     Yes      1446
No internet service  No       1413
                     Yes       113
Yes                  No       1734
                     Yes       310
Name: customerID, dtype: int64
```

In [41]:
```
df.groupby(['TechSupport', 'Churn'])['customerID'].count().plot()
```

Out[41]:
```
<Axes: xlabel='TechSupport,Churn'>
```



In [42]:
```
sns.countplot(x = 'TechSupport', data = df, hue = 'Churn')
plt.show()
```



In [43]:
```
"""
The visualization shows that large number of customers had no technical support and mo
The teleco has to work on improving and providing technical support to its customers
"""
```

Out[43]:
```
'\nThe visualization shows that large number of customers had no technical support an
d most of those customers churned services\nThe teleco has to work on improving and p
roviding technical support to its customers\n'
```

In [44]:
```
df.columns
```

Out[44]:
```
Index(['customerID', 'gender', 'SeniorCitizen', 'Partner', 'Dependents',
       'tenure', 'PhoneService', 'MultipleLines', 'InternetService',
       'OnlineSecurity', 'OnlineBackup', 'DeviceProtection', 'TechSupport',
       'StreamingTV', 'StreamingMovies', 'Contract', 'PaperlessBilling',
       'PaymentMethod', 'MonthlyCharges', 'TotalCharges', 'Churn'],
      dtype='object')
```

In [45]:
```
"""
Impact of contract on churn
"""
```

Out[45]:
```
'\nImpact of contract on churn\n'
```

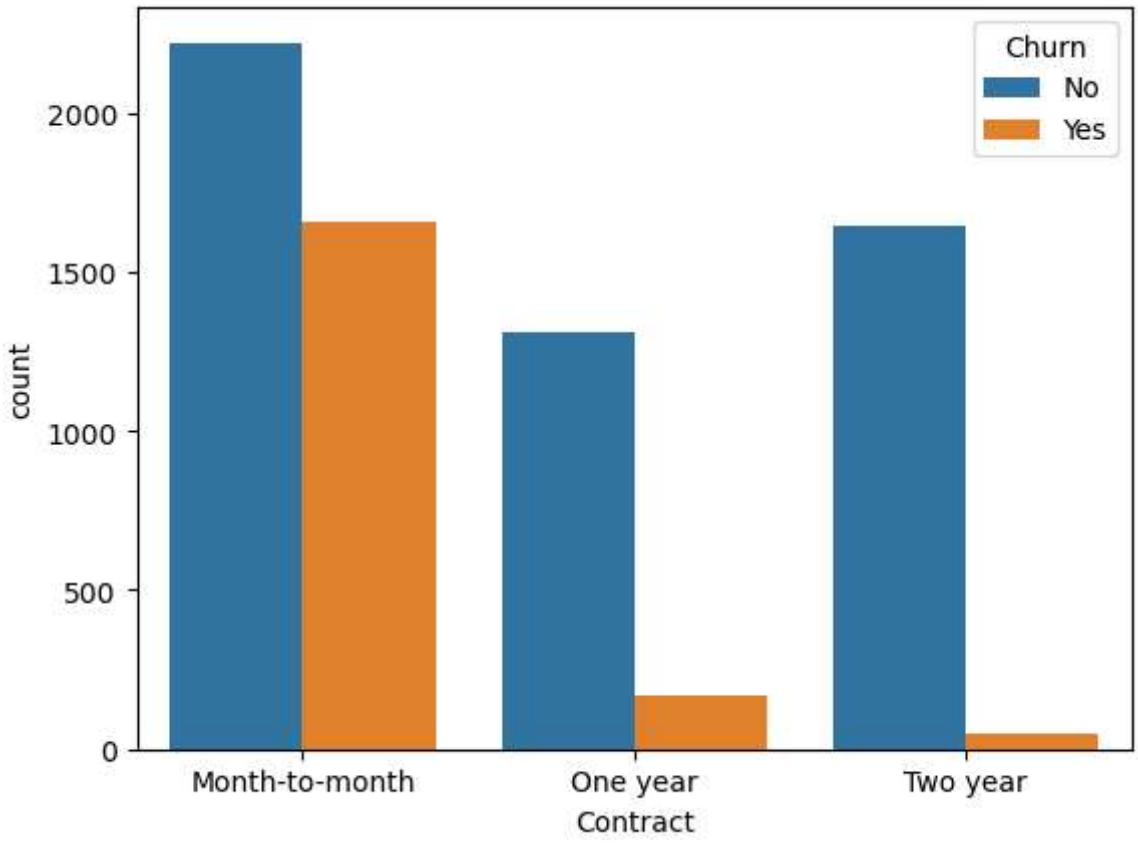
In [46]:
```
df.groupby('Contract')['customerID'].count()
```

Out[46]:
```
Contract
Month-to-month    3875
One year          1473
Two year          1695
Name: customerID, dtype: int64
```

In [47]:
```
df.groupby(['Contract', 'Churn'])['customerID'].count()
```

Out[47]:
```
Contract        Churn
Month-to-month  No       2220
                Yes      1655
One year        No       1307
                Yes       166
Two year        No       1647
                Yes        48
Name: customerID, dtype: int64
```

In [48]:
```
sns.countplot(x = 'Contract', data = df, hue = 'Churn')
plt.show()
```



In [49]:
```
"""
The visualization show that month-to-month had the highest number of customers and als
"""
```

Out[49]:
```
'\nThe visualization show that month-to-month had the highest number of customers and
also had the highest rate of churn servies\n'
```

In [50]:
```
"""
Correlation
"""
```

Out[50]:
```
'\nCorrelation\n'
```

In [51]:
```
df.corr()
```

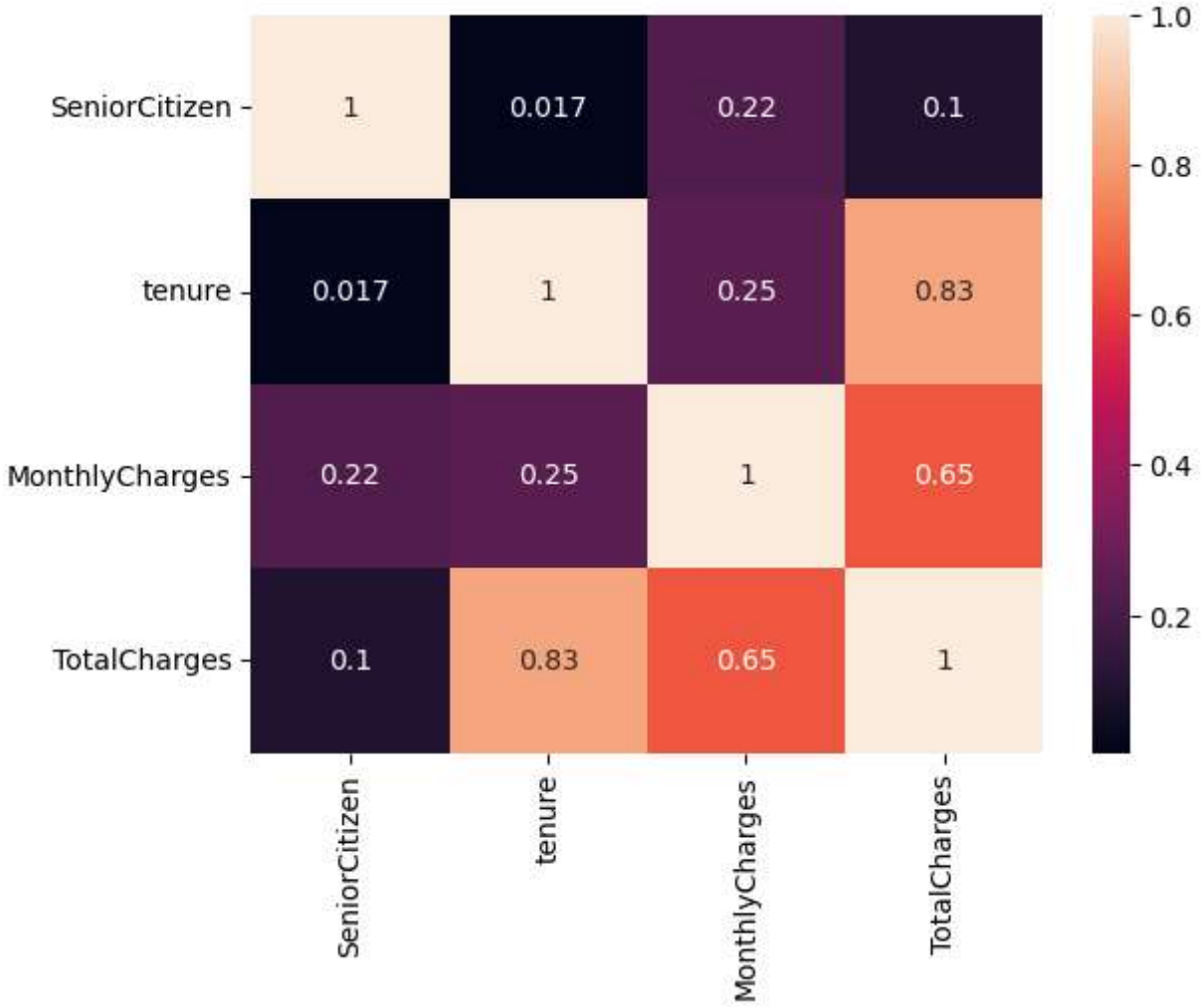
```
C:\Users\USER\AppData\Local\Temp\ipykernel_9428\1134722465.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  df.corr()
```

Out[51]:

	SeniorCitizen	tenure	MonthlyCharges	TotalCharges
SeniorCitizen	1.000000	0.016567	0.220173	0.102411
tenure	0.016567	1.000000	0.247900	0.825880
MonthlyCharges	0.220173	0.247900	1.000000	0.651065
TotalCharges	0.102411	0.825880	0.651065	1.000000

In [52]:
```
sns.heatmap(df.corr(), annot = True)
plt.show()
```

```
C:\Users\USER\AppData\Local\Temp\ipykernel_9428\3614615914.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  sns.heatmap(df.corr(), annot = True)
```



In [53]:
```
"""
Each attritube is highly correlated to itself
Tenure and TotalCharges have the next high correlation
```

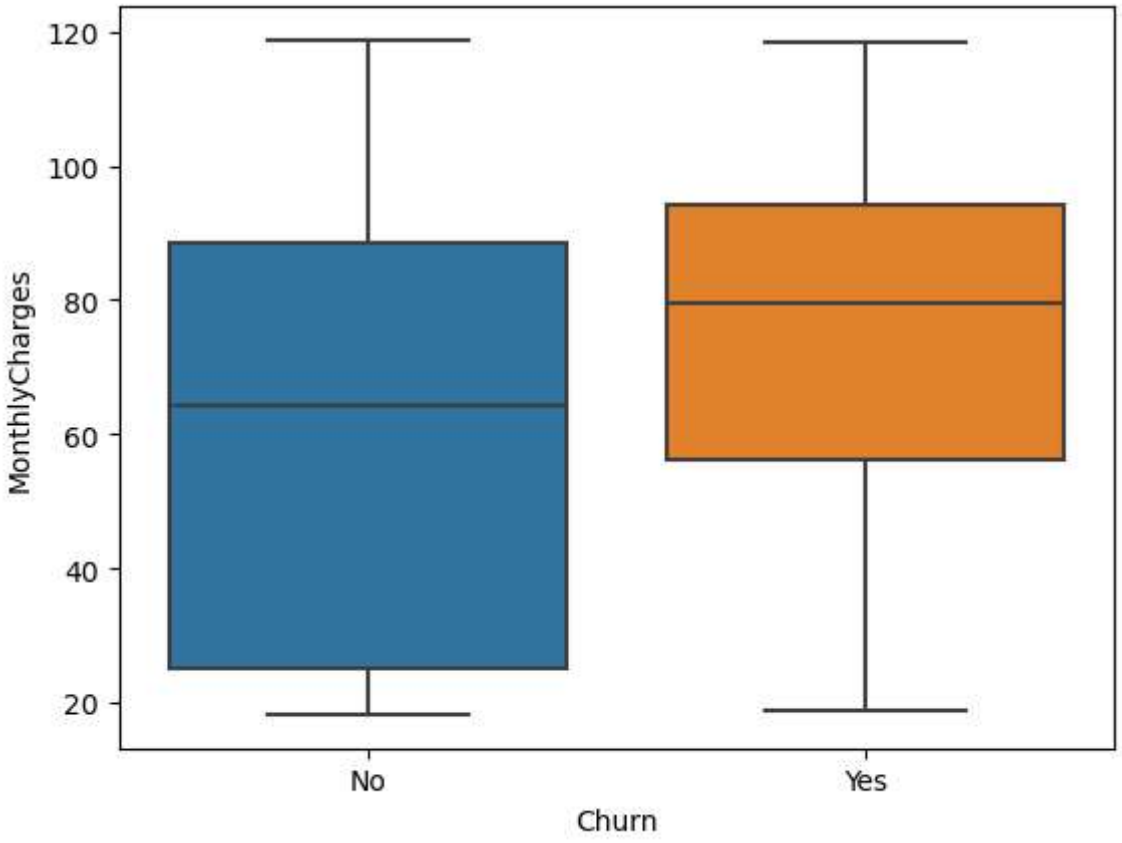
```
SeniorCitizen and Tenure have the least correlation between each other
"""
```

Out[53]:
```
'\nEach attritube is highly correlated to itself\nTenure and TotalCharges have the ne
xt high correlation\nSeniorCitizen and Tenure have the least correlation between each
other\n'
```

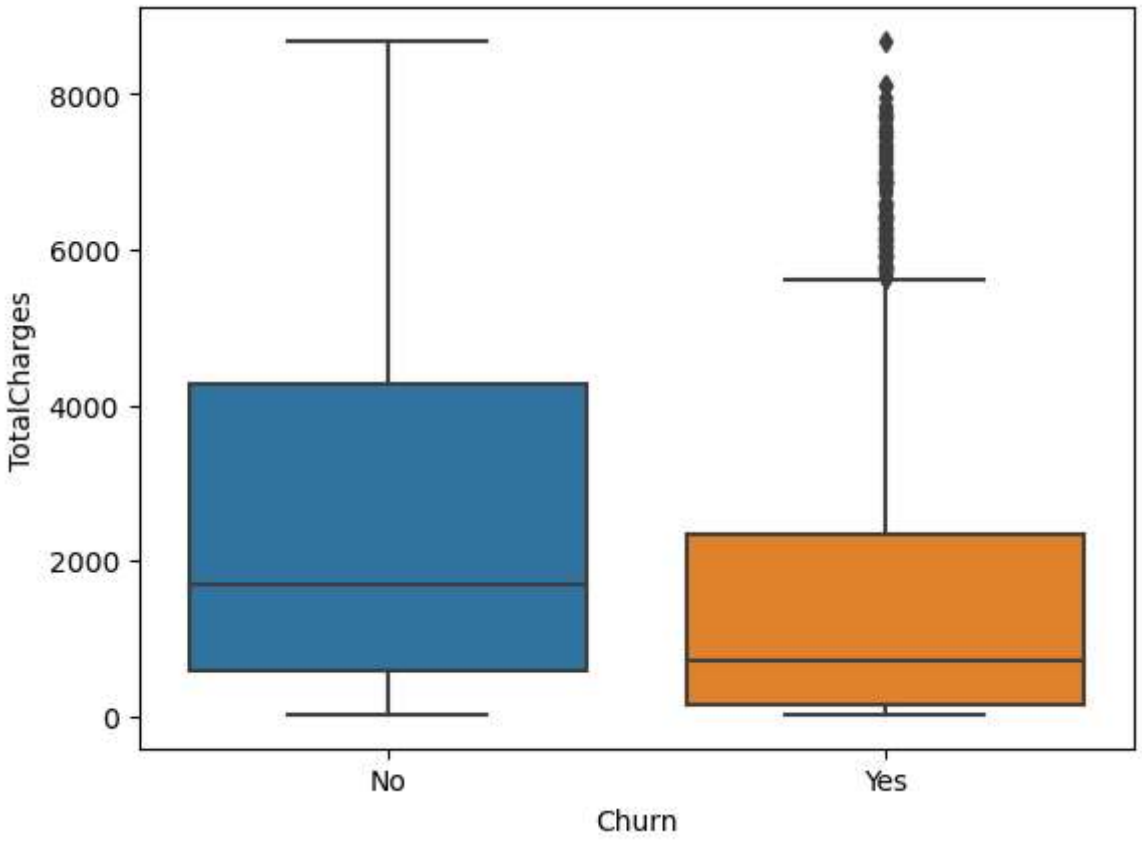
In [54]:
```
"""
Boxplot of Churn on MonthlyCharges and TotalCharges
"""
```

Out[54]:
```
'\nBoxplot of Churn on MonthlyCharges and TotalCharges\n'
```

In [55]:
```
sns.boxplot(x = 'Churn', y = 'MonthlyCharges', data = df)
plt.show()
```



In [56]:
```
sns.boxplot(x = 'Churn', y = 'TotalCharges', data = df)
plt.show()
```



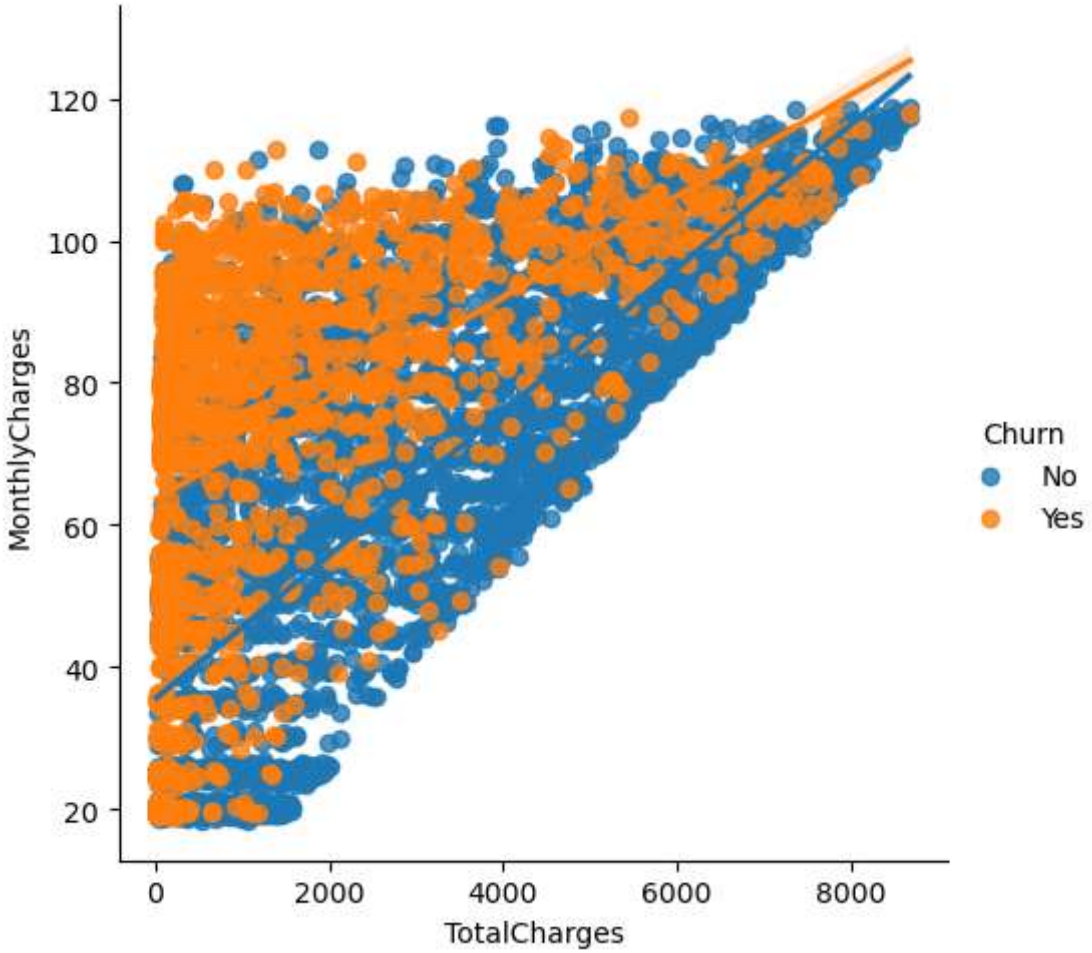
In [58]:
```
"""
From the visualization TotalCharges has outliers
"""
```

Out[58]:
```
'\nFrom the visualization TotalCharges has outliers\n'
```

In [59]:
```
"""
Linear relationship between TotalCharges and MonthlyCharges
"""
```

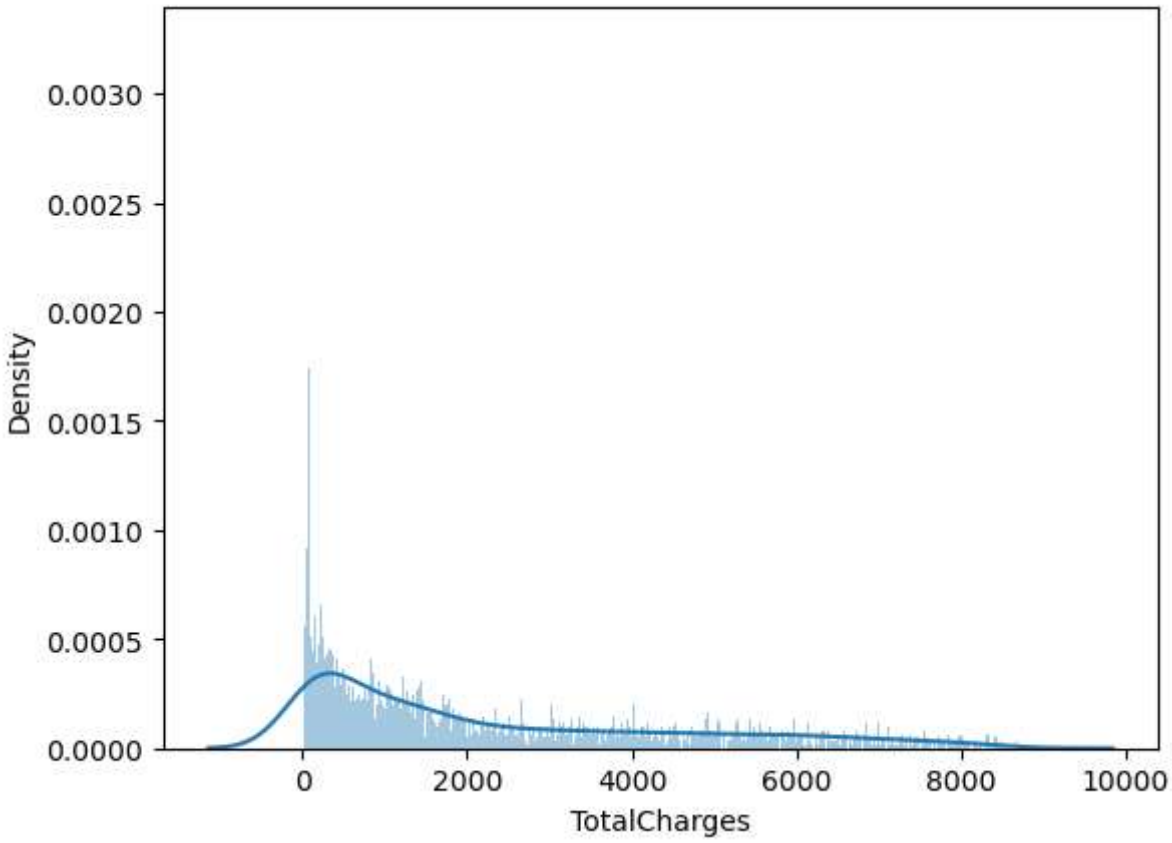
Out[59]:
```
'\nLinear relationship between TotalCharges and MonthlyCharges\n'
```

In [64]:
```
sns.lmplot(x = 'TotalCharges', y = 'MonthlyCharges', data = df, hue = 'Churn')
plt.show()
```



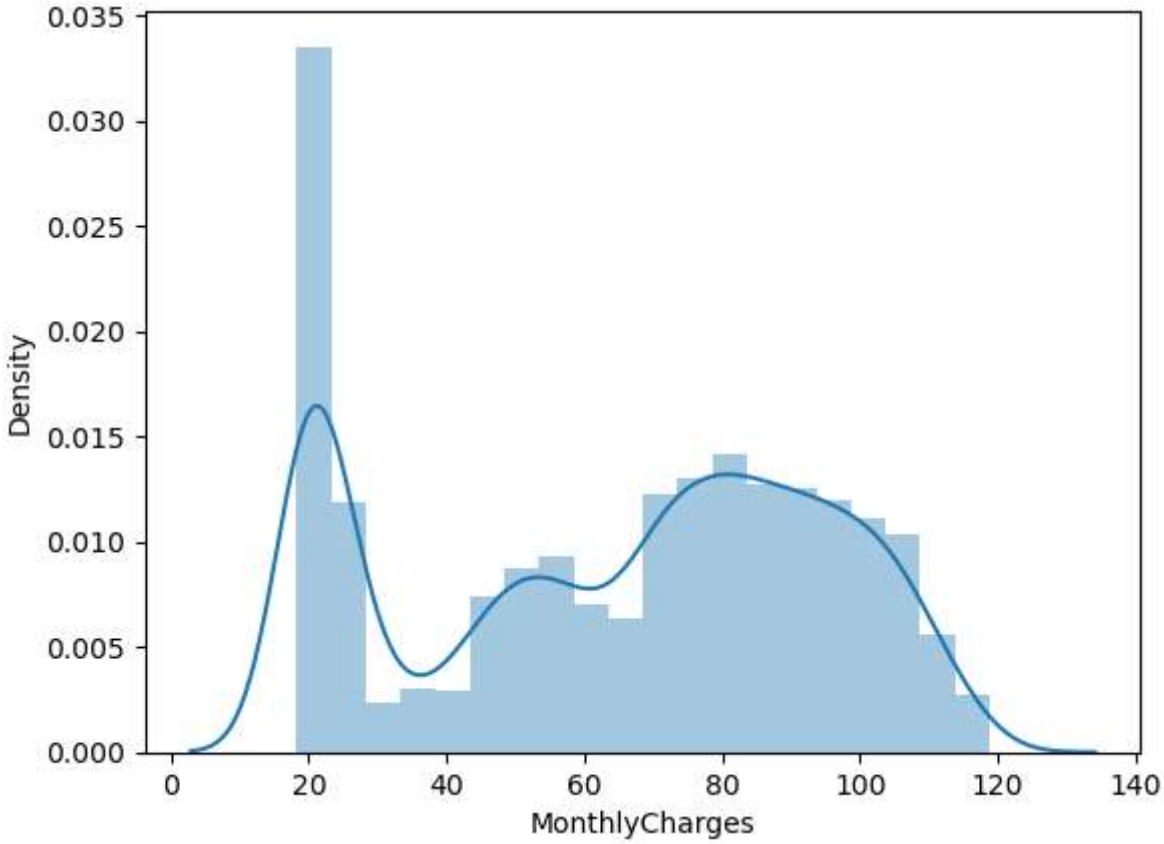
In [67]:
```
sns.distplot(df.TotalCharges, bins = 1000)
plt.show()
```

```
C:\Users\USER\AppData\Local\Temp\ipykernel_9428\3809537282.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df.TotalCharges, bins = 1000)
```



In [73]:
```
sns.distplot(df.MonthlyCharges, bins = 20)
plt.show()
```

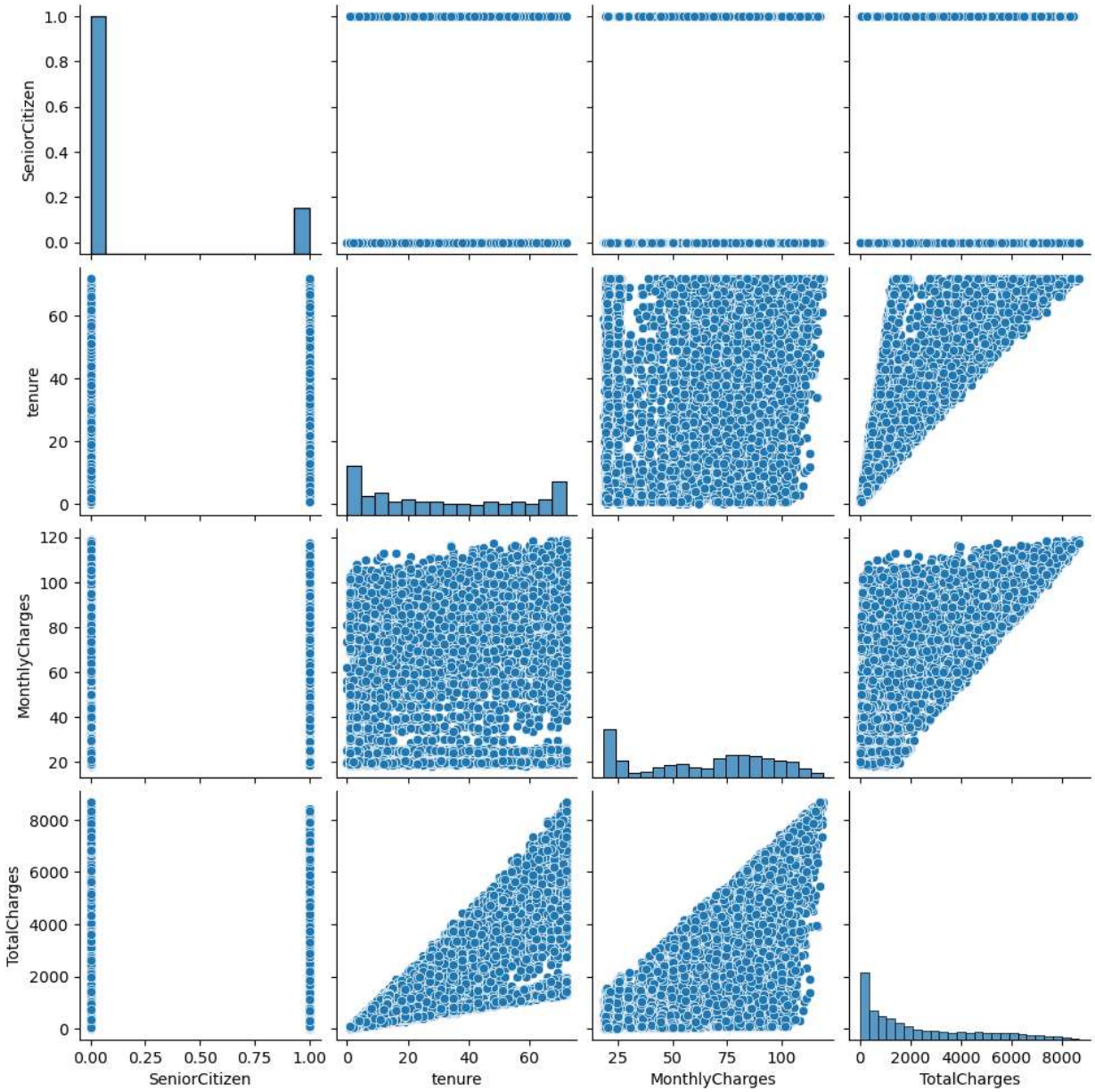
```
C:\Users\USER\AppData\Local\Temp\ipykernel_9428\1557555406.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df.MonthlyCharges, bins = 20)
```



In [69]:
```
df['TotalCharges']
```

Out[69]:
```
0         29.85
1       1889.50
2        108.15
3       1840.75
4        151.65
         ...
7038    1990.50
7039    7362.90
7040     346.45
7041     306.60
7042    6844.50
Name: TotalCharges, Length: 7043, dtype: float64
```

In [75]:
```
sns.pairplot(data = df)
plt.show()
```


SeniorCitizen
tenure
MonthlyCharges
TotalCharges
SeniorCitizen
tenure
MonthlyCharges
TotalCharges

In [ ]:

In [ ]:

In [ ]: