churnprediction

March 28, 2023

1 Importing all Necessary Libraries

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

2 Data Ingestion

[2]:
```
df=pd.read_csv(r'C:
↪\Users\PS4Z\Downloads\archive\WA_Fn-UseC_-Telco-Customer-Churn.csv')
```

[3]:
```
#seeing how the data looks like
df.head()
```

[3]:
```
   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
3  7795-CFOCW    Male              0      No         No      45           No
4  9237-HQITU  Female              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
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        1889.5    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]
```

3 Understanding Data

[4]:
```python
#shape of data
print('Data shape:',df.shape)
```

```
Data shape: (7043, 21)
```

[5]:
```python
#finding null values in data
df.isnull().sum()
```

[5]:
```
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
```

Observations:No Null values

[6]:
```python
#Getting information about data; null counts and data types of data 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
```

[7]:
```python
#list of column names
df.columns
```

[7]:
```
Index(['customerID', 'gender', 'SeniorCitizen', 'Partner', 'Dependents',
       'tenure', 'PhoneService', 'MultipleLines', 'InternetService',
       'OnlineSecurity', 'OnlineBackup', 'DeviceProtection', 'TechSupport',
       'StreamingTV', 'StreamingMovies', 'Contract', 'PaperlessBilling',
       'PaymentMethod', 'MonthlyCharges', 'TotalCharges', 'Churn'],
      dtype='object')
```

[8]:
```python
#getting data types of each column header
df.dtypes
```

[8]:
```
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          object
Churn                 object
dtype: object
```

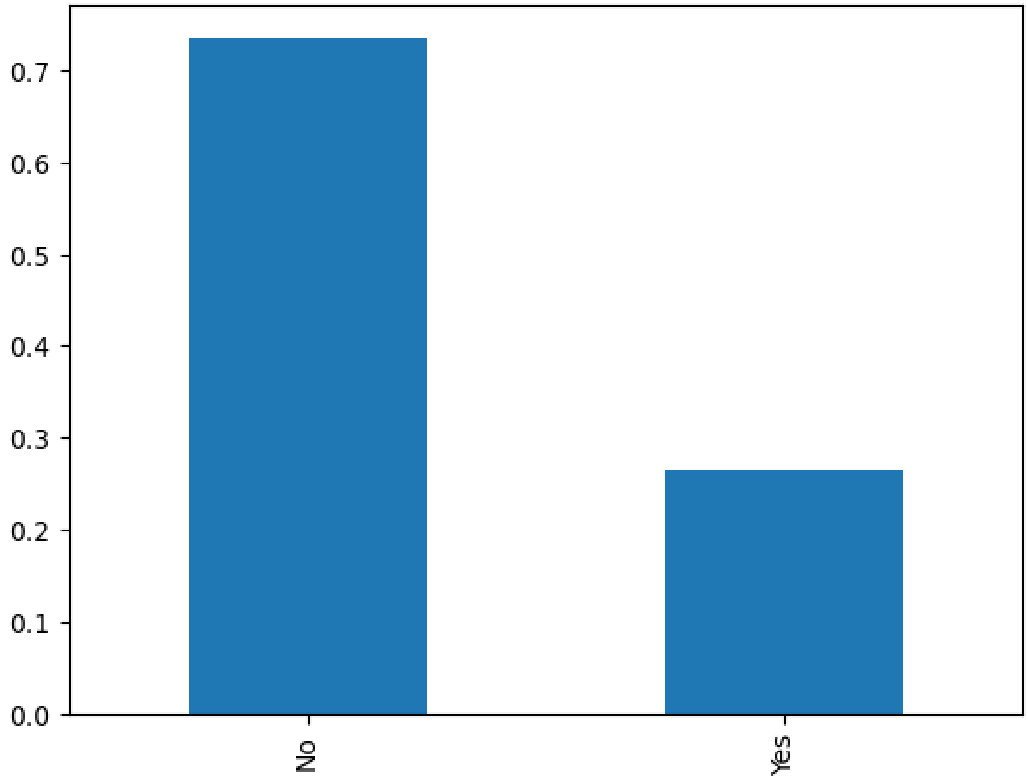
[9]:
```
#checking for duplicate values
df.duplicated().sum()
```

[9]:
```
0
```

Observations:No duplicate Values

4 Understanding Imbalanced Data

[10]:
```
#Plotting Bar Plot for target variable
df['Churn'].value_counts(normalize=True).plot(kind='bar');
```



Observation:we can see above data is imbalanced, more class belongs to no than yes

[11]:
```
#getting a sample from data
df.sample(7)
```

[11]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	\
236	0621-JFHOL	Female	0	No	No	10	
4714	0016-QLJIS	Female	0	Yes	Yes	65	
6471	0859-YGKFW	Male	0	Yes	Yes	18	
375	7156-MXBJE	Female	0	No	No	43	
4908	3957-LXOLK	Female	1	No	No	28	
1270	8780-IHCRN	Male	0	Yes	Yes	63	
3205	3810-DVDQQ	Female	0	Yes	Yes	72	

	PhoneService	MultipleLines	InternetService	OnlineSecurity	…	\
236	No	No phone service	DSL	No	…	
4714	Yes	Yes	DSL	Yes	…	
6471	Yes	No	No	No internet service	…	
375	Yes	Yes	DSL	No	…	
4908	Yes	Yes	Fiber optic	No	…	
1270	Yes	Yes	No	No internet service	…	
3205	Yes	Yes	Fiber optic	Yes	…	

```
        DeviceProtection          TechSupport          StreamingTV  \
236                   No                  Yes                   No
4714                 Yes                  Yes                  Yes
6471  No internet service  No internet service  No internet service
375                  Yes                  Yes                  Yes
4908                 Yes                   No                  Yes
1270  No internet service  No internet service  No internet service
3205                 Yes                  Yes                  Yes

         StreamingMovies        Contract PaperlessBilling  \
236                   No        Two year              Yes
4714                 Yes        Two year              Yes
6471  No internet service        One year               No
375                  Yes        One year               No
4908                 Yes  Month-to-month              Yes
1270  No internet service        Two year               No
3205                 Yes        Two year              Yes

                  PaymentMethod MonthlyCharges  TotalCharges Churn
236                Mailed check          29.60        299.05    No
4714               Mailed check          90.45        5957.9    No
6471  Bank transfer (automatic)          20.05         345.9    No
375     Credit card (automatic)          85.10       3662.25    No
4908           Electronic check         106.15        3152.5   Yes
1270    Credit card (automatic)          24.65        1574.5    No
3205  Bank transfer (automatic)         117.60        8308.9    No

[7 rows x 21 columns]
```

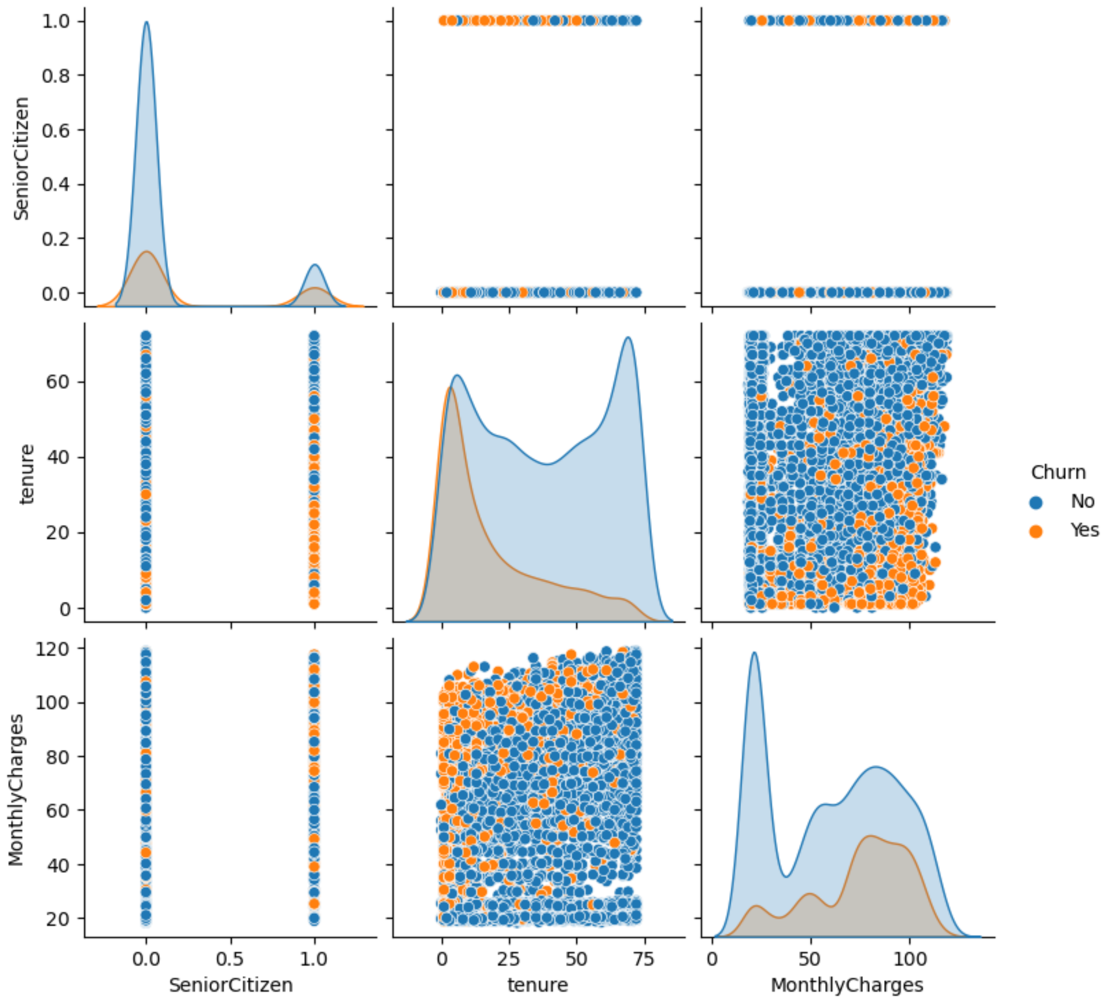
[12]:
```python
#Getting 5 point summary for all numercial features
df.describe()
```

[12]:

	SeniorCitizen	tenure	MonthlyCharges
count	7043.000000	7043.000000	7043.000000
mean	0.162147	32.371149	64.761692
std	0.368612	24.559481	30.090047
min	0.000000	0.000000	18.250000
25%	0.000000	9.000000	35.500000
50%	0.000000	29.000000	70.350000
75%	0.000000	55.000000	89.850000
max	1.000000	72.000000	118.750000

5 Visualizing the Data

[13]:
```
#pairplot shows graphical representation of all numerical features with one␣
↪another with target variable in legend
sns.pairplot(data=df,hue='Churn');
```

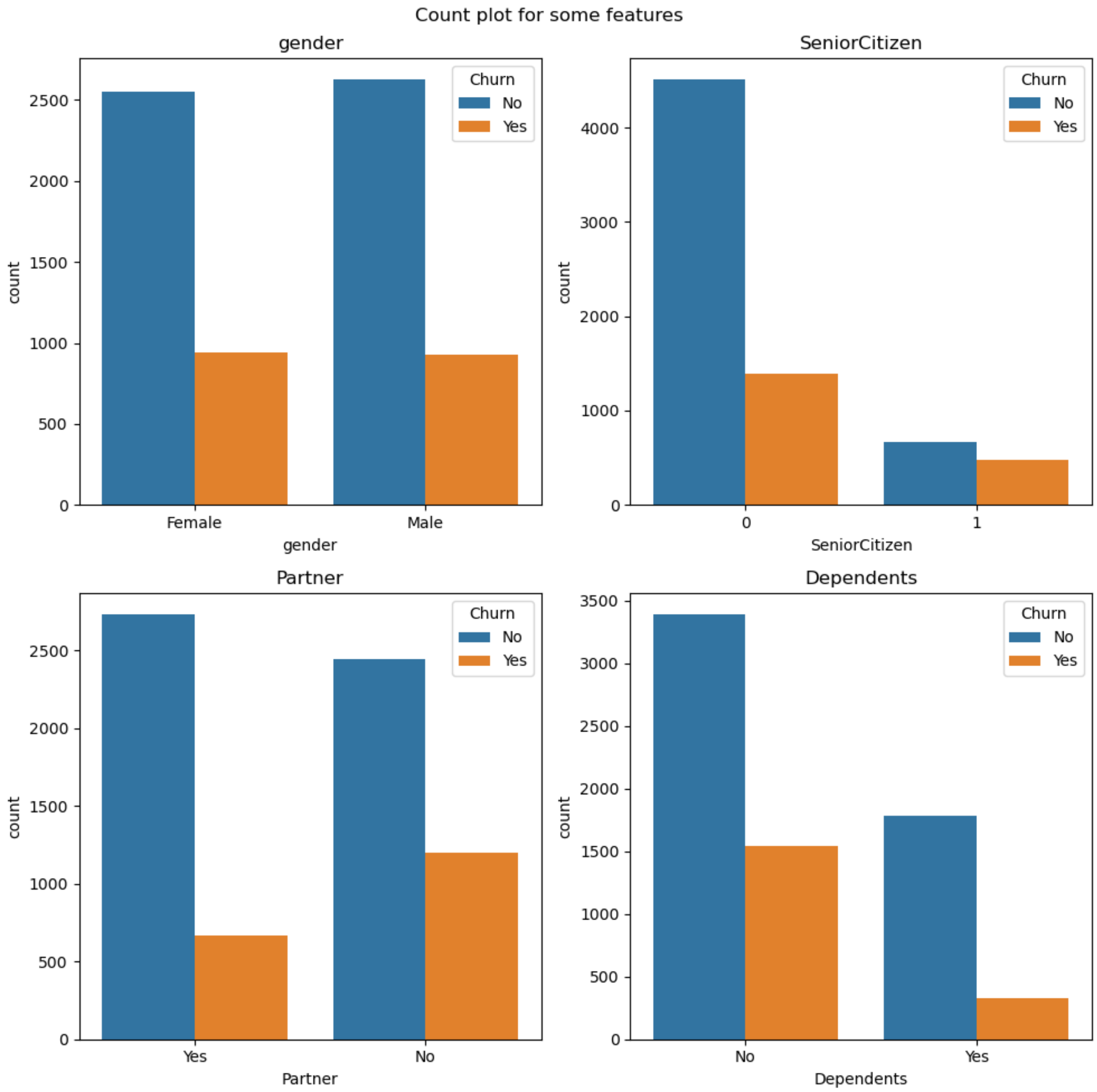


[14]:
```
feat_df=df[['gender','SeniorCitizen','Partner','Dependents']]
feat=feat_df.columns
feat
```

[14]: Index(['gender', 'SeniorCitizen', 'Partner', 'Dependents'], dtype='object')

[15]:
```
#plotting features based on Target variable as hue to draw observations
plt.figure(figsize=(10,10))
plt.suptitle('Count plot for some features')
for a in range(0,len(feat)):
```

```
    plt.subplot(2,2,a+1)
    sns.countplot(df[feat[a]],hue=df['Churn'])
    plt.title(label=feat[a])
    plt.tight_layout();
```

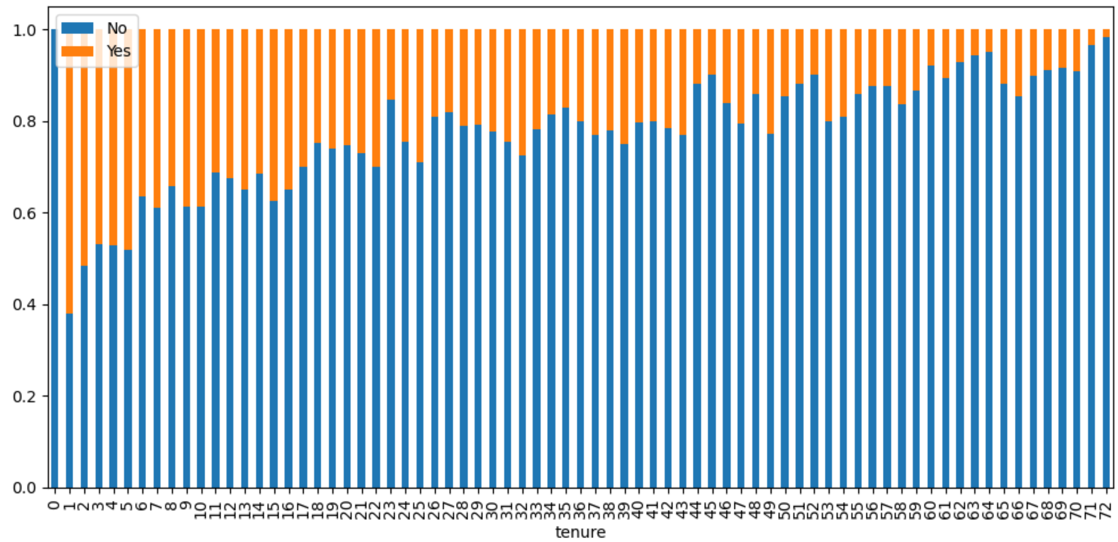


Observations:

1. No significant relation between Churn and gender
2. One who is not Senior Citizen is more likely to be churned than a Senior Citizen
3. One who don't have Partners got most churned.
4. One who doesn't have dependents got most churned

[16]:
```
#plotting features based on Target variable as hue to draw observations
plt.figure(figsize=(10,10))
exp=pd.crosstab(df['tenure'],df['Churn'])
exp.div(exp.sum(1).astype(float),axis=0).
 ↪plot(kind="bar",stacked=True,figsize=(10,5))
plt.legend(loc=0)
plt.tight_layout();
```

```
<Figure size 1000x1000 with 0 Axes>
```



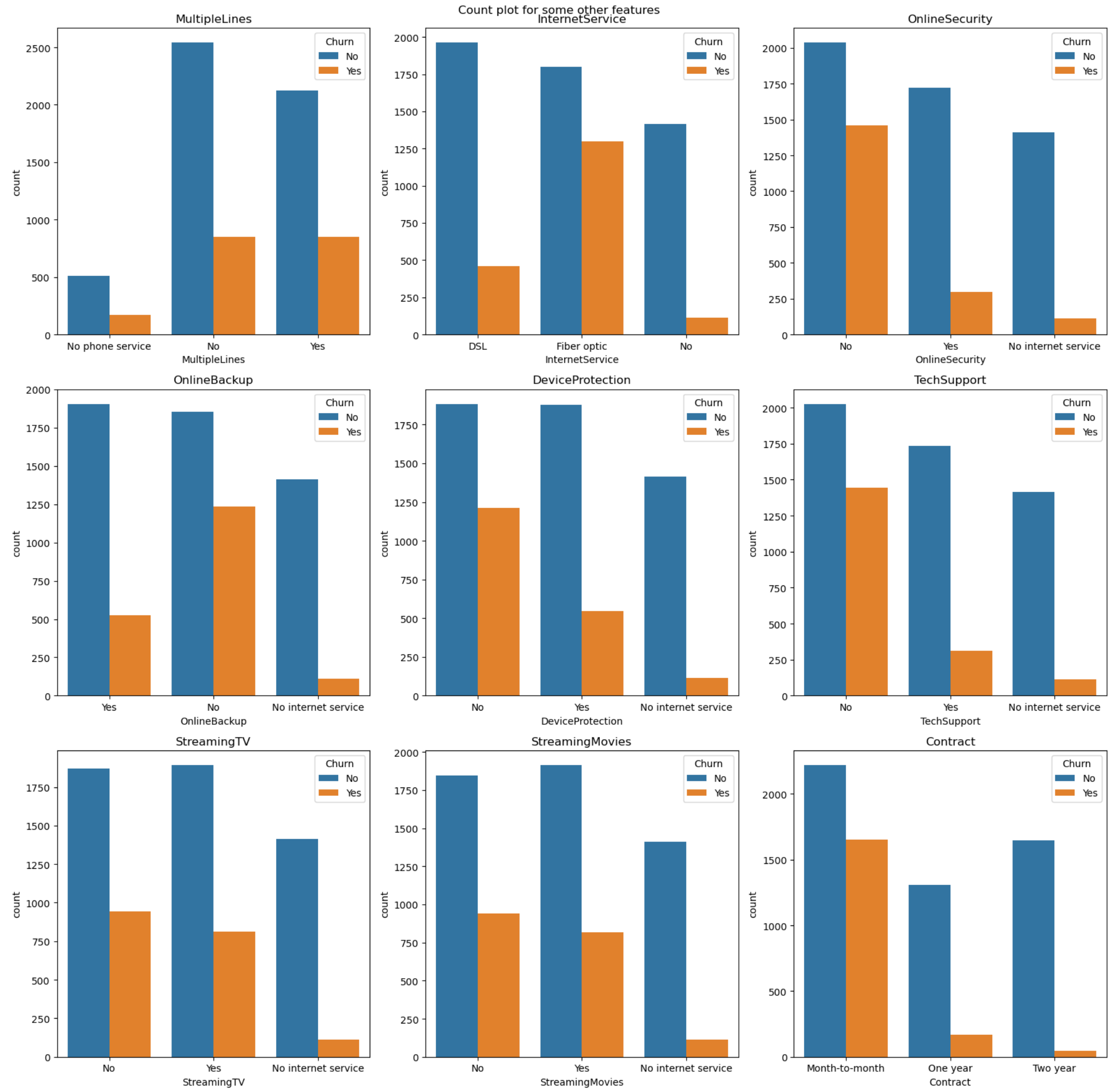
Observations:new customers have a maximum churning rate, we can convert these months into years so it would be easy to target the new customers.

[17]:
```
feat1_df=df[['MultipleLines','InternetService','OnlineSecurity','OnlineBackup','DeviceProtecti
               'StreamingMovies','Contract']]
feat1=feat1_df.columns
feat1
```

[17]:
```
Index(['MultipleLines', 'InternetService', 'OnlineSecurity', 'OnlineBackup',
       'DeviceProtection', 'TechSupport', 'StreamingTV', 'StreamingMovies',
       'Contract'],
      dtype='object')
```

[18]:
```
#plotting features based on Target variable as hue to draw observations
plt.figure(figsize=(16,16))
plt.suptitle('Count plot for some other features')
for a in range(0,len(feat1)):
    plt.subplot(3,3,a+1)
    sns.countplot(df[feat1[a]],hue=df['Churn'])
```

```
    plt.title(label=feat1[a])
    plt.tight_layout();
```



Observations:

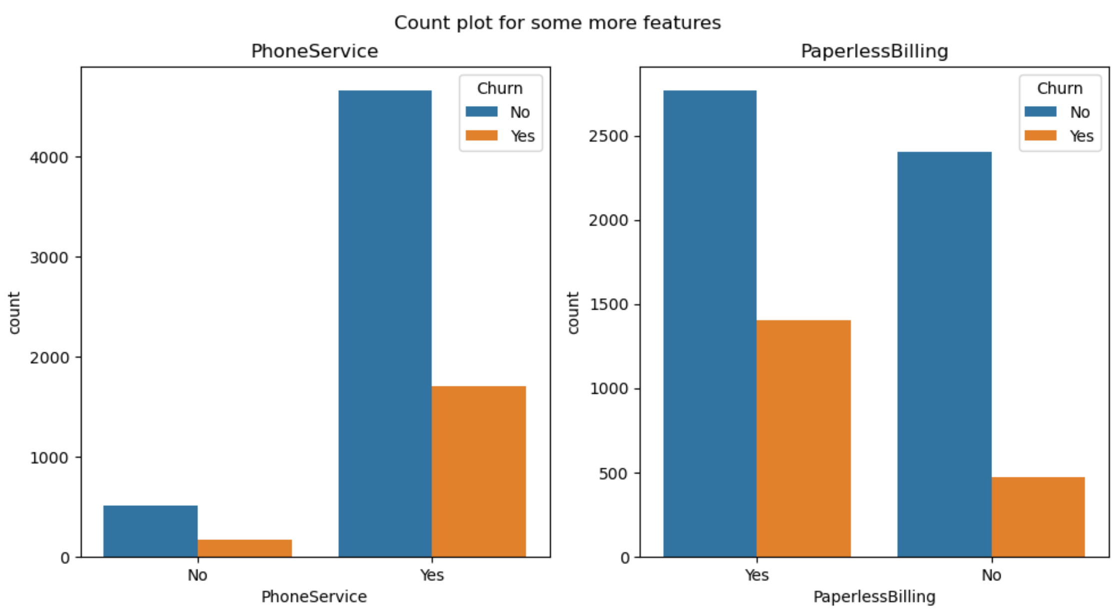
1. Those who has taken fiber optic internet service more likely to be churn.
2. We know the consequences of Cyber vulnerability, so it's obvious that customer will trust us if we will give them full security.
3. The customers who are not getting online backup services are more likely to be churned.
4. One who has not got Device protection guarantee are more likely to be churned.
5. One who has no technology support are more likely to be churned.

6. There is no significant change between the churning rate of with StreamingTV and without StreamingTV services, but at some point one who has not this service are more likely to be churned.
7. There is no significant change between the churning rate of with StreamingMovies and without StreamingMovies services. but at some point one who has not this service are more likely to be churned.
8. We have to focus on the retaintion of Month-Month Customers by providing them good quality services.
9. No significant change between churning rate of Customes having multiple line services or not, also no significance without phone services as well.

[19]:
```
feat2_df=df[['PhoneService','PaperlessBilling']]
feat2=feat2_df.columns
feat2
```

[19]:
```
Index(['PhoneService', 'PaperlessBilling'], dtype='object')
```

[20]:
```
#plotting features based on Target variable as hue to draw observations
plt.figure(figsize=(10,10))
plt.suptitle('Count plot for some more features')
for a in range(0,len(feat2)):
    plt.subplot(2,2,a+1)
    sns.countplot(df[feat2[a]],hue=df['Churn'])
    plt.title(label=feat2[a])
    plt.tight_layout();
```

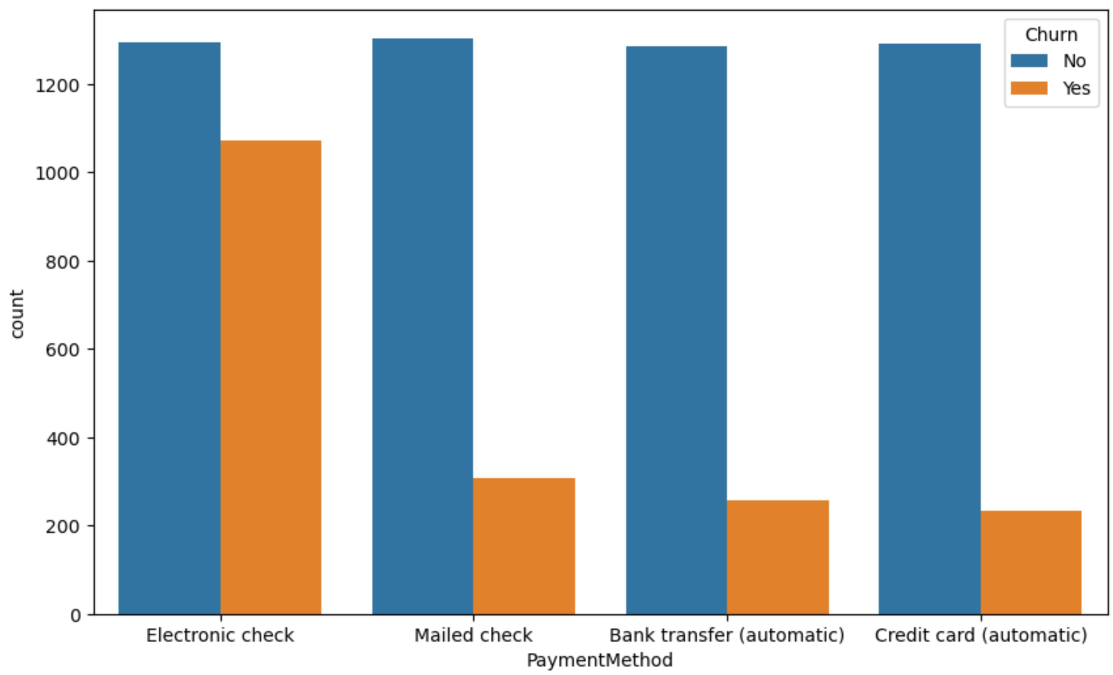


Observations:

1. One who has Phone Services are more likely to be churned than those who do not.
2. One who has PaperlessBilling are more likely to be chured than those who do not.

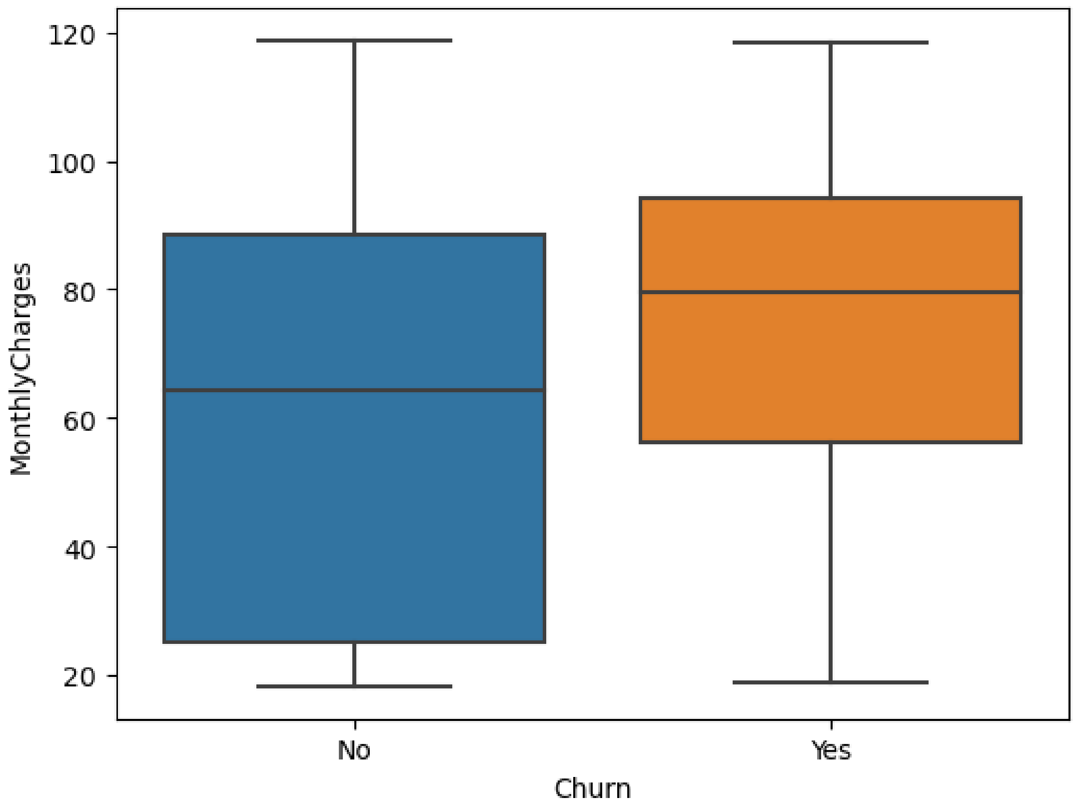
[21]:
```
#for feature PaymentMethod
plt.figure(figsize=(10,6))
sns.countplot(df['PaymentMethod'],hue=df['Churn'])
```

[21]: `<AxesSubplot:xlabel='PaymentMethod', ylabel='count'>`



Observation:Most of customers who has chosen Electronic payment method more likely to be churned.

[22]:
```
#plotting feature MonthlyCharges
sns.boxplot(y=df['MonthlyCharges'],x=df['Churn']);
```



Observations:one who paying more are likely to be churned.

[23]:
```
#Changing data type to float and replacing blank values to 0 in feature␣
↪TotalCharges
df['TotalCharges']=df['TotalCharges'].replace(' ',0)
df['TotalCharges']=pd.to_numeric(df['TotalCharges'],downcast="float")
```

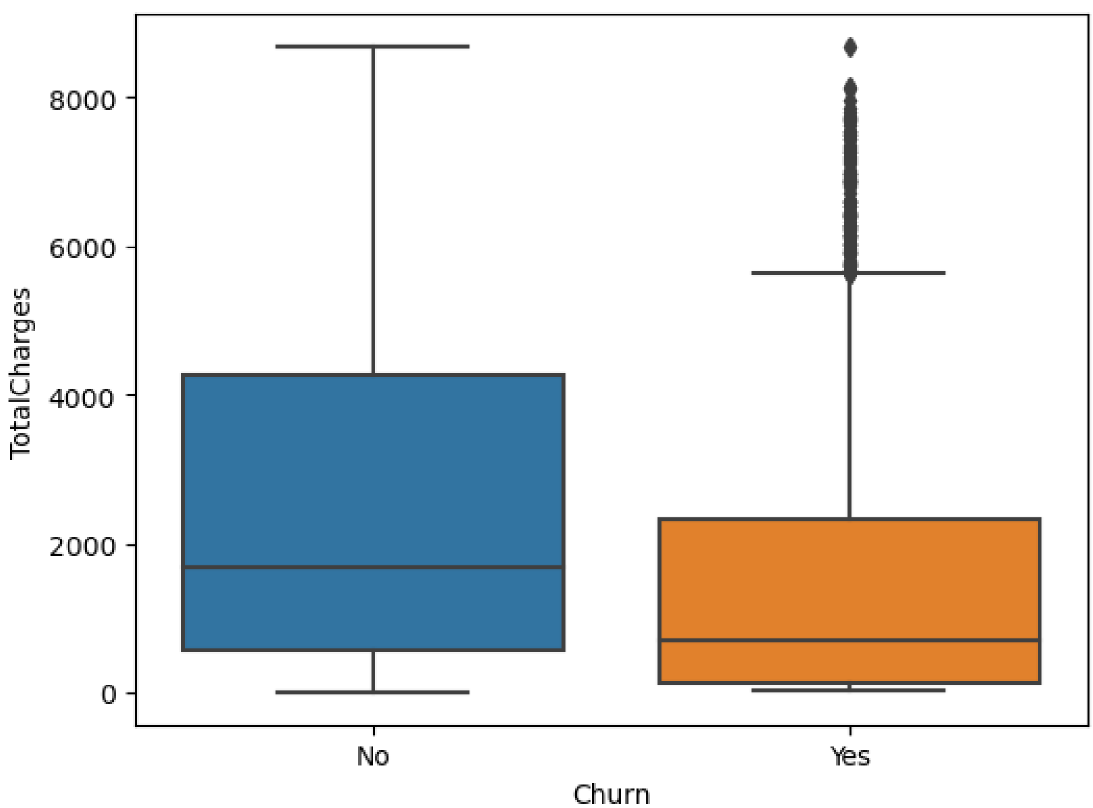
[24]:
```
df.head()
```

[24]:
```
   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
3  7795-CFOCW    Male              0      No         No      45           No
4  9237-HQITU  Female              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
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     29.850000     No
1              Mailed check          56.95   1889.500000     No
2              Mailed check          53.85    108.150002    Yes
3 Bank transfer (automatic)          42.30   1840.750000     No
4          Electronic check          70.70    151.649994    Yes

[5 rows x 21 columns]
```

[25]:

```python
sns.boxplot(y=df['TotalCharges'],x=df['Churn']);
```



Observations:The outliers present in the total charges With respect to churn rate. Will take care of it in next step.

6 Feature Engineering

[26]:
```
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
```

[27]:
```
# Convert all the categorical features into numerical

# define class

encode=LabelEncoder()

df['gender']=encode.fit_transform(df['gender'])
df['Partner']=encode.fit_transform(df['Partner'])
df['Dependents']=encode.fit_transform(df['Dependents'])
df['PhoneService']=encode.fit_transform(df['PhoneService'])
df['MultipleLines']=encode.fit_transform(df['MultipleLines'])
df['InternetService']=encode.fit_transform(df['InternetService'])
df['OnlineSecurity']=encode.fit_transform(df['OnlineSecurity'])
df['OnlineBackup']=encode.fit_transform(df['OnlineBackup'])
df['DeviceProtection']=encode.fit_transform(df['DeviceProtection'])
df['TechSupport']=encode.fit_transform(df['TechSupport'])
df['StreamingTV']=encode.fit_transform(df['StreamingTV'])
df['StreamingMovies']=encode.fit_transform(df['StreamingMovies'])
df['Contract']=encode.fit_transform(df['Contract'])
df['PaperlessBilling']=encode.fit_transform(df['PaperlessBilling'])
df['PaymentMethod']=encode.fit_transform(df['PaymentMethod'])
df['Churn']=encode.fit_transform(df['Churn'])
```

[28]:
```
# Convert tenure feature into 3 category (we have taken 2 year difference␣
↪according to the previous Analysis.)
# 0-24 Months-->1, 25-48 Months--->2 and else is 3

df['tenure']=df['tenure'].map(lambda x: 1 if x<=24 else 2 if x<=48 else 3)
```

[29]:
```
#define class

scale=StandardScaler()

df['MonthlyCharges']=scale.fit_transform(df['MonthlyCharges'].values.
↪reshape(-1,1))
df['TotalCharges']=scale.fit_transform(df['TotalCharges'].values.reshape(-1,1))
```

7 Handling Outliers

[30]:
```
# df.groupby('Churn')['TotalCharges'].mean()
mean_=df.groupby('Churn')['TotalCharges'].mean()
mean_
```

[30]:
```
Churn
0    0.119198
1   -0.329978
Name: TotalCharges, dtype: float32
```

[31]:
```
mean=1531.796143
df1=df[(df['Churn']=='Yes')&(df['TotalCharges']>=mean)]['TotalCharges'].
 ↪map(lambda x: mean if x>mean else x)
df['TotalCharges'].update(df1)
```

8 Checking for Multicollinearity

[32]:
```
#checking correlation between all numerical features in the dataset
df.corr()
```

[32]:
```
                    gender  SeniorCitizen   Partner  Dependents    tenure  \
gender            1.000000      -0.001874 -0.001808    0.010517  0.006577
SeniorCitizen    -0.001874       1.000000  0.016479   -0.211185  0.018821
Partner          -0.001808       0.016479  1.000000    0.452676  0.348668
Dependents        0.010517      -0.211185  0.452676    1.000000  0.141257
tenure            0.006577       0.018821  0.348668    0.141257  1.000000
PhoneService     -0.006488       0.008576  0.017706   -0.001762  0.004960
MultipleLines    -0.006739       0.146185  0.142410   -0.024991  0.316073
InternetService  -0.000863      -0.032310  0.000891    0.044590 -0.031103
OnlineSecurity   -0.015017      -0.128221  0.150828    0.152166  0.294287
OnlineBackup     -0.012057      -0.013632  0.153130    0.091015  0.348974
DeviceProtection  0.000549      -0.021398  0.166330    0.080537  0.339041
TechSupport      -0.006825      -0.151268  0.126733    0.133524  0.297290
StreamingTV      -0.006421       0.030776  0.137341    0.046885  0.268137
StreamingMovies  -0.008743       0.047266  0.129574    0.021321  0.276032
Contract          0.000126      -0.142554  0.294806    0.243187  0.626061
PaperlessBilling -0.011754       0.156530 -0.014877   -0.111377  0.005004
PaymentMethod     0.017352      -0.038551 -0.154798   -0.040292 -0.343300
MonthlyCharges   -0.014569       0.220173  0.096848   -0.113890  0.233516
TotalCharges     -0.000080       0.103006  0.317504    0.062078  0.789548
Churn            -0.008612       0.150889 -0.150448   -0.164221 -0.318469

                  PhoneService  MultipleLines  InternetService  \
gender               -0.006488      -0.006739        -0.000863
SeniorCitizen         0.008576       0.146185        -0.032310
```

Partner	0.017706	0.142410	0.000891
Dependents	-0.001762	-0.024991	0.044590
tenure	0.004960	0.316073	-0.031103
PhoneService	1.000000	-0.020538	0.387436
MultipleLines	-0.020538	1.000000	-0.109216
InternetService	0.387436	-0.109216	1.000000
OnlineSecurity	-0.015198	0.007141	-0.028416
OnlineBackup	0.024105	0.117327	0.036138
DeviceProtection	0.003727	0.122318	0.044944
TechSupport	-0.019158	0.011466	-0.026047
StreamingTV	0.055353	0.175059	0.107417
StreamingMovies	0.043870	0.180957	0.098350
Contract	0.002247	0.110842	0.099721
PaperlessBilling	0.016505	0.165146	-0.138625
PaymentMethod	-0.004184	-0.176793	0.086140
MonthlyCharges	0.247398	0.433576	-0.323260
TotalCharges	0.113214	0.452577	-0.175755
Churn	0.011942	0.038037	-0.047291

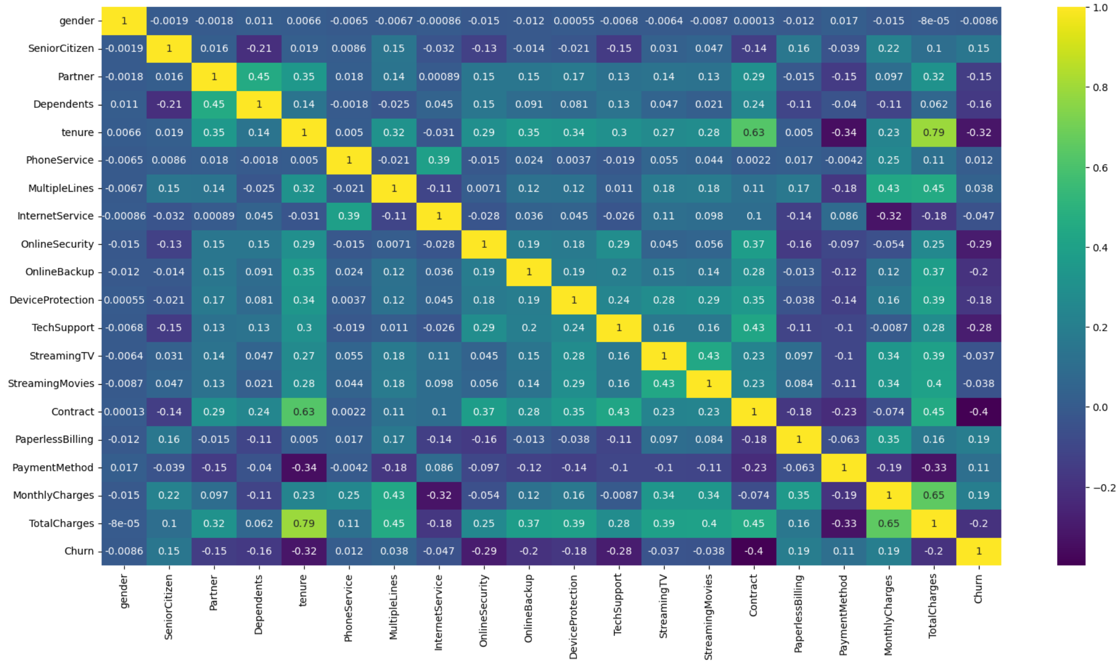
	OnlineSecurity	OnlineBackup	DeviceProtection	TechSupport \
gender	-0.015017	-0.012057	0.000549	-0.006825
SeniorCitizen	-0.128221	-0.013632	-0.021398	-0.151268
Partner	0.150828	0.153130	0.166330	0.126733
Dependents	0.152166	0.091015	0.080537	0.133524
tenure	0.294287	0.348974	0.339041	0.297290
PhoneService	-0.015198	0.024105	0.003727	-0.019158
MultipleLines	0.007141	0.117327	0.122318	0.011466
InternetService	-0.028416	0.036138	0.044944	-0.026047
OnlineSecurity	1.000000	0.185126	0.175985	0.285028
OnlineBackup	0.185126	1.000000	0.187757	0.195748
DeviceProtection	0.175985	0.187757	1.000000	0.240593
TechSupport	0.285028	0.195748	0.240593	1.000000
StreamingTV	0.044669	0.147186	0.276652	0.161305
StreamingMovies	0.055954	0.136722	0.288799	0.161316
Contract	0.374416	0.280980	0.350277	0.425367
PaperlessBilling	-0.157641	-0.013370	-0.038234	-0.113600
PaymentMethod	-0.096726	-0.124847	-0.135750	-0.104670
MonthlyCharges	-0.053878	0.119777	0.163652	-0.008682
TotalCharges	0.253224	0.374410	0.387897	0.275625
Churn	-0.289309	-0.195525	-0.178134	-0.282492

	StreamingTV	StreamingMovies	Contract	PaperlessBilling \
gender	-0.006421	-0.008743	0.000126	-0.011754
SeniorCitizen	0.030776	0.047266	-0.142554	0.156530
Partner	0.137341	0.129574	0.294806	-0.014877
Dependents	0.046885	0.021321	0.243187	-0.111377
tenure	0.268137	0.276032	0.626061	0.005004

```
PhoneService          0.055353          0.043870  0.002247          0.016505
MultipleLines         0.175059          0.180957  0.110842          0.165146
InternetService       0.107417          0.098350  0.099721         -0.138625
OnlineSecurity        0.044669          0.055954  0.374416         -0.157641
OnlineBackup          0.147186          0.136722  0.280980         -0.013370
DeviceProtection      0.276652          0.288799  0.350277         -0.038234
TechSupport           0.161305          0.161316  0.425367         -0.113600
StreamingTV           1.000000          0.434772  0.227116          0.096642
StreamingMovies       0.434772          1.000000  0.231226          0.083700
Contract              0.227116          0.231226  1.000000         -0.176733
PaperlessBilling      0.096642          0.083700 -0.176733          1.000000
PaymentMethod        -0.104234         -0.111241 -0.227543         -0.062904
MonthlyCharges        0.336706          0.335459 -0.074195          0.352150
TotalCharges          0.391470          0.398066  0.446855          0.158574
Churn                -0.036581         -0.038492 -0.396713          0.191825

                  PaymentMethod  MonthlyCharges  TotalCharges     Churn
gender                 0.017352       -0.014569     -0.000080 -0.008612
SeniorCitizen         -0.038551        0.220173      0.103006  0.150889
Partner               -0.154798        0.096848      0.317504 -0.150448
Dependents            -0.040292       -0.113890      0.062078 -0.164221
tenure                -0.343300        0.233516      0.789548 -0.318469
PhoneService          -0.004184        0.247398      0.113214  0.011942
MultipleLines         -0.176793        0.433576      0.452577  0.038037
InternetService        0.086140       -0.323260     -0.175755 -0.047291
OnlineSecurity        -0.096726       -0.053878      0.253224 -0.289309
OnlineBackup          -0.124847        0.119777      0.374410 -0.195525
DeviceProtection      -0.135750        0.163652      0.387897 -0.178134
TechSupport           -0.104670       -0.008682      0.275625 -0.282492
StreamingTV           -0.104234        0.336706      0.391470 -0.036581
StreamingMovies       -0.111241        0.335459      0.398066 -0.038492
Contract              -0.227543       -0.074195      0.446855 -0.396713
PaperlessBilling      -0.062904        0.352150      0.158574  0.191825
PaymentMethod          1.000000       -0.193407     -0.330918  0.107062
MonthlyCharges        -0.193407        1.000000      0.651174  0.193356
TotalCharges          -0.330918        0.651174      1.000000 -0.198324
Churn                  0.107062        0.193356     -0.198324  1.000000
```

[33]:

```
#plotting correlation into heatmap
plt.figure(figsize=(20,10))
sns.heatmap(data=df.corr(),cmap='viridis',annot=True);
```



Observations:Tenure is highly correlated with Contract and TotalCharges features. We have to delete this feature to avoid multicollinearity problem.

9 Separating target variable(Dependent) from Independent variables

[34]:
```
x=df.drop(['customerID','Churn','tenure'],axis=1)
y=df['Churn']
```

[35]:
```
# We have Imbalanced Data and we have to do sampling to avoid this problem
# we have two method so for 1] Under Sampling, 2] Oversampling
# We will go for Oversampling.

from imblearn.over_sampling import RandomOverSampler
ros =RandomOverSampler()
x_sample,y_sample=ros.fit_resample(x, y)
```

10 Train_Test_Split

[36]:
```
# Model Selection
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x_sample,y_sample,test_size=0.
↪2,random_state=101)
```

[37]:
```
x_train.head()
```

[37]:
```
      gender  SeniorCitizen  Partner  Dependents  PhoneService  MultipleLines  \
3773       0              0        0           0             0              1
6666       0              1        1           0             1              2
682        1              0        0           0             1              0
6333       0              1        0           0             1              2
4027       0              0        1           0             0              1

      InternetService  OnlineSecurity  OnlineBackup  DeviceProtection  \
3773                0               0             0                 0
6666                0               0             2                 2
682                 0               2             0                 0
6333                1               0             0                 2
4027                0               0             0                 0

      TechSupport  StreamingTV  StreamingMovies  Contract  PaperlessBilling  \
3773            0            0                0         0                 0
6666            2            2                2         0                 0
682             2            2                0         0                 1
6333            2            2                2         1                 1
4027            0            0                0         0                 1

      PaymentMethod  MonthlyCharges  TotalCharges
3773              3       -1.314870     -0.994662
6666              2        0.634418      0.013441
682               1       -0.012021     -0.919462
6333              2        1.370594      1.836056
4027              2       -1.339797     -0.792777
```

[38]:
```
#getting shape of training data
x_train.shape,y_train.shape
```

[38]:
```
((8278, 18), (8278,))
```

[39]:
```
#getting shape of testing data
x_test.shape,y_test.shape
```

[39]:
```
((2070, 18), (2070,))
```

11 Logistic Regression Model

[40]:
```
#fitting the model
from sklearn.linear_model import LogisticRegression
lr=LogisticRegression(max_iter=1000)
lr.fit(x_train,y_train)
```

[40]: LogisticRegression(max_iter=1000)

[41]:
```
#prediction
pred=lr.predict(x_test)
```

12 Perforance Metrics

[42]:
```
#performance metrics
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
```

[43]:
```
print('MSE:',mean_squared_error(y_test,pred))
print('MAE:',mean_absolute_error(y_test,pred))
```

```
MSE: 0.23478260869565218
MAE: 0.23478260869565218
```

13 Random Forest Model

[44]:
```
#fitting the model
from sklearn.ensemble import RandomForestClassifier
rf=RandomForestClassifier()
rf.fit(x_train,y_train)
```

[44]: RandomForestClassifier()

[45]:
```
#prediction
pred1=rf.predict(x_test)
```

14 Model Evaluation

[46]:
```
# Model Evaluation
from sklearn.metrics import classification_report, confusion_matrix
report1=classification_report(y_test,pred1)
matrix1=confusion_matrix(y_test,pred1)
```

[47]:
```
report1=classification_report(y_test,pred1)
matrix1=confusion_matrix(y_test,pred1)
```

[48]:
```
print('Classification Report:\n',report1)
print('Confusion Matrix :\n',matrix1)
```

```
Classification Report:
               precision    recall  f1-score   support
```

```
              0       0.94      0.83      0.88      1052
              1       0.85      0.94      0.89      1018

       accuracy                           0.89      2070
      macro avg       0.89      0.89      0.89      2070
   weighted avg       0.89      0.89      0.89      2070

Confusion Matrix :
 [[878 174]
 [ 60 958]]
```

15 Hyperpatameter Tuning

[49]:
```python
from sklearn.model_selection import RandomizedSearchCV
```

[50]:
```python
params={'n_estimators':[i for i in  range(100,2000,200)],
      'max_depth':[1,2,4,5,10,15,20,30,35,40],
       'min_samples_split':[1,2,4,5,10,15,20],
       'min_samples_leaf':[1,2,6,8,10,15,20,25,30]}

clf=RandomForestClassifier()

model=RandomizedSearchCV(clf,param_distributions=params,cv=3)

model.fit(x_train,y_train)
```

[50]:
```
RandomizedSearchCV(cv=3, estimator=RandomForestClassifier(),
                   param_distributions={'max_depth': [1, 2, 4, 5, 10, 15, 20,
                                                      30, 35, 40],
                                        'min_samples_leaf': [1, 2, 6, 8, 10, 15,
                                                             20, 25, 30],
                                        'min_samples_split': [1, 2, 4, 5, 10,
                                                              15, 20],
                                        'n_estimators': [100, 300, 500, 700,
                                                         900, 1100, 1300, 1500,
                                                         1700, 1900]})
```

[51]:
```python
model.best_score_
```

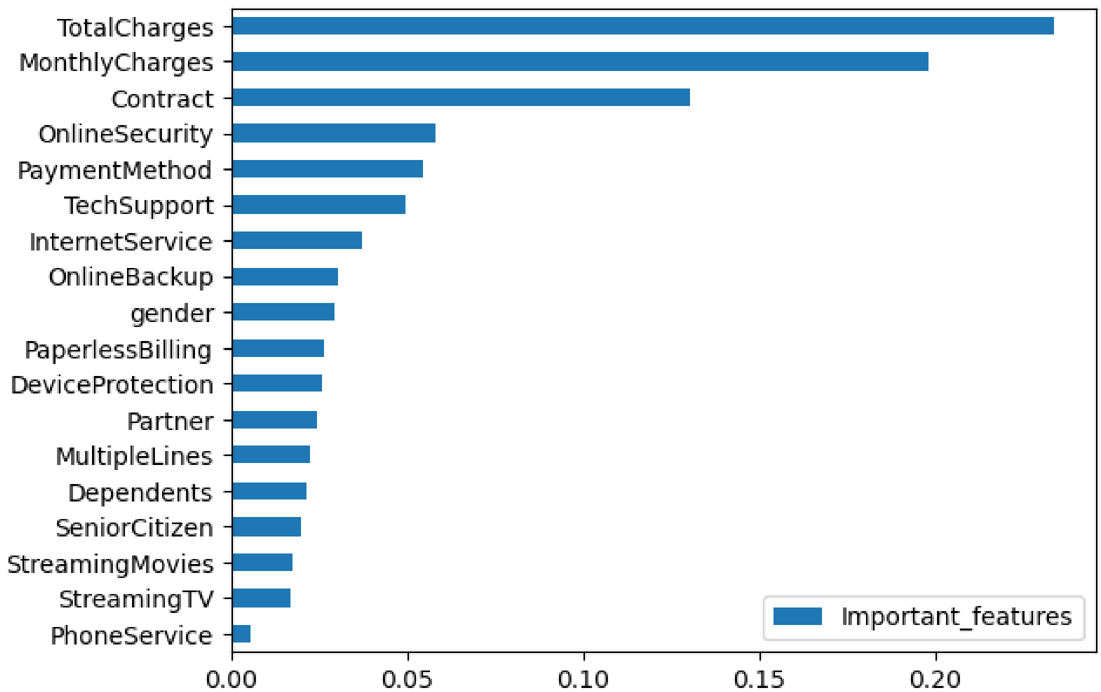
[51]:
```
0.796205960991958
```

Observations:We can see hyperparameter-tuning is not improving the accuracy, still we got the good accuracy.

We got almost 90% F1-Score without Hyperparameter-Tuning.

[52]:
```
features=pd.DataFrame({'Important_features':rf.feature_importances_},index=x.
↪columns)
```

[53]:
```
features.sort_values(by='Important_features',ascending=True).plot(kind='barh');
```



[54]:
```
import pickle
filename='churn_rfc.pkl'
pickle.dump(rf,open(filename,'wb'))
```

[55]:
```
x.head()
```

[55]:

	gender	SeniorCitizen	Partner	Dependents	PhoneService	MultipleLines	\
0	0	0	1	0	0	1	
1	1	0	0	0	1	0	
2	1	0	0	0	1	0	
3	1	0	0	0	0	1	
4	0	0	0	0	1	0	

	InternetService	OnlineSecurity	OnlineBackup	DeviceProtection	\
0	0	0	2	0	
1	0	2	0	2	
2	0	2	2	0	
3	0	2	0	2	
4	1	0	0	0	

```
   TechSupport  StreamingTV  StreamingMovies  Contract  PaperlessBilling  \
0            0            0                0         0                 1
1            0            0                0         1                 0
2            0            0                0         0                 1
3            2            0                0         1                 0
4            0            0                0         0                 1

   PaymentMethod  MonthlyCharges  TotalCharges
0              2       -1.160323     -0.992611
1              3       -0.259629     -0.172165
2              3       -0.362660     -0.958066
3              0       -0.746535     -0.193672
4              2        0.197365     -0.938874
```

[56]:
```
X=[[0,0,0,0,1,2,1,0,0,2,0,2,2,0,1,0,70.8,60.8]]

rf.predict(X)
```

[56]:
```
array([0])
```

[57]:
```
df.head()
```

[57]:
```
   customerID  gender  SeniorCitizen  Partner  Dependents  tenure  \
0  7590-VHVEG       0              0        1           0       1
1  5575-GNVDE       1              0        0           0       2
2  3668-QPYBK       1              0        0           0       1
3  7795-CFOCW       1              0        0           0       2
4  9237-HQITU       0              0        0           0       1

   PhoneService  MultipleLines  InternetService  OnlineSecurity  …  \
0             0              1                0               0  …
1             1              0                0               2  …
2             1              0                0               2  …
3             0              1                0               2  …
4             1              0                1               0  …

   DeviceProtection  TechSupport  StreamingTV  StreamingMovies  Contract  \
0                 0            0            0                0         0
1                 2            0            0                0         1
2                 0            0            0                0         0
3                 2            2            0                0         1
4                 0            0            0                0         0

   PaperlessBilling  PaymentMethod  MonthlyCharges  TotalCharges  Churn
0                 1              2       -1.160323     -0.992611      0
1                 0              3       -0.259629     -0.172165      0
```

```
2                1           3     -0.362660    -0.958066     1
3                0           0     -0.746535    -0.193672     0
4                1           2      0.197365    -0.938874     1

[5 rows x 21 columns]
```

[58]:

```
pred1
```

[58]:

```
array([0, 0, 0, …, 1, 1, 1])
```

[59]:

```
x_test.head()
```

[59]:

```
      gender  SeniorCitizen  Partner  Dependents  PhoneService  MultipleLines  \
800        0              1        1           0             1              2
4215       1              0        1           0             1              2
41         0              0        1           1             1              2
5461       0              0        0           0             1              0
9375       1              0        0           0             1              0

      InternetService  OnlineSecurity  OnlineBackup  DeviceProtection  \
800                 1               0             0                 0
4215                1               0             2                 2
41                  0               2             2                 0
5461                0               2             0                 0
9375                1               0             0                 0

      TechSupport  StreamingTV  StreamingMovies  Contract  PaperlessBilling  \
800             0            0                0         2                 1
4215            2            2                2         2                 1
41              0            2                0         2                 1
5461            2            0                2         2                 1
9375            0            2                0         0                 1

      PaymentMethod  MonthlyCharges  TotalCharges
800               1        0.361883      1.381637
4215              2        1.443713      2.453359
41                1        0.147511      1.143818
5461              3        0.011244      0.691341
9375              2        0.514768     -0.942007
```