Written by Juhi Mishra

Predicting the Customer churn in a Telecom industry

Now a days as we all know that for any company customer churn is a bog loss, as an existing customers are far less expensive than having new one. Not only this even through existing customers a company can have more referrals and can acquire new customers at a very low cost. While hiring new customer means to know customer taste, demand and expectation also it will be more expensive compare to have an existing one . So for any company it become more important to retain their existing customer by providing some additional benefits . Not only this a company should always focus on how well they know their customers their pros and cons so that they can understand their issues and can resolve the same on time. Here we are going to predict customer churn for telecom industry as this industry has the barrier to entry for switching services are so low. I had taken this customer data from IBM Sample Data Sets with the aim of building and comparing several customer churn prediction models.
—The telecommunications industry is made up of cable companies, internet service providers, satellite companies, and telephone companies. Telecommunications is defined as communicating over a distance. The industry's origin can be traced to postal courier services.

Importing the Important Libraries

```
# Importng important libraries
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder

import warnings
warnings.filterwarnings('ignore')

from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import
train_test_split,GridSearchCV
from sklearn.metrics import
accuracy_score,confusion_matrix,roc_curve,roc_auc_score
from sklearn.model_selection import cross_val_score
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.metrics import classification_report
from sklearn.metrics import plot_roc_curve
```

Get the Dataset(CSV form)

```
df = pd.read_csv("/Users/juhimishra/Downloads/
Telecom_customer_churn.csv")
df.head()
```

This dataset is a classification based problem as our target consist of binary information which is either in yes/no so we need to build our model accordingly. Also need to check whether our data is imbalanced dataset or not as our target is also a categorical column and mostly in categorical column we should check whether there dataset is a balanced or imbalanced dataset.

For all columns visualisation will use pd.set_option as

```
pd. set_option("display.max_columns", None)
```

EDA(Explanatory Data Analysis)

```
print('No of rows:',df.shape[0])
print('no of columns:',df.shape[1])
```

```
No of rows: 7043
no of columns: 21
```

This data consist of total no of rows as 7043 and total columns are 21.

```
# Finding null values, object and int/float in dataset
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
```

This data has 20 features and 1 target variable that is Churn which is a categorical column. Also we came across that this is a classification problem as our target consist binary information which is either yes or no. From this info method we analysed that there are only two columns which are in integer form and 1 column which is in float form, rest all are having string/object Dtype. This we need to convert into integer or float form so that our machine can understand the input for further process.

Below Listed the detail of column with short descriptions :

Feature Variable :

1. Customer Id = Id given to customer by company for unique identification. Name can be same but customer id used by any company to have unique identification for each client.
2. Gender = Male and Female
3. Senior Citizen = No of Senior citizen customers company is having
4. Partner = Whether any customer is in partnership or not
5. Dependents = Whether any customer having dependents or not
6. Tenure = For how long customers are having contract with company

7. Phone Service = This comes under additional benefit . Customer getting phone service by company
8. MultipleLines = Customers getting single connection or multiple lines by company. But this is applicable only to those who are having phone service.
9. Internet Service = Whether customer getting internet service facility or not
10. Online Security = whether online security given to customer or not. This facility is related to those who are getting internet service.
11.Online Backup = Online backup provided to customer or not. Again we can see here that Internet service, online security and backup are interrelated to each other. That means the customer
Who are getting internet service will have the chance to get these facilities too but not mandatory.
12. Device Protection = Whether company is providing device protection customers to not. This also comes under additional benefit.
13. Tech Support = Tech support given to customer or not
14. Streaming TV = Streaming TV facility provided to customer
15. Streaming Movies = facility provided to customer
16. Contract =. For what period the contract has been decided between company and client. Within the contract period the client can't move to other company or can find any other opportunity as per the company law either it will b considered as breach of contract.

17. Paperless billing = Whether the company is having this facility or not. Now a days most of the company do have paperless billing so that they can reduce physical document as well as it's easy to keep record for future.

18. Payment Method = What payment method (mode of payment) do company have.
19. Monthly Charges = Customer paying on a monthly basis to the company as decided during entering to contract.
20. Total charges = Total charges that customer need to pay to the company as per their contract.

21. Churn = Whether the customer is moving or not from the company

To check the static data of customer churn used describe method. With the help of describe method we can see all continuous data static details along with missing values, mean, std and Quantile detail.

```
# Checking static data using describe method
df.describe()
```

	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

As describe data shows only continous / integer form column so we can see here only three columns and rest all are in object form

From above static data following observations recorded:

1. No null values in any of the column
2. 64% are paying monthly charges and for tenure we can say that 32% of the customers are for longer tenure in a company which is very less.
3. As we can see that senior citizen column are having two values i.e., 0 and 1 so we can consider it as categorical column.

As above with the help of describe method we can only find the detail of continuous variables not categorical variable so will use isna method . Using isna method can find the missing values even if any in categorical columns.

```
df.isna().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
```

No null values even in categorical column too

Let's have analysis using visualisation technique:

Univariate analysis

First will compare gender means no. of male and female customer do company have.

Checked unique value for gender

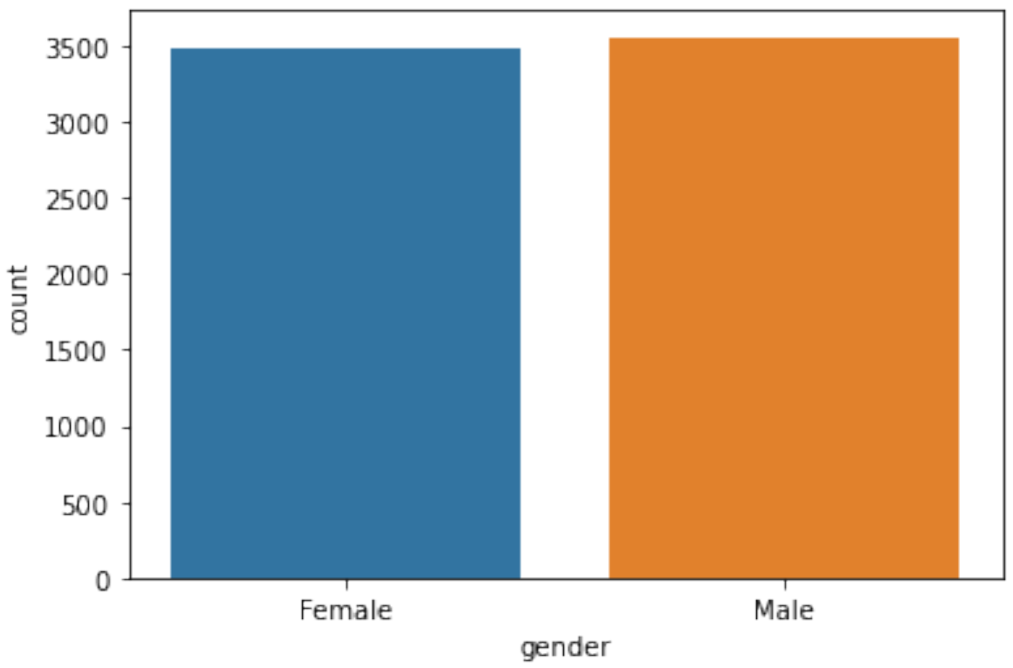
```
df['gender'].unique()
```

```
df['gender'].value_counts()
Male      3555
Female    3488
Name: gender, dtype: int64
```

Same thing shown using count plot

```
sns.countplot('gender',data=df )
```

```
AxesSubplot:xlabel='gender', ylabel='count'>
```



```
gender_per = ((3555-3488)/3555)*100
gender_per
1.88%
```

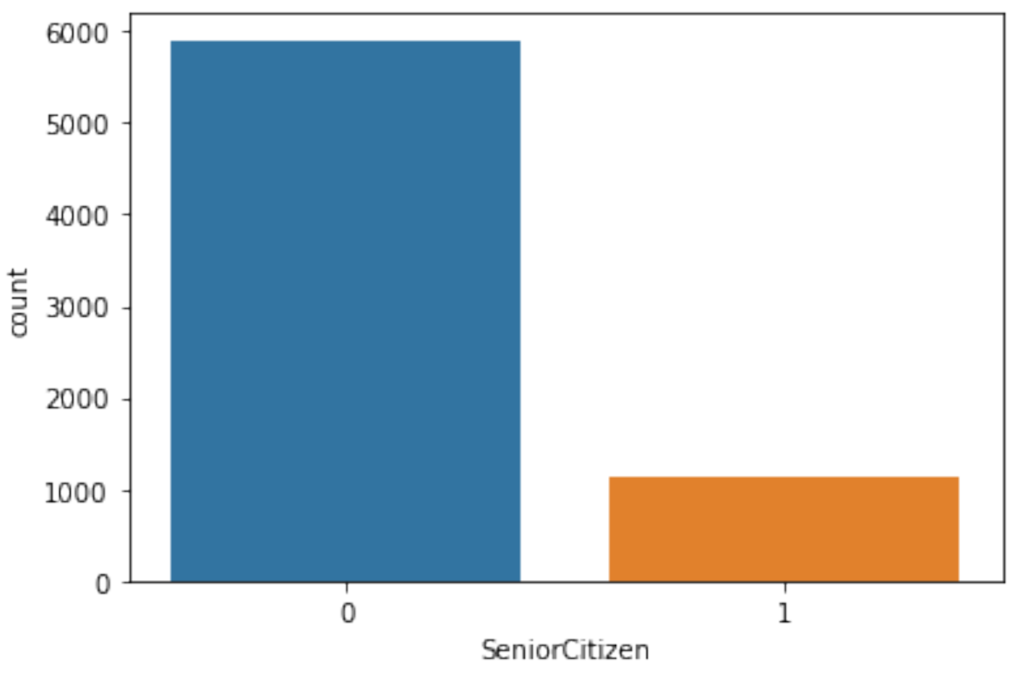
Compare to male female are approx 1.9% less in industry so not much gap

```
df['SeniorCitizen'].unique()
```

```
array([0, 1])
```

```
sns.countplot('SeniorCitizen',data=df)
```

```
<AxesSubplot:xlabel='SeniorCitizen', ylabel='count'>
```

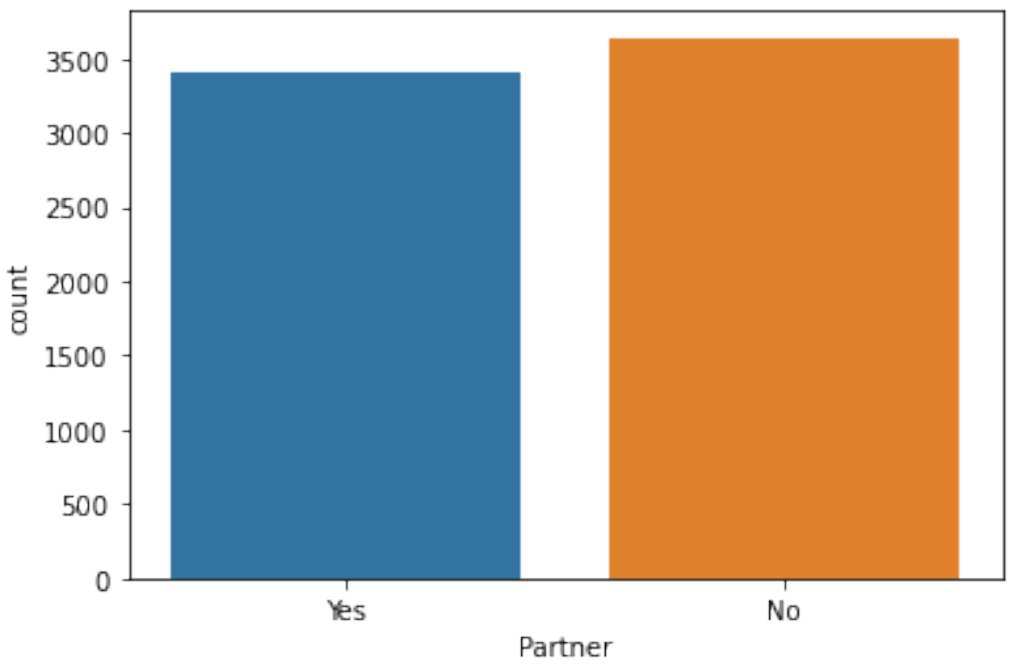


No. of seniourcitizen customers are less as we can see with the help of countplot here 0 stand for non senior citizen and 1 stands for senior citizen

```
df['Partner'].unique()
array(['Yes', 'No'], dtype=object)
```

```
sns.countplot('Partner',data=df)
```

```
<AxesSubplot:xlabel='Partner', ylabel='count'>
```



```
df['Partner'].value_counts()
No      3641
Yes     3402
Name: Partner, dtype: int64
```

```
partner_per = ((3641-3402)/3641)*100
partner_per
```

```
6.56%
```

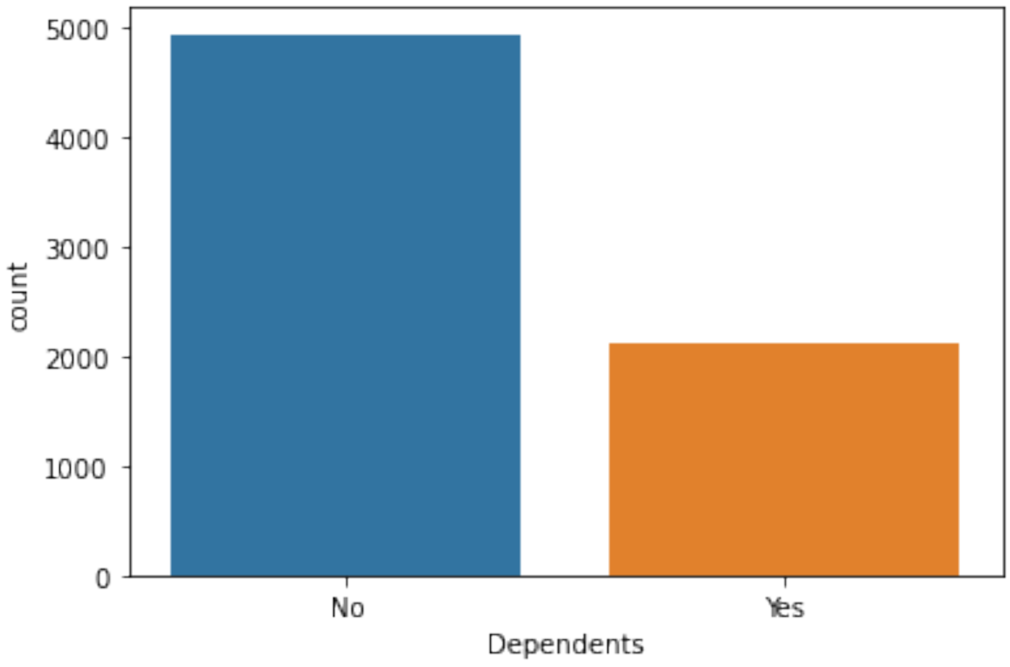
As we can see above got only 6.56% of customers who are not in partnership that means most of the customers are doing along with partner. With this we can say that concept of partnership in telecom industry are higher.

```
df['Dependents'].unique()
```

```
array(['No', 'Yes']
```

```
sns.countplot('Dependents',data=df)
```

```
<AxesSubplot:xlabel='Dependents', ylabel='count'>
```



```
df['Dependents'].value_counts()
```

```
No      4933
Yes     2110
Name: Dependents, dtype: int64
```

almost more than half of the customers are not having dependents.

```
df['tenure'].unique()
```

```
array([ 1, 34,  2, 45,  8, 22, 10, 28, 62, 13, 16,
58, 49, 25, 69, 52, 71,
       21, 12, 30, 47, 72, 17, 27,  5, 46, 11, 70,
63, 43, 15, 60, 18, 66,
        9,  3, 31, 50, 64, 56,  7, 42, 35, 48, 29,
65, 38, 68, 32, 55, 37,
       36, 41,  6,  4, 33, 67, 23, 57, 61, 14, 20,
53, 40, 59, 24, 44, 19,
       54, 51, 26,  0, 39])
```

```
Here min tenure we can see is for 1 month and maximum
tenure are 72 months i.e., 6 years which is really a
good time period for any customer and company to
built a good reputation and relation with each other.
Also from this we can consider that the customers who
are for such a longer period will have minimum
chances to switch to some other company.
```

One of the useful analysis we can say as we observed here, that customer with longer tenure stick more to the company that means company should focus to convince their customer to have longer period contract either by building good terms with them or by giving some additional benefits.

So below will observe how many customers do enjoy additional benefits and what are the impacts do company have.

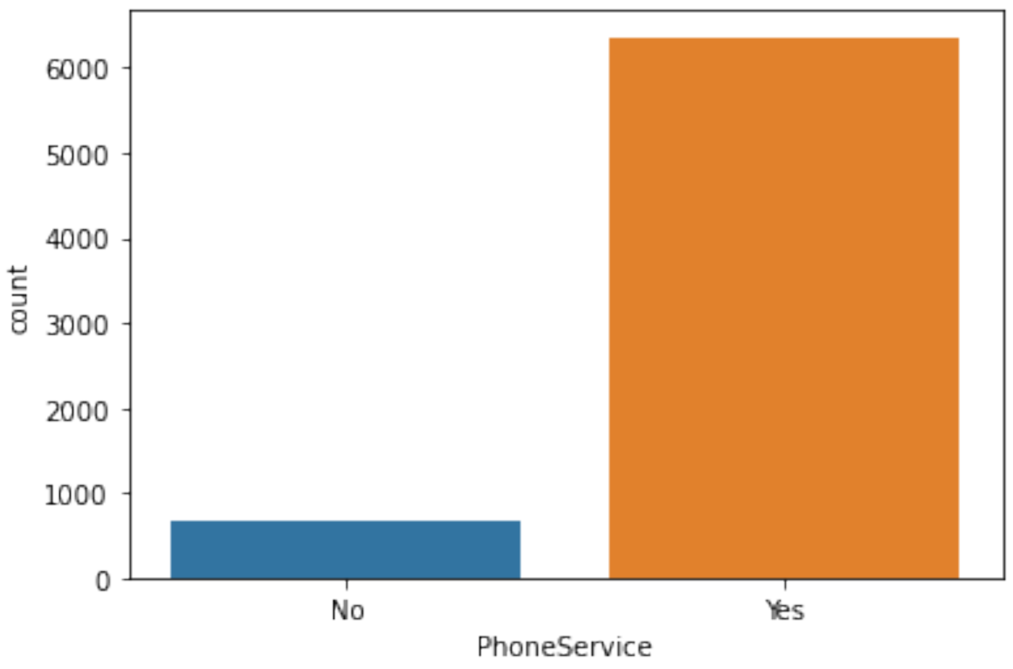
Univariate analysis

```
df['PhoneService'].unique()
```

```
array(['No', 'Yes']
```

```
sns.countplot('PhoneService',data=df)
```

```
<AxesSubplot:xlabel='PhoneService', ylabel='count'>
```



as we can see that most of the customers are getting phone services.

```
df['PhoneService'].value_counts()
```

```
Yes     6361
No       682
```

```
ph_per = ((6361-682)/6361)*100
ph_per
```

```
89.27%
```

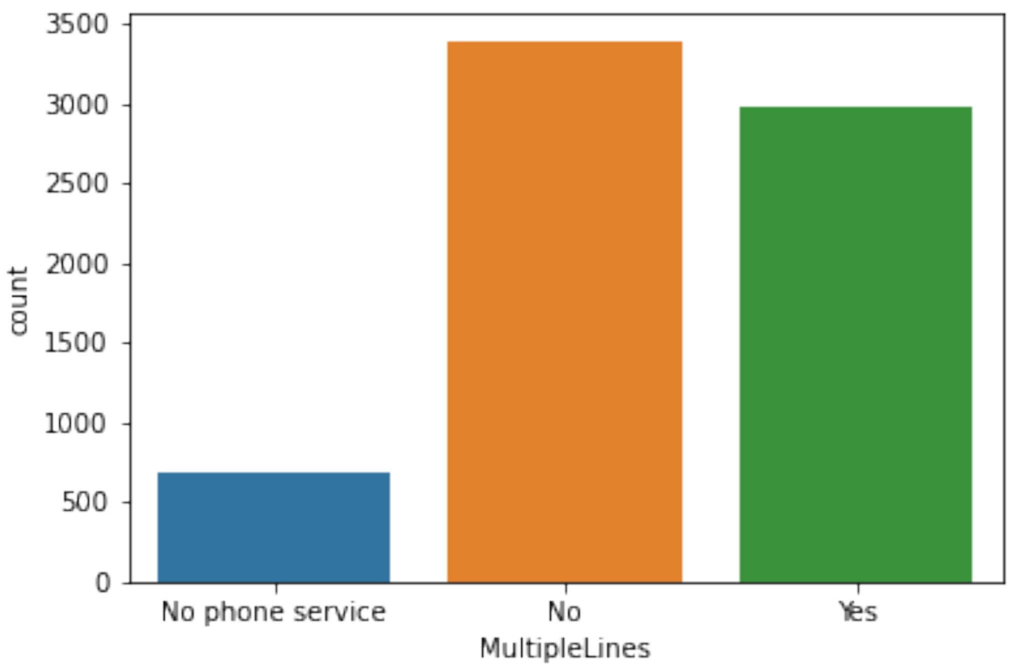
Only 11% of the customers are not getting phone services from telecom industry

```
df['MultipleLines'].unique()
```

```
array(['No phone service', 'No', 'Yes']
As we can see that there is one option given no phone
service that means phone service and multiple lines
are interrelated to each other . If phone service
provided to any customer will have maximum chances
that multiplelines service will also be given to
them. Let’s see using count plot
```

```
sns.countplot('MultipleLines',data=df)
```

```
<AxesSubplot:xlabel='MultipleLines', ylabel='count'>
```



```
df['MultipleLines'].value_counts()
```

```
No                    3390
Yes                   2971
No phone service       682
```

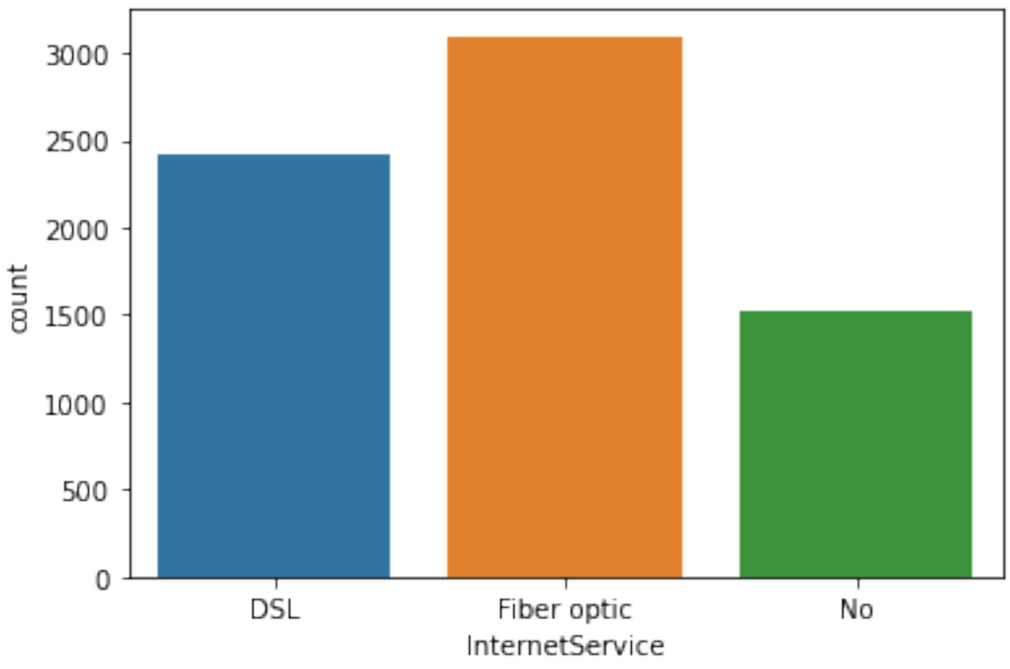
Multiple lines are given only to few customers almost only half of the customers are enjoying this service. Above we seen that almost 90% of the customers were getting phone service but multiple lines service are not provided to even 50% of the customers.

```
df['InternetService'].unique()
```

```
array(['DSL', 'Fiber optic', 'No']
sns.countplot('InternetService',data=df)
```

Out[33]:

```
<AxesSubplot:xlabel='InternetService',
ylabel='count'>
```



```
df['InternetService'].value_counts()
```

```
Fiber optic     3096
DSL             2421
No              1526
```

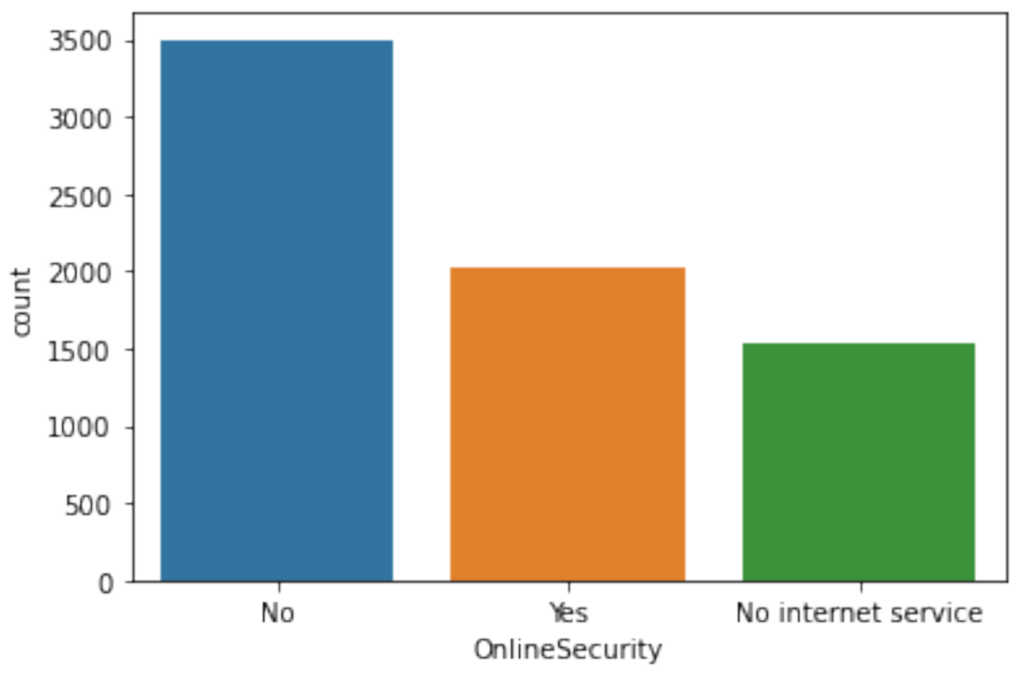
Most of the customers are getting Fiber optic internet service but there are 1526 customers who doesn't get any service from the industry

```
df['OnlineSecurity'].unique()
```

```
array(['No', 'Yes', 'No internet service']
```

```
sns.countplot('OnlineSecurity',data=df)
```

```
<AxesSubplot:xlabel='OnlineSecurity', ylabel='count'>
```



```
df['OnlineSecurity'].value_counts()
```

```
No                     3498
Yes                    2019
No internet service    1526
```

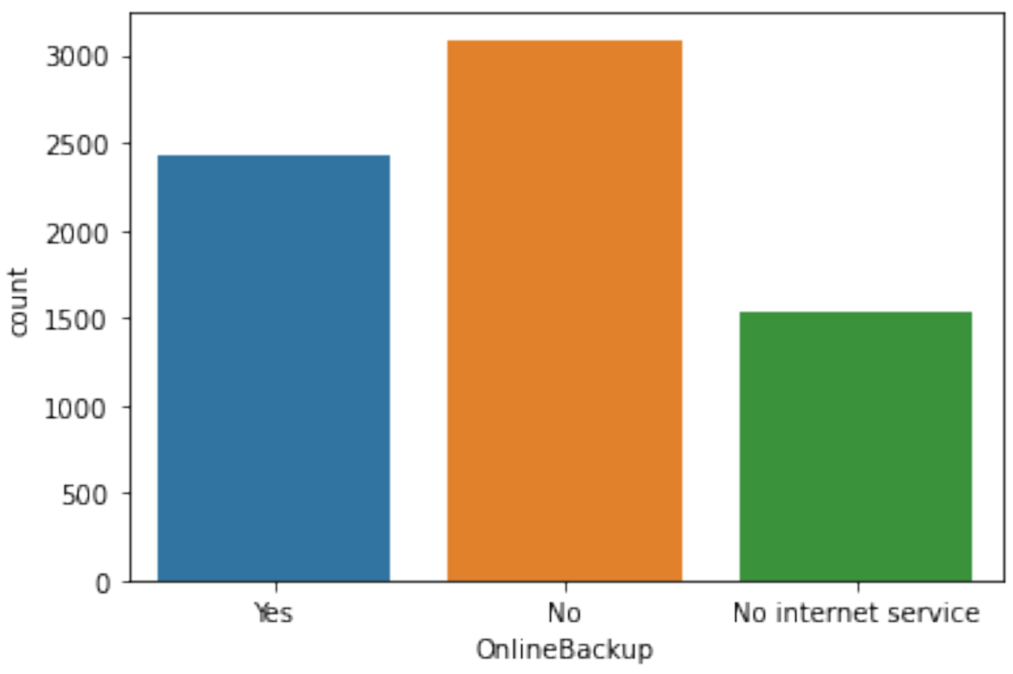
Here also we can see that only 2019 customers are getting online securities and above we already got the data that 1526 customers are not getting internet services so online service will definately be not there for them

```
df['OnlineBackup'].unique()
```

```
array(['Yes', 'No', 'No internet service']
```

```
sns.countplot('OnlineBackup',data=df)
```

```
<AxesSubplot:xlabel='OnlineBackup', ylabel='count'>
```



```
df['OnlineBackup'].value_counts()
```

```
No                     3088
Yes                    2429
No internet service    1526
```

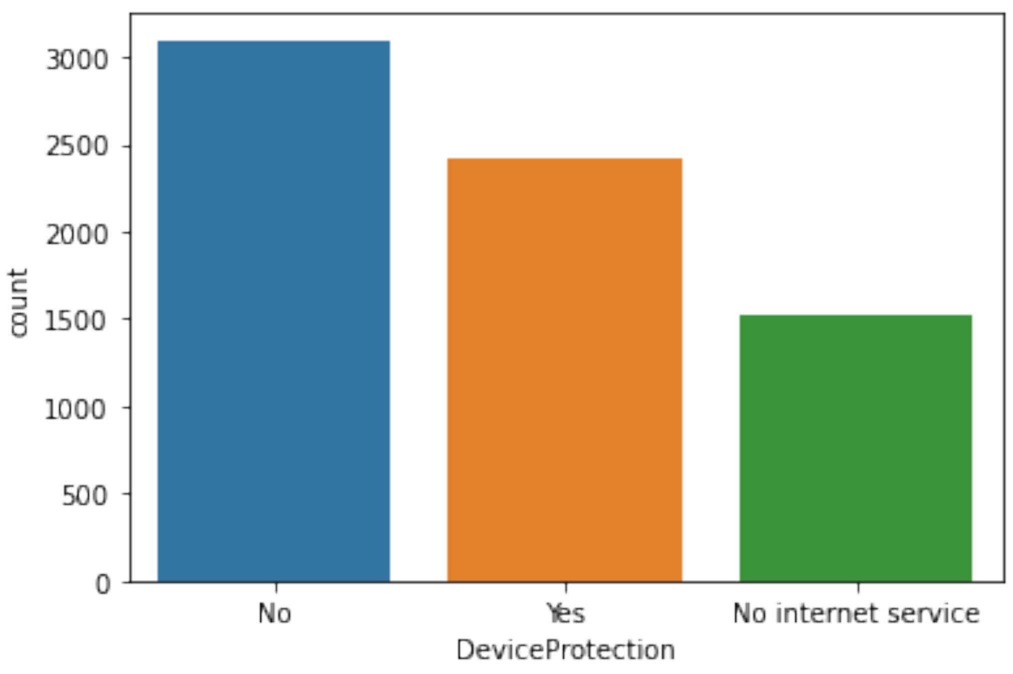
Even online backup are also given to few of the customers

```
df['DeviceProtection'].unique()
```

```
array(['No', 'Yes', 'No internet service']
```

```
sns.countplot('DeviceProtection',data=df)
```

```
<AxesSubplot:xlabel='DeviceProtection',
ylabel='count'>
```



```
df['DeviceProtection'].value_counts()
```

```
No                     3095
Yes                    2422
No internet service    1526
```

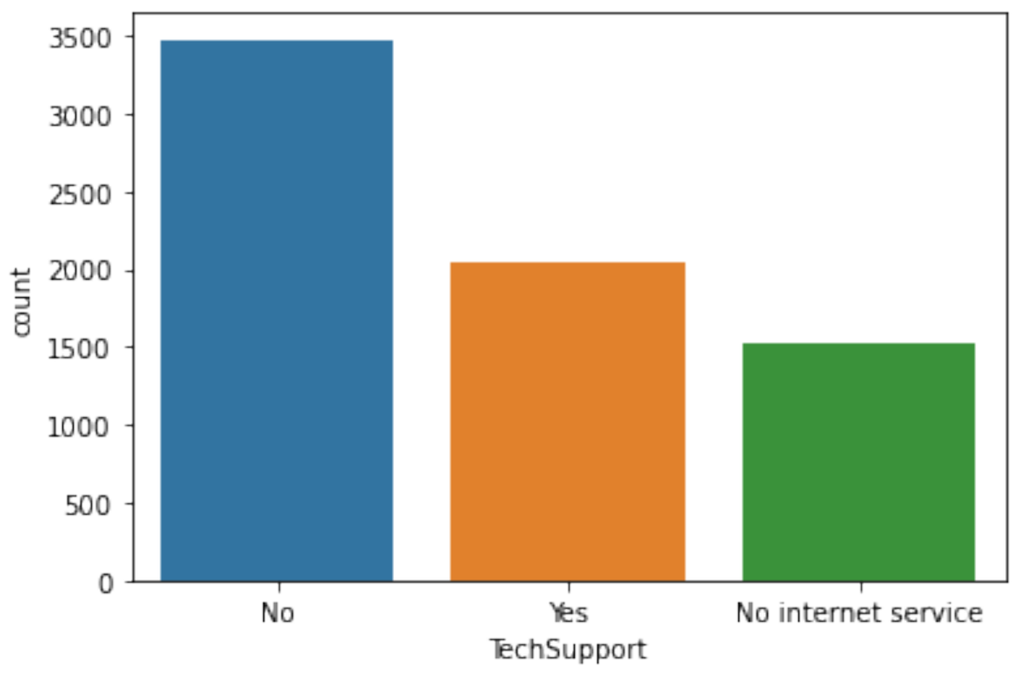
Every online services we can see from above are interrelated to each other as here also we can see that only half of the customers are getting device protection.

```
df['TechSupport'].unique()
```

```
array(['No', 'Yes', 'No internet service']
```

```
sns.countplot('TechSupport',data=df)
```

```
AxesSubplot:xlabel='TechSupport', ylabel='count'>
```



```
df['TechSupport'].value_counts()
```

```
No                     3473
Yes                    2044
No internet service    1526
```

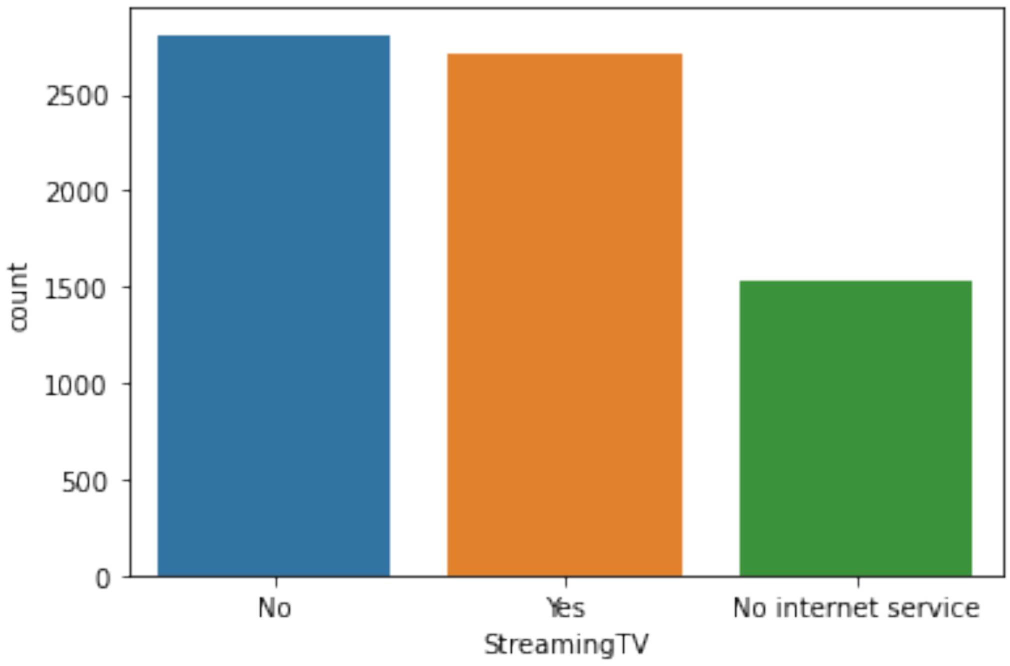
From here we can analyse that from internet services to tech support are given to half of the customers by telecom industry. Which may be a reason for churn for many of the customers.

```
df['StreamingTV'].unique()
```

```
array(['No', 'Yes', 'No internet service']
```

```
sns.countplot('StreamingTV',data=df)
```

```
<AxesSubplot:xlabel='StreamingTV', ylabel='count'>
```



```
df['StreamingTV'].value_counts()
```

```
No                     2810
Yes                    2707
No internet service    1526
```

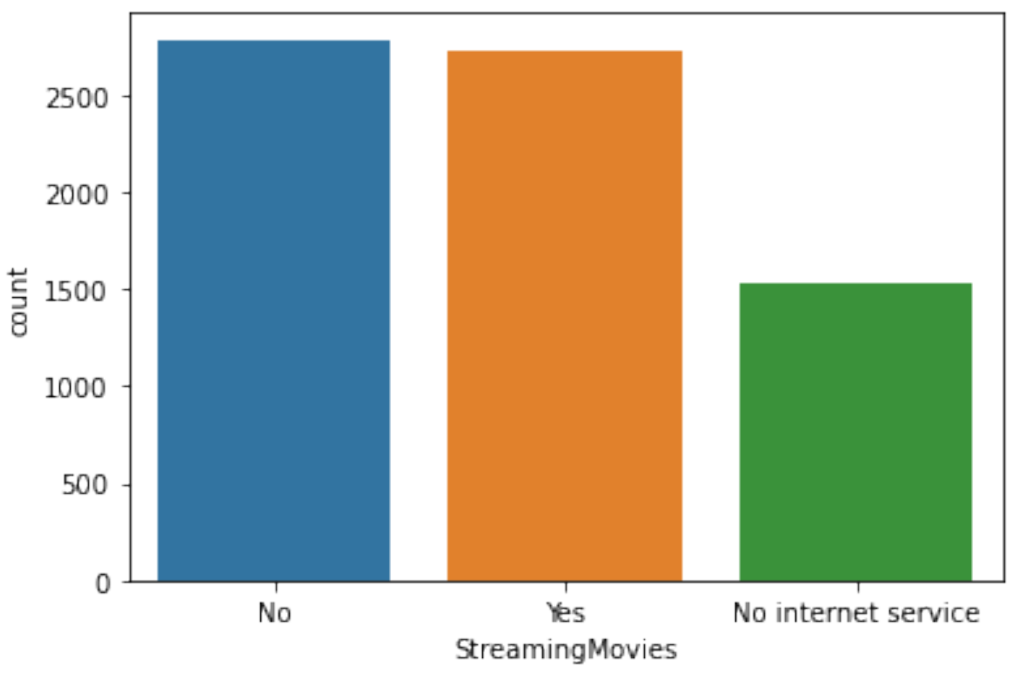
Here also we analyze the same thing that half of the customers are enjyoing this service

```
df['StreamingMovies'].unique()
```

```
array(['No', 'Yes', 'No internet service']
```

```
sns.countplot('StreamingMovies',data=df)
```

```
<AxesSubplot:xlabel='StreamingMovies',
ylabel='count'>
```



```
df['StreamingMovies'].value_counts()
```

```
No                     2785
Yes                    2732
No internet service    1526
```

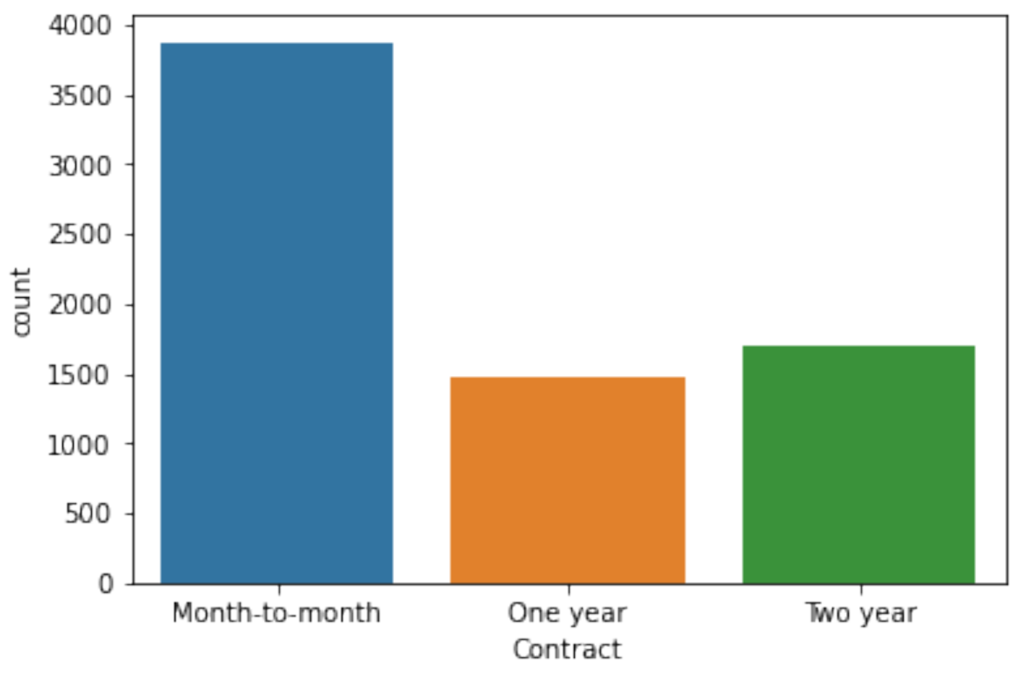
same analysis as above .Industry are providing almost all the facilities mostly to same customers.

```
df['Contract'].unique()
```

```
array(['Month-to-month', 'One year', 'Two year']
```

```
sns.countplot('Contract',data=df)
```

```
<AxesSubplot:xlabel='Contract', ylabel='count'>
```



```
df['Contract'].value_counts()
```

```
Month-to-month    3875
Two year          1695
One year          1473
```

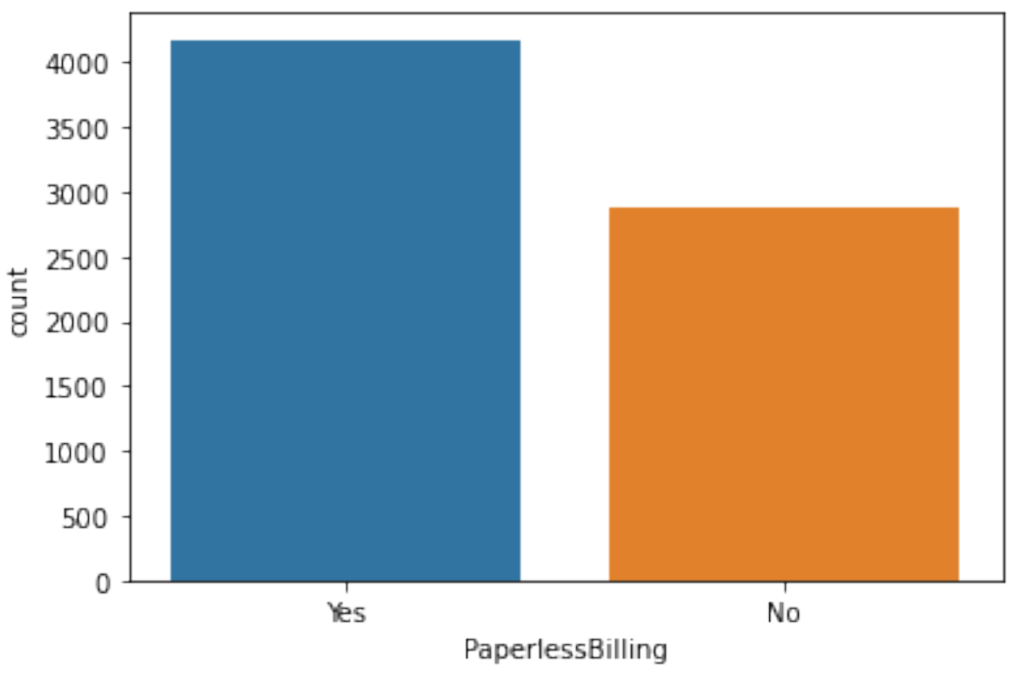
Here we can see that monthly contract are higher than longer one that means most of the customers are doing contract for short period only and above we seen that one of the reason for churn was short term contract.

```
df['PaperlessBilling'].unique()
```

```
array(['Yes', 'No']
```

```
sns.countplot('PaperlessBilling',data=df)
```

```
<AxesSubplot:xlabel='PaperlessBilling',
ylabel='count'>
```



```
df['PaperlessBilling'].value_counts()
```

```
Yes    4171
No     2872
```

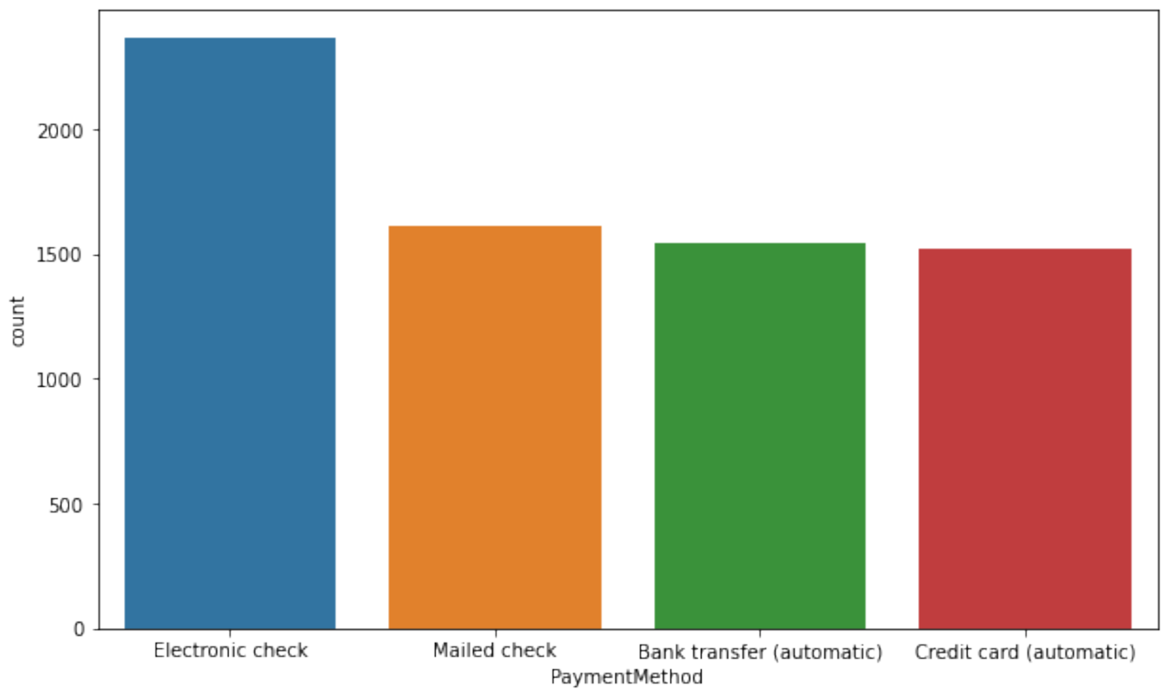
For most of the customers paperless billing protocol being followed. This one will not only help company to save some costing but also easy and safe to keep records for future reference . Might be possible that paper may get lost while having records on system will be more safer.

```
df['PaymentMethod'].unique()
```

```
array(['Electronic check', 'Mailed check', 'Bank
transfer (automatic)',
       'Credit card (automatic)']
```

```
plt.figure(figsize=(10,6))
sns.countplot('PaymentMethod',data=df)
```

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



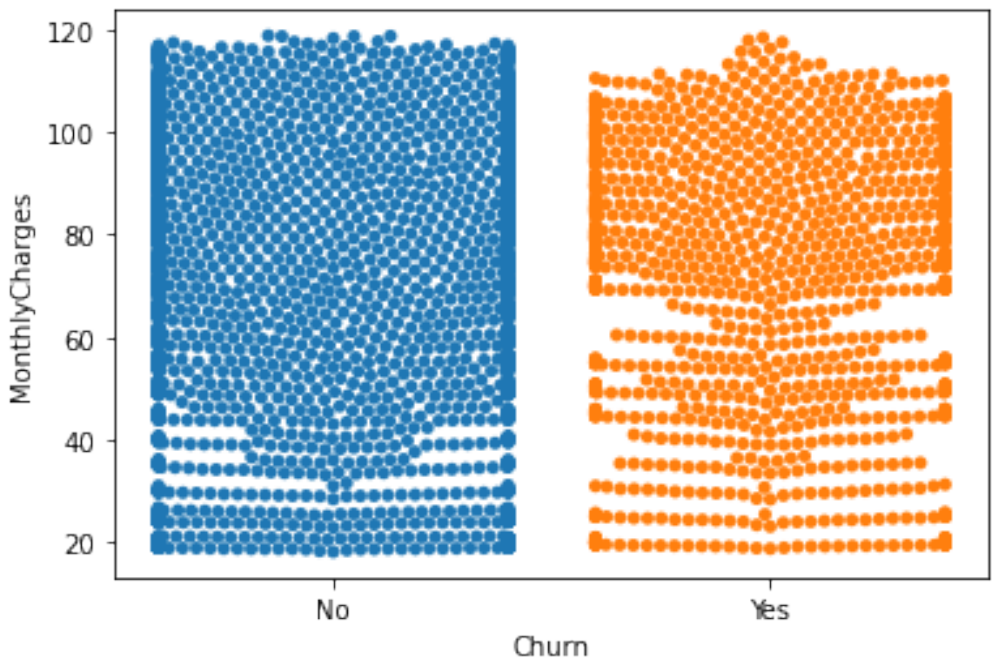
```
df['PaymentMethod'].value_counts()
```

```
Electronic check             2365
Mailed check                 1612
Bank transfer (automatic)    1544
Credit card (automatic)      1522
```

Maximum customers are following electronic check method for payment. While credit card and bank transfer mode of payment having almost same no of customers and least used by them.

```
sns.swarmplot(x = 'Churn',y='MonthlyCharges',data=df)
```

```
<AxesSubplot:xlabel='Churn', ylabel='MonthlyCharges'>
```



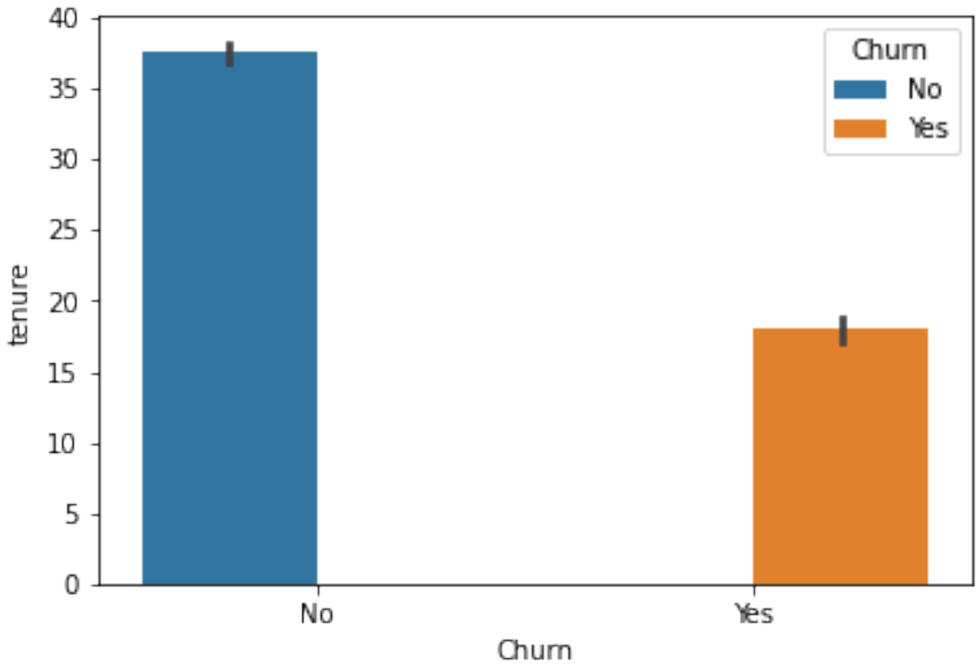
Almost industry are charging monthly charges for half of their customers as we can see above that both yes and no are are almost on the same label.

Bivariate Analysis

Comparing churn vs tenure using churn as hue to have better clarity

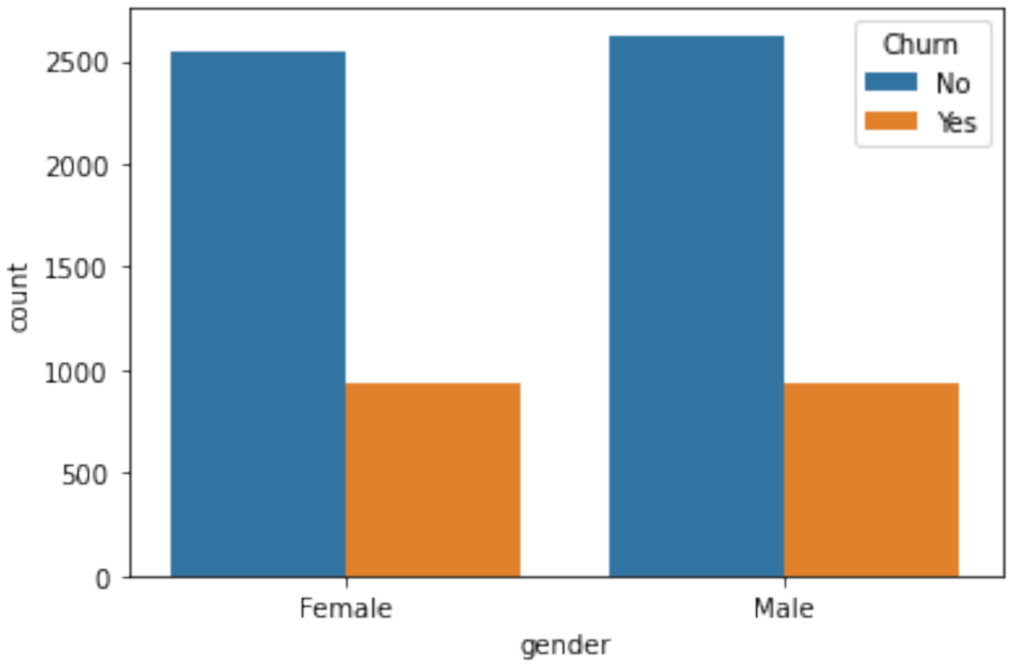
```
sns.barplot(x='Churn',y = 'tenure',hue =
'Churn',data=df)
```

```
<AxesSubplot:xlabel='Churn', ylabel='tenure'>
```



```
sns.countplot(x='gender',hue='Churn',data=df)
```

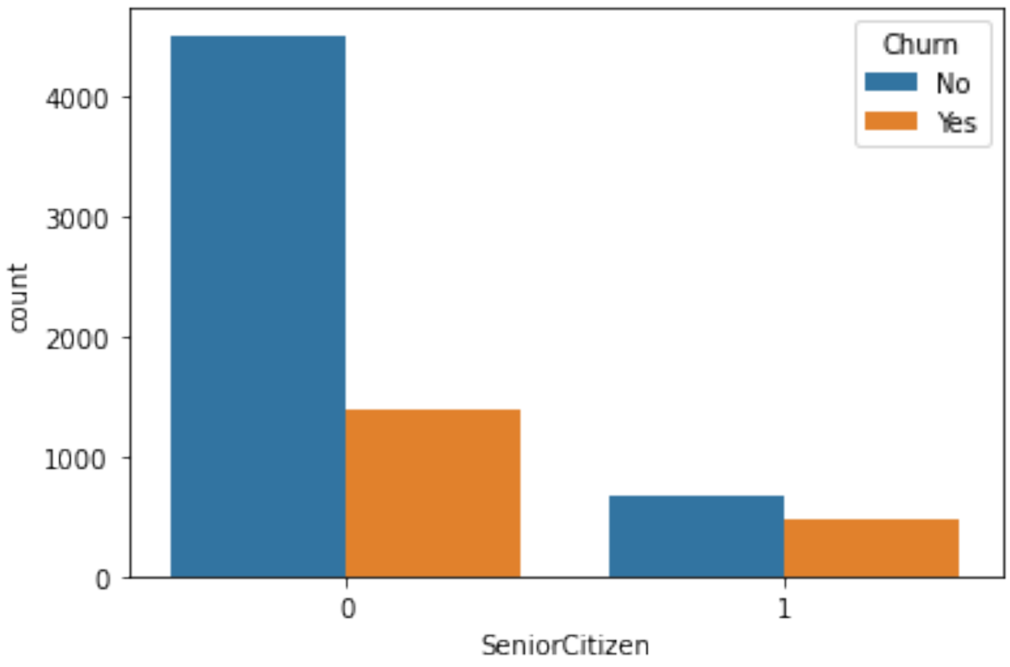
```
<AxesSubplot:xlabel='gender', ylabel='count'>
```



As we can see churn for all the gender are equal either for male or female. Really doesn't matter whether customer is male or female.

```
sns.countplot(x='SeniorCitizen',hue='Churn',data=df)
```

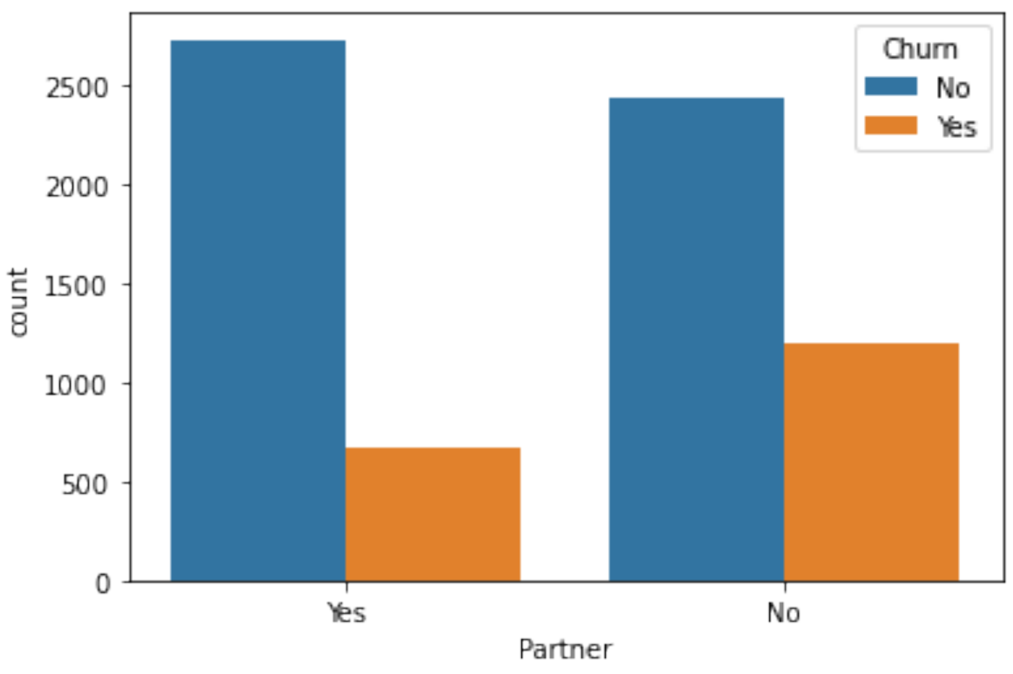
```
<AxesSubplot:xlabel='SeniorCitizen', ylabel='count'>
```



no of senior citizen are less churning compare to non senior citizen. That means youngsters are looking for either more profit or more opportunities in terms of either additional services or some other services such as may be not satisfied with company terms and condition or services being provided to them.

```
sns.countplot(x='Partner',hue = 'Churn',data=df)
```

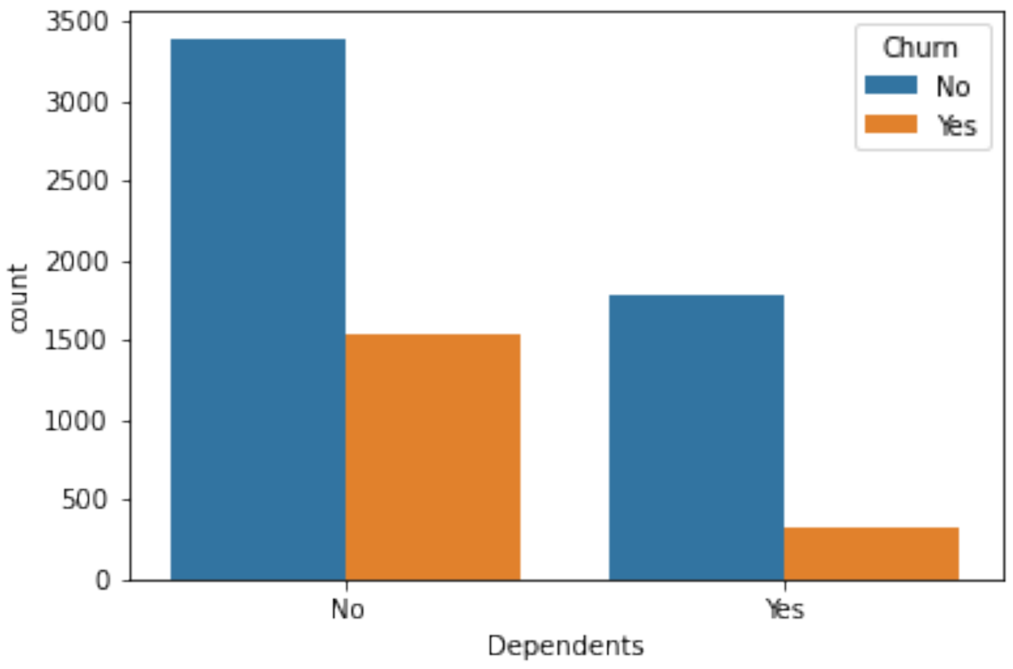
```
<AxesSubplot:xlabel='Partner', ylabel='count'>
```



Those who are not in partnership are shifting more that means we can consider by mutual consent they are moving to some other company.

```
sns.countplot(x='Dependents',hue='Churn',data=df)
```

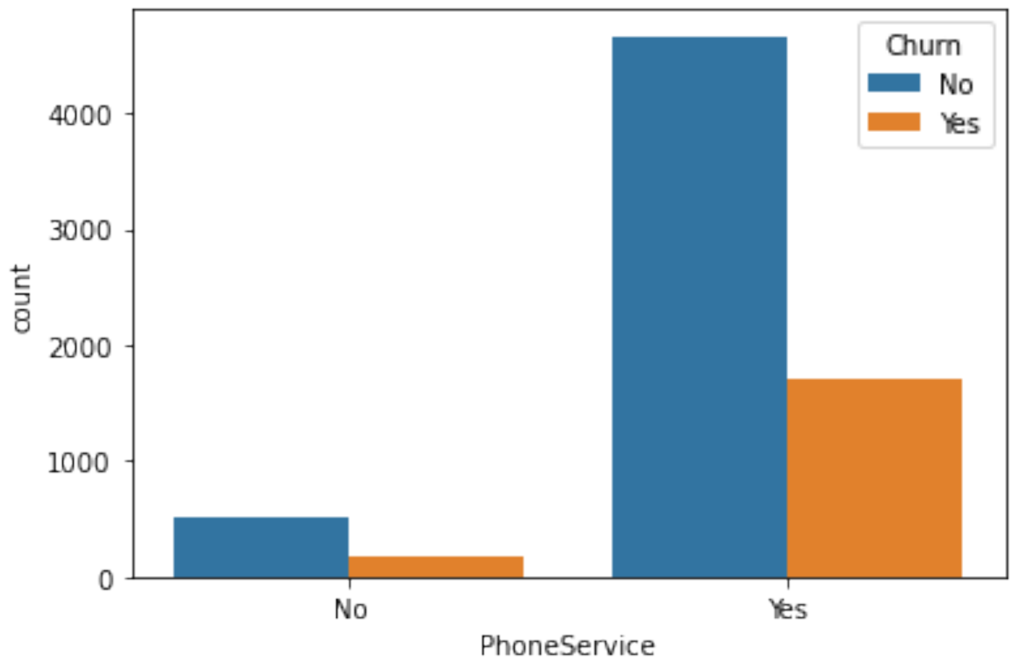
```
<AxesSubplot:xlabel='Dependents', ylabel='count'>
```



Having dependents are not moving to other option compare to those who are independent.

```
sns.countplot(x='PhoneService',hue='Churn',data=df)
```

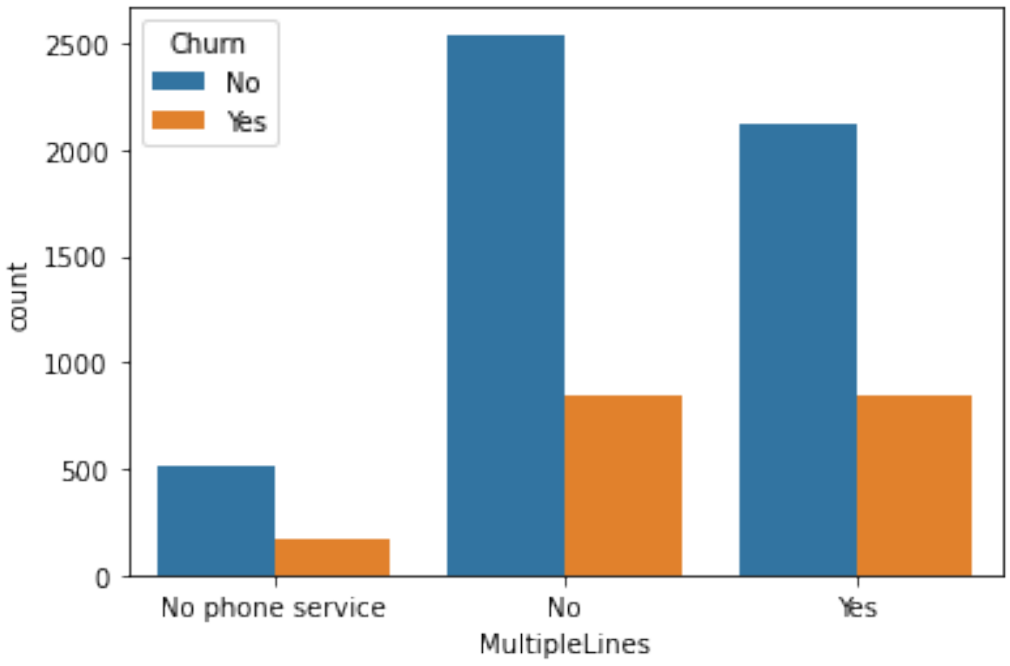
```
<AxesSubplot:xlabel='PhoneService', ylabel='count'>
```



providing ph services to customers are profitable for industry as we can see they are moving less

```
sns.countplot(x='MultipleLines',hue='Churn',data=df)
```

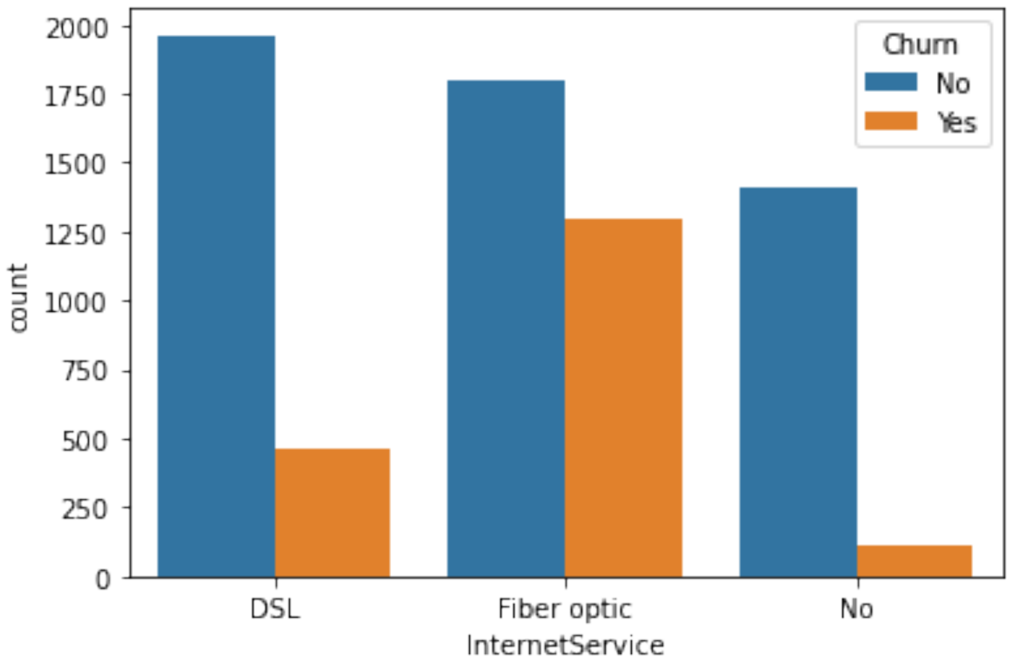
```
<AxesSubplot:xlabel='MultipleLines', ylabel='count'>
```



From above we can see that providing additional services can attract customers and they stick to the particular client.

```
sns.countplot(x='InternetService',hue='Churn',data=df
)
```

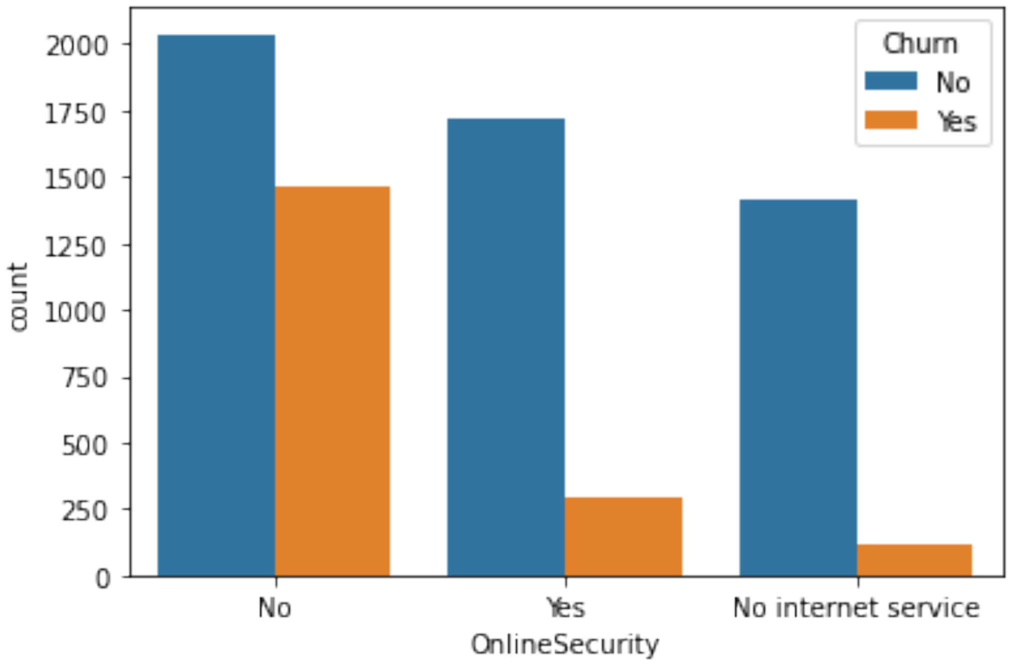
```
<AxesSubplot:xlabel='InternetService',
ylabel='count'>
```



DSL services provided more compare to others and as we observed above the same we can see here too

```
sns.countplot(x='OnlineSecurity',hue='Churn',data=df)
```

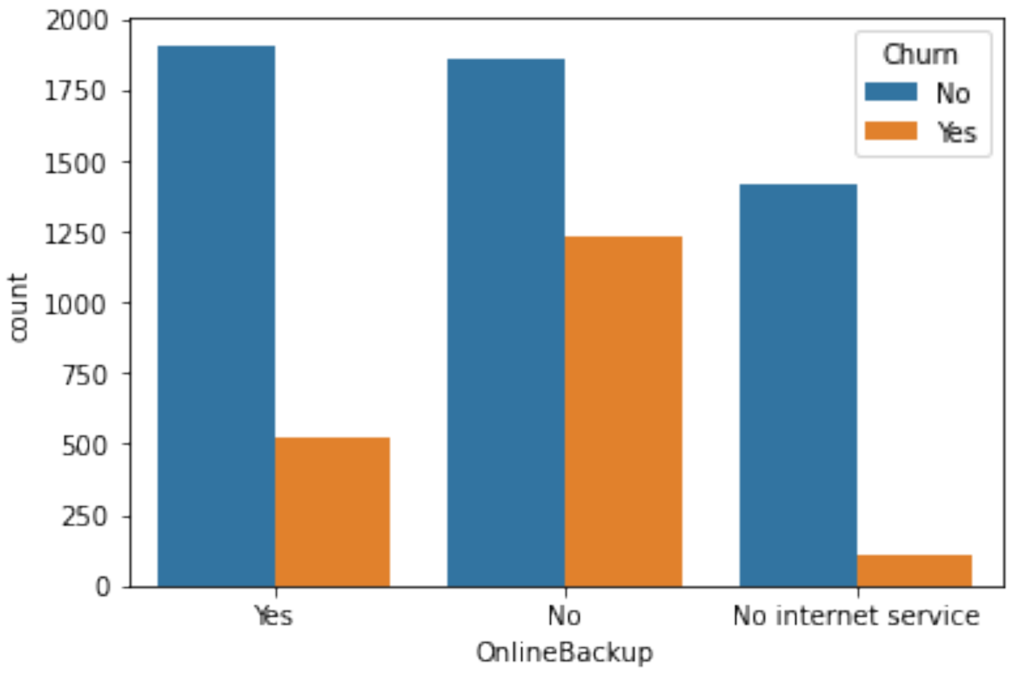
```
<AxesSubplot:xlabel='OnlineSecurity', ylabel='count'>
```



Same observation providing additional services can retain customers to stick to one of the company

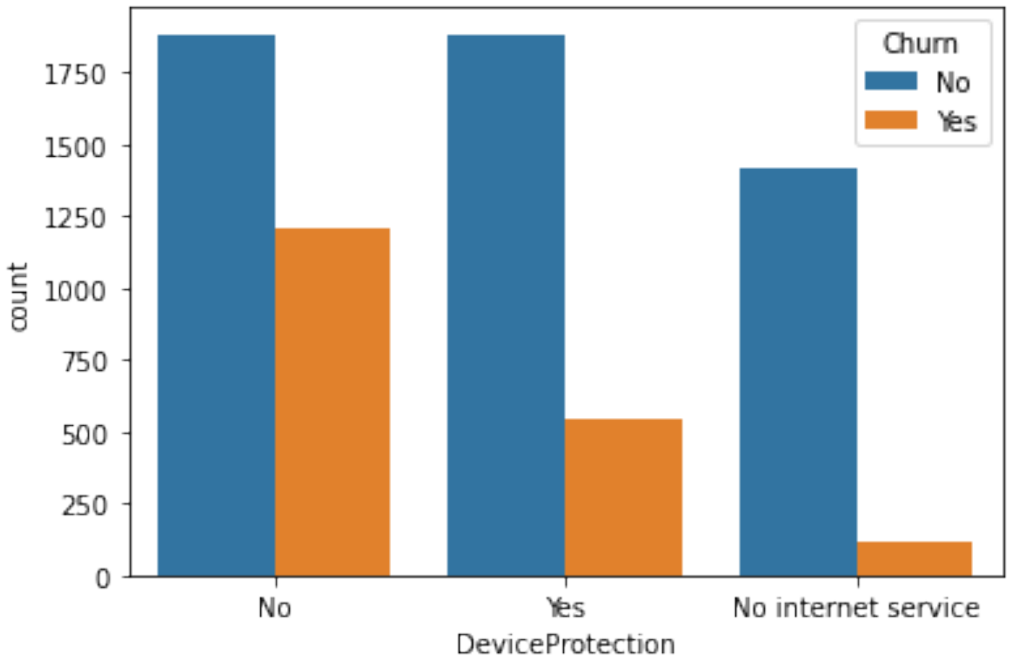
```
sns.countplot(x='OnlineBackup',hue='Churn',data=df)
```

```
<AxesSubplot:xlabel='OnlineBackup', ylabel='count'>
```



```
sns.countplot(x='DeviceProtection',hue='Churn',data=d
f)
```

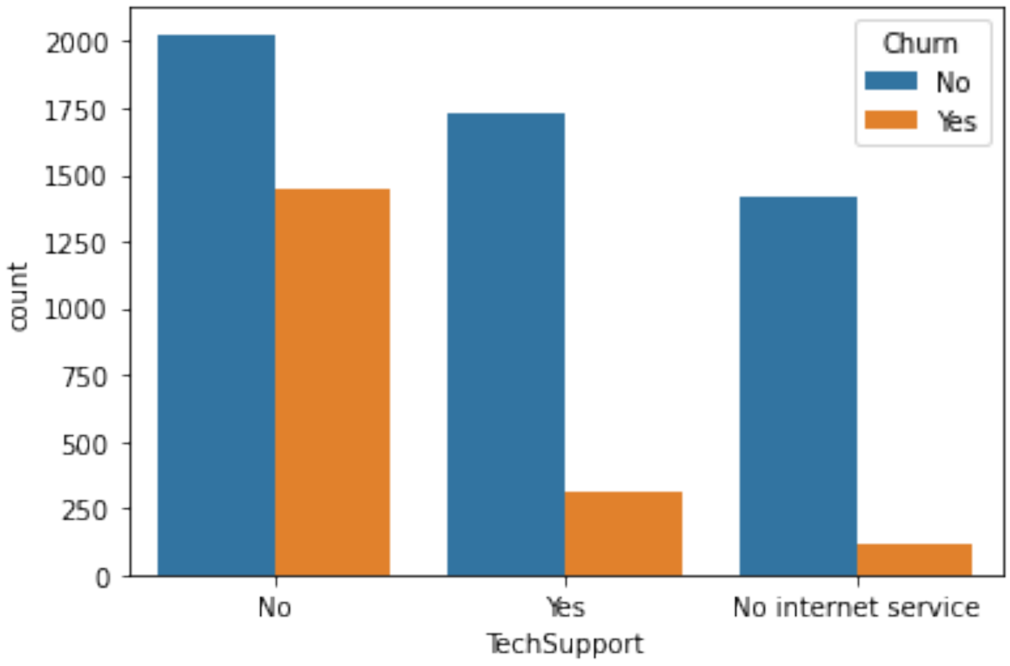
```
<AxesSubplot:xlabel='DeviceProtection', ylabel='count'>
```



Here we can see that only 50% of the customers are getting device protection service and as a result company is loosing more customers

```
sns.countplot(x='TechSupport',hue='Churn',data=df)
```

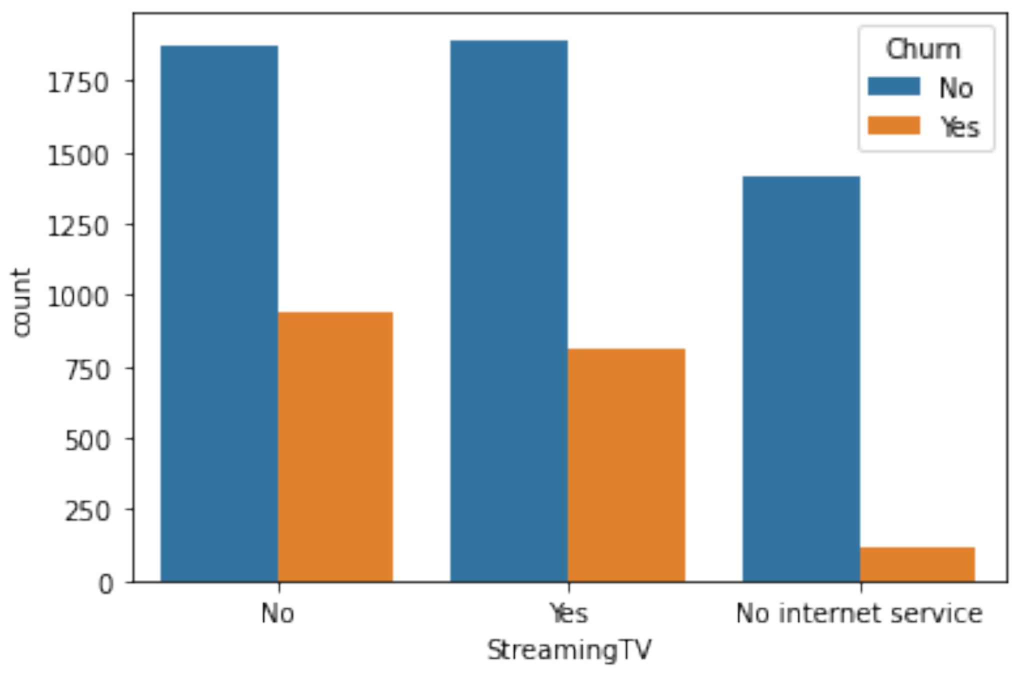
```
<AxesSubplot:xlabel='TechSupport', ylabel='count'>
```



Tech support are also given to less than 50% customers . So we can see that how much the customers are getting services really effective for the company.

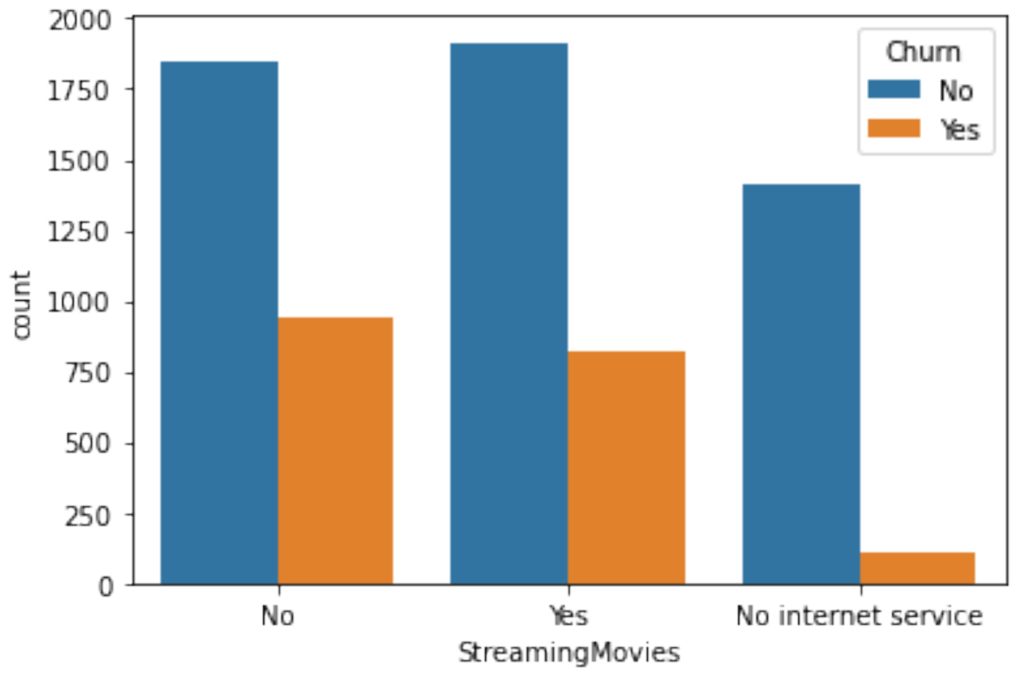
```
sns.countplot(x='StreamingTV',hue='Churn',data=df)
```

```
<AxesSubplot:xlabel='StreamingTV', ylabel='count'>
```



```
sns.countplot(x='StreamingMovies',hue='Churn',data=df
)
```

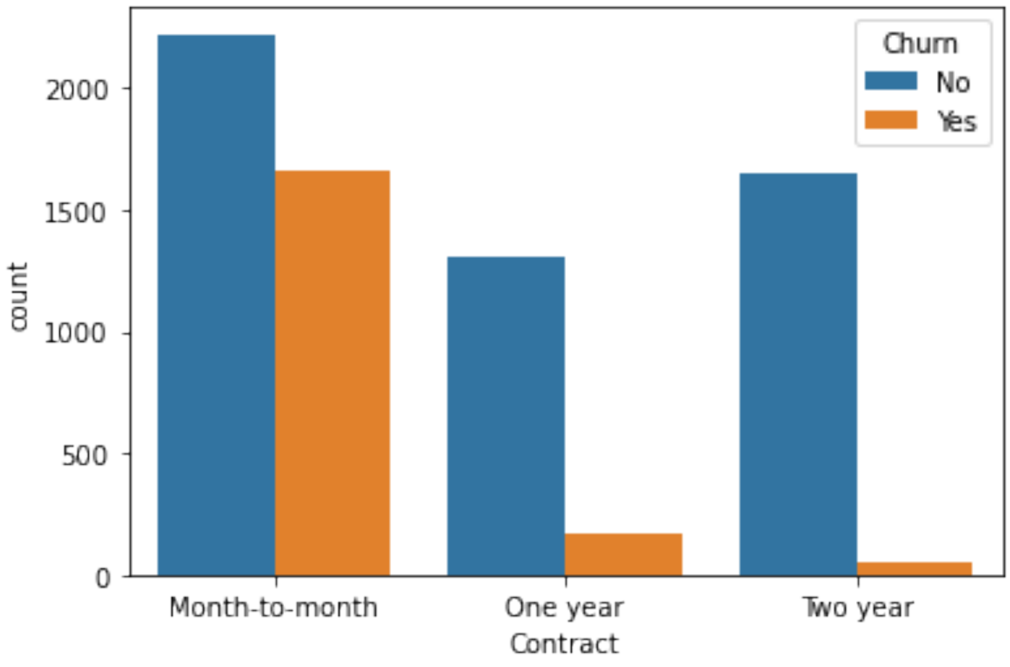
```
<AxesSubplot:xlabel='StreamingMovies',
ylabel='count'>
```



Though this service is given to lil more customers compare to other services still result are same

```
sns.countplot(x='Contract',hue='Churn',data=df)
```

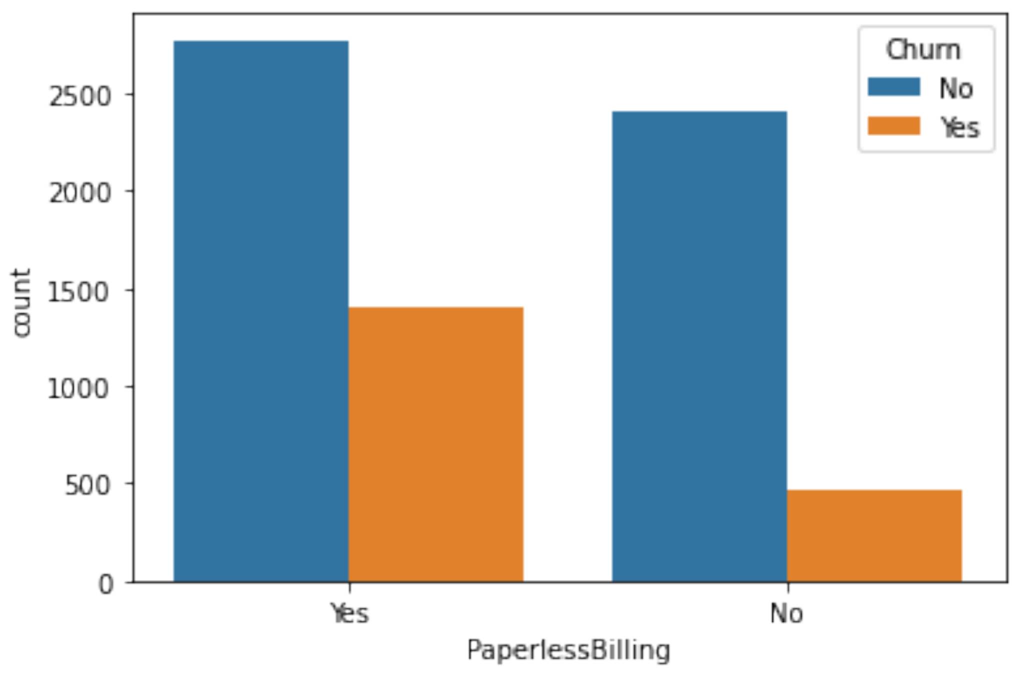
```
<AxesSubplot:xlabel='Contract', ylabel='count'>
```



Here we can see that most of the customers are on a monthly basis contract so the attrition is more but those who are for longer period having less churn and this is definately loss for company. So they should look into this and try convert monthly customers for having longer period contract.

```
sns.countplot(x='PaperlessBilling',hue='Churn',data=d
f)
```

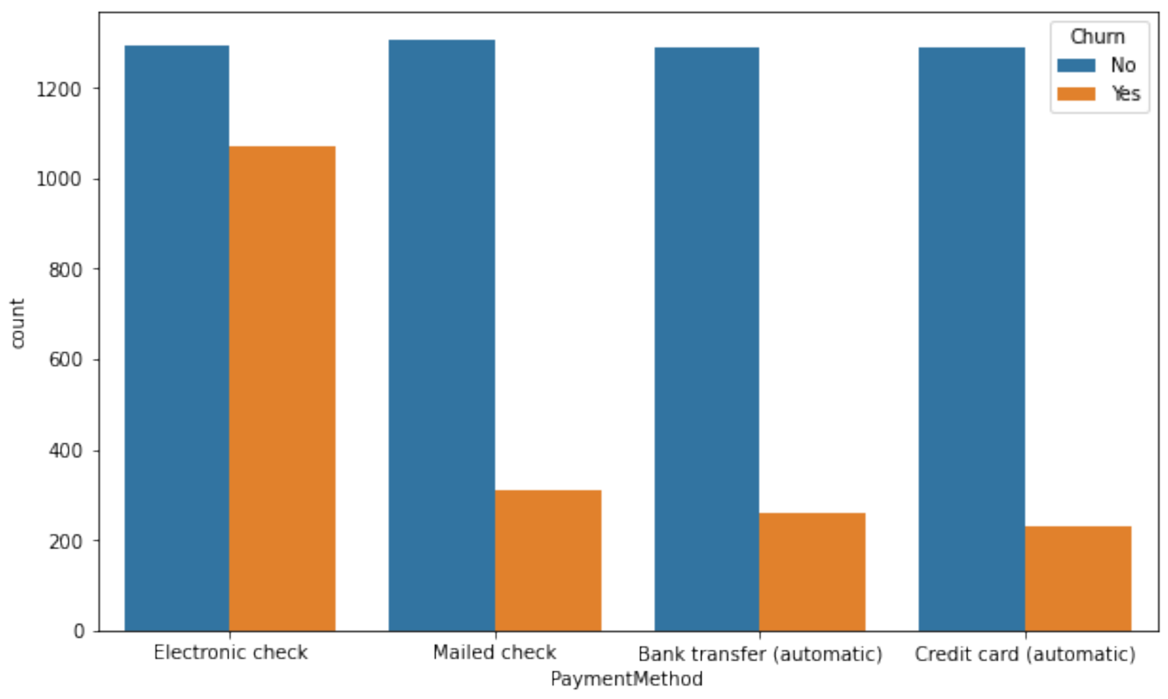
```
<AxesSubplot:xlabel='PaperlessBilling',
ylabel='count'>
```



Paperless billing are not only economical but also easy to keep record which are also helpful to have less churn.

```
plt.figure(figsize=(10,6))
sns.countplot(x='PaymentMethod',hue='Churn',data=df)
```

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



Here we can see that almost all the customers are almost on same label for using payment mode but in electronic check no of loosing cutomers are more compare to others

Data Processing

Will drop Customer id from the dataset because it does not contribute anything related to probability of whether customer will churn or not.

```
# dropping customer id column as not required for our
prediction
df.drop(columns=['customerID'],inplace=True)
```

Feature Engineering

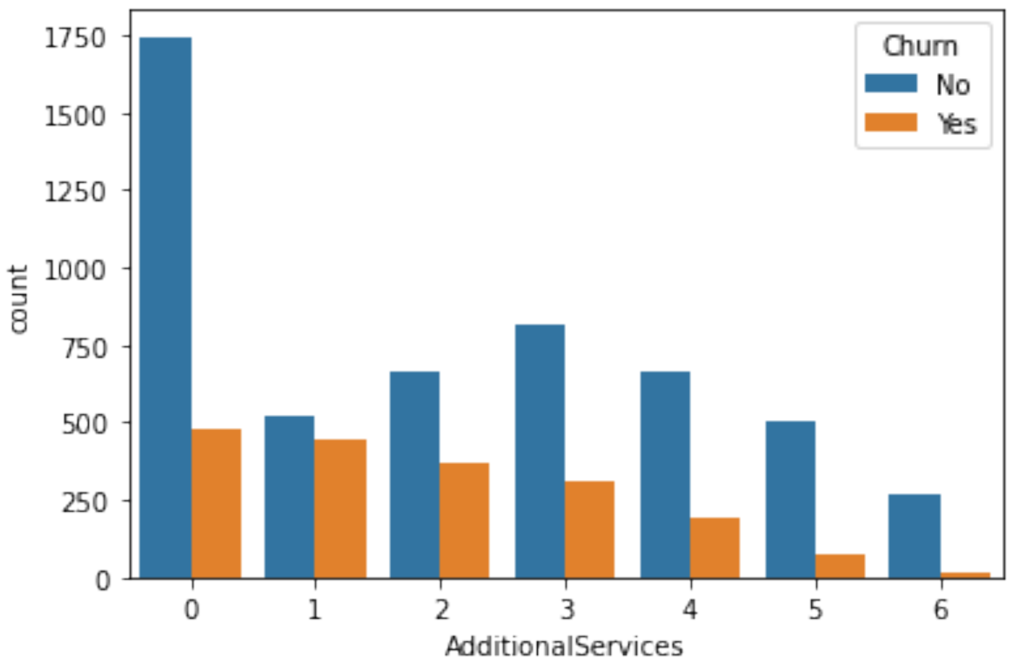
Applying labelencoder for all binary cases (ie.,yes/no)

LabelEncoder is one of the feature engineering we use to convert object to integer form which will help our machine algorithm to understand and interpret for further processing. While using LabelEncoder our object get changed to integer form and no of column will also not increase which usually we can see in one hot encoder.

```
# Adding all the additional services
df['AdditionalServices'] =
(df[['OnlineSecurity','OnlineBackup','DeviceProtectio
n','TechSupport','StreamingTV'
                              ,'StreamingMovies']]==
'Yes').sum(axis=1)
```

```
sns.countplot(x='AdditionalServices',hue='Churn',data
=df)
```

```
<AxesSubplot:xlabel='AdditionalServices',
ylabel='count'>
```



Customers who are having high additional services are having less churning rate compare to those who gets less additional services

```
features_le =
['gender','Partner','Dependents','PhoneService','Pape
rlessBilling']
def Label_Encoding (features,df):
    for i in features:
        df[i]=df[i].map({'Yes':1,'No':1})
    return
Label_Encoding
(['Partner','Dependents','PhoneService','PaperlessBil
ling',],df)
df['gender'] =
df['gender'].map({'Female':1,'Male':0})
df['Churn']=df['Churn'].replace({'Yes’:1,’No’:0})
```

```
le=LabelEncoder()
df1 = le.fit_transform(df['TotalCharges'])
pd.Series(df1)
df['TotalCharges']=df1
```

All the binary for dataset has been changed to numerical form using label encoder.

Will use one hot encoder for those column having more than two objects.

```
features_ohe =
['MultipleLines','InternetService','OnlineSecurity','
OnlineBackup','DeviceProtection','TechSupport',

'StreamingTV','StreamingMovies','Contract','PaymentMe
thod','AdditionalServices']
df = pd.get_dummies(df,columns=features_ohe)
```

```
df.head()
```

After that we checked the changed we made using encoding techniques.

Let's cheque dataset shape after encoding

```
df.shape
```

```
(7043, 48)
```

Here we can see that no columns has been increased previously including customer id it was 21 and after removal it become 20 . Here we can see 48 columns due to one hot encoder we used and as we know that using one hot encoder no of columns will increase.

Will check skewness and outliers if any in dataset

Skewness in any dataset will impact a lot while performing the model and outliers and skewness are interrelated to each other so we need to remove both if our data consist the same.

```
df.skew()
```

```
gender                                   0.019031
SeniorCitizen                            1.833633
Partner                                  0.000000
Dependents                               0.000000
tenure                                   0.239540
PhoneService                             0.000000
```

```
PaperlessBilling                          0.000000
MonthlyCharges                           -0.220524
TotalCharges                              0.015857
Churn                                     1.063031
MultipleLines_No                          0.074752
MultipleLines_No phone service            2.727153
MultipleLines_Yes                         0.316610
InternetService_DSL                       0.658113
InternetService_Fiber optic               0.243494
InternetService_No                        1.375769
OnlineSecurity_No                         0.013350
OnlineSecurity_No internet service        1.375769
OnlineSecurity_Yes                        0.943722
OnlineBackup_No                           0.248142
OnlineBackup_No internet service          1.375769
OnlineBackup_Yes                          0.652817
DeviceProtection_No                       0.244075
DeviceProtection_No internet service      1.375769
DeviceProtection_Yes                      0.657450
TechSupport_No                            0.027554
TechSupport_No internet service           1.375769
TechSupport_Yes                           0.924630
StreamingTV_No                            0.412686
StreamingTV_No internet service           1.375769
StreamingTV_Yes                           0.475581
StreamingMovies_No                        0.427838
StreamingMovies_No internet service       1.375769
StreamingMovies_Yes                       0.460199
Contract_Month-to-month                  -0.201829
Contract_One year                         1.430637
Contract_Two year                         1.213561
PaymentMethod_Bank transfer (automatic)   1.357605
PaymentMethod_Credit card (automatic)     1.379837
PaymentMethod_Electronic check            0.695541
PaymentMethod_Mailed check                1.290981
AdditionalServices_0                      0.796376
AdditionalServices_1                      2.109916
AdditionalServices_2                      1.997895
AdditionalServices_3                      1.868106
AdditionalServices_4                      2.325157
AdditionalServices_5                      3.070300
```

```
AdditionalServices_6                        4.674465
```

```
Skewness can be removed only for continuous column.
But in this dataset no skewness found so no need to
treat.
```

Checking Outliers

Outliers are extreme values that fall a long way outside of the other observations.
Here checking outliers using quantile method as having only three continuous columns i.e.,
`['MonthlyCharges','TotalCharges','tenure']`

```
con_features =
['MonthlyCharges','TotalCharges','tenure']
df_num = df[con_features]
df_num.describe()
Q1 = df_num.quantile(0.25)
Q3 = df_num.quantile(0.75)
IQR = Q3-Q1
IQR
((df_num <(Q1-1.5*IQR)) | (df_num >
(Q3+1.5*IQR))).any()
```

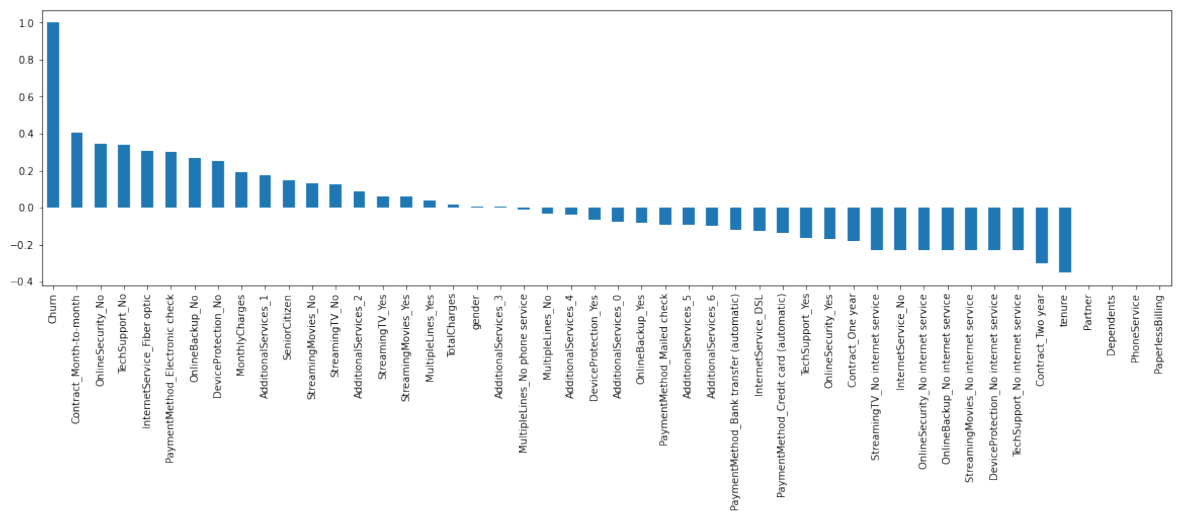
```
MonthlyCharges    False
TotalCharges      False
tenure            False
```

No outliers in any of the continuous columns.

Will check correlation between feature and target

```
plt.figure(figsize=(15,10))
df.corr()
['Churn'].sort_values(ascending=False).plot(kind='bar
',figsize=(20,5))
```

```
<AxesSubplot:>
```



correlation is what - Two features (variables) can be positively correlated with each other. It means that when the value of one variable increase then the value of the other variable(s) also increases.
Two features (variables) can be negatively correlated with each other. It means that when the value of one variable increase then the value of the other variable(s) decreases.
Two features (variables) are not correlated with each other. It means that when the value of one variable increase or decrease then the value of the other variable(s) doesn't increase or decreases.
And The value of the coefficient of correlation will always lie between -1 and +1, it means, there is perfect positive correlation between the variables if coefficient having +1 while having negative correlation if coefficient is -1 and if the coefficient value is 0 that means no relation

Here we looking how's the correlation between feature and target. None of the variables perfectly lies on 0 scale that means either positive or negative correlation exist. Here Churn we can see that having positive correlation it is because if we compare same variable so definitely it will perfect positive correlation that only we can see here.But other feature as we can see maximum lies between -4 to +4. That means no need to delete any of the column as none of the columns are positively or negatively correlated.

Balancing Data

Above we observed that our dataset is an imbalanced dataset so we

need to balance the same either using oversampling or undersampling method . But using under sampling method we may loose lot many dataset which may affect our model performance so here I will use oversampling method using SMOTE.
Below we will see that what our target do have within the dataset using unique value

```
df['Churn'].unique()
array(['No', 'Yes']
```

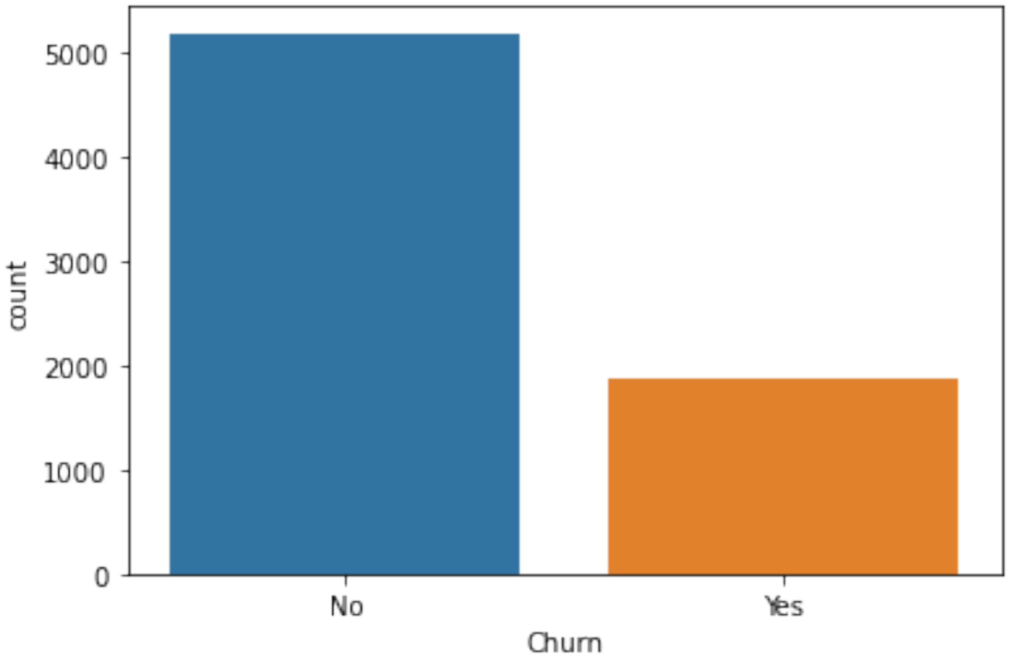
So our target do have two values yes and no which we converted above using replace method. Now will check using count method how much our dataset is imbalanced.

```
df['Churn'].value_counts()

No      5174
Yes     1869
```

```
sns.countplot('Churn',data=df)

<AxesSubplot:xlabel='Churn', ylabel='count'>
```



So here churning are less as we can see above plot as well as data which shows that 36% of the customer are moving out from the company. But definitely % are not less . Company should look into this and try to find out the reason for churning.

```
## Splitting target and feature variable into x and y
x = df.drop(columns=['Churn'])
y= df['Churn']
```

All the feature are in x variable and target in y variable.

```
x.shape
(7043, 47)
```

```
y.shape
(7043,)
```

```
from imblearn.over_sampling import SMOTE
sm = SMOTE()
```

```
x1,y1 = sm.fit_resample(x,y)
```

```
y1.value_counts()

1    5174
0    5174
```

```
Balanced the dataset using SMOTE method. So here we
did oversampling
```

Scaling the dataset and model Training

Here will scale all the feature dataset before proceeding for model training as all the feature column should be on same scale so that our machine can train the model without any hurdle. Scaling is not applicable for target variable.

```
scaler = StandardScaler()
x_scaler = scaler.fit_transform(x1)
x_scaler
```

```
array([[ 1.18525687, -0.39309561,  0.        , ...,
-0.31479705,
        -0.24345249, -0.16950071],
       [-0.84369897, -0.39309561,  0.        , ...,
-0.31479705,
        -0.24345249, -0.16950071],
       [-0.84369897, -0.39309561,  0.        , ...,
-0.31479705,
        -0.24345249, -0.16950071],
       ...,
       [-0.84369897, -0.39309561,  0.        , ...,
-0.31479705,
        -0.24345249, -0.16950071],
       [ 1.18525687, -0.39309561,  0.        , ...,
-0.31479705,
        -0.24345249, -0.16950071],
       [-0.84369897, -0.39309561,  0.        , ...,
3.1766498 ,
        -0.24345249, -0.16950071]])
```

Finding Best Random State

```
maxaccu = 0
maxrs = 0

for i in range(1,500):
    x1_train,x1_test,y1_train,y1_test =
train_test_split(x_scaler,y1,test_size =
0.30,random_state = i)
    dt = DecisionTreeClassifier()
    dt.fit(x1_train,y1_train)
    pred = dt.predict(x1_test)
    acc = accuracy_score(y1_test,pred)
    if acc>maxaccu:
        maxaccu=acc
        maxrs=i
print("Best Accuracy score is:",maxaccu,"On Random
state: ",maxrs)
```

```
Best Accuracy score is: 0.8238325281803542 On Random
state:  437
```

Here we got accuracy score as 82% approx which is on 437 best random state.

Training the model

```
x1_train,x1_test,y1_train,y1_test=train_test_split(x_
scaler,y1,random_state=i,test_size=0.30)
```

Training data is 70% and testing data we given as 30% based on this our model will be trained on 70% data and will do the prediction accordingly how far they can understand the dataset.

```
DTC = DecisionTreeClassifier()
DTC.fit(x1_train,y1_train)
pred = DTC.predict(x1_test)
acc = classification_report(y1_test,pred)
print(acc)
```

```
                  precision    recall  f1-score
support

           0       0.80      0.82      0.81      1513
           1       0.83      0.80      0.81      1592

    accuracy                           0.81      3105
   macro avg       0.81      0.81      0.81      3105
weighted avg       0.81      0.81      0.81      3105
```

First model I used was DecisionTreeClassifier which is giving accuracy score as 81%, precision - 80%, recall - 82%, f1-score - 81% . Here I used accuracy score as a matrix to predict how well my model is performing on a given based dataset and as a result we can see that our model is predicting 82% of the time, customers churning correctly. Moreover got recall and precision score too which doesn't have much difference from our matrix prediction score. F1 score which is a combined score of precision and recall is also predicting almost the same percentage of customer churn.
Here we didn't get much difference in score of F1 because precision and recall are almost giving same prediction as a result otherwise the F-score is computed with the harmonic mean of precision and recall. Note that it assigns much more weight to low values. As a result of that, the classifier will only get a high F-score, if both recall and precision are high.

```
print(cross_val_score(DTC,x_scaler,y1,cv=5).mean())
```

```
0.7969803611163646
```

Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample.That is, to use a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model.The purpose of cross–validation is to test the ability of a machine learning model to predict new data. It is also used to flag problems like overfitting or selection bias and gives insights on how the model will generalize to an independent dataset.

Here we got cross validation score as 79% on an average so difference between accuracy score and cross validation is approx 2% which is not much. Lets see other model performance and based on that will decide which model is performing best for this dataset.

```
RFC = RandomForestClassifier()
RFC.fit(x1_train,y1_train)
pred =RFC.predict(x1_test)
acc = classification_report(y1_test,pred)
print(acc)
```

```
              precision    recall  f1-score   support

           0       0.82      0.87      0.84      1513
           1       0.87      0.82      0.84      1592

    accuracy                           0.84      3105
   macro avg       0.84      0.84      0.84      3105
weighted avg       0.85      0.84      0.84      3105
```

random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction. Random forest has nearly the same hyperparameters as a decision tree or a bagging classifier. As from above definition we can understand that Random forest can work much better than decision tree as it works on multiple decision trees and merge them together to have more accurate result so here we can see little improvement in score compare to DecisionTree Classifier. Accuracy score is 84% almost 3% more compare to DecisionTree Classifier. Precision score is 82% and recall which really shows a good percentage which is 87%. So here compare to precision recall is having higher score which in affect we can see change in f1 score that is 84% as it takes the mean of both. But f1 score is not perfect as it favours classifiers that have similar precision and recall and this becomes problem as sometimes we want high precision and sometimes high recall. And as we know that sometimes an increasing precision results in a decreasing recall and vice versa which is also known as precision/ recall tradeoff.

```
print(cross_val_score(RFC,x_scaler,y1,cv=5).mean())
```

```
0.8324482176504787
```

Here even cross validation score is better compare to DecisionTree cross validation score. And the difference between randomForest and Cross Validation are almost negligible which is we can 0.7% approx.

```
SV = SVC()
SV.fit(x1_train,y1_train)
pred = SV.predict(x1_test)
acc = classification_report(y1_test,pred)
print(acc)
```

```
              precision    recall  f1-score   support

           0       0.83      0.87      0.85      1513
           1       0.87      0.82      0.85      1592

    accuracy                           0.85      3105
   macro avg       0.85      0.85      0.85      3105
weighted avg       0.85      0.85      0.85      3105
```

SVC is such type of class which is capable of performing binary and multi-class classification on a dataset. It is a nonparametric clustering algorithm that does not make any assumption on the number or shape of the clusters in the data. It work best for low dimensional data. Compare to both the above model here we can see that SVC accuracy score is better which is 85% not much difference from

DecisionTreeClassifier still having best performance ,may be due to our target variable is in binary form and as per the above definition SVC works on binary and multi classification.
Even precision score is 83%, Recall - 87% not having any difference compare to RandomForestClassifier and f1-score is 85% .

```
print(cross_val_score(SV,x_scaler,y1,cv=5).mean())
```

```
0.8260704720943861
```

But cross validation score is not so good as compare to other model performance. Difference between accuracy score and cross validation score is approx 2.5% , that means here our best model is RandomForestClassifier with a very less or minor difference between CV score Acc_score. So will do hyper parameter tuning for the same.

Let’s plot ROC AUC Curve to have better glance on each model

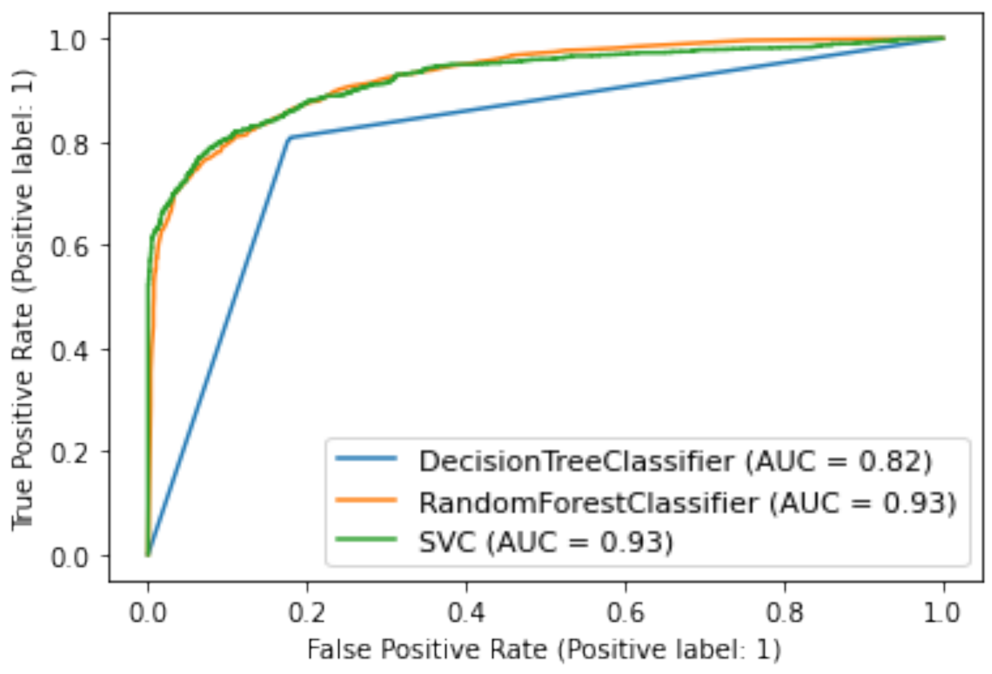
```
disp = plot_roc_curve(DTC,x1_test,y1_test)

plot_roc_curve(RFC,x1_test,y1_test,ax = disp.ax_) # 
ax = axes with confusion mtrix

plot_roc_curve(SV,x1_test,y1_test,ax = disp.ax_)

plt.legend(prop={'size':11}, loc = 'lower right')

plt.show()
```



Another way to evaluate and compare binary classifier is provided by the ROC AUC Curve. This curve plots the true positive rate (also called recall) against the false positive rate (ratio of incorrectly classified negative instances).
Here RandomForest Classifier seems to be performing better compare to other models.

Hyperparameter Tuning

As our best model for this dataset which is giving not only higher score but less difference between CV score and there accuracy score is RandomForest Classifier, so will do hyper parameter tuning for RandomForestClassifier.

```
# RandomForestClassifier
param = {'n_estimators':[100,500,700,900],
        'criterion':['gini','entropy'],
        'max_depth':[10,20,30,40,50],
        'max_features':['auto','sqrt','log2'],
        'class_weight':
['balanced','balanced_subsample']}
```

Choosing parameters for our model to get trained so that we can get some improvement in our score if possible.

```
GC = GridSearchCV(RFC,param,cv=5)
```

Passing the model for which we want to do Grid Search CV along with parameters we choose and CV score.

```
GC.fit(x1_train,y1_train)
```

```
GridSearchCV(cv=5,
estimator=RandomForestClassifier(),
             param_grid={'class_weight': ['balanced',
'balanced_subsample'],
                          'criterion': ['gini',
'entropy'],
                          'max_depth': [10, 20, 30,
40, 50],
                          'max_features': ['auto',
'sqrt', 'log2'],
                          'n_estimators': [100, 500,
700, 900]})
```

```
Training our model
```

```
GC.best_params_
```

```
{'class_weight': 'balanced',
 'criterion': 'entropy',
 'max_depth': 20,
 'max_features': 'sqrt',
 'n_estimators': 700}
```

Best Parameters which Gridsearch CV passed after going through the dataset are mentioned above. As we know that Grid-search is used to find the optimal hyperparameters of a model which results in the most 'accurate' predictions.

```
final_rfc =
RandomForestClassifier(class_weight='balanced',criter
ion= 'entropy',max_depth = 20, max_features =
'sqrt',n_estimators = 700)
final_rfc.fit(x1_train,y1_train)
```

```
pred = final_rfc.predict(x1_test)
acc = accuracy_score(y1_test,pred)
print(acc*100)
85.1207729468599
```

```
Final score after doing GridSearchCV we got as 85%
approx. Only 1% improvement in our model performance
we can see after doing hyper parameter Tuning.

Next will save the model using Joblib so that if want
to load in future for any changes or improvement we
can do the same.
```

Save the Model

```
import joblib
joblib.dump(final_log,'FinalmodelAttrition.pkl')
```

Summary

Dataset on which we started was whether customer is churning from the company or not if yes then in what percentage. So for this first I import all the important libraries which was required to perform all the necessary steps. Next imported the dataset on which I was suppose to work using pandas.And after analysing the dataset came across that this is a classification based dataset and we need to perform accordingly.Then did all the necessary EDA to get the better insight of the dataset so that will able to understand what exactly our data is and what are the steps we can proceed further for having better model performance and more accurate score. During EDA (evaluating data analysis) we got lot many important information which will really be helpful for the company to curb the churn of customers. Likewise as we seen that tenure is one of the important role playing here. Customer for longer period are moving less compare to customer who are for shorter period so company should think of convincing their customer for having longer period contract. Which will not only be helpful in curbing churn but a company can also build good relation with their customer by knowing them in a better way and providing good service and benefits.Next we seen that customer who were independent, young generation were more volatile so this is also one of the important factor we can say that youngsters looking for more opportunities and better scope to have more

profit which may lure them to move frequently. Next concern was additional services which being provided to customers by company , so here we observed that customers who are getting more additional services stick more to the company and those who are getting less are churning more. In this scenario company can retain those who are really an asset for them by giving additional services and other benefits, so that it will not only help company to retain there existing customer but will also be cost effective as they don't have to spend more on marketing and advertising to search again for new customers , also knowing them and their expectations will be time taking. These are the important points which company should think of and take an appropriate steps accordingly. During EDA used seaboard and matplotlib for visualisation analysis. In data preprocessing there were not much to do as this dataset were not having any null / missing values . Did feature engineering for columns who were in string / object form using Label Encoder and one hot encoder. As our machine algorithm doesn't understand string / object format. This data even doesn't have any skewness or outliers. But here as our dataset was an imbalanced dataset so we need to balance the same using either oversampling or under sampling method. Here I used SMOTE as doing under sampling may cost loss of lot many data which may affect our model performance. Next before training the model did splitting and stored all the feature in x variable and target into y variable. Scaled all the data to bring down to same unit as it might be possible that some columns may have different units, due to which model would have difficulty to understand and train the data.
After finding best random state on which our model would be trained started training the model by segregating 70% to training set and 30% to testing set. Here I used three algorithm to predict accuracy score which are
1. DecisionTreeClassifier
2. RandomForest Classifier
3. SVC (Support Vector Classifier)

On the basis of these three models did prediction, how good our data being trained and fitted to predict. Among these three RandomForestClassifier performance was best though Accuracy score for SVC was higher still difference between CV score and Accuracy score was low for RandomForest Classifier. After this did Hyperparameter Tuning but only 1% increase we were able to see. So from this I can say that there were more scope to have much better score than what we got by including more parameters to grid search cv or by doing more preprocessing either by checking if any feature can be removed or by doing some more EDA any other improvement could have been done .