Mini Project 2

(2019-2020)

On

Customer Churn Prediction

Mid Term Report

Institute of Engineering & Technology

ANKUR OMAR
(171500051)

Supervised By:
Mrs. Harvinder Kour
(Technical Trainer)
Department of Computer Engineering & Applications
GLA University
Mathura-281406, India

INDEX

ABSTRACT

This mid-term report documents the amount of work done in the Mini Project during this date. The report first shall give an overview of the tasks completed during this period with technical details. Then the report shall be analyzed. Report shall also elaborate on the future works which are still to be persuaded as an advancement of current work. I have tried my best to keep the report simple yet technically correct.

ABOUT THE PROJECT

In this project we will trying to predict the customer churn(responses) by the help of machine learning technology. The objective of the project is to reducing customer churn by identifying the potential churn customer and take proactive actions to make them stay.

SYSTEM REQUIREMENTS

- Hardware Requirement**:** laptop
- Software Requirement**:** Anaconda – Jupyter, Idle Python (3.7 64 bit)
- Implementation Language: Python

DATASET DESCRIPTION

- The data given is in the form of a comma-separated values files with customer id and their corresponding gender. The training dataset is a csv file of type customer id, churn, gender where the customer id is a unique integer identifying the customer, churn is either 'Yes' or 'No'. Similarly, the test dataset is a csv file of type customer id, gender.

- The data set contains different types of features or columns. the features like customer id, gender, customer services , churn, etc. these features help to identify the customer churn. in our data set, there are 21 columns and approximately 7000 rows.

PROCESS DIAGRAM

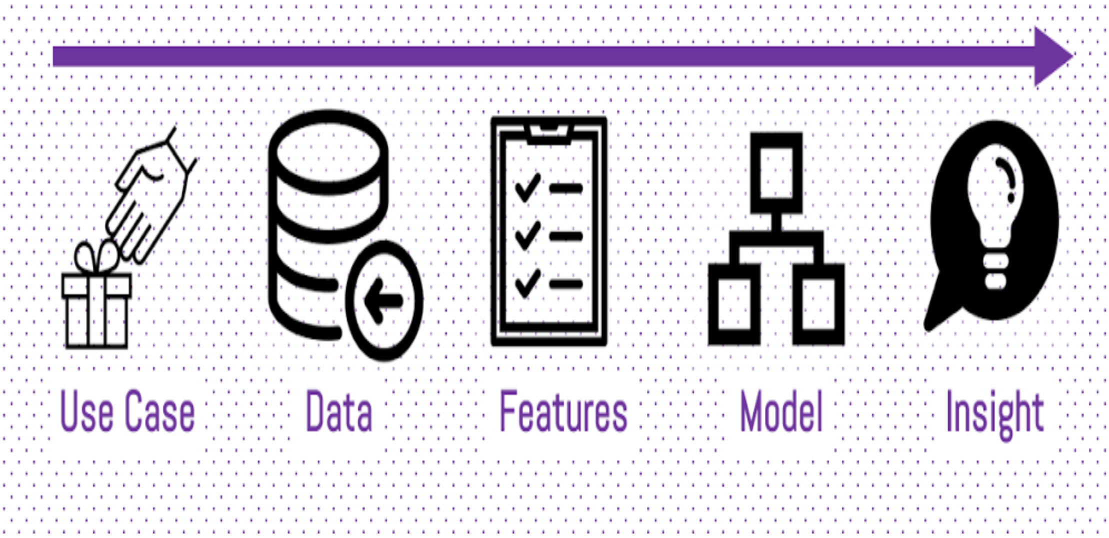


Fig:1

NECESSARY LIBRARIES

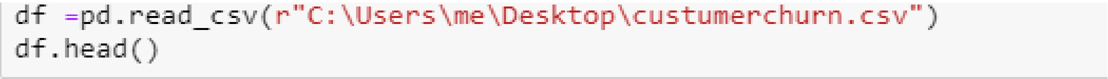
- Import numpy
- Import pandas as pd
- Import matplotlib
- Import seaborn

AREA OF DEVELOPMENT

This project deals with the area of Machine learning technology. The whole Project run on Jupyter Notebook with some important inbuilt python libraries which is help full to implement the machine learning concept. It is also being developed with major machine learning algorithms like Logistic Regression, Naïve byes, K-nearest neobhour,Decision Tree, Random forest .

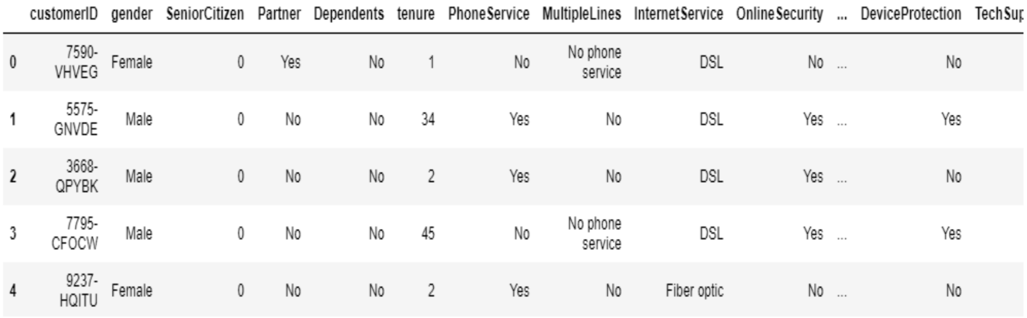
IMPLEMENTATION

1. First we import the data by the help of pandas library and view the some top of the information given in the dataset.



```
df =pd.read_csv(r"C:\Users\me\Desktop\custumerchurn.csv")
df.head()
```

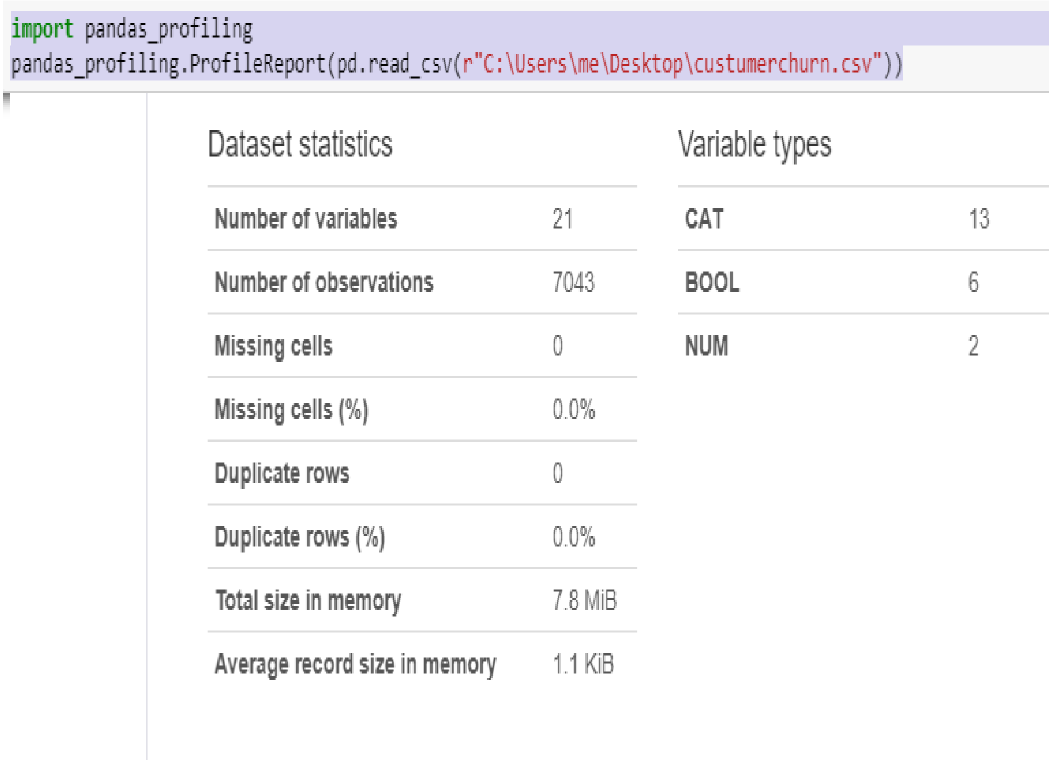
Output:



	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	TechSup
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	...	No	
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	...	Yes	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	...	No	
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes	...	Yes	
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	...	No	

2: By using import pandas_profiling library we view the dataset with all columns which have in dataset and check missing values, and also check the data type of each columns. After that we found there is no missing values and the number of categorical columns is more as compare to numeric columns.

Output:-



```
import pandas_profiling
pandas_profiling.ProfileReport(pd.read_csv(r"C:\Users\me\Desktop\custumerchurn.csv"))
```

Dataset statistics

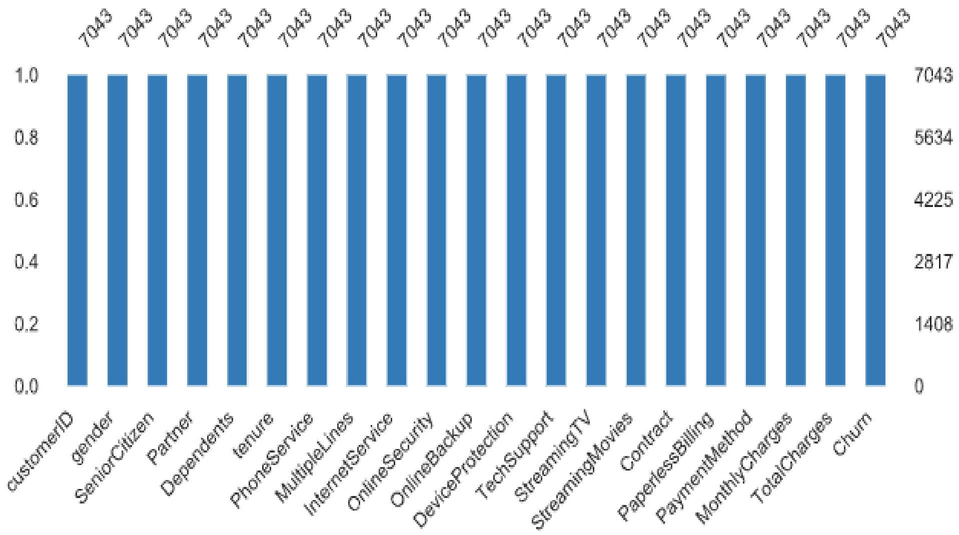
Number of variables	21
Number of observations	7043
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	7.8 MiB
Average record size in memory	1.1 KiB

Variable types

CAT	13
BOOL	6
NUM	2

3: Checking Missing Values:

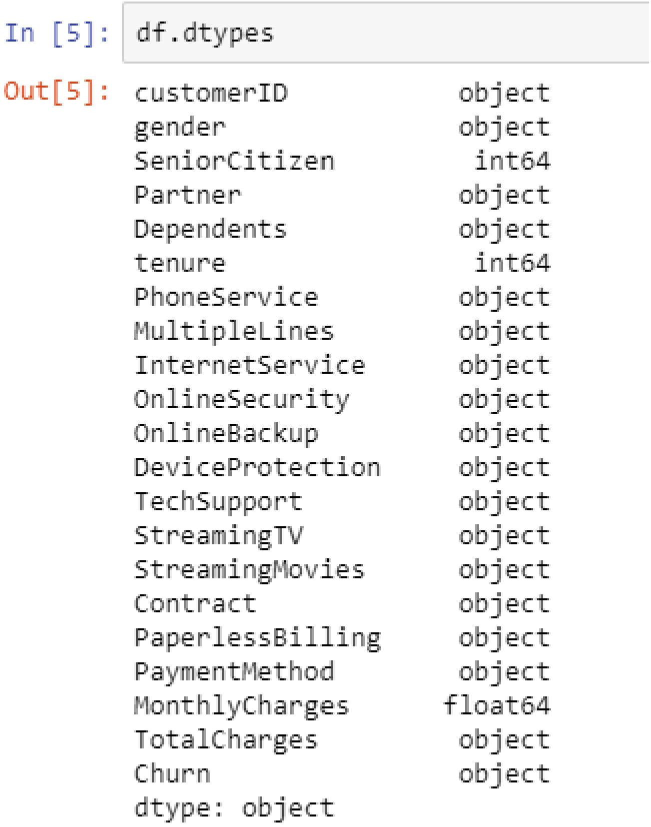
There is no null or missing values in the dataset. You can see the values in each columns.



4:- Check data type of each columns:

Here you can see the number of object data types is more.

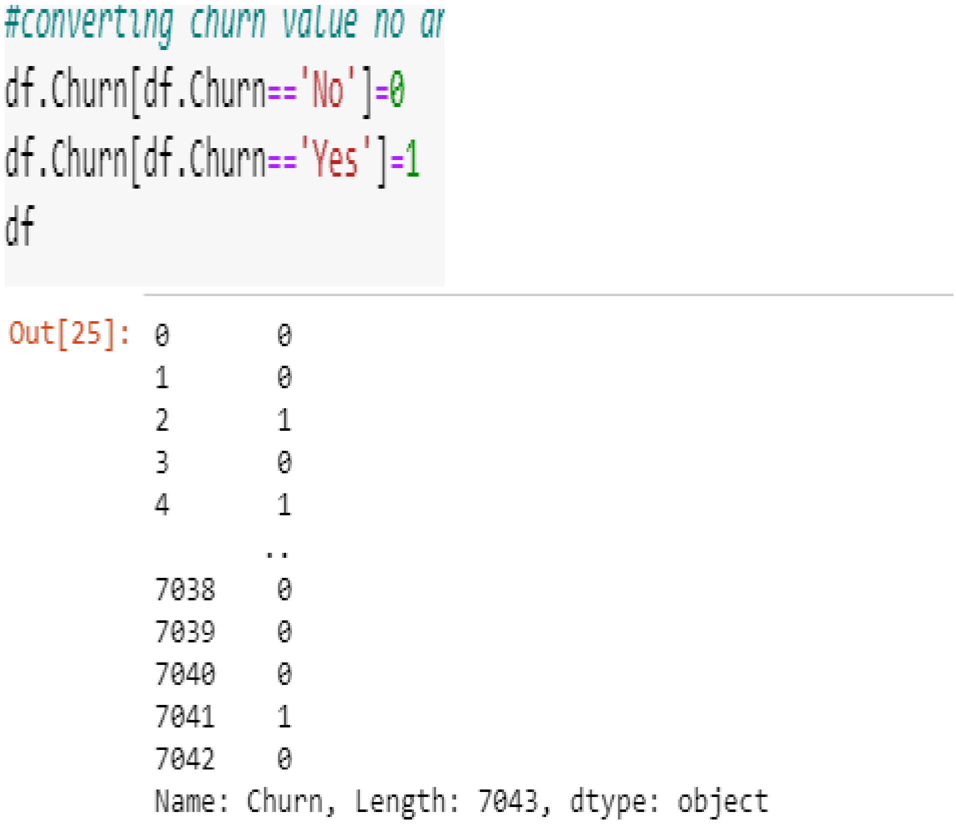
And there are only two integer type columns and only one float data type but here Total Charges column is object but we convert it from object data type to float data type. here object type is categorical features which show a class of ‘YES” and “NO”.



```
In [5]: df.dtypes

Out[5]: 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
```

Feature Scaling:-

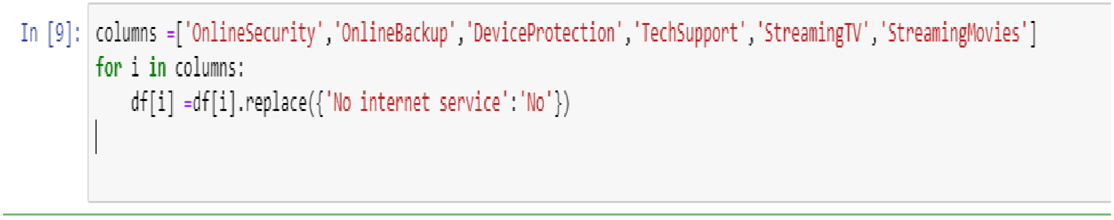
1:- The first step in feature scaling is to convert our target feature which is “Churn”, in to 0 or 1 form because our machine learning model will not be able to understand the string type values so we need to convert “yes” or “no” type values in to 0 or 1 form.



```
#converting churn value no ar
df.Churn[df.Churn=='No']=0
df.Churn[df.Churn=='Yes']=1
df
```

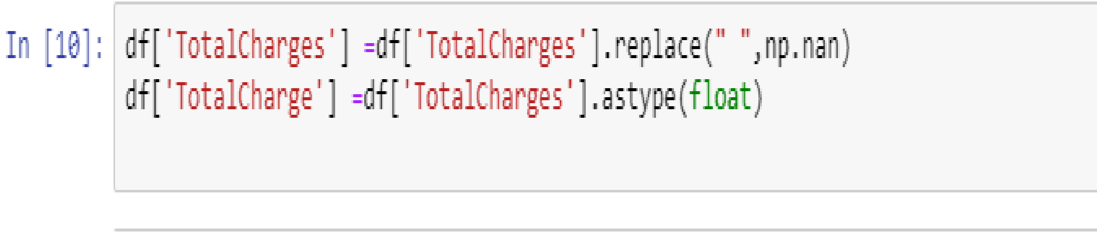
```
Out[25]: 0       0
         1       0
         2       1
         3       0
         4       1
                ..
         7038    0
         7039    0
         7040    0
         7041    1
         7042    0
         Name: Churn, Length: 7043, dtype: object
```

2:- In columns “OnlineSecurity”, “OnlineBackup”, “DeviceProtection”,”TechSupport”,”StremmingTv”,”StremmingMovie” have some values like “No Internet Service”, we convert this value with “No” in these columns.



```
In [9]: columns =['OnlineSecurity','OnlineBackup','DeviceProtection','TechSupport','StreamingTV','StreamingMovies']
        for i in columns:
            df[i] =df[i].replace({'No internet service':'No'})
```

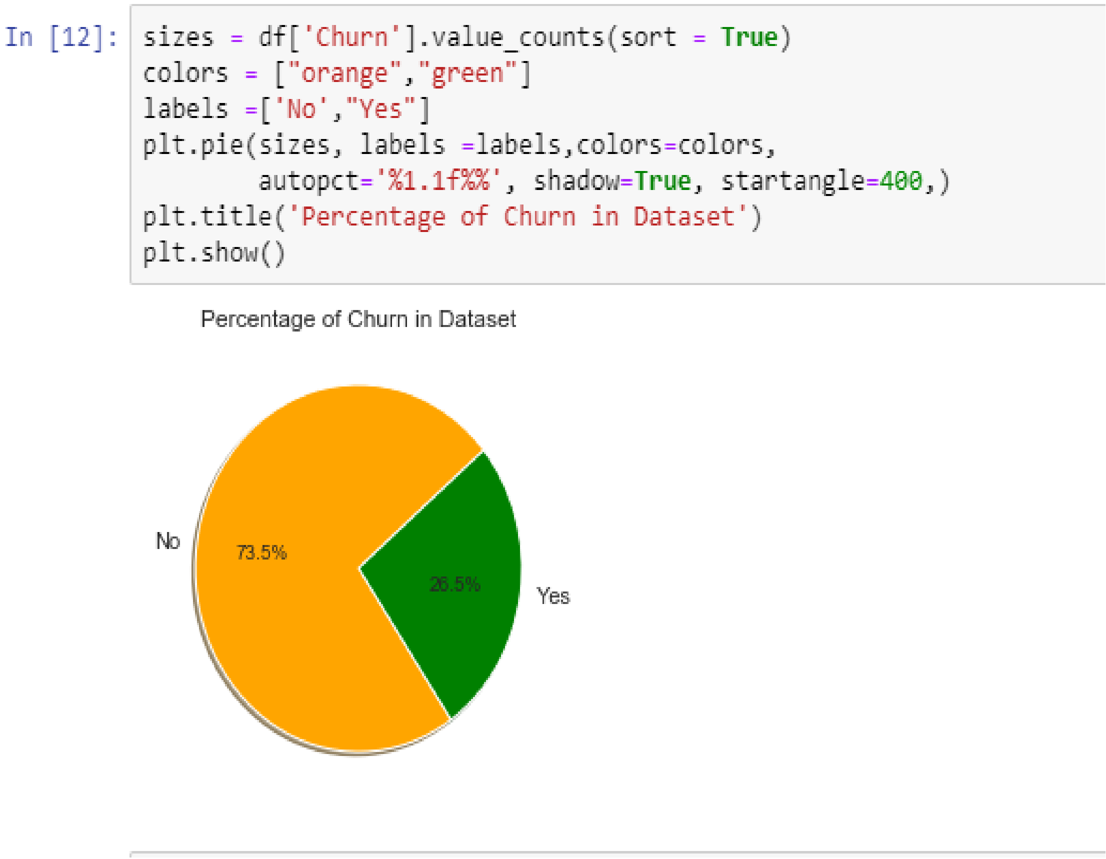
3:- The column “Total Charges” contain some spaces we replace all the spaces with null values and convert “Total Charges” feature to float values.



```
In [10]: df['TotalCharges'] =df['TotalCharges'].replace(" ",np.nan)
         df['TotalCharge'] =df['TotalCharges'].astype(float)
```

DATA VISUALIZATION

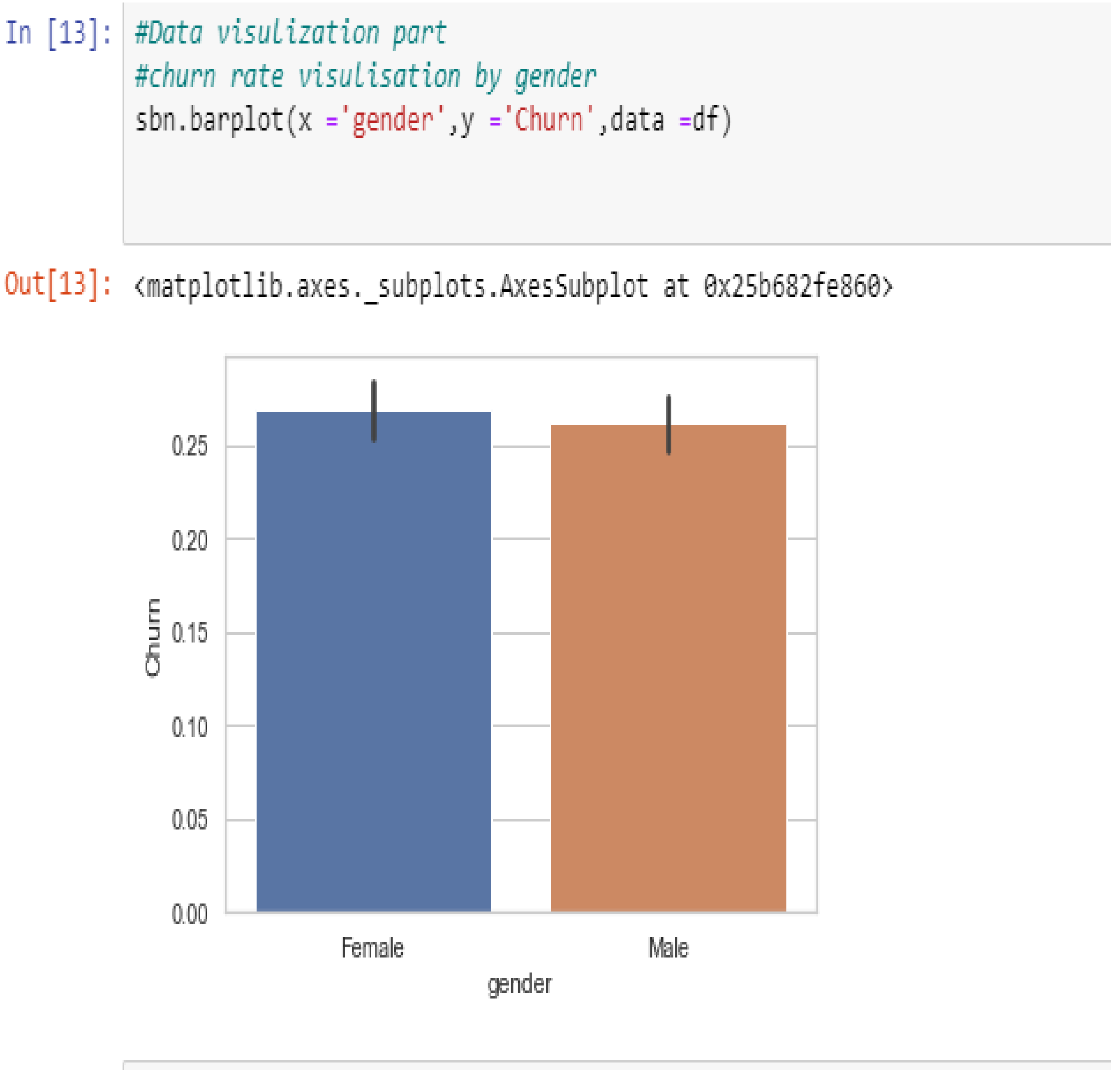
1:- First we visualize the output column data in form of graph –



```
In [12]: sizes = df['Churn'].value_counts(sort = True)
         colors = ["orange","green"]
         labels =['No',"Yes"]
         plt.pie(sizes, labels =labels,colors=colors,
                 autopct='%1.1f%%', shadow=True, startangle=400,)
         plt.title('Percentage of Churn in Dataset')
         plt.show()
```

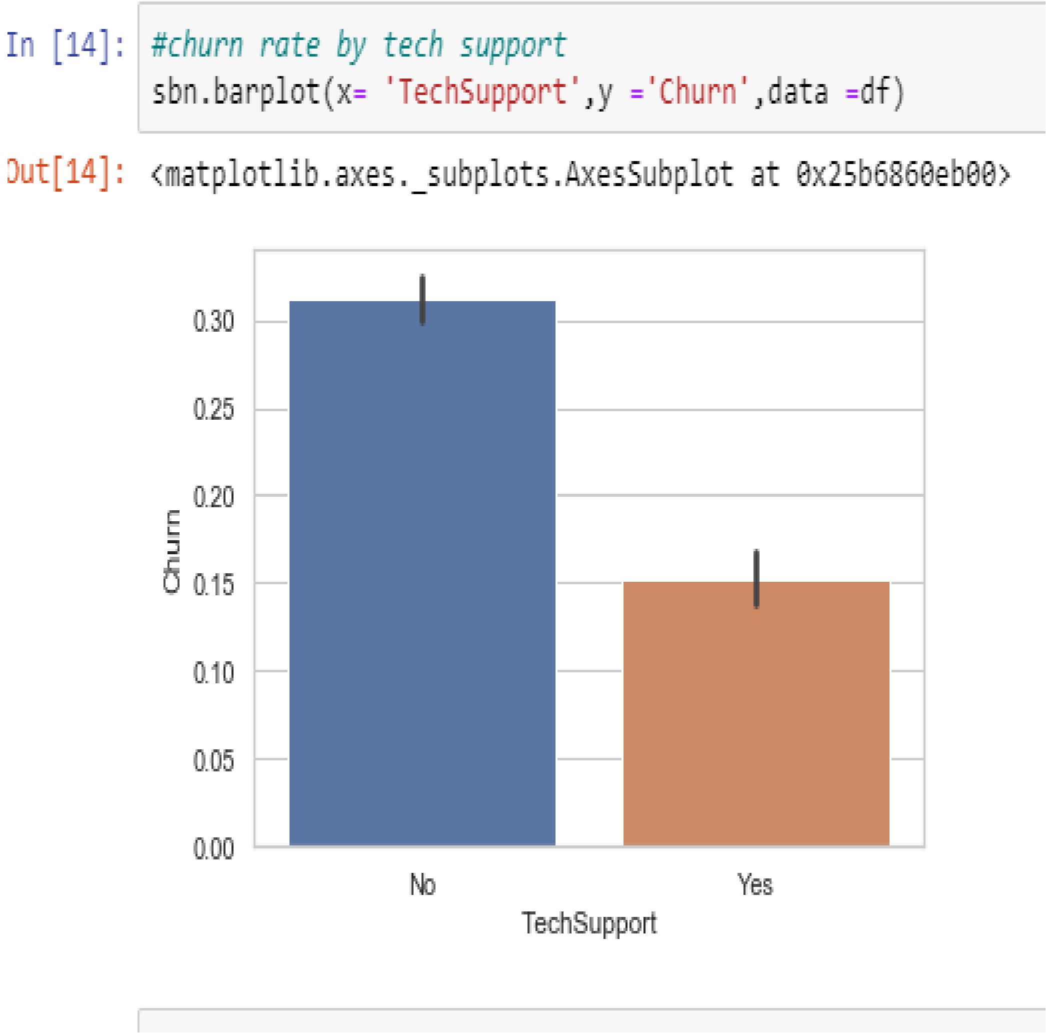
The above show the percentage of churn customer in a form of pie graph. As we can see that the number of customer who are not churned is more as compare to who are churned. There are 73.5% customer are not churn and 26.5% customer who are churn.

2:- After that we generate the graph between gender and churn –



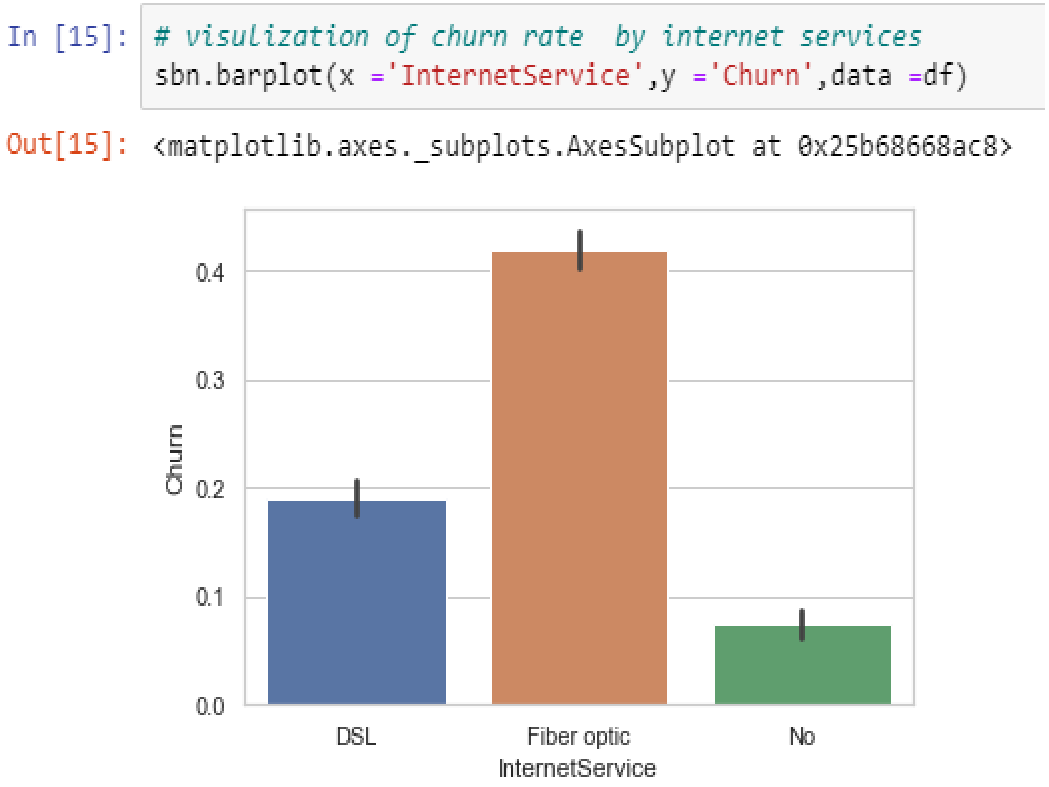
```
In [13]: #Data visulization part
         #churn rate visulisation by gender
         sbn.barplot(x ='gender',y ='Churn',data =df)

Out[13]: <matplotlib.axes._subplots.AxesSubplot at 0x25b682fe860>
```

3:- we show the graph between churn rate and tech support-



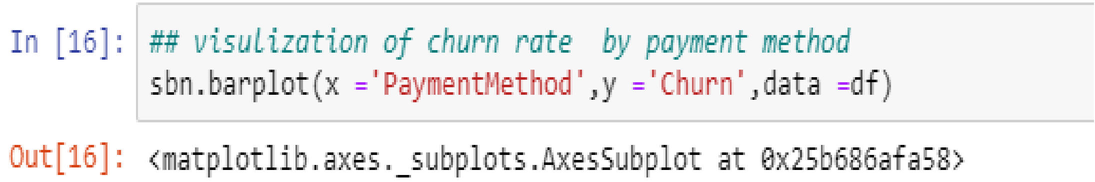
```
In [14]: #churn rate by tech support
         sbn.barplot(x= 'TechSupport',y ='Churn',data =df)

Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0x25b6860eb00>
```

4:- we show the graph between churn rate and internet services-

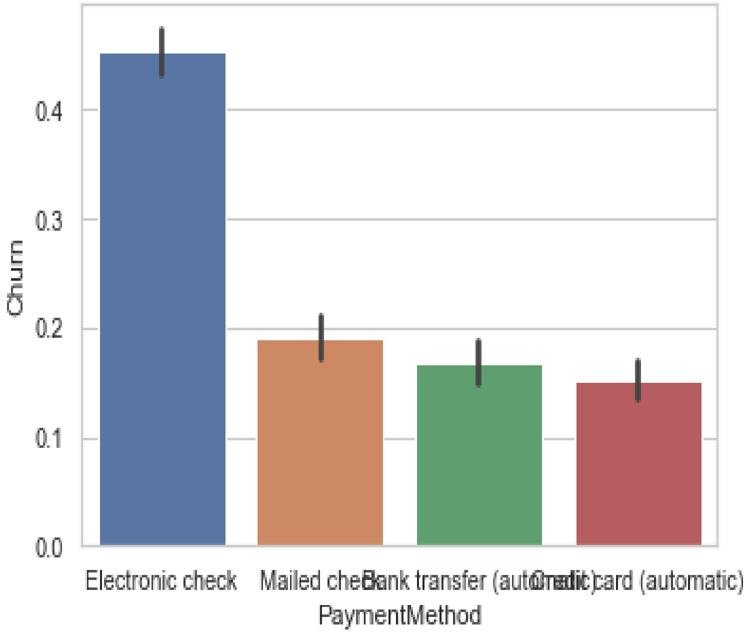


```
In [15]: # visulization of churn rate  by internet services
         sbn.barplot(x ='InternetService',y ='Churn',data =df)

Out[15]: <matplotlib.axes._subplots.AxesSubplot at 0x25b68668ac8>
```

5:- we show the graph between churn rate and payment method-



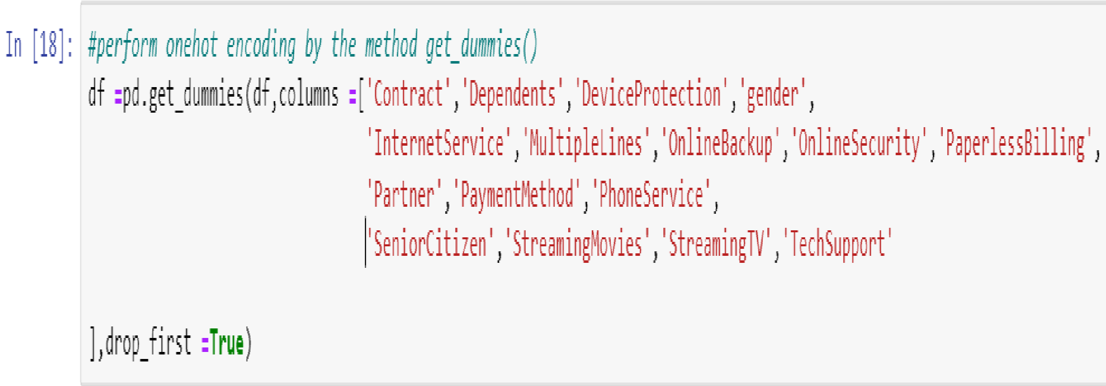
```
In [16]: ## visulization of churn rate  by payment method
         sbn.barplot(x ='PaymentMethod',y ='Churn',data =df)

Out[16]: <matplotlib.axes._subplots.AxesSubplot at 0x25b686afa58>
```



ONEHOT ENCODING

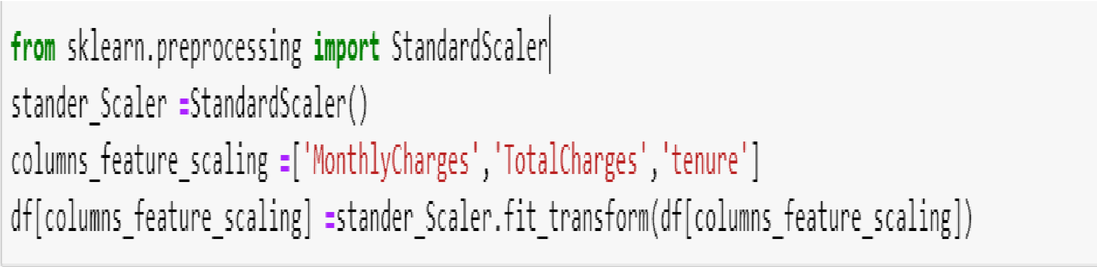
A one hot encoding allows the representation of categorical data to be more expressive. Many machine learning algorithms cannot work with categorical data directly. The categories must be converted into numbers. This is required for both input and output variables that are categorical. We could use an integer encoding directly, rescaled where needed. This may work for problems where there is a natural ordinal relationship between the categories, and in turn the integer values, such as labels for temperature ‘cold’, warm’, and ‘hot’.

In our data set there are many features are in form of categories like “yes” and “No” form. So we need to apply the one hot encoding concept in our categorical features. It done by the method get_dummies() it make some dummies features and we drop first feature from dummies features which would made by get_dummies() method.

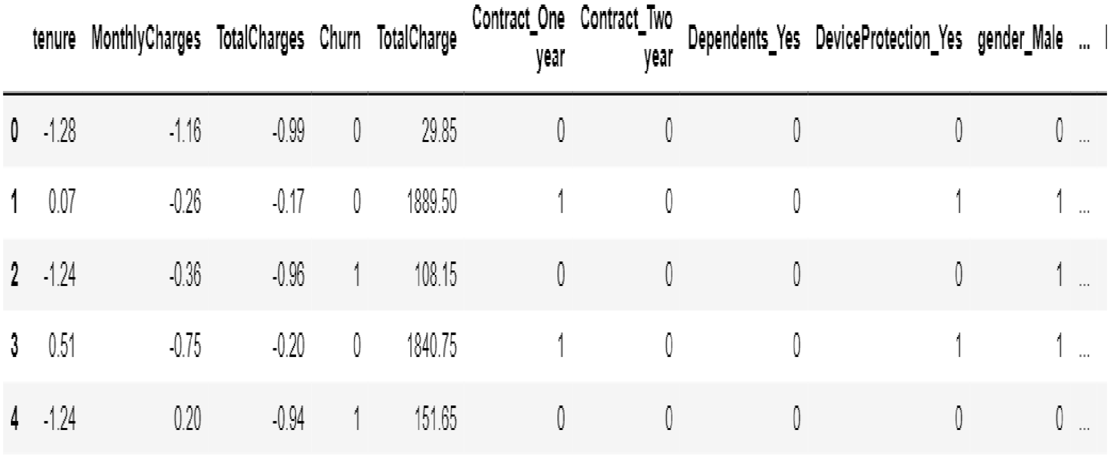


```
In [18]: #perform onehot encoding by the method get_dummies()
         df =pd.get_dummies(df,columns =['Contract','Dependents','DeviceProtection','gender',
                                         'InternetService','MultipleLines','OnlineBackup','OnlineSecurity','PaperlessBilling',
                                         'Partner','PaymentMethod','PhoneService',
                                         'SeniorCitizen','StreamingMovies','StreamingTV','TechSupport'


         ],drop_first =True)
```

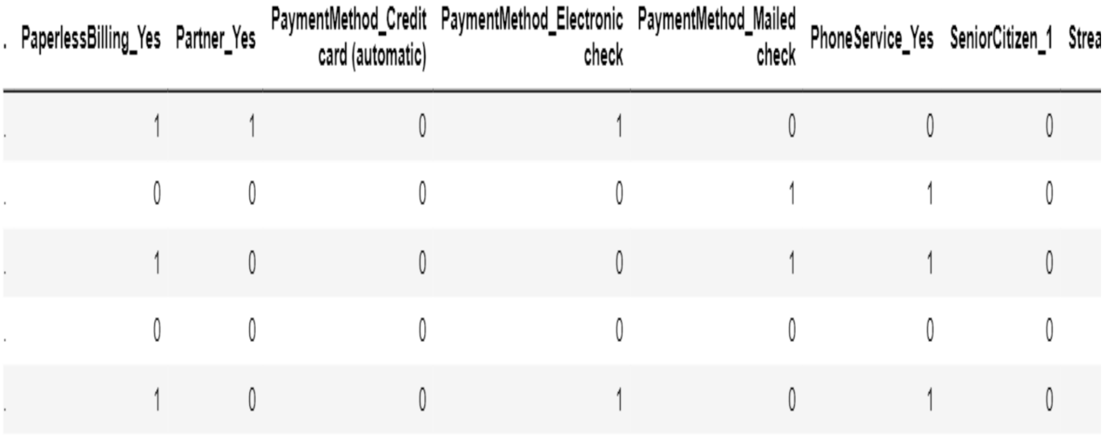
Convert MonthlyCharges and TotalCharges and Tenure in same categorical values range-



```
from sklearn.preprocessing import StandardScaler
stander_Scaler =StandardScaler()
columns_feature_scaling =['MonthlyCharges','TotalCharges','tenure']
df[columns_feature_scaling] =stander_Scaler.fit_transform(df[columns_feature_scaling])
```



	tenure	MonthlyCharges	TotalCharges	Churn	TotalCharge	Contract_One year	Contract_Two year	Dependents_Yes	DeviceProtection_Yes	gender_Male	...	I
0	-1.28	-1.16	-0.99	0	29.85	0	0	0	0	0	...	
1	0.07	-0.26	-0.17	0	1889.50	1	0	0	1	1	...	
2	-1.24	-0.36	-0.96	1	108.15	0	0	0	0	1	...	
3	0.51	-0.75	-0.20	0	1840.75	1	0	0	1	1	...	
4	-1.24	0.20	-0.94	1	151.65	0	0	0	0	0	...	



.	PaperlessBilling_Yes	Partner_Yes	PaymentMethod_Credit card (automatic)	PaymentMethod_Electronic check	PaymentMethod_Mailed check	PhoneService_Yes	SeniorCitizen_1	Strea
.	1	1	0	1	0	0	0	
.	0	0	0	0	1	1	0	
.	1	0	0	0	1	1	0	
.	0	0	0	0	0	0	0	
.	1	0	0	1	0	1	0	

CONCLUSION

- We have finished the data preprocessing, data visualization, remove unnecessary features , and fill all the categorical features string values with some discrete values like 0 or 1. By the help of above steps we make our dataset cleanness.

- further we split our data set in to two part one part is training set and the other part is testing set. After training and testing we feed the training data to the model after trained the model we check the model performance by the help of testing set. Check the model accuracy by the help of different machine learning algorithms like logistic regression , decision tree, random forest, naïve byes, k-nearest neobhour etc.

REFERENCE

- https://www.geeksforgeeks.org/machine-learning/
- https://expertsystem.com/blog/machine-learning/
- https://www.coursera.org/learn/machine-learning
- https://towardsdatascience.com/machine-learning-an-introduction-23b84d51e6d0