

# ASSIGNMENT-02

## DATA VISUALIZATION AND PRE PROCESSING

Assignment Date 22 September 2022
Student Name Boomiha
Student Roll Number 113219071003
Maximum Marks 2 Marks

1. Download the dataset: Dataset  
Dataset downloaded in csv form.

2. Load the dataset.

```
import pandas as pd
df = pd.read_csv("/content/drive/MyDrive/IBM Assignments/Churn_Modelling.csv")
```

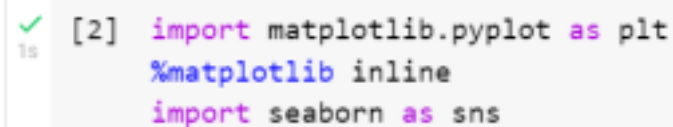
A screenshot of a Jupyter Notebook cell. On the left, there is a green checkmark and the text '1s'. The code in the cell is: 

```
import pandas as pd
df = pd.read_csv("/content/drive/MyDrive/IBM Assignments/Churn_Modelling.csv")
```

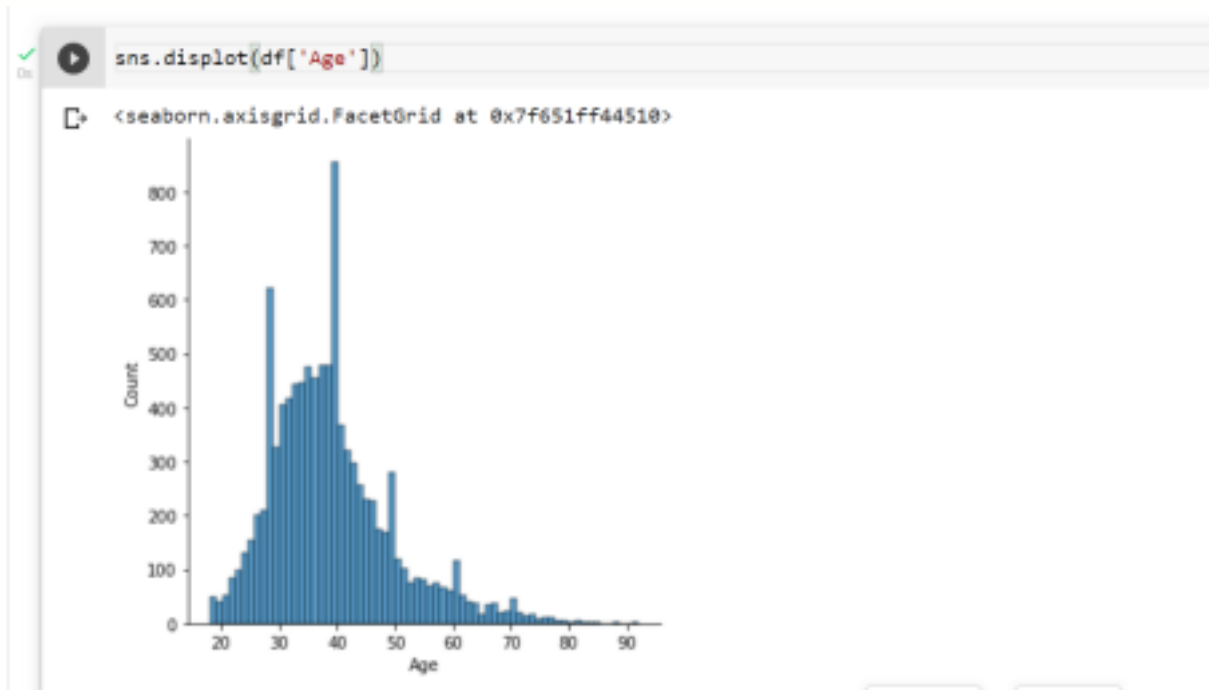
3. Perform Below Visualizations.

- Univariate Analysis

```
sns.displot(df['Age'])
```

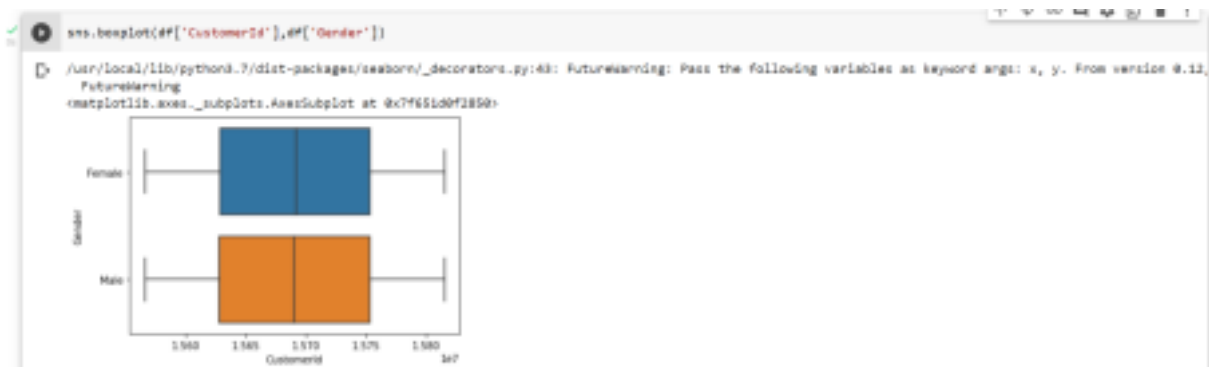
A screenshot of a Jupyter Notebook cell. On the left, there is a green checkmark and the text '1s'. The code in the cell is: 

```
[2] import matplotlib.pyplot as plt
    %matplotlib inline
    import seaborn as sns
```

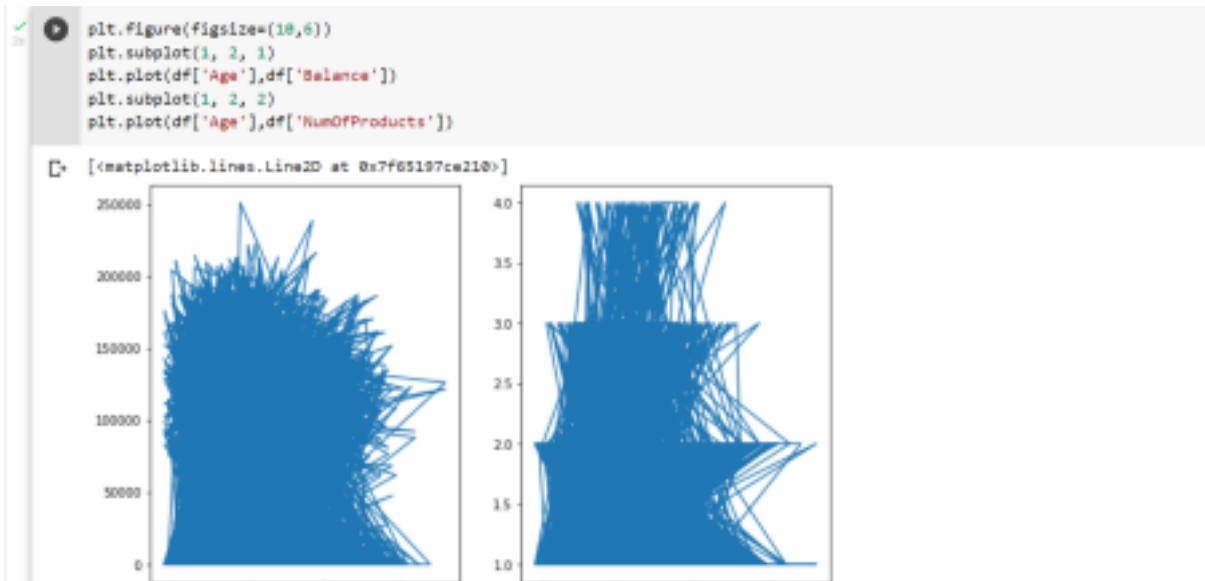


- Bi - Variate Analysis

```
sns.boxplot(df['CustomerId'],df['Gender'])
```



- Multi - Variate Analysis



4. Perform descriptive statistics on the dataset.

```
df.describe()
```

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary
count	10000.00000	1.000000e+04	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000
mean	5000.50000	1.566894e+07	650.528600	38.621800	5.812800	76485.986288	1.530200	0.705580	0.515100	100066.238881
std	2886.89595	7.193819e+04	96.853289	10.487806	2.892174	82397.405202	0.581054	0.455840	0.489797	57510.482818
min	1.00000	1.556670e+07	350.000000	18.000000	0.000000	0.000000	1.000000	0.000000	0.000000	11.580000
25%	2500.75000	1.562853e+07	584.000000	32.000000	3.000000	0.000000	1.000000	0.000000	0.000000	51002.110000
50%	5000.50000	1.566894e+07	652.000000	37.000000	5.000000	87196.540000	1.000000	1.000000	1.000000	100163.915000
75%	7500.25000	1.575323e+07	718.000000	44.000000	7.000000	127944.240000	2.000000	1.000000	1.000000	140388.247500
max	10000.00000	1.581589e+07	850.000000	62.000000	10.000000	250896.090000	4.000000	1.000000	1.000000	199662.480000

Mean:

```
df.mean()
```

```

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only')
"""Entry point for launching an IPython kernel.
RowNumber      5.000589e+01
CustomerId     1.566894e+07
CreditScore    6.505286e+02
Age            3.862180e+01
Tenure         5.812800e+00
Balance        7.648599e+04
NumOfProducts  1.530200e+00
HasCrCard      7.055800e-01
IsActiveMember 5.151000e-01
EstimatedSalary 1.000662e+05
Exited         2.837888e-01
dtype: object

```

5. Handle the Missing values.

```
df.isnull().sum()

RowNumber      0
CustomerId      0
Surname         0
CreditScore     0
Geography      0
Gender          0
Age            0
Tenure         0
Balance        0
NumOfProducts  0
HasCrCard       0
IsActiveMember  0
EstimatedSalary 0
Exited          0
dtype: int64
```

6. Find the outliers and replace the outliers

Finding Outliers:

Using Boxplot



Using method

```
[83] qnt = df.quantile(q=[0.25,0.75])
qnt
```

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
0.25	2500.75	15629528.25	584.0	32.0	3.0	0.00	1.0	0.0	0.0	51062.1100	0.0
0.75	7500.25	15753233.75	718.0	44.0	7.0	127644.24	2.0	1.0	1.0	149388.2475	0.0

```
iqn = qnt.loc[0.75]-qnt.loc[0.25]
iqn
```

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
	4999.5000	124785.5000	134.0000	12.0000	4.0000	127644.2400	1.0000	1.0000	1.0000	98386.1375	0.0000

dtype: float64

```

lower = qnt.loc[0.25]-1.5*Iqr
print("Lower bound:",lower)
upper = qnt.loc[0.75]+1.5*Iqr
print("Upper bound:",upper)

Lower bound: RowNumber      -4.988588e+03
CustomerId      1.594147e+07
CreditScore      3.838888e+02
Age      1.488888e+01
Tenure      -3.888888e+00
Balance      -1.914664e+05
NumOfProducts      -5.000000e-01
HasCrCard      -1.500000e+00
IsActiveMember      -1.500000e+00
EstimatedSalary      -9.657710e+04
Exited      0.000000e+00
dtype: float64
Upper bound: RowNumber      1.499950e+04
CustomerId      1.594029e+07
CreditScore      9.198888e+02
Age      6.200000e+01
Tenure      1.388888e+01
Balance      3.191106e+05
NumOfProducts      3.500000e+00
HasCrCard      2.500000e+00
IsActiveMember      2.500000e+00
EstimatedSalary      2.968675e+05
Exited      0.000000e+00
dtype: float64

```

Replacing Outliers:

```

''' replacing outliers '''
df['Balance'] = np.where(df['Balance']>127644,0.00,df['Balance'])

```

7. Check for Categorical columns and perform encoding.

Categorical columns: Geography,Gender

```

[98] from sklearn.preprocessing import LabelEncoder
labelencoder_df = LabelEncoder()
df['Geography'] = labelencoder_df.fit_transform(df['Geography'])

df.head()

```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
0	1	15634602	Hargrave	619	0	Female	42	2	0.00	1	1	1	101348.88	
1	2	15647311	Hill	608	2	Female	41	1	83807.86	1	0	1	112542.58	
2	3	15619304	Onio	502	0	Female	42	8	0.00	3	1	0	113831.57	
3	4	15701354	Boni	689	0	Female	39	1	0.00	2	0	0	93826.63	
4	5	15737888	Michell	850	2	Female	43	2	125510.82	1	1	1	79084.10	

```

df['Gender'] = labelencoder_df.fit_transform(df['Gender'])

df.head(7)

```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
0	1	15634602	Hargrave	619	0	0	42	2	0.00	1	1	1	101348.88	
1	2	15647311	Hill	608	2	0	41	1	83807.86	1	0	1	112542.58	
2	3	15619304	Onio	502	0	0	42	8	0.00	3	1	0	113831.57	
3	4	15701354	Boni	689	0	0	39	1	0.00	2	0	0	93826.63	
4	5	15737888	Michell	850	2	0	43	2	125510.82	1	1	1	79084.10	
5	6	15574012	Chu	645	2	1	44	8	113795.78	2	1	0	146756.71	
6	7	15602531	Berkett	622	0	1	50	7	0.00	2	1	1	10682.80	

8. Split the data into dependent and independent variables.



9. Scale the independent variables



10. Split the data into training and testing

