ASSIGNMENT-02

DATA VISUALIZATION AND PRE PROCESSING

Assignment Date	22 September 2022
Student Name	SAHANA J M
Student Roll Number	113219071033
Maximum Marks	2 Marks

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

2. Load the dataset.

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

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

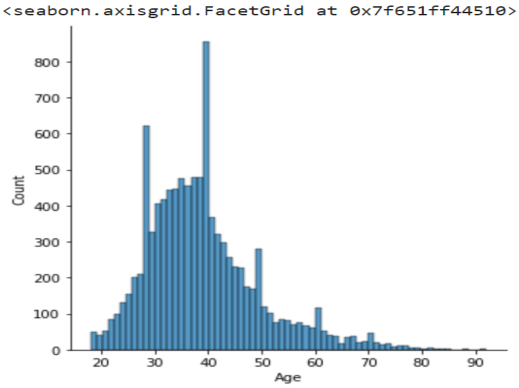
3. Perform Below Visualizations.

● Univariate Analysis – deals with single column

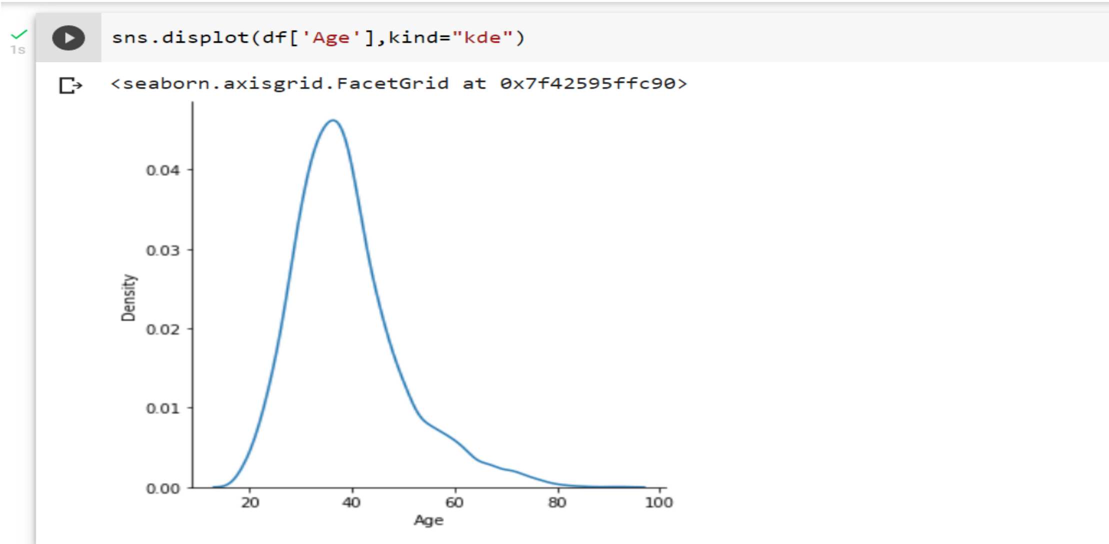
```
[2] import matplotlib.pyplot as plt
    %matplotlib inline
    import seaborn as sns
```

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

```
sns.displot(df['Age'])
&lt;seaborn.axisgrid.FacetGrid at 0x7f651ff44510&gt;
```

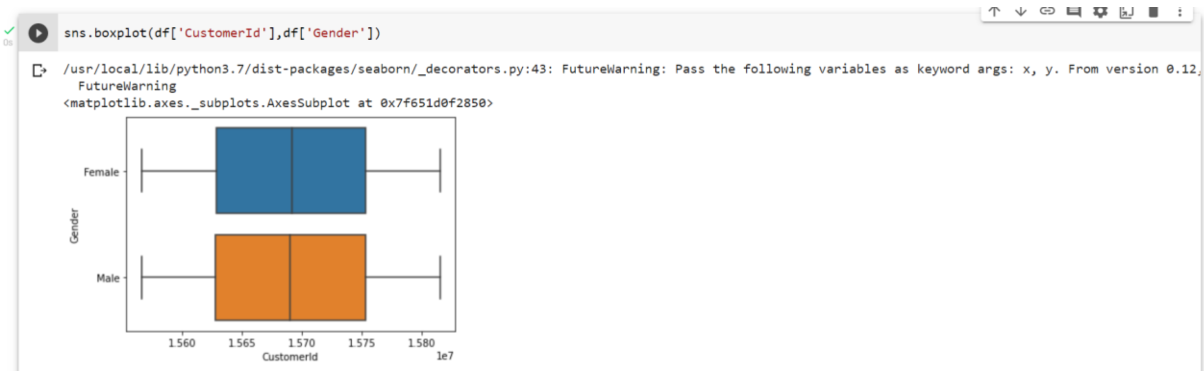


```
sns.displot(df['Age'],kind="kde")
```

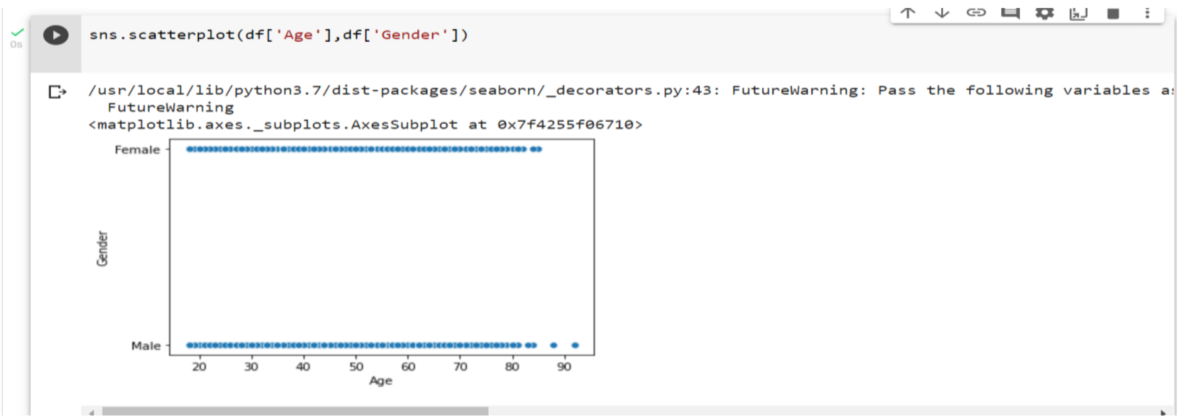


● Bi - Variate Analysis – deals with 2 columns of data

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



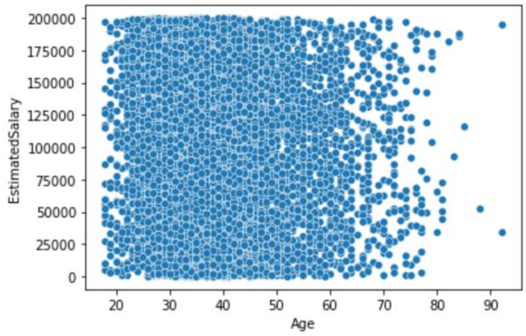

```
sns.scatterplot(df['Age'],df['Gender'])
```



```
sns.scatterplot(df['Age'],df['EstimatedSalary'])
```

```
sns.scatterplot(df['Age'],df['EstimatedSalary'])
```

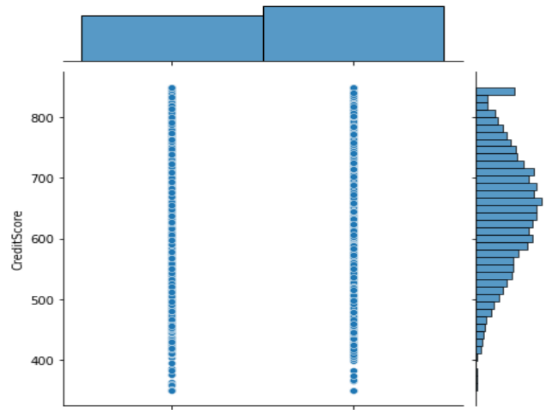
```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables a
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f4255dd5890>
```




```
sns.jointplot(df['Gender'],df['CreditScore'])
```

```
sns.jointplot(df['Gender'],df['CreditScore'])
```

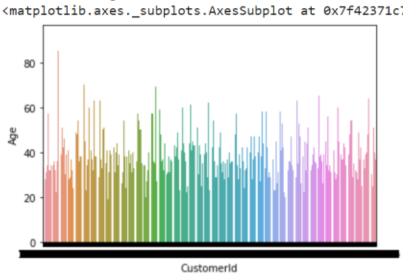
```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables a
  FutureWarning
<seaborn.axisgrid.JointGrid at 0x7f4253ee0750>
```




```
sns.barplot(df['CustomerId'],df['Age'])
```

```
sns.barplot(df['CustomerId'],df['Age'])
```

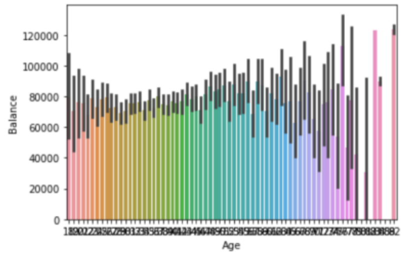
```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.1
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f42371c7a90>
```



```
sns.barplot(df['Age'],df['Balance'])
```

```
sns.barplot(df['Age'],df['Balance'])
```

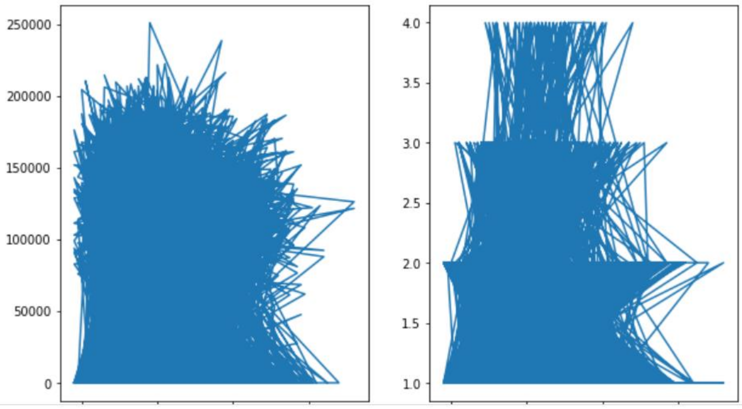
```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f421f34f050>
```



● Multi - Variate Analysis – deals with multiple columns

```
plt.figure(figsize=(10,6))
plt.subplot(1, 2, 1)
plt.plot(df['Age'],df['Balance'])
plt.subplot(1, 2, 2)
plt.plot(df['Age'],df['NumOfProducts'])
```

```
[<matplotlib.lines.Line2D at 0x7f65197ce210>]
```



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	10000.00000	10000.000000	10000.000000
mean	5000.50000	1.569094e+07	650.528800	38.921800	5.012800	76485.889288	1.530200	0.70550	0.515100	100090.239881
std	2886.89568	7.193619e+04	96.653299	10.487806	2.892174	62397.405202	0.581654	0.45584	0.499797	57510.492818
min	1.00000	1.556570e+07	350.000000	18.000000	0.000000	0.000000	1.000000	0.00000	0.000000	11.580000
25%	2500.75000	1.562853e+07	584.000000	32.000000	3.000000	0.000000	1.000000	0.00000	0.000000	51002.110000
50%	5000.50000	1.569074e+07	652.000000	37.000000	5.000000	97198.540000	1.000000	1.00000	1.000000	100193.915000
75%	7500.25000	1.575323e+07	718.000000	44.000000	7.000000	127644.240000	2.000000	1.00000	1.000000	149388.247500
max	10000.00000	1.581569e+07	850.000000	92.000000	10.000000	250898.090000	4.000000	1.00000	1.000000	199992.480000

Mean : df.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          5.000500e+03
CustomerId         1.569094e+07
CreditScore        6.505288e+02
Age                3.892180e+01
Tenure             5.012800e+00
Balance            7.648589e+04
NumOfProducts      1.530200e+00
HasCrCard          7.055000e-01
IsActiveMember     5.151000e-01
EstimatedSalary    1.000902e+05
Exited             2.037000e-01
dtype: float64
```

Median: df.median()

```
df.median()

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_onl
  """Entry point for launching an IPython kernel.
RowNumber          5.000500e+03
CustomerId         1.569074e+07
CreditScore        6.520000e+02
Age                3.700000e+01
Tenure             5.000000e+00
Balance            9.719854e+04
NumOfProducts      1.000000e+00
HasCrCard          1.000000e+00
IsActiveMember     1.000000e+00
EstimatedSalary    1.001939e+05
Exited             0.000000e+00
dtype: float64
```

Standard Deviation: df.std()

```
df.std()

/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           2886.895680
CustomerId         71936.186123
CreditScore           96.653299
Age                   10.487806
Tenure                 2.892174
Balance            62397.405202
NumOfProducts          0.581654
HasCrCard              0.455840
IsActiveMember         0.499797
EstimatedSalary    57510.492818
Exited                 0.402769
dtype: float64
```

5. Handle the Missing values.

df.isnull().sum()

```
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
```

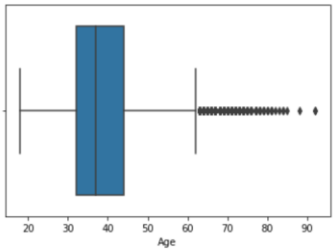
6. Find the outliers and replace the outliers

Outliers: The values which are different from others or less relative to others.

Using Boxplot

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

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f651c4bcfd0>
```



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	15628528.25	584.0	32.0	3.0	0.00	1.0	0.0	0.0	51002.1100	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

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

```
RowNumber            4999.5000
CustomerId         124705.5000
CreditScore           134.0000
Age                    12.0000
Tenure                  4.0000
Balance            127644.2400
NumOfProducts           1.0000
HasCrCard               1.0000
IsActiveMember          1.0000
EstimatedSalary     98386.1375
Exited                  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         -4.998500e+03
CustomerId          1.544147e+07
CreditScore         3.830000e+02
Age                 1.400000e+01
Tenure             -3.000000e+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         9.190000e+02
Age                 6.200000e+01
Tenure              1.300000e+01
Balance             3.191106e+05
NumOfProducts       3.500000e+00
HasCrCard           2.500000e+00
IsActiveMember      2.500000e+00
EstimatedSalary     2.969675e+05
Exited              0.000000e+00
dtype: float64
```

Replacing Outliers:

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

```
''' 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

Encoding changes the values to numerical forms such as 0,1

```
[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	Ex
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	113931.57	
3	4	15701354	Boni	699	0	Female	39	1	0.00	2	0	0	93826.63	
4	5	15737888	Mitchell	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	E
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	113931.57	
3	4	15701354	Boni	699	0	0	39	1	0.00	2	0	0	93826.63	
4	5	15737888	Mitchell	850	2	0	43	2	125510.82	1	1	1	79084.10	
5	6	15574012	Chu	645	2	1	44	8	113755.78	2	1	0	149756.71	
6	7	15592531	Bartlett	822	0	1	50	7	0.00	2	1	1	10062.80	

8. Split the data into dependent and independent variables.

```
[105] X = df.iloc[:, :-1].values
      print(X)

[[1 15634602 'Hargrave' ... 1 1 101348.88]
 [2 15647311 'Hill' ... 0 1 112542.58]
 [3 15619304 'Onio' ... 1 0 113931.57]
 ...
 [9998 15584532 'Liu' ... 0 1 42085.58]
 [9999 15682355 'Sabbatini' ... 1 0 92888.52]
 [10000 15628319 'Walker' ... 1 0 38190.78]]
```

```
Y = df.iloc[:, -1].values
print(Y)

[1 0 1 ... 1 1 0]
```

9. Scale the independent variables

```
[115] from sklearn.preprocessing import scale
      Y = scale(Y)
```

```
Y
```

```
array([ 1.97716468, -0.50577476,  1.97716468, ...,  1.97716468,
        1.97716468, -0.50577476])
```

10. Split the data into training and testing

```
Y_train
```

```
array([-0.50577476, -0.50577476, -0.50577476, ..., -0.50577476,
       -0.50577476,  1.97716468])
```

```
Y_test
```

```
array([-0.50577476,  1.97716468, -0.50577476, ..., -0.50577476,
       -0.50577476, -0.50577476])
```

```
X_train
```

```
array([[7390, 15676909, 'Mishin', ..., 1, 0, 163830.64],
       [9276, 15749265, 'Carslaw', ..., 1, 1, 57098.0],
       [2996, 15582492, 'Moore', ..., 1, 0, 185630.76],
       ...,
       [3265, 15574372, 'Hoolan', ..., 1, 0, 181429.87],
       [9846, 15664035, 'Parsons', ..., 1, 1, 148750.16],
       [2733, 15592816, 'Udokamma', ..., 1, 0, 118855.26]], dtype=object)
```

```
X_test
```

```
array([[9395, 15615753, 'Upchurch', ..., 1, 1, 192852.67],
       [899, 15654700, 'Fallaci', ..., 1, 0, 128702.1],
       [2399, 15633877, 'Morrison', ..., 1, 1, 75732.25],
       ...,
       [9550, 15772604, 'Chiemezie', ..., 1, 0, 141533.19],
       [2741, 15787699, 'Burke', ..., 1, 1, 11276.48],
       [6691, 15579223, 'Niu', ..., 1, 0, 192950.6]], dtype=object)
```