Assignment -2
Python Programming

Assignment Date	**25 September 2022**
Student Name	**ABIMANYU**
Student Roll Number	**720819106002**
Maximum Marks	**2 Marks**

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
```

2.Loading the data Set

```
df=pd.read_csv("Churn_Modelling.csv")
df
```

```
      RowNumber  CustomerId    Surname  CreditScore Geography  Gender  Age  \
0             1    15634602   Hargrave          619    France  Female   42
1             2    15647311       Hill          608     Spain  Female   41
2             3    15619304       Onio          502    France  Female   42
3             4    15701354       Boni          699    France  Female   39
4             5    15737888   Mitchell          850     Spain  Female   43
...         ...         ...        ...          ...       ...     ...  ...
9995       9996    15606229   Obijiaku          771    France    Male   39
9996       9997    15569892  Johnstone          516    France    Male   35
9997       9998    15584532        Liu          709    France  Female   36
9998       9999    15682355  Sabbatini          772   Germany    Male   42
9999      10000    15628319     Walker          792    France  Female   28

      Tenure    Balance  NumOfProducts  HasCrCard  IsActiveMember  \
0          2       0.00              1          1               1
1          1   83807.86              1          0               1
2          8  159660.80              3          1               0
3          1       0.00              2          0               0
4          2  125510.82              1          1               1
...      ...        ...            ...        ...             ...
9995       5       0.00              2          1               0
9996      10   57369.61              1          1               1
9997       7       0.00              1          0               1
```

```
9998       3   75075.31              2          1               0
9999       4  130142.79              1          1               0

      EstimatedSalary  Exited
0           101348.88       1
1           112542.58       0
2           113931.57       1
3            93826.63       0
4            79084.10       0
...               ...     ...
9995         96270.64       0
9996        101699.77       0
9997         42085.58       1
9998         92888.52       1
9999         38190.78       0

[10000 rows x 14 columns]
```

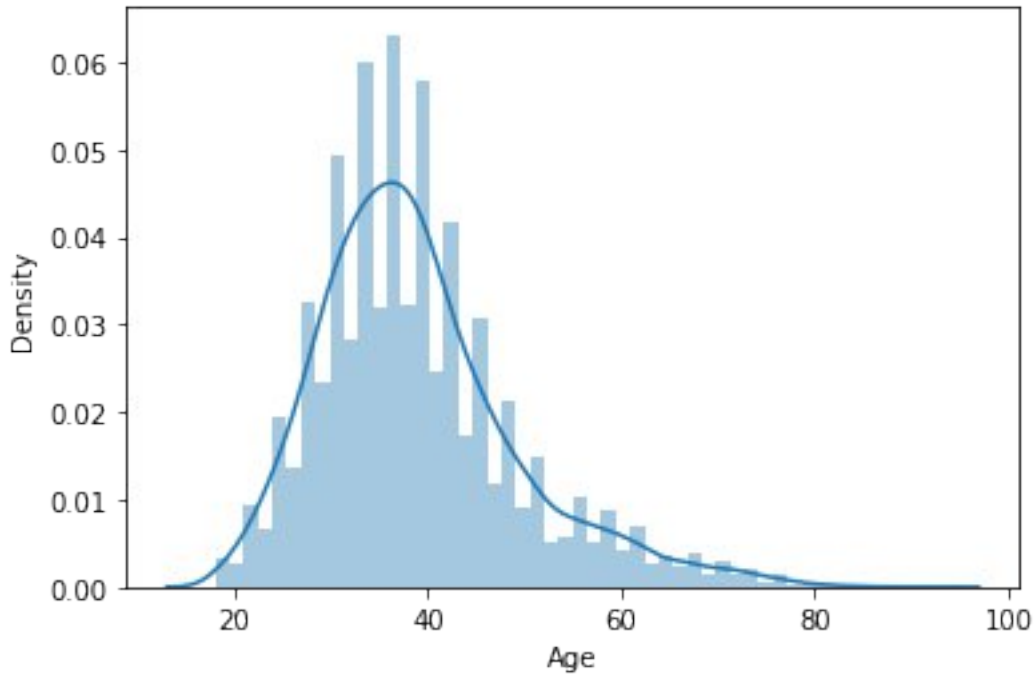
3.Visulaizatoin

3.1 Univariate Analysis

```
sns.distplot(df.Age)
```

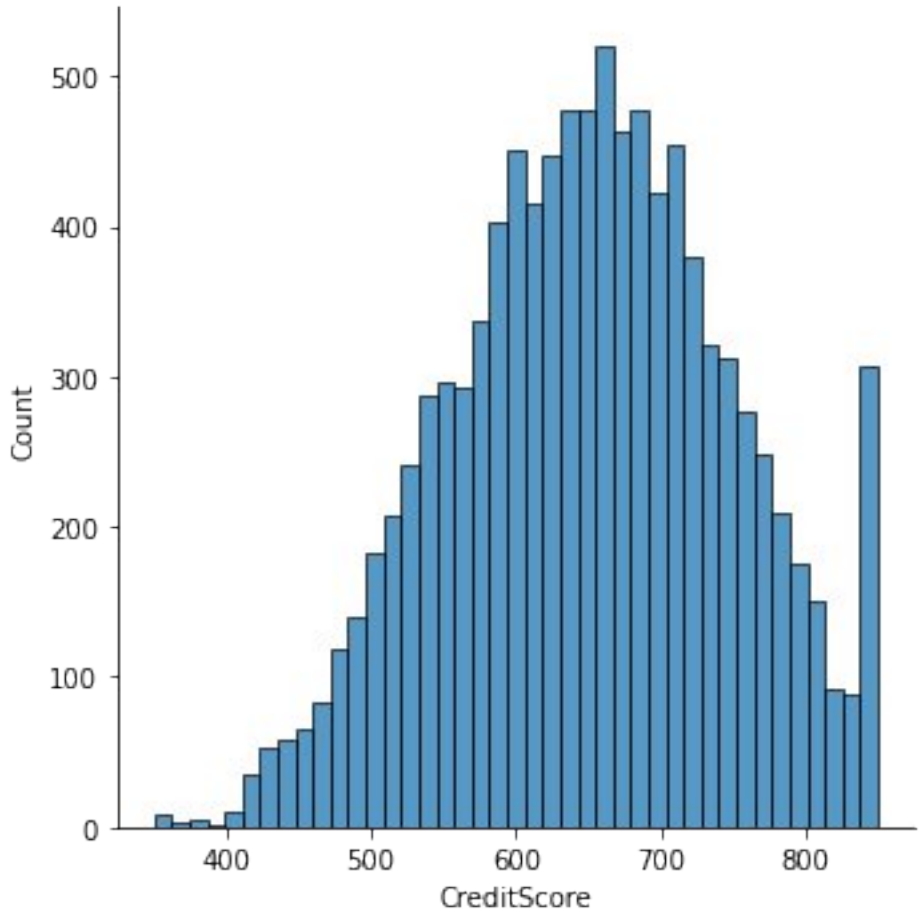
```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2619:
FutureWarning: `distplot` is a deprecated function and will be removed in a
future version. Please adapt your code to use either `displot` (a figure-
level function with similar flexibility) or `histplot` (an axes-level
function for histograms).
  warnings.warn(msg, FutureWarning)
```

```
<AxesSubplot:xlabel='Age', ylabel='Density'>
```



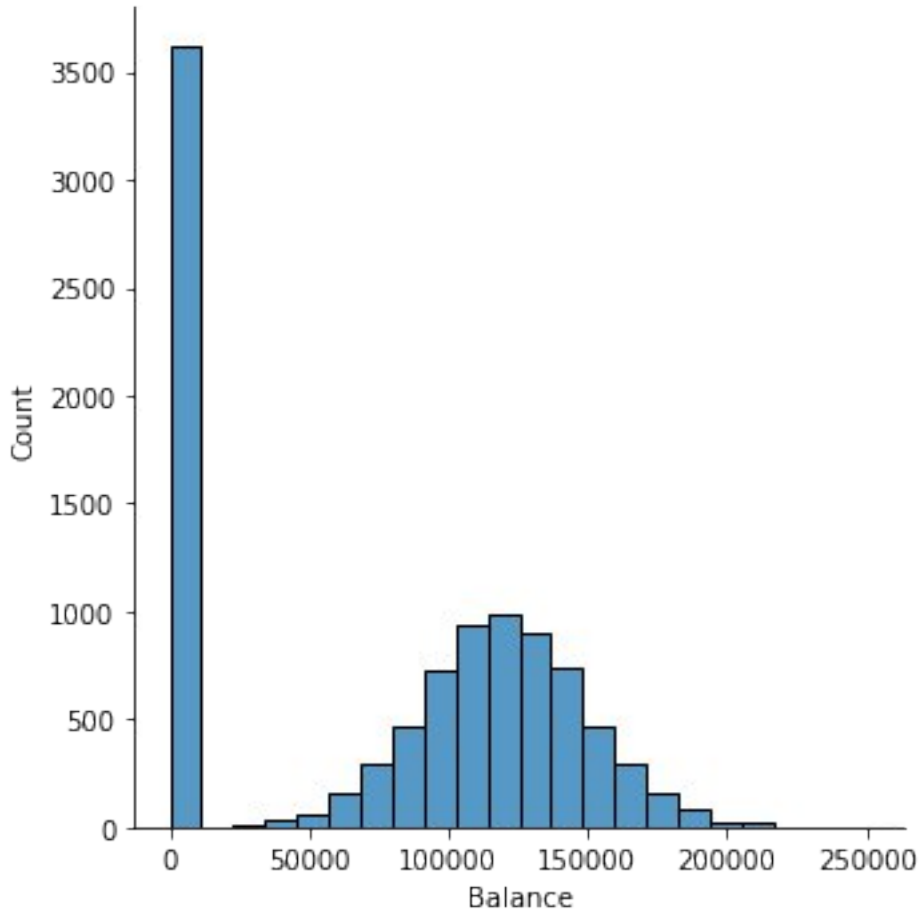

```
sns.displot(df.CreditScore)
```

```
<seaborn.axisgrid.FacetGrid at 0x1d3d827f550>
```




```
sns.displot(df.Balance)
```

```
<seaborn.axisgrid.FacetGrid at 0x1d3d40c6700>
```

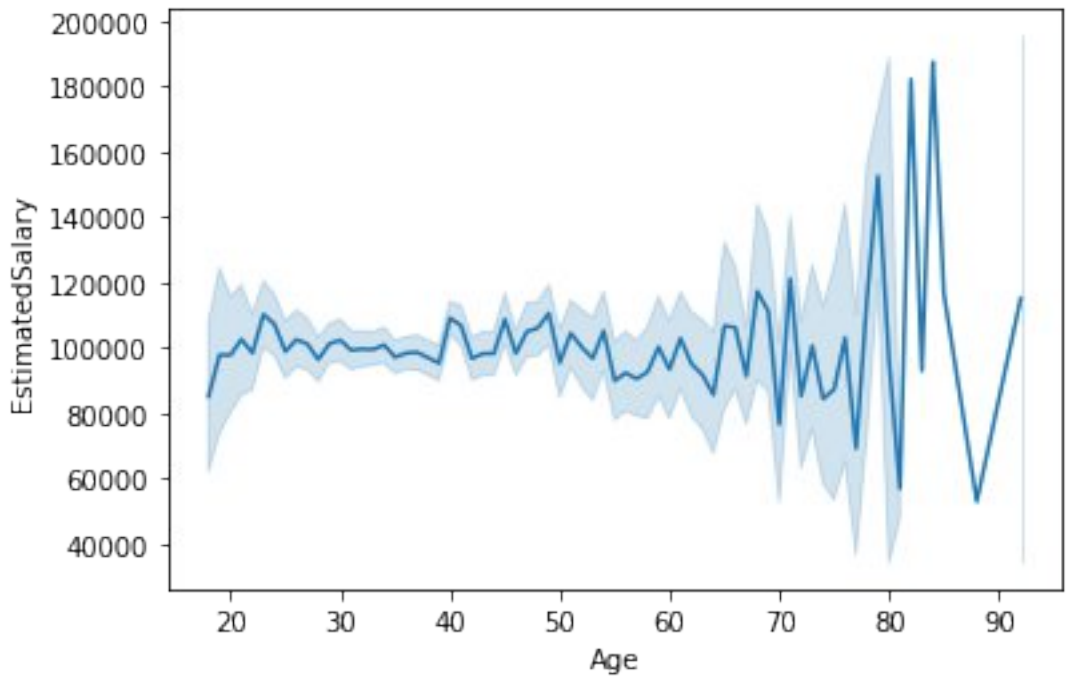


3.2 Bi - Variate analysis

```
sns.lineplot(df.Age,df.EstimatedSalary)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From
version 0.12, the only valid positional argument will be `data`, and passing
other arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

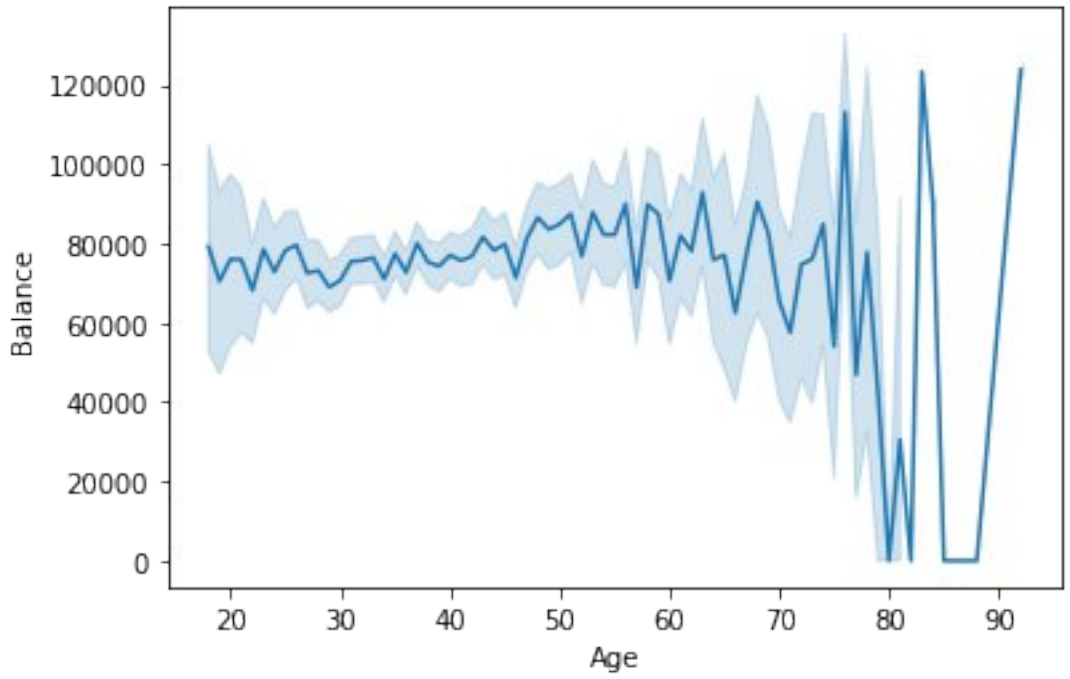
```
<AxesSubplot:xlabel='Age', ylabel='EstimatedSalary'>
```




```
sns.lineplot(df.Age,df.Balance)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From
version 0.12, the only valid positional argument will be `data`, and passing
other arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

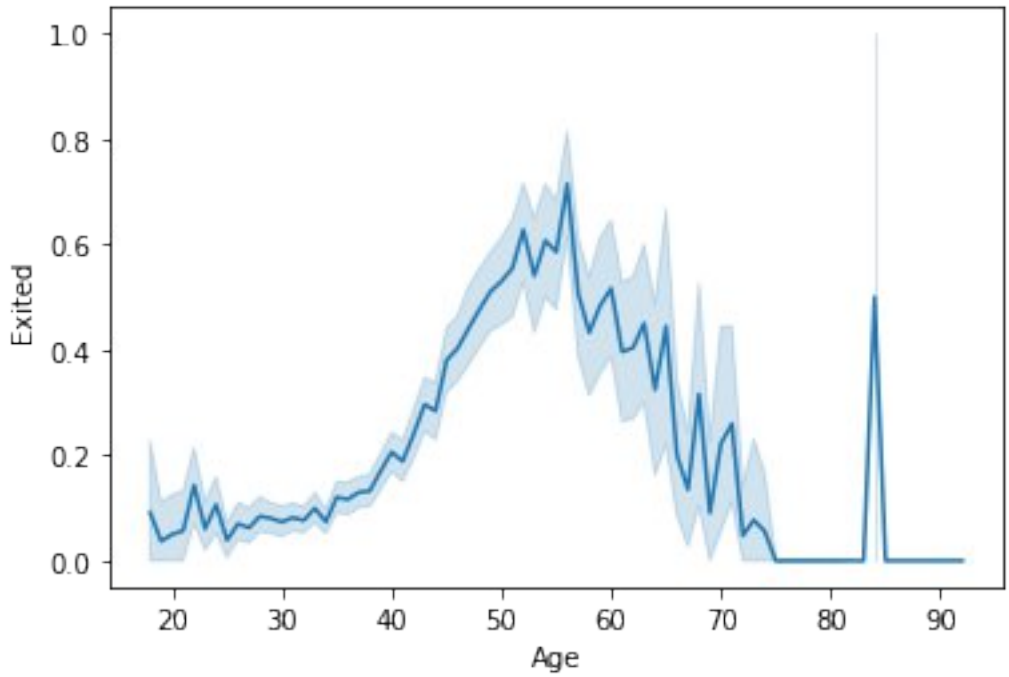
```
<AxesSubplot:xlabel='Age', ylabel='Balance'>
```



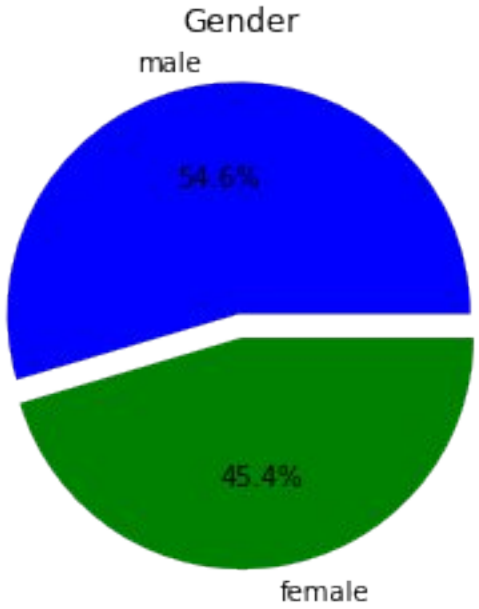

```
sns.lineplot(df.Age,df.Exited)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From
version 0.12, the only valid positional argument will be `data`, and passing
other arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

```
<AxesSubplot:xlabel='Age', ylabel='Exited'>
```



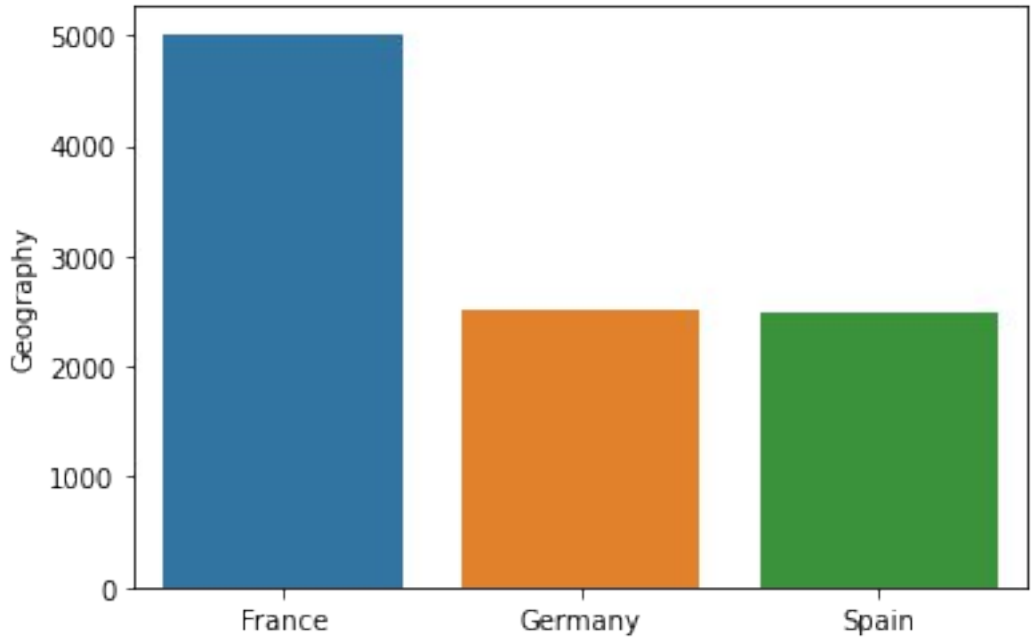

```
plt.pie(df.Gender.value_counts(),[0.1,0],labels=["male","female"],autopct="%1
.1f%%",colors=["blue","green"])
plt.title("Gender")
plt.show()
```




```
sns.barplot(df.Geography.value_counts().index,df.Geography.value_counts())
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From
version 0.12, the only valid positional argument will be `data`, and passing
other arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

```
<AxesSubplot:ylabel='Geography'>
```



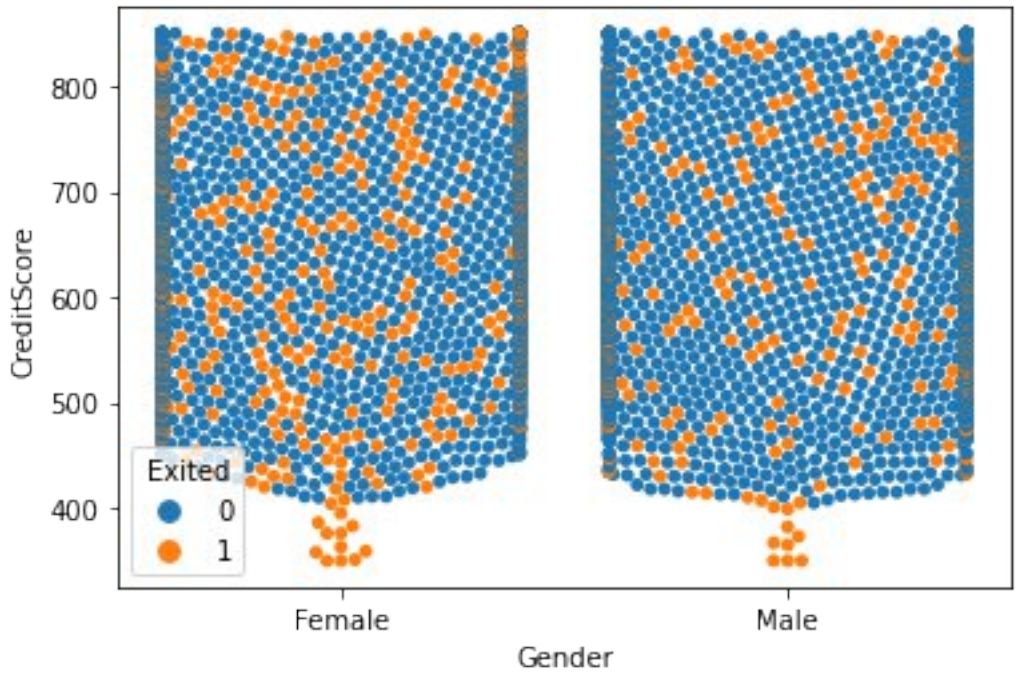
3.3 Multi-Variate analysis

```
sns.swarmplot(df['Gender'], df['CreditScore'], hue = df['Exited'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From
version 0.12, the only valid positional argument will be `data`, and passing
other arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\categorical.py:1296:
UserWarning: 80.0% of the points cannot be placed; you may want to decrease
the size of the markers or use stripplot.
```

```
  warnings.warn(msg, UserWarning)
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\categorical.py:1296:
UserWarning: 83.1% of the points cannot be placed; you may want to decrease
the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
```

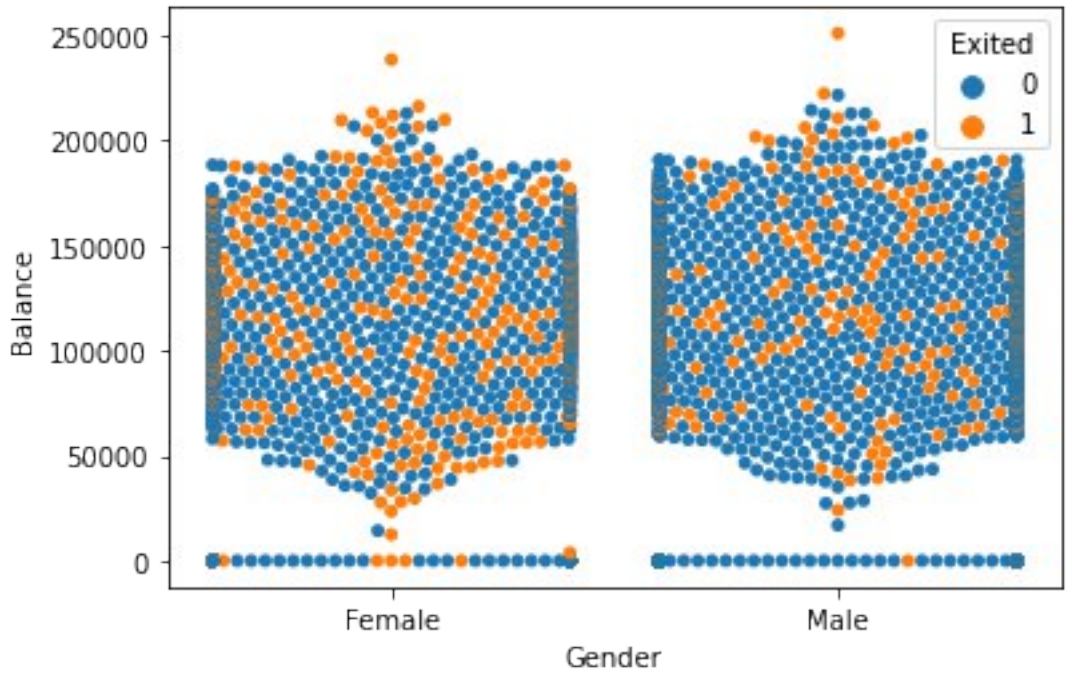
```
<AxesSubplot:xlabel='Gender', ylabel='CreditScore'>
```




```
sns.swarmplot(df['Gender'], df['Balance'], hue = df['Exited'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variables as keyword args: x, y. From
version 0.12, the only valid positional argument will be `data`, and passing
other arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\categorical.py:1296:
UserWarning: 85.1% of the points cannot be placed; you may want to decrease
the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\categorical.py:1296:
UserWarning: 87.3% of the points cannot be placed; you may want to decrease
the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
```

```
<AxesSubplot:xlabel='Gender', ylabel='Balance'>
```



4.Perform the discriptive Statistics

```
df.describe()
```

```
          RowNumber    CustomerId   CreditScore           Age        Tenure  \
count  10000.00000  1.000000e+04  10000.000000  10000.000000  10000.000000
mean    5000.50000  1.569094e+07    650.528800     38.921800      5.012800
std     2886.89568  7.193619e+04     96.653299     10.487806      2.892174
min        1.00000  1.556570e+07    350.000000     18.000000      0.000000
25%     2500.75000  1.562853e+07    584.000000     32.000000      3.000000
50%     5000.50000  1.569074e+07    652.000000     37.000000      5.000000
75%     7500.25000  1.575323e+07    718.000000     44.000000      7.000000
max    10000.00000  1.581569e+07    850.000000     92.000000     10.000000

              Balance  NumOfProducts    HasCrCard  IsActiveMember  \
count   10000.000000   10000.000000  10000.00000    10000.000000
mean    76485.889288       1.530200      0.70550        0.515100
std     62397.405202       0.581654      0.45584        0.499797
min         0.000000       1.000000      0.00000        0.000000
25%         0.000000       1.000000      0.00000        0.000000
50%     97198.540000       1.000000      1.00000        1.000000
75%    127644.240000       2.000000      1.00000        1.000000
max    250898.090000       4.000000      1.00000        1.000000

       EstimatedSalary        Exited
count     10000.000000  10000.000000
```

```
mean      100090.239881      0.203700
std        57510.492818      0.402769
min           11.580000      0.000000
25%        51002.110000      0.000000
50%       100193.915000      0.000000
75%       149388.247500      0.000000
max       199992.480000      1.000000
```

5.Handling the missing Values

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

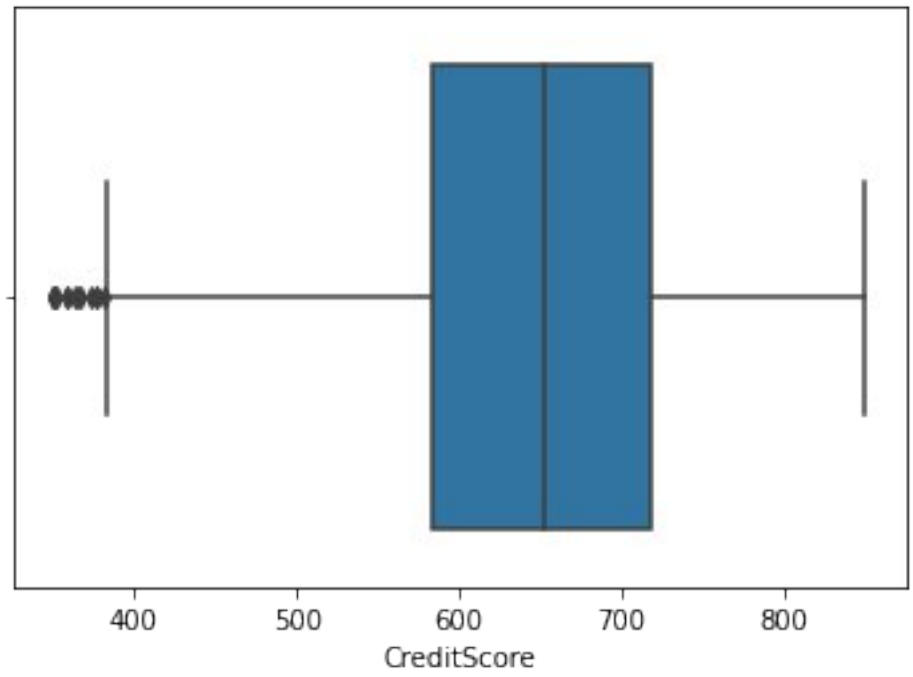
```
RowNumber          0
CustomerId         0
Surname            0
CreditScore        0
Geography          0
Gender             0
Age                0
Tenure             0
Balance            0
NumOfProducts      0
HasCrCard          0
IsActiveMember     0
EstimatedSalary    0
Exited             0
dtype: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

6.Finding Outliers

```
sns.boxplot(df.CreditScore)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

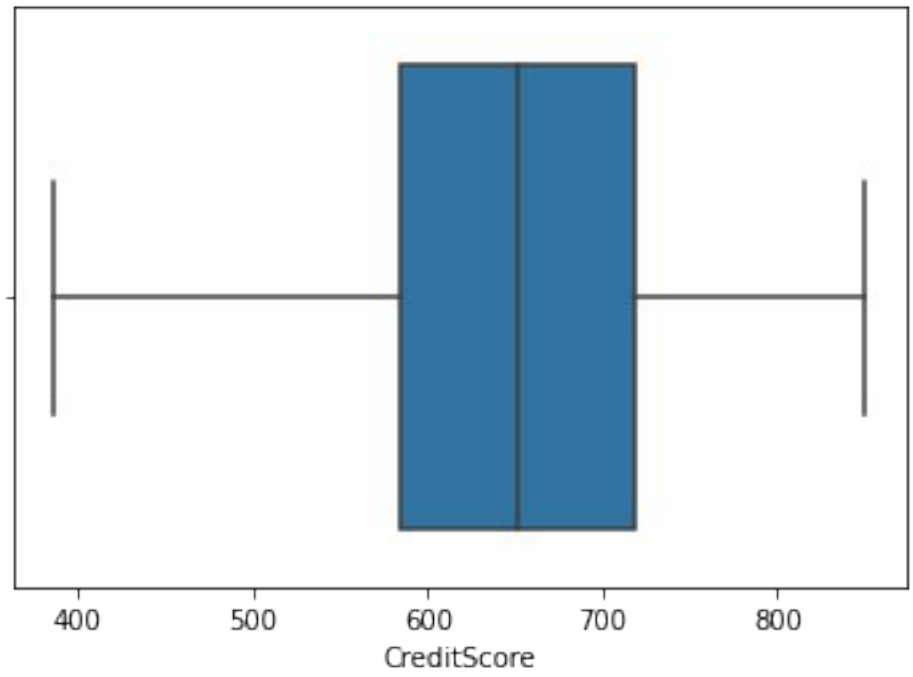
```
<AxesSubplot:xlabel='CreditScore'>
```




```
q1_A=df.CreditScore.quantile(0.25)
q3_A=df.CreditScore.quantile(0.75)
IQR_A=q3_A-q1_A

upper_limit_A=q3_A+1.5*IQR_A
lower_limit_A=q1_A-1.5*IQR_A
```

Replacing outliers for Creditscore

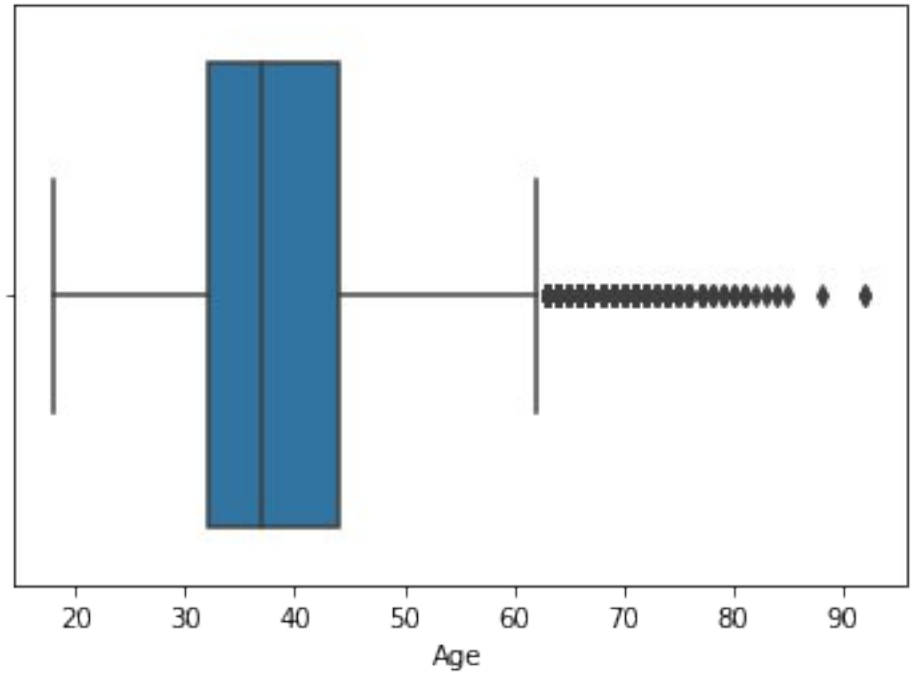
```
df['CreditScore']=np.where(df.CreditScore<=lower_limit_A,df.CreditScore.media
n(),df.CreditScore)

sns.boxplot(df.CreditScore)

C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(

<AxesSubplot:xlabel='CreditScore'>
```




```
# for Age Column
sns.boxplot(df.Age)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

```
<AxesSubplot:xlabel='Age'>
```




```
a99=df.Age.quantile(0.94)
a99
```

```
58.0
```

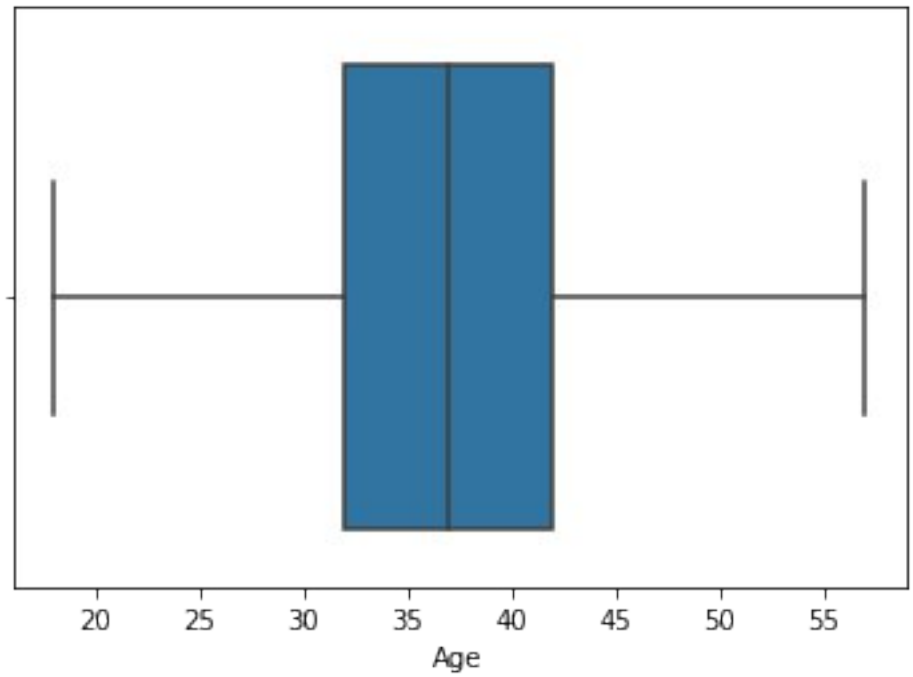
replacing outliers for age

```
df["Age"]=np.where(df.Age>=a99,df.Age.median(),df.Age)
```

```
sns.boxplot(df.Age)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

```
<AxesSubplot:xlabel='Age'>
```



7. Check for Categorical columns and perform encoding

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
df.Gender=le.fit_transform(df.Gender)
df.Geography=le.fit_transform(df.Geography)
```

```
df.head()
```

```
   RowNumber  CustomerId   Surname  CreditScore  Geography  Gender   Age  \
0          1    15634602  Hargrave        619.0          0       0  42.0
1          2    15647311      Hill        608.0          2       0  41.0
2          3    15619304      Onio        502.0          0       0  42.0
3          4    15701354      Boni        699.0          0       0  39.0
4          5    15737888  Mitchell        850.0          2       0  43.0

   Tenure    Balance  NumOfProducts  HasCrCard  IsActiveMember  \
0       2       0.00              1          1               1
1       1   83807.86              1          0               1
2       8  159660.80              3          1               0
3       1       0.00              2          0               0
4       2  125510.82              1          1               1

   EstimatedSalary  Exited
0        101348.88       1
1        112542.58       0
```

```
2        113931.57       1
3         93826.63       0
4         79084.10       0
```

8. Split the data into dependent and independent variables

```
x=df.drop(["RowNumber","CustomerId","Surname"],axis="columns")
x
```

```
      CreditScore  Geography  Gender   Age  Tenure    Balance  NumOfProducts
\
0           619.0          0       0  42.0       2       0.00              1
1           608.0          2       0  41.0       1   83807.86              1
2           502.0          0       0  42.0       8  159660.80              3
3           699.0          0       0  39.0       1       0.00              2
4           850.0          2       0  43.0       2  125510.82              1
...           ...        ...     ...   ...     ...        ...            ...
9995        771.0          0       1  39.0       5       0.00              2
9996        516.0          0       1  35.0      10   57369.61              1
9997        709.0          0       0  36.0       7       0.00              1
9998        772.0          1       1  42.0       3   75075.31              2
9999        792.0          0       0  28.0       4  130142.79              1

      HasCrCard  IsActiveMember  EstimatedSalary  Exited
0             1               1        101348.88       1
1             0               1        112542.58       0
2             1               0        113931.57       1
3             0               0         93826.63       0
4             1               1         79084.10       0
...         ...             ...              ...     ...
9995          1               0         96270.64       0
9996          1               1        101699.77       0
9997          0               1         42085.58       1
9998          1               0         92888.52       1
9999          1               0         38190.78       0

[10000 rows x 11 columns]
```

```
y=df.Exited
y
```

```
0       1
1       0
2       1
3       0
4       0
       ..
9995    0
9996    0
9997    1
9998    1
9999    0
Name: Exited, Length: 10000, dtype: int64
```

9. Scale the independent variables

```
from sklearn.preprocessing import scale
x_scaled=pd.DataFrame(scale(x),columns=x.columns)
```

10. Split the data into training and testing

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x_scaled,y,test_size=0.3,rando
m_state=10)
```

```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
rfc.score(x_test,y_test)
```

```
1.0
```