ASSIGNMENT - 2

Downloaded the given dataset

```
import pandas as pd
import seaborn as sns
import numpy as np
from matplotlib import pyplot as plt
%matplotlib inline
```

uploaded the given dataset

```
url = 'https://drive.google.com/file/d/160K6XcuYDyRBPGj-
JsqThkyFoJhCvOWy/view?usp=sharing'
path = 'https://drive.google.com/uc?
export=download&id='+url.split('/')[-2]
df= pd.read_csv(path)

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
```

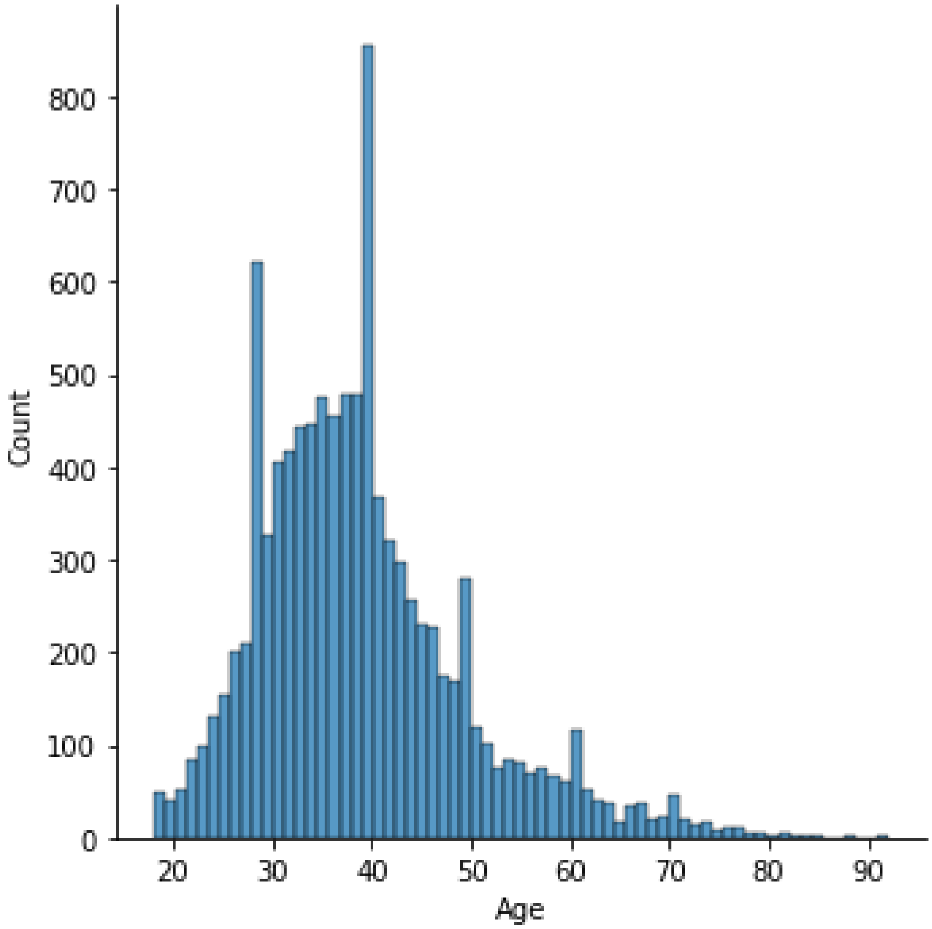
```
...      ...        ...            ...        ...             ...
9995       5       0.00              2          1               0
9996      10   57369.61              1          1               1
9997       7       0.00              1          0               1
9998       3   75075.31              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]
```

perform below visualizations.

1.Univariate Analysis

```
sns.displot(df.Age)
```

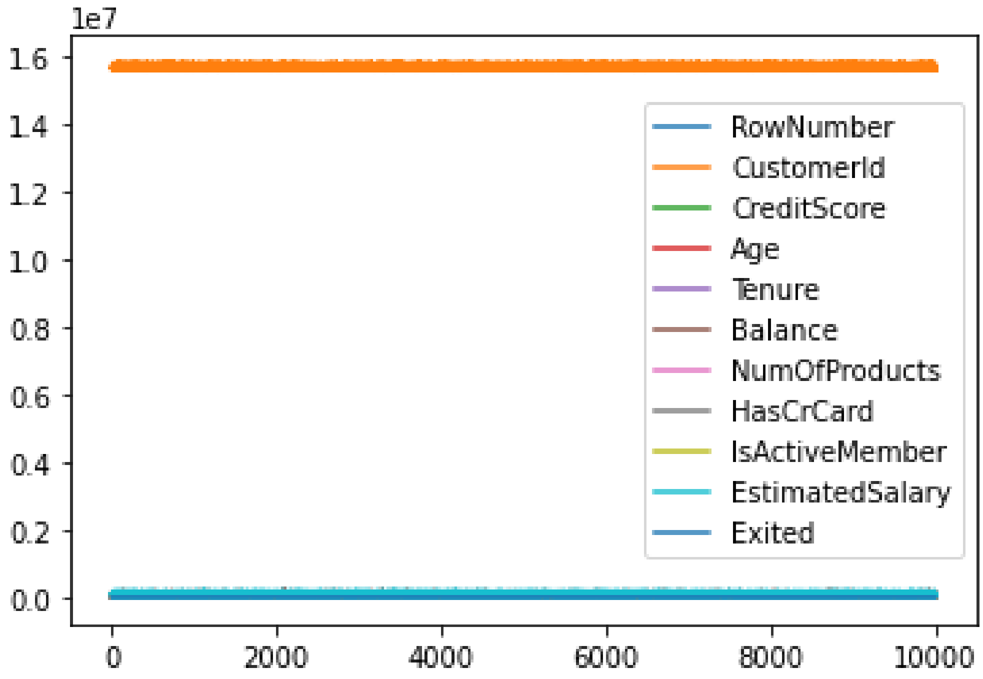
```
<seaborn.axisgrid.FacetGrid at 0x7f5cf9021b10>
```



2.Bi-Variate Analysis

```
df.plot.line()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5cf8e8d3d0>
```



3.MultiVariate Analysis

```
pip install seaborn
```

```
Looking in indexes: https://pypi.org/simple, https://us-
python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: seaborn in
/usr/local/lib/python3.7/dist-packages (0.11.2)
Requirement already satisfied: matplotlib>=2.2 in
/usr/local/lib/python3.7/dist-packages (from seaborn) (3.2.2)
Requirement already satisfied: scipy>=1.0 in
/usr/local/lib/python3.7/dist-packages (from seaborn) (1.7.3)
Requirement already satisfied: pandas>=0.23 in
/usr/local/lib/python3.7/dist-packages (from seaborn) (1.3.5)
Requirement already satisfied: numpy>=1.15 in
/usr/local/lib/python3.7/dist-packages (from seaborn) (1.21.6)
Requirement already satisfied: kiwisolver>=1.0.1 in
/usr/local/lib/python3.7/dist-packages (from matplotlib>=2.2->seaborn)
(1.4.4)
Requirement already satisfied: cycler>=0.10 in
/usr/local/lib/python3.7/dist-packages (from matplotlib>=2.2->seaborn)
(0.11.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!
=2.1.6,>=2.0.1 in /usr/local/lib/python3.7/dist-packages (from
matplotlib>=2.2->seaborn) (3.0.9)
Requirement already satisfied: python-dateutil>=2.1 in
/usr/local/lib/python3.7/dist-packages (from matplotlib>=2.2->seaborn)
(2.8.2)
```

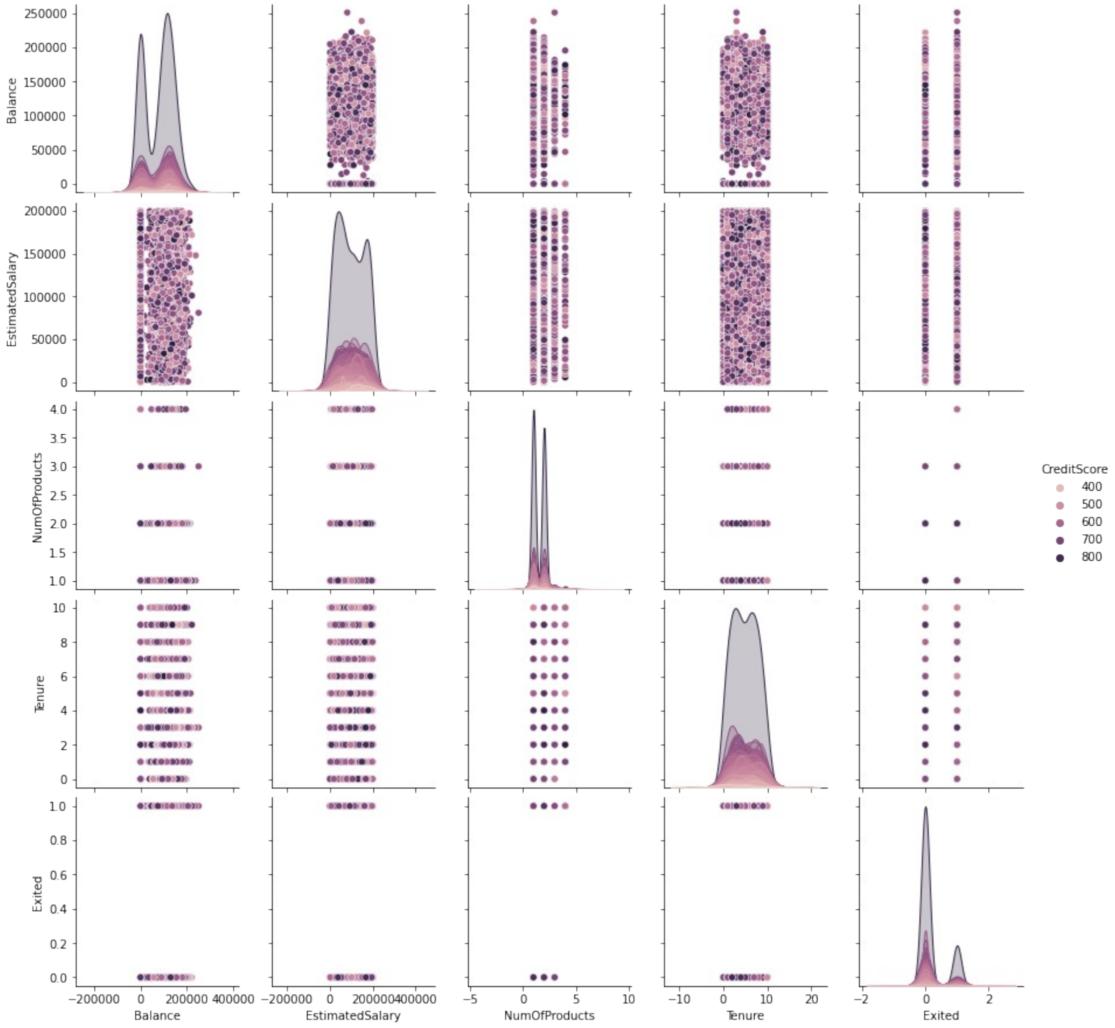
```
Requirement already satisfied: typing-extensions in
/usr/local/lib/python3.7/dist-packages (from kiwisolver>=1.0.1-
>matplotlib>=2.2->seaborn) (4.1.1)
Requirement already satisfied: pytz>=2017.3 in
/usr/local/lib/python3.7/dist-packages (from pandas>=0.23->seaborn)
(2022.2.1)
Requirement already satisfied: six>=1.5 in
/usr/local/lib/python3.7/dist-packages (from python-dateutil>=2.1-
>matplotlib>=2.2->seaborn) (1.15.0)
```

```
import seaborn as sns
```

```
plt.figure(figsize=(4,4))
sns.pairplot(data=df[["Balance","CreditScore","EstimatedSalary","NumOf
Products","Tenure","Exited"]],hue="CreditScore")
```

```
<seaborn.axisgrid.PairGrid at 0x7f5cf301c710>
```

```
<Figure size 288x288 with 0 Axes>
```



Perform descriptive statistics on the dataset

```
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
```

Handle the missing values

```
url = 'https://drive.google.com/file/d/160K6XcuYDyRBPGj-
JsqThkyFoJhCvOWy/view?usp=sharing'
path = 'https://drive.google.com/uc?
export=download&id='+url.split('/')[-2]
```

```
df= pd.read_csv(path)
pd.isnull(df["Age"])
```

```
0       False
1       False
2       False
3       False
4       False
        ...
9995    False
9996    False
9997    False
9998    False
9999    False
Name: Age, Length: 10000, dtype: bool
```

Find the outliers and replace the outliers

```
df["Age"]=np.where(df["Age"]>10,np.median,df["Age"])
df["Age"]
```

```
0       <function median at 0x7f5d15042b00>
1       <function median at 0x7f5d15042b00>
2       <function median at 0x7f5d15042b00>
3       <function median at 0x7f5d15042b00>
4       <function median at 0x7f5d15042b00>
                       ...
9995    <function median at 0x7f5d15042b00>
9996    <function median at 0x7f5d15042b00>
9997    <function median at 0x7f5d15042b00>
9998    <function median at 0x7f5d15042b00>
9999    <function median at 0x7f5d15042b00>
Name: Age, Length: 10000, dtype: object
```

Check for categorical columns and perform encoding.

```
from sklearn.preprocessing import LabelEncoder
df['Gender'].unique()
```

```
array(['Female', 'Male'], dtype=object)
```

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

```
2736    1
4076    1
8015    1
4068    1
1311    1
       ..
1313    1
```

```
5472    1
3785    1
4225    1
2497    1
Name: Gender, Length: 10000, dtype: int64
```

```
encoding=LabelEncoder()
df["Gender"]=encoding.fit_transform(df.iloc[:,1].values)
df
```

```
      RowNumber  CustomerId    Surname  CreditScore Geography  Gender  
\
0             1    15634602   Hargrave          619    France    2736

1             2    15647311       Hill          608     Spain    3258

2             3    15619304       Onio          502    France    2104

3             4    15701354       Boni          699    France    5435

4             5    15737888   Mitchell          850     Spain    6899

...         ...         ...        ...          ...       ...     ...

9995       9996    15606229   Obijiaku          771    France    1599

9996       9997    15569892  Johnstone          516    France     161

9997       9998    15584532        Liu          709    France     717

9998       9999    15682355  Sabbatini          772   Germany    4656

9999      10000    15628319     Walker          792    France    2497


                                      Age  Tenure    Balance  
NumOfProducts  \
0     <function median at 0x7f5d15042b00>       2       0.00
1
1     <function median at 0x7f5d15042b00>       1   83807.86
1
2     <function median at 0x7f5d15042b00>       8  159660.80
3
3     <function median at 0x7f5d15042b00>       1       0.00
2
4     <function median at 0x7f5d15042b00>       2  125510.82
1
...                                   ...     ...        ...
...
```

```
9995  <function median at 0x7f5d15042b00>       5       0.00
2
9996  <function median at 0x7f5d15042b00>      10   57369.61
1
9997  <function median at 0x7f5d15042b00>       7       0.00
1
9998  <function median at 0x7f5d15042b00>       3   75075.31
2
9999  <function median at 0x7f5d15042b00>       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 14 columns]
```

Split the data into dependent and independent variables

```
x=df.iloc[:,:-2].values
print(x)
```

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

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

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

Scale the independent variables

```
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
```

```
df[["RowNumber"]] =scaler.fit_transform(df[["RowNumber"]])
print(df)
```

```
      RowNumber  CustomerId    Surname  CreditScore Geography  Gender  
\
0        0.0000    15634602   Hargrave          619    France    2736   

1        0.0001    15647311       Hill          608     Spain    3258   

2        0.0002    15619304       Onio          502    France    2104   

3        0.0003    15701354       Boni          699    France    5435   

4        0.0004    15737888   Mitchell          850     Spain    6899   

...         ...         ...        ...          ...       ...     ...   

9995     0.9996    15606229   Obijiaku          771    France    1599   

9996     0.9997    15569892  Johnstone          516    France     161   

9997     0.9998    15584532        Liu          709    France     717   

9998     0.9999    15682355  Sabbatini          772   Germany    4656   

9999     1.0000    15628319     Walker          792    France    2497   


                                      Age  Tenure    Balance  
NumOfProducts  \
0     <function median at 0x7f5d15042b00>       2       0.00  
1  
1     <function median at 0x7f5d15042b00>       1   83807.86  
1  
2     <function median at 0x7f5d15042b00>       8  159660.80  
3  
3     <function median at 0x7f5d15042b00>       1       0.00  
2  
4     <function median at 0x7f5d15042b00>       2  125510.82  
1  
...                                   ...     ...        ...  
...  
9995  <function median at 0x7f5d15042b00>       5       0.00  
2  
9996  <function median at 0x7f5d15042b00>      10   57369.61  
1  
9997  <function median at 0x7f5d15042b00>       7       0.00  
1  
9998  <function median at 0x7f5d15042b00>       3   75075.31  
```

```
2
9999  <function median at 0x7f5d15042b00>       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 14 columns]
```

Spilt the data into training and testing

```
from sklearn.model_selection import train_test_split
train_size=0.8
X=df.drop(columns=['Age']).copy()
Y=df['Age']
X_train,X_rem,Y_train,Y_rem=train_test_split(X,Y,train_size=0.8)
test_size=0.5
X_valid,X_test,Y_valid,Y_test=train_test_split(X_rem,Y_rem,test_size=0
.5)
print(X_train.shape),print(Y_train.shape)
print(X_valid.shape),print(Y_valid.shape)
print(X_test.shape),print(Y_test.shape)
```

```
(8000, 13)
(8000,)
(1000, 13)
(1000,)
(1000, 13)
(1000,)

(None, None)
```