Assignment 2

Assigment Date	08 November 2022
Student Name	Surya V S
Student Register Number	620619106039
Maximum Marks	2

1.Importing package

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

2.Loading dataset

```
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]
```
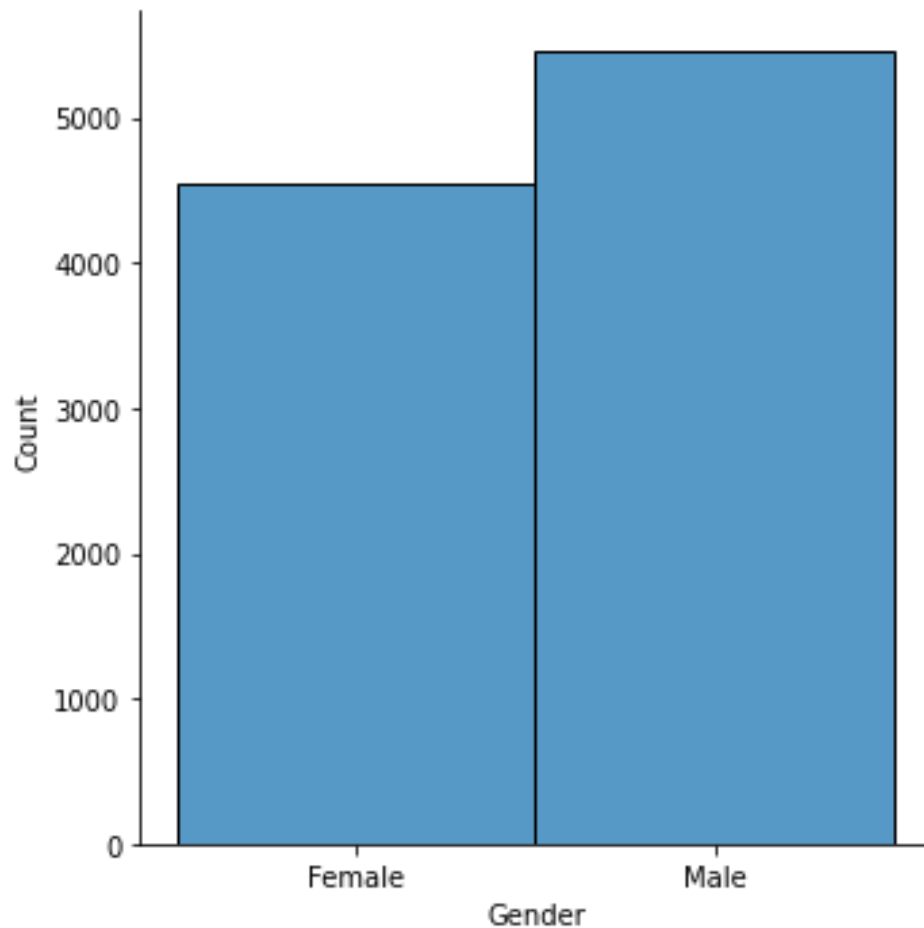
3.Visualizations
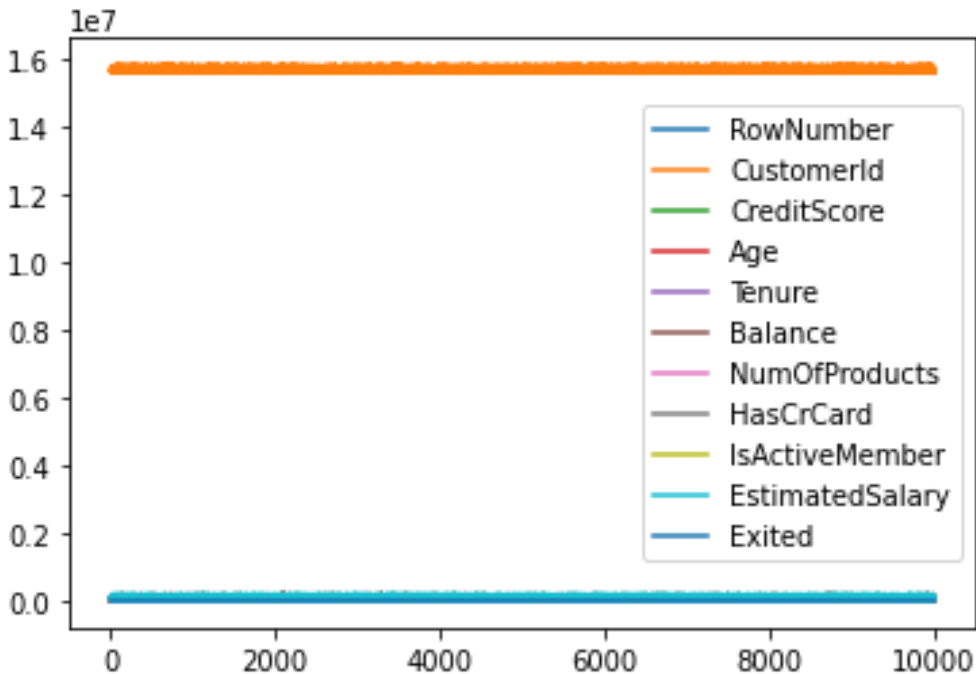
a) Univariate Analysis

```
sns.displot(df.Gender)
```

```
<seaborn.axisgrid.FacetGrid at 0x7f1cc8cc7710>
```


b) Bi-Variate Analysis

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

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1cc8a68f10>
```

c) Multi-Variate Analysis

```python
sns.lmplot("Tenure","NumOfProducts",df,hue="NumOfProducts",
fit_reg=False);
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variables as keyword args: x, y,
data. 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.
  FutureWarning
```
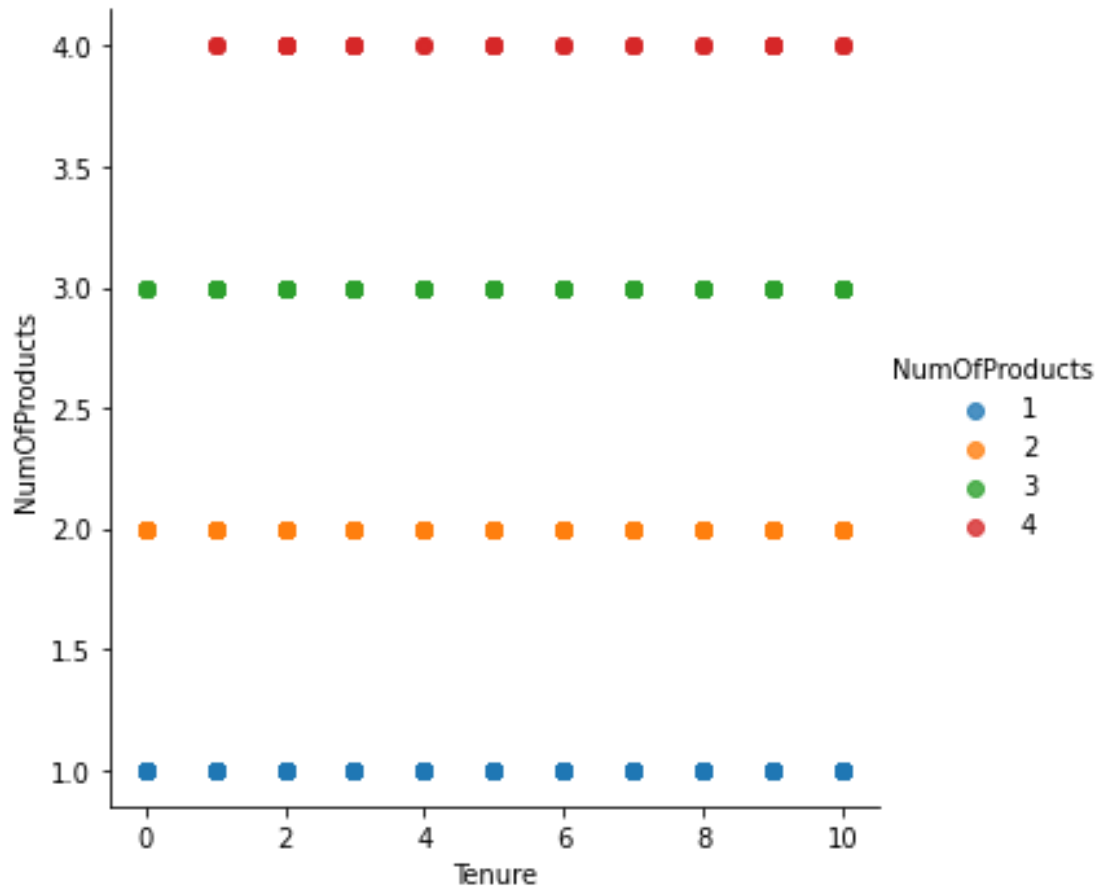
4. Perform descriptive statistic on the dataset

```
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
```

5. Handle the Missing values

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

pd.isnull(data["Gender"])
```

```
0       False
1       False
2       False
3       False
4       False
        ...
9995    False
9996    False
9997    False
9998    False
9999    False
Name: Gender, Length: 10000, dtype: bool
```
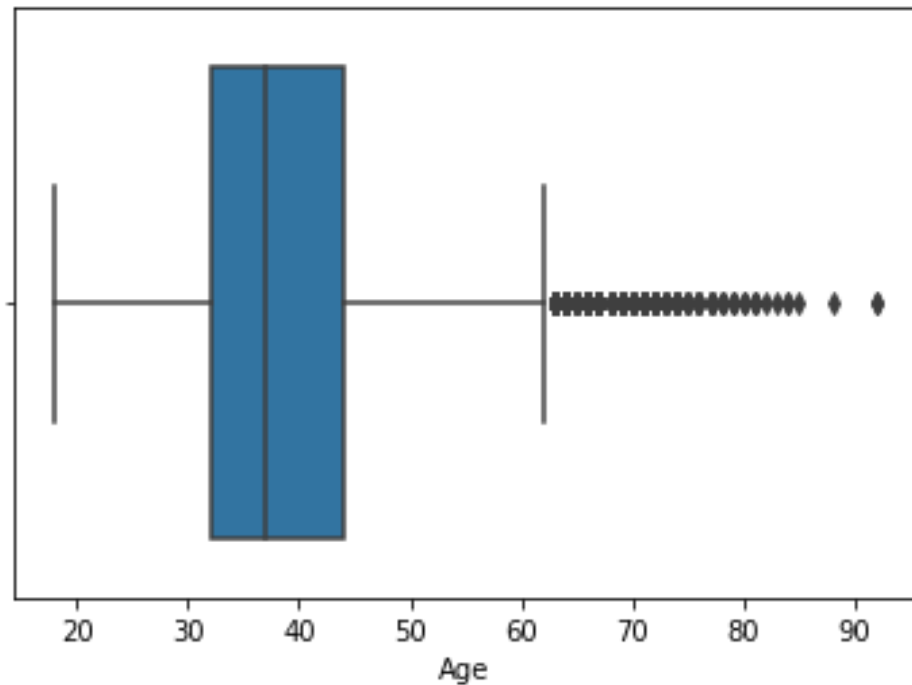
6. Find the outliers and replace the outliers

```
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, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1cc5a28b90>
```



```python
df['Age']=np.where(df['Age']>50,40,df['Age'])
```

```python
df['Age']
```

```
0       42
1       41
2       42
3       39
4       43
        ..
9995    39
9996    35
9997    36
9998    42
9999    28
Name: Age, Length: 10000, dtype: int64
```
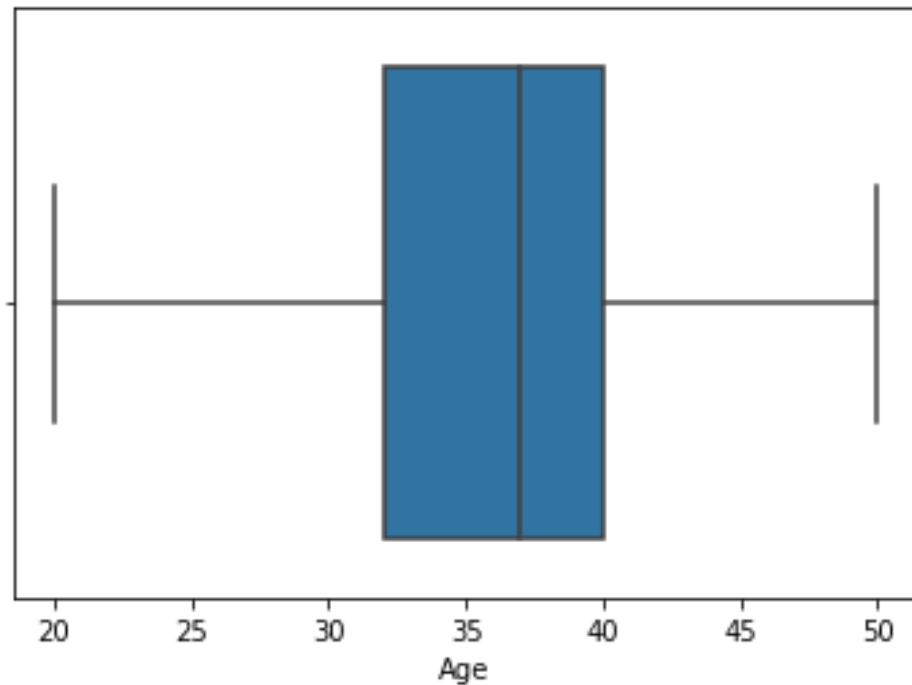
```python
sns.boxplot(df['Age'])
```

```
/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, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f1cc592b3d0>


```python
df['Age']=np.where(df['Age']<20,35,df['Age'])
```

```python
df['Age']
```

```
0        42
1        41
2        42
3        39
4        43
         ..
9995     39
9996     35
9997     36
9998     42
9999     28
Name: Age, Length: 10000, dtype: int64
```

7. Check for Categorical columns and perform encoding

```
pd.get_dummies(df, columns=["Gender", "Age"], prefix=["Age",
"Gender"]).head()
```

```
   RowNumber   CustomerId    Surname   CreditScore Geography   Tenure
Balance   \
0          1     15634602   Hargrave           619    France        2
0.00
1          2     15647311       Hill           608     Spain        1
83807.86
2          3     15619304       Onio           502    France        8
159660.80
3          4     15701354       Boni           699    France        1
0.00
4          5     15737888   Mitchell           850     Spain        2
125510.82

   NumOfProducts   HasCrCard   IsActiveMember   ...   Gender_41   Gender_42
\
0              1           1                1   ...           0           1
1              1           0                1   ...           1           0
2              3           1                0   ...           0           1
3              2           0                0   ...           0           0
4              1           1                1   ...           0           0

   Gender_43   Gender_44   Gender_45   Gender_46   Gender_47   Gender_48   \
0          0           0           0           0           0           0
1          0           0           0           0           0           0
2          0           0           0           0           0           0
3          0           0           0           0           0           0
4          1           0           0           0           0           0

   Gender_49   Gender_50
0          0           0
1          0           0
2          0           0
3          0           0
4          0           0

[5 rows x 45 columns]
```

8.Split the data into dependent and independent variables

a) Split the data into Independent variables.

```
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]]
```

b) Split the data into Dependent variables.

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

```
print(Y)
```

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

9. Scale the independent variables

```
import pandas as pd

from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()

df[["CustomerId"]] = scaler.fit_transform(df[["CustomerId"]])

print(df)
```

```
     RowNumber  CustomerId   Surname  CreditScore Geography  Gender
Age  \
0            1    0.275616  Hargrave          619    France  Female
42
1            2    0.326454      Hill          608     Spain  Female
41
2            3    0.214421      Onio          502    France  Female
42
3            4    0.542636      Boni          699    France  Female
39
4            5    0.688778  Mitchell          850     Spain  Female
43
...        ...         ...       ...          ...       ...     ...
...
```

```
9995      9996    0.162119   Obijiaku       771    France     Male
39
9996      9997    0.016765  Johnstone       516    France     Male
35
9997      9998    0.075327        Liu       709    France   Female
36
9998      9999    0.466637  Sabbatini       772   Germany     Male
42
9999     10000    0.250483     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]
```

10. Split the data into training and testing

```
from sklearn.model_selection import train_test_split

train_size=0.8

X = df.drop(columns = ['Tenure']).copy()
```

```python
y = df['Tenure']

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)
```