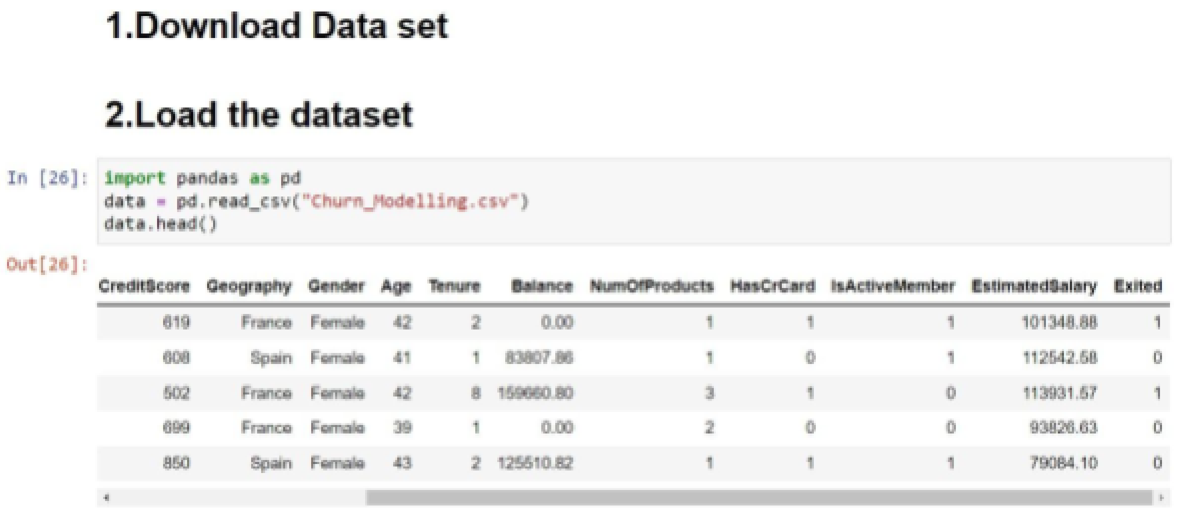
Assignment - 02

Data visualization and pre-processing

Assignment Date	26 September 2022
Student Name	Deepa.S
Student Roll Number	820419106009
Maximum Marks	2 Marks



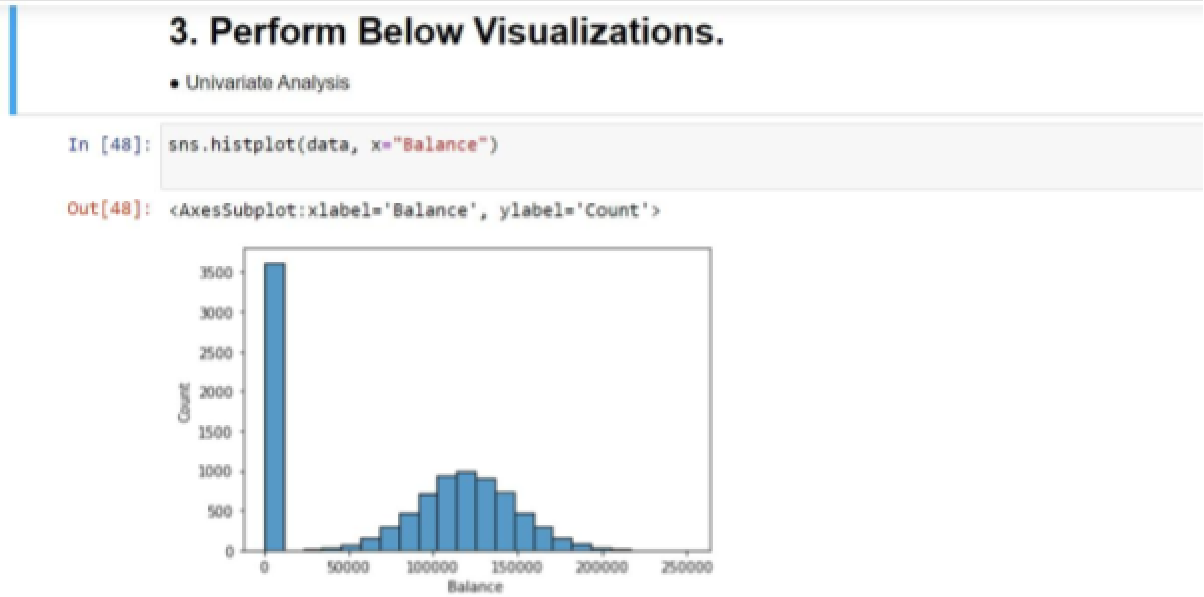
1.Download Data set

2.Load the dataset

```
In [26]: import pandas as pd
         data = pd.read_csv("Churn_Modelling.csv")
         data.head()
```

Out[26]:

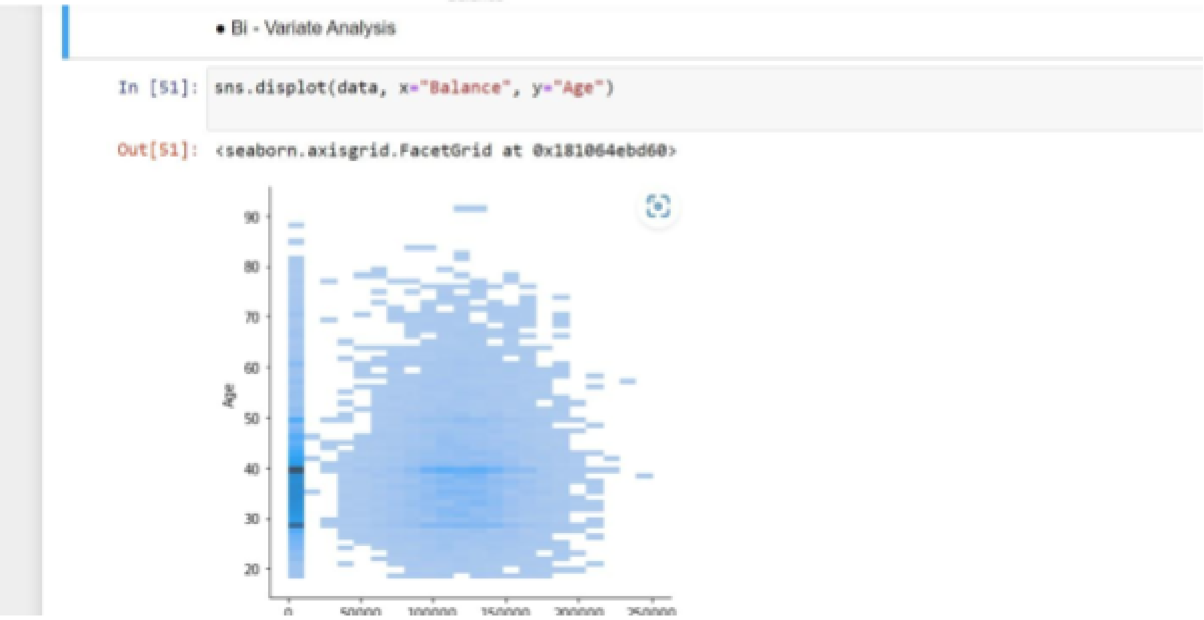
CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
619	France	Female	42	2	0.00	1	1	1	101348.88	1
608	Spain	Female	41	1	83807.86	1	0	1	112542.58	0
502	France	Female	42	8	159660.80	3	1	0	113931.57	1
699	France	Female	39	1	0.00	2	0	0	93826.63	0
850	Spain	Female	43	2	125510.82	1	1	1	79084.10	0



3. Perform Below Visualizations.

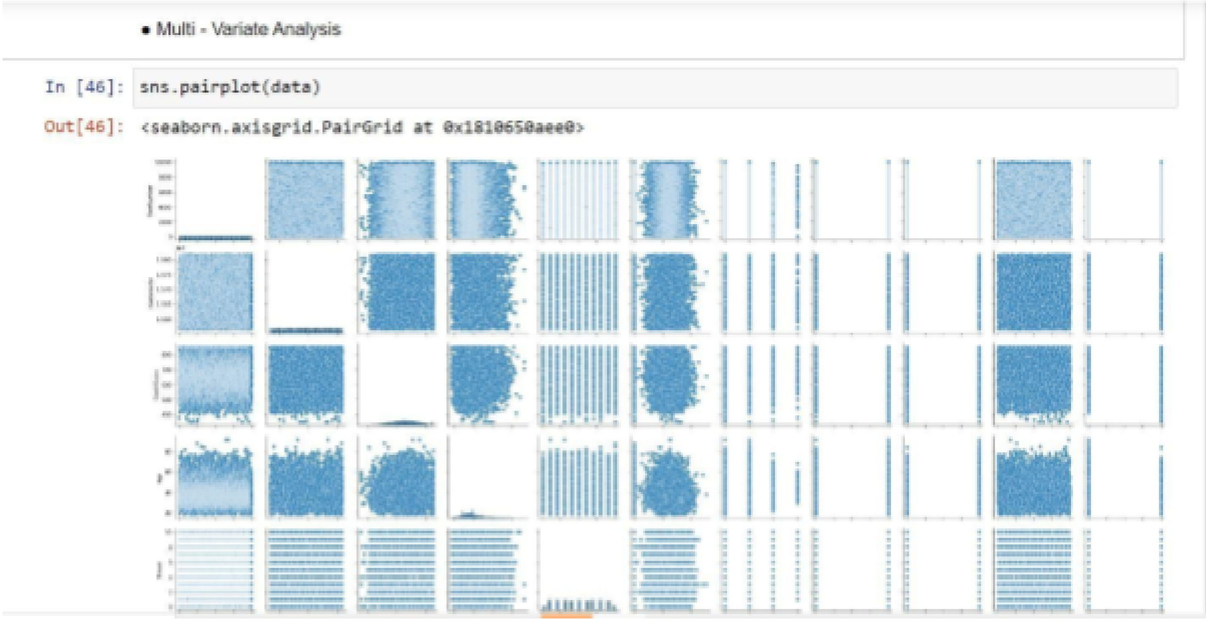
- Univariate Analysis

```
In [48]: sns.histplot(data, x="Balance")

Out[48]: <AxesSubplot:xlabel='Balance', ylabel='Count'>
```



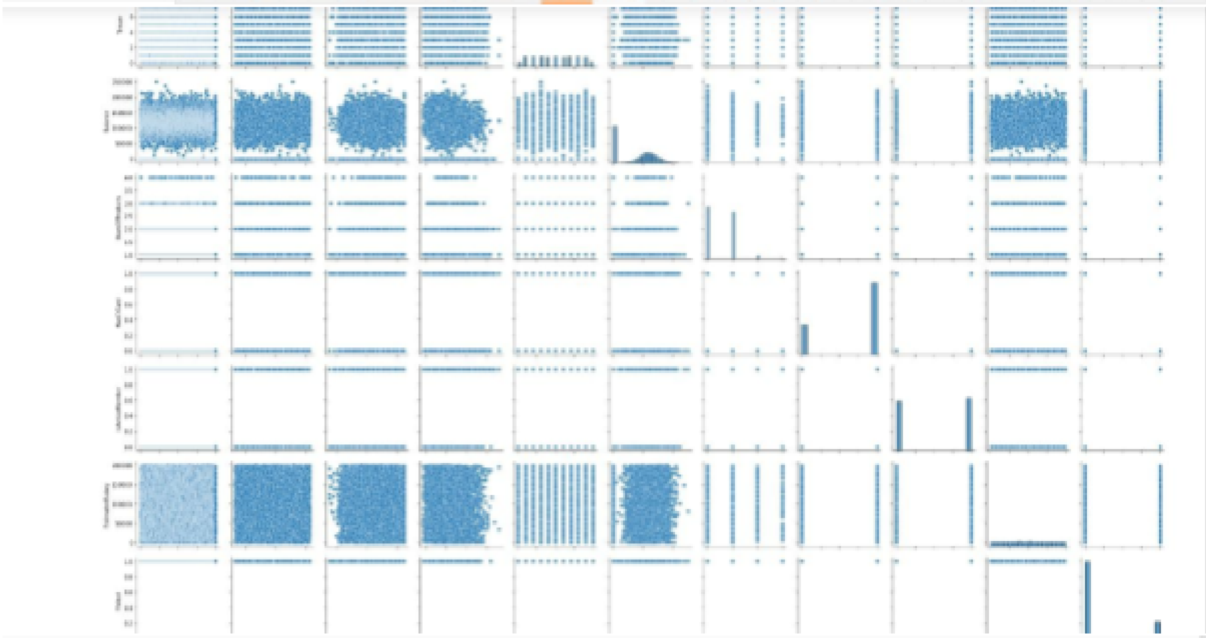
- Bi - Variate Analysis

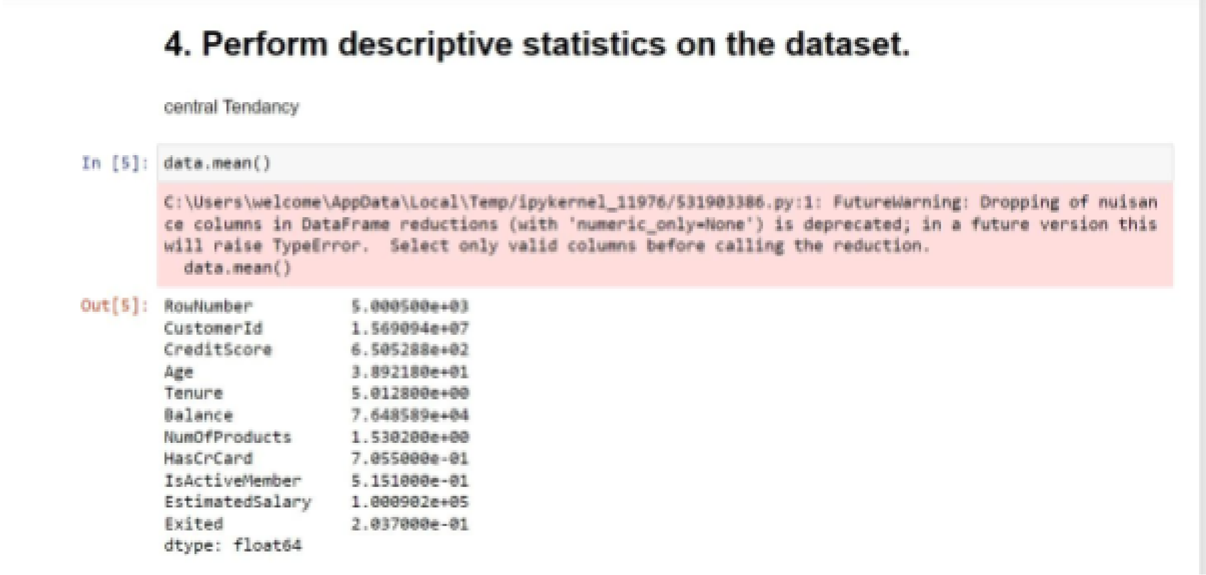
```
In [51]: sns.displot(data, x="Balance", y="Age")

Out[51]: <seaborn.axisgrid.FacetGrid at 0x181064ebd60>
```



- Multi - Variate Analysis

```
In [46]: sns.pairplot(data)

Out[46]: <seaborn.axisgrid.PairGrid at 0x1810650aee0>
```

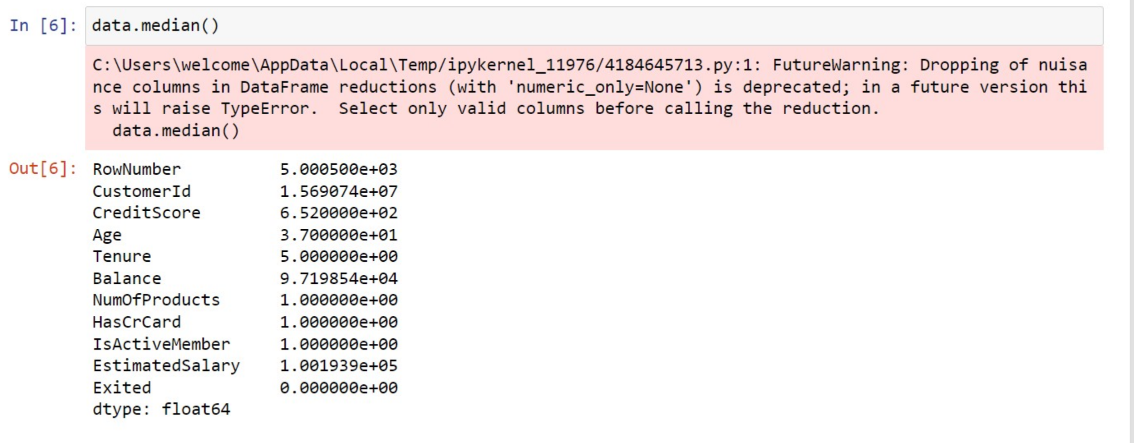




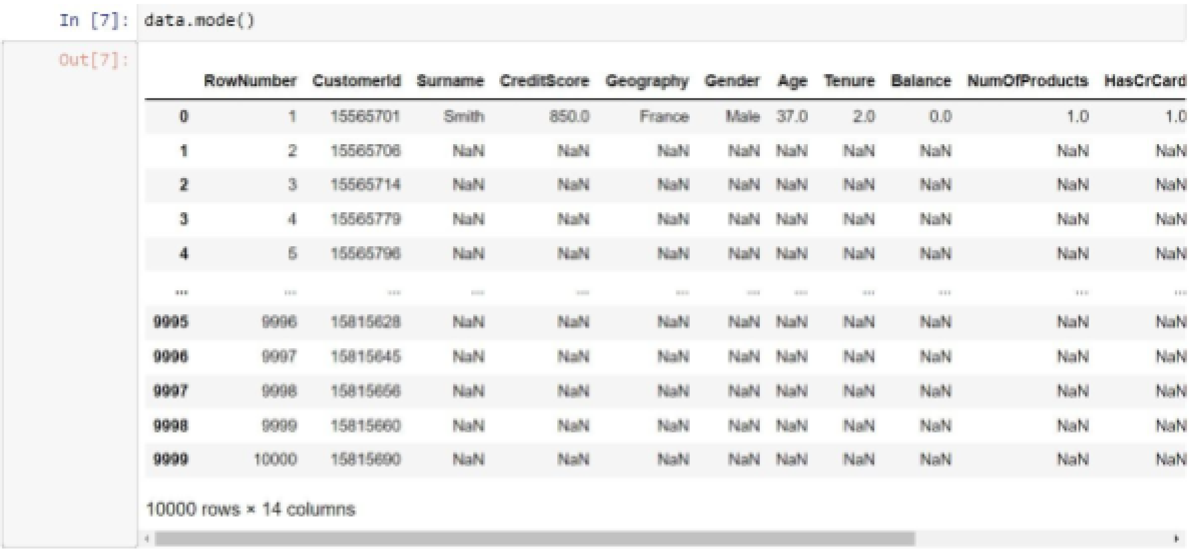
4. Perform descriptive statistics on the dataset.

central Tendancy

```
In [5]: data.mean()

C:\Users\welcome\AppData\Local\Temp/ipykernel_11976/531903386.py:1: FutureWarning: Dropping of nuisan
ce columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this
will raise TypeError.  Select only valid columns before calling the reduction.
  data.mean()

Out[5]: 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
```



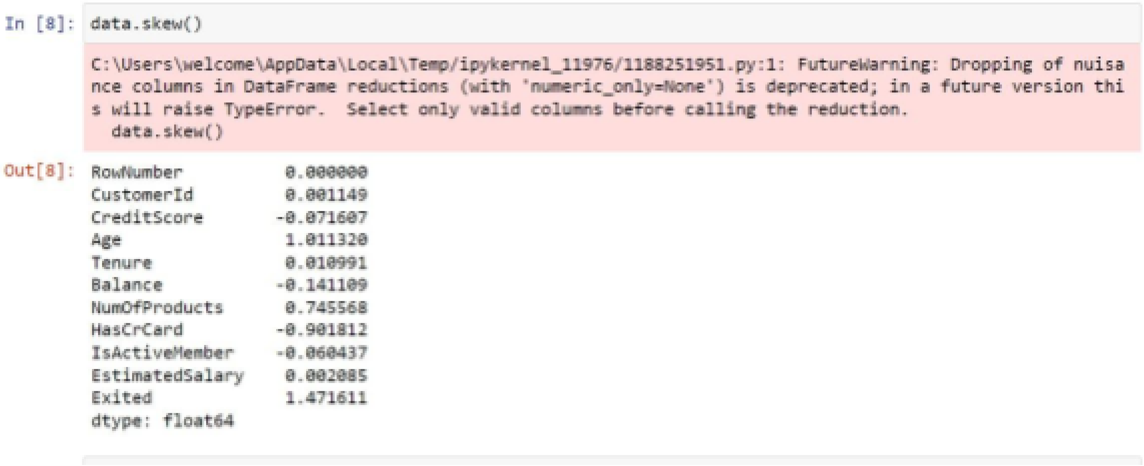
```
In [6]: data.median()

C:\Users\welcome\AppData\Local\Temp/ipykernel_11976/4184645713.py:1: FutureWarning: Dropping of nuisa
nce columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version thi
s will raise TypeError.  Select only valid columns before calling the reduction.
  data.median()

Out[6]: 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
```



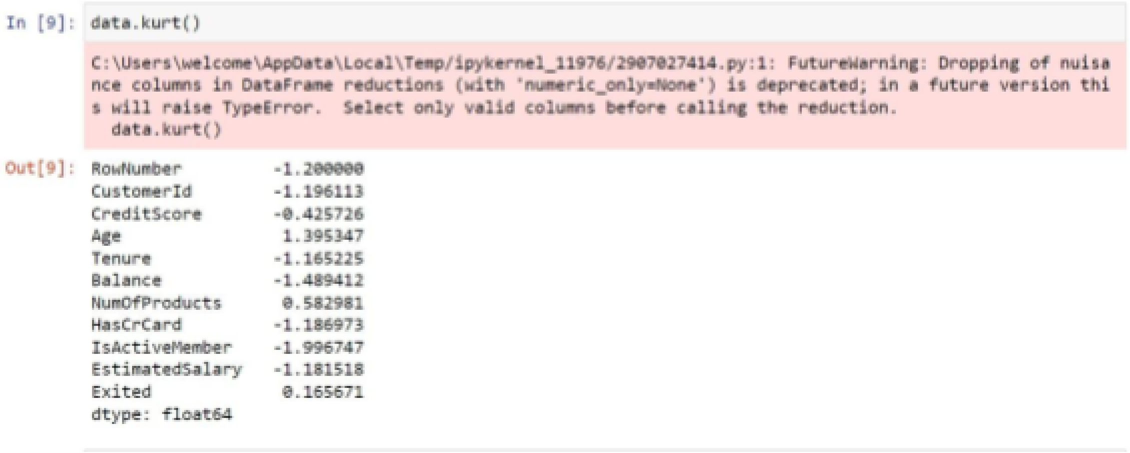
```
In [7]: data.mode()

Out[7]:
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard
0	1	15565701	Smith	850.0	France	Male	37.0	2.0	0.0	1.0	1.0
1	2	15565706	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	3	15565714	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	4	15565779	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	5	15565796	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...
9995	9996	15815628	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
9996	9997	15815645	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
9997	9998	15815656	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
9998	9999	15815660	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
9999	10000	15815690	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

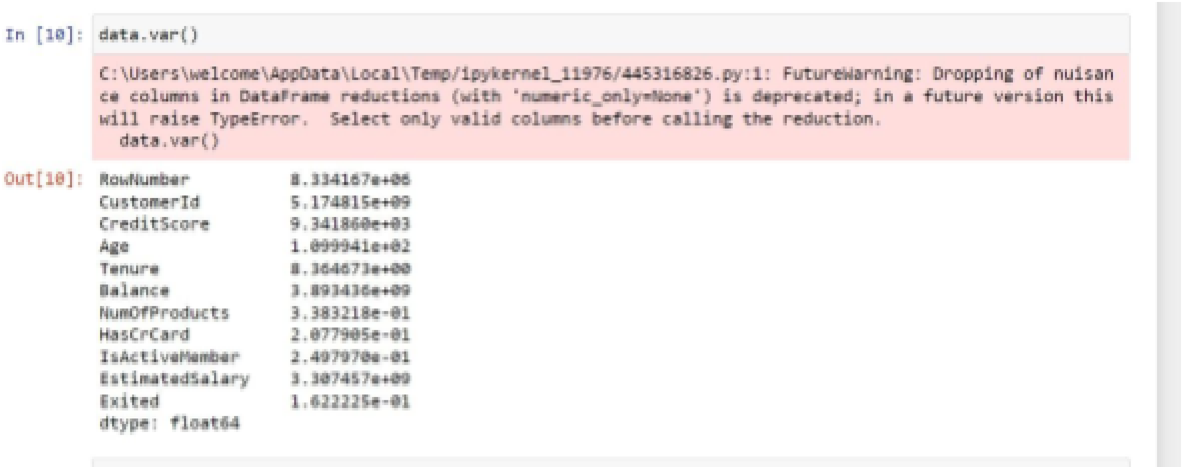
10000 rows × 14 columns



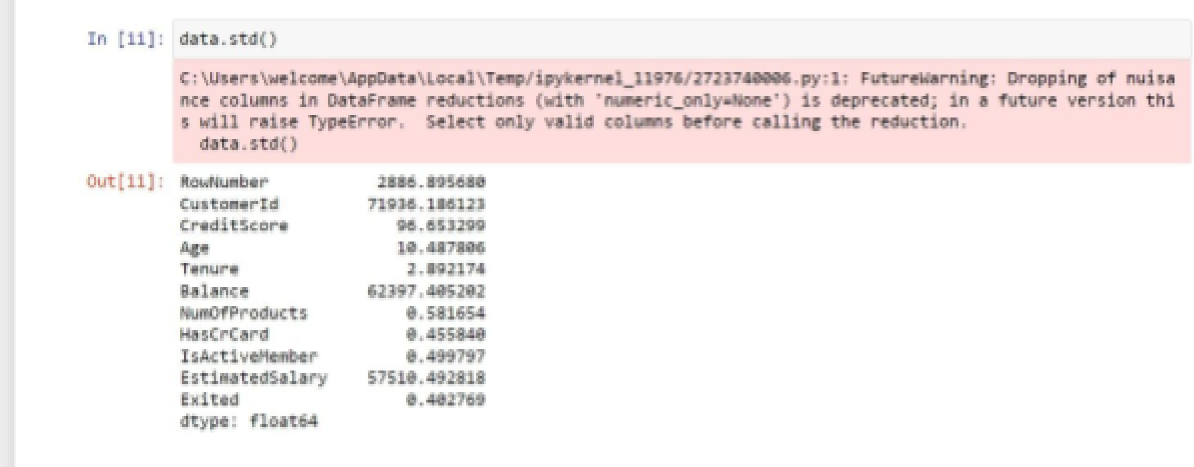
```
In [8]: data.skew()

C:\Users\welcome\AppData\Local\Temp/ipykernel_11976/1188251951.py:1: FutureWarning: Dropping of nuisa
nce columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version thi
s will raise TypeError.  Select only valid columns before calling the reduction.
  data.skew()

Out[8]: RowNumber          0.000000
        CustomerId         0.001149
        CreditScore       -0.071607
        Age                1.011320
        Tenure             0.010991
        Balance           -0.141109
        NumOfProducts      0.745568
        HasCrCard         -0.901812
        IsActiveMember    -0.060437
        EstimatedSalary    0.002085
        Exited             1.471611
        dtype: float64
```



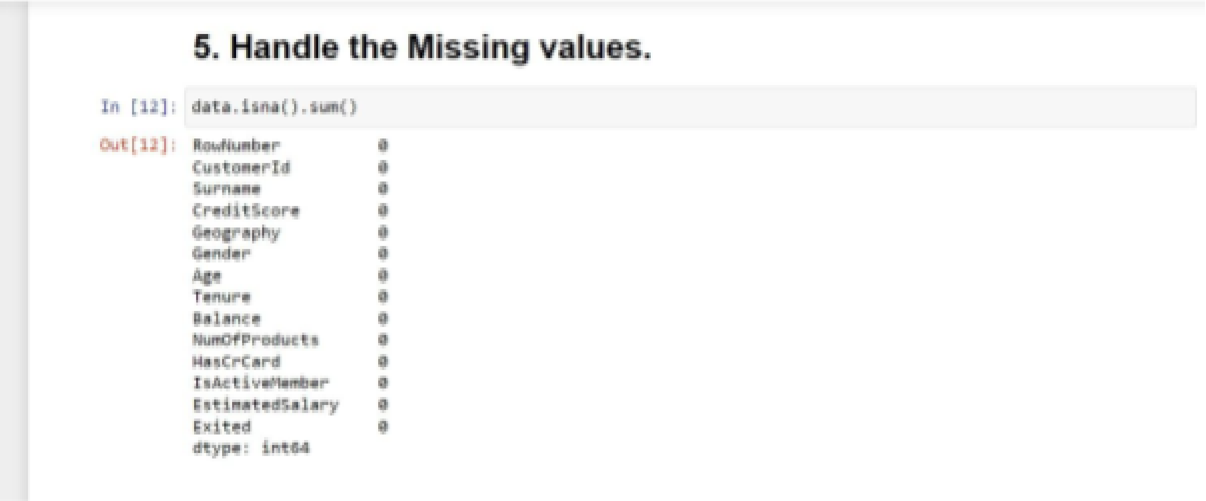
```
In [9]: data.kurt()

C:\Users\welcome\AppData\Local\Temp/ipykernel_11976/2907027414.py:1: FutureWarning: Dropping of nuisa
nce columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version thi
s will raise TypeError.  Select only valid columns before calling the reduction.
  data.kurt()

Out[9]: RowNumber         -1.200000
        CustomerId        -1.196113
        CreditScore       -0.425726
        Age                1.395347
        Tenure            -1.165225
        Balance           -1.489412
        NumOfProducts      0.582981
        HasCrCard         -1.186973
        IsActiveMember    -1.996747
        EstimatedSalary   -1.181518
        Exited             0.165671
        dtype: float64
```



```
In [10]: data.var()

C:\Users\welcome\AppData\Local\Temp/ipykernel_11976/445316826.py:1: FutureWarning: Dropping of nuisan
ce columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this
will raise TypeError.  Select only valid columns before calling the reduction.
  data.var()

Out[10]: RowNumber          8.334167e+06
         CustomerId         5.174815e+09
         CreditScore        9.341860e+03
         Age                1.099941e+02
         Tenure             8.364673e+00
         Balance            3.893436e+09
         NumOfProducts      3.383218e-01
         HasCrCard          2.077905e-01
         IsActiveMember     2.497970e-01
         EstimatedSalary    3.307457e+09
         Exited             1.622225e-01
         dtype: float64
```

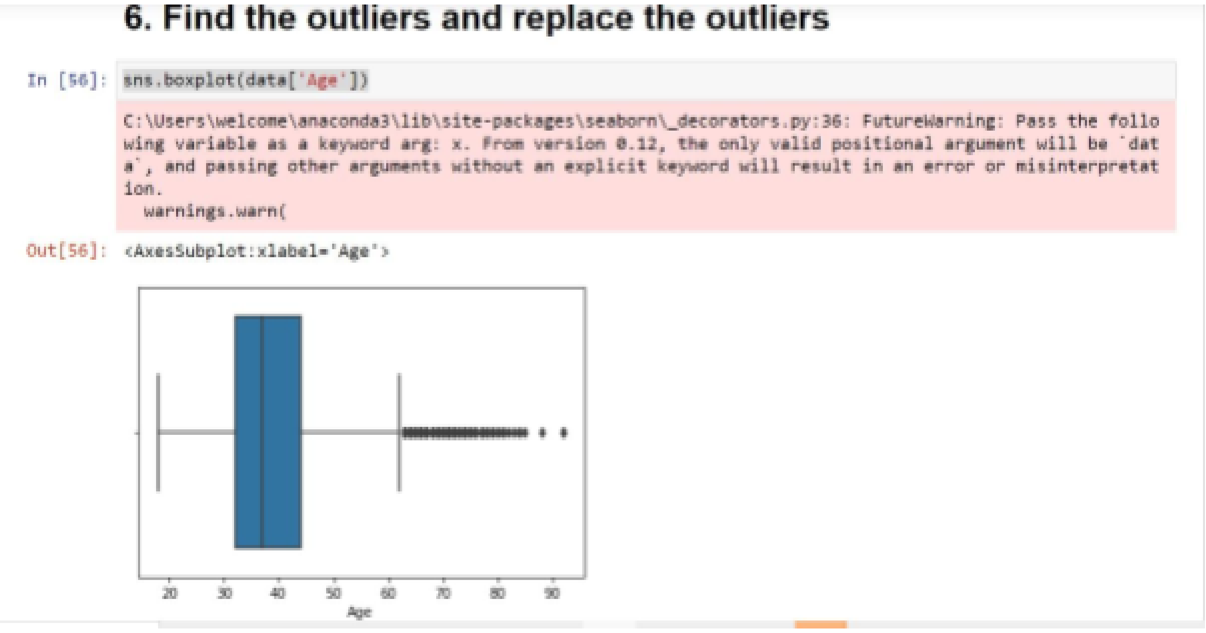


```
In [11]: data.std()

C:\Users\welcome\AppData\Local\Temp/ipykernel_11976/2723740006.py:1: FutureWarning: Dropping of nuisa
nce columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version thi
s will raise TypeError.  Select only valid columns before calling the reduction.
  data.std()

Out[11]: 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
```



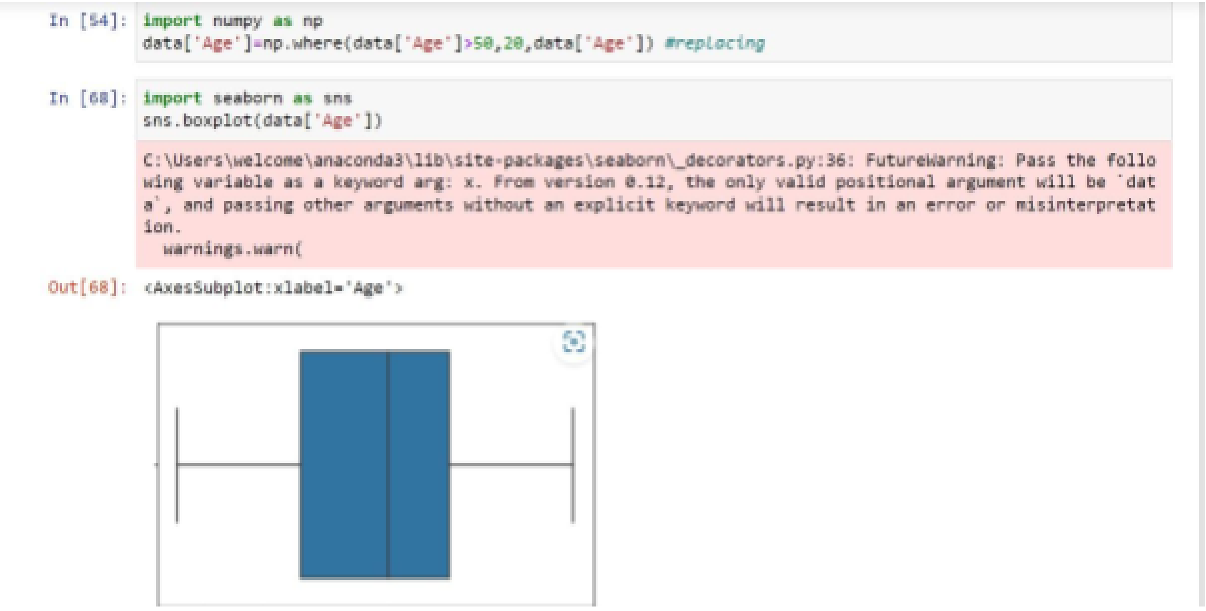
5. Handle the Missing values.

```
In [12]: data.isna().sum()

Out[12]: 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

```
In [56]: sns.boxplot(data['Age'])

C:\Users\welcome\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the follo
wing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `dat
a`, and passing other arguments without an explicit keyword will result in an error or misinterpretat
ion.
  warnings.warn(

Out[56]: <AxesSubplot:xlabel='Age'>
```

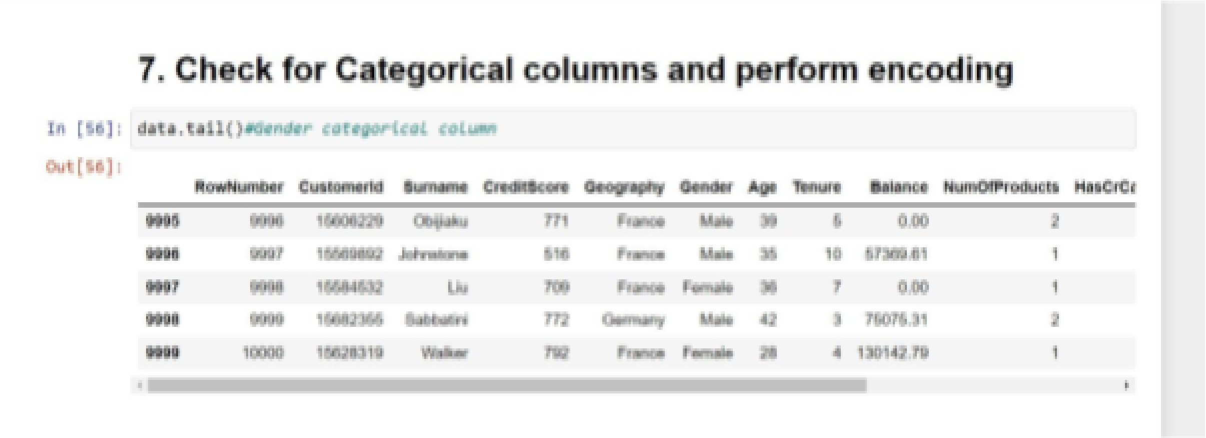


```
In [54]: import numpy as np
         data['Age']=np.where(data['Age']>50,20,data['Age']) #replacing
```

```
In [68]: import seaborn as sns
         sns.boxplot(data['Age'])
```

```
C:\Users\welcome\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the follo
wing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `dat
a`, and passing other arguments without an explicit keyword will result in an error or misinterpretat
ion.
  warnings.warn(
```

```
Out[68]: <AxesSubplot:xlabel='Age'>
```

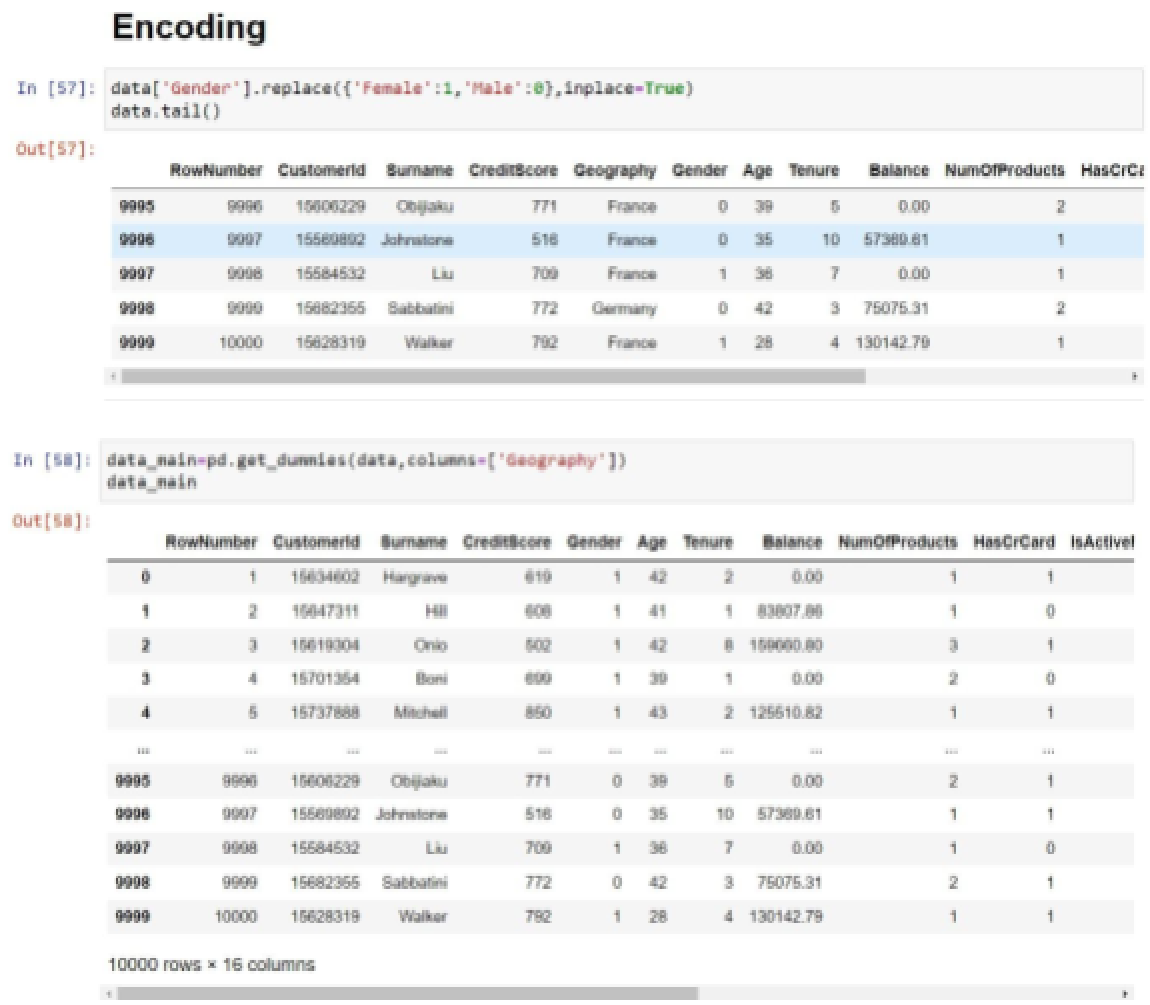


7. Check for Categorical columns and perform encoding

```
In [56]: data.tail()#Gender categorical column
```

Out[56]:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCa
9995	9996	15606229	Obijiaku	771	France	Male	39	5	0.00	2	
9996	9997	15569892	Johnstone	516	France	Male	35	10	57369.61	1	
9997	9998	15584532	Liu	709	France	Female	36	7	0.00	1	
9998	9999	15682355	Sabbatini	772	Germany	Male	42	3	75075.31	2	
9999	10000	15628319	Walker	792	France	Female	28	4	130142.79	1	



Encoding

```
In [57]: data['Gender'].replace({'Female':1,'Male':0},inplace=True)
         data.tail()
```

Out[57]:

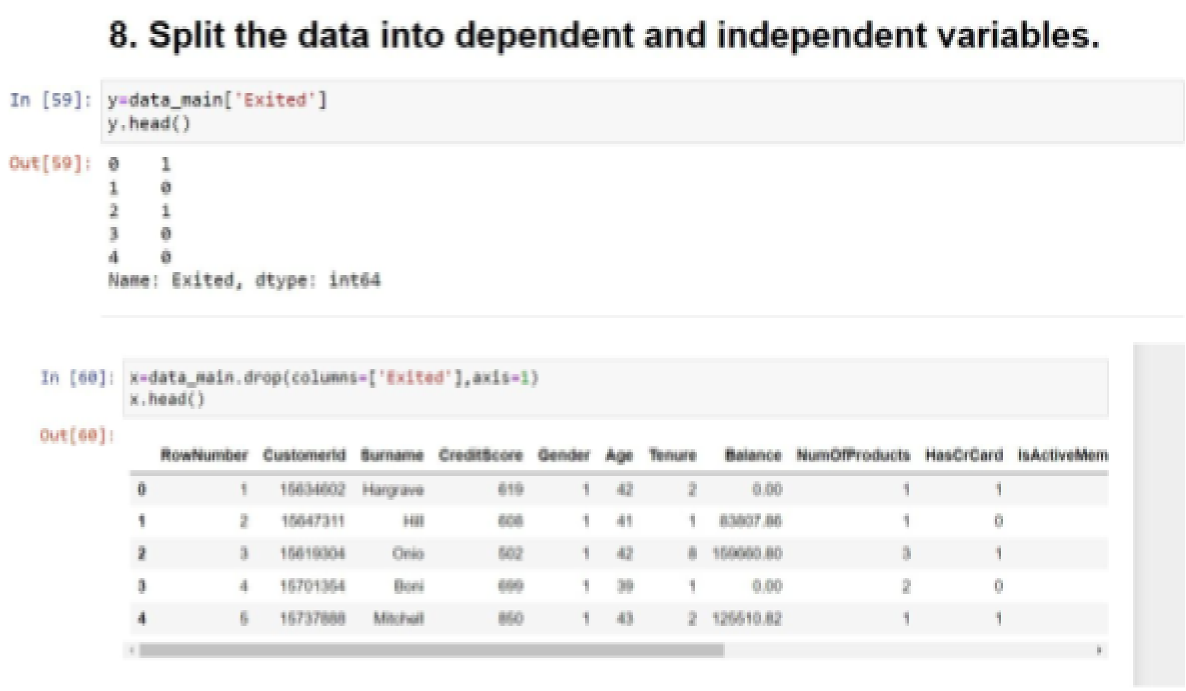
	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCa
9995	9996	15606229	Obijiaku	771	France	0	39	5	0.00	2	
9996	9997	15569892	Johnstone	516	France	0	35	10	57369.61	1	
9997	9998	15584532	Liu	709	France	1	36	7	0.00	1	
9998	9999	15682355	Sabbatini	772	Germany	0	42	3	75075.31	2	
9999	10000	15628319	Walker	792	France	1	28	4	130142.79	1	

```
In [58]: data_main=pd.get_dummies(data,columns=['Geography'])
         data_main
```

Out[58]:

	RowNumber	CustomerId	Surname	CreditScore	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveI
0	1	15634602	Hargrave	619	1	42	2	0.00	1	1	
1	2	15647311	Hill	608	1	41	1	83807.86	1	0	
2	3	15619304	Onio	502	1	42	8	159660.80	3	1	
3	4	15701354	Boni	699	1	39	1	0.00	2	0	
4	5	15737888	Mitchell	850	1	43	2	125510.82	1	1	
...	...	...	...	...	...	...	...	...	...	...	
9995	9996	15606229	Obijiaku	771	0	39	5	0.00	2	1	
9996	9997	15569892	Johnstone	516	0	35	10	57369.61	1	1	
9997	9998	15584532	Liu	709	1	36	7	0.00	1	0	
9998	9999	15682355	Sabbatini	772	0	42	3	75075.31	2	1	
9999	10000	15628319	Walker	792	1	28	4	130142.79	1	1	

10000 rows × 16 columns

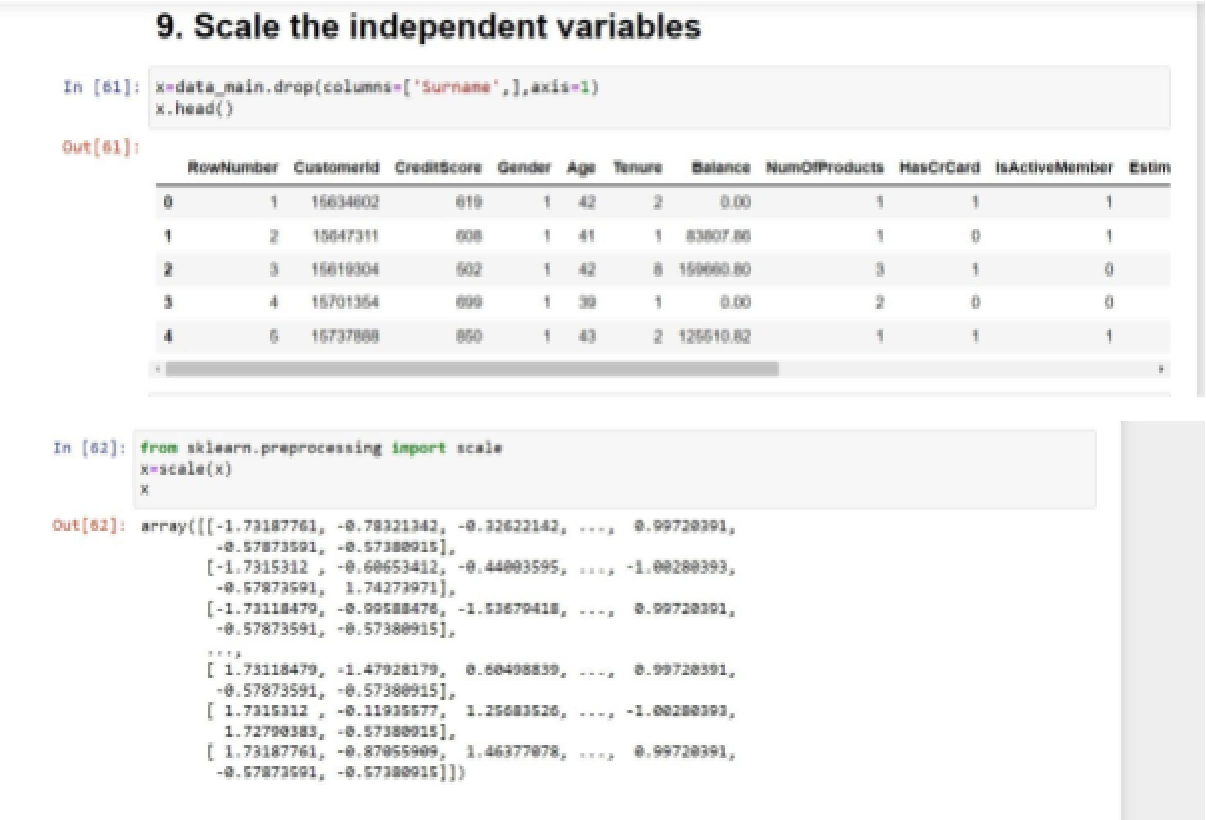


8. Split the data into dependent and independent variables.

```
In [59]: y=data_main['Exited']
         y.head()

Out[59]: 0    1
         1    0
         2    1
         3    0
         4    0
         Name: Exited, dtype: int64
```

```
In [60]: x=data_main.drop(columns=['Exited'],axis=1)
         x.head()

Out[60]:
```

	RowNumber	CustomerId	Surname	CreditScore	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMem
0	1	15634602	Hargrave	619	1	42	2	0.00	1	1	
1	2	15647311	Hill	608	1	41	1	83807.86	1	0	
2	3	15619304	Onio	502	1	42	8	159660.80	3	1	
3	4	15701354	Boni	699	1	39	1	0.00	2	0	
4	5	15737888	Mitchell	850	1	43	2	125510.82	1	1	

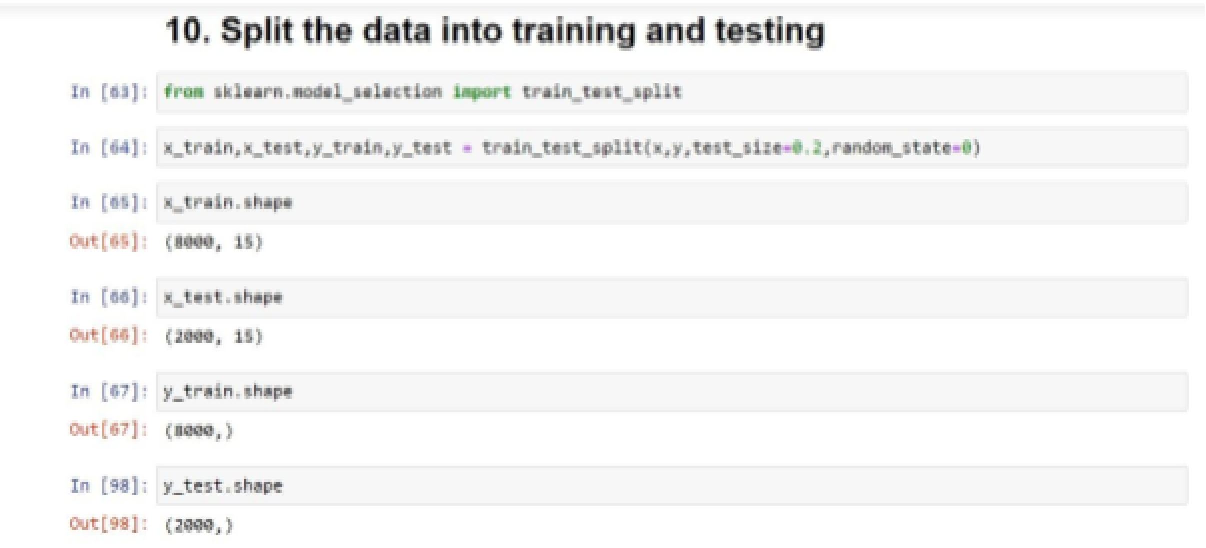


9. Scale the independent variables

```
In [61]: x=data_main.drop(columns=['Surname',],axis=1)
         x.head()

Out[61]:
```

	RowNumber	CustomerId	CreditScore	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	Estim
0	1	15634602	619	1	42	2	0.00	1	1	1	
1	2	15647311	608	1	41	1	83807.86	1	0	1	
2	3	15619304	502	1	42	8	159660.80	3	1	0	
3	4	15701354	699	1	39	1	0.00	2	0	0	
4	5	15737888	850	1	43	2	125510.82	1	1	1	

```
In [62]: from sklearn.preprocessing import scale
         x=scale(x)
         x

Out[62]: array([[-1.73187761, -0.78321342, -0.32622142, ...,  0.99720391,
                 -0.57873591, -0.57380915],
                [-1.7315312 , -0.60653412, -0.44003595, ..., -1.00280393,
                 -0.57873591,  1.74273971],
                [-1.73118479, -0.99588476, -1.53679418, ...,  0.99720391,
                 -0.57873591, -0.57380915],
                ...,
                [ 1.73118479, -1.47928179,  0.60498839, ...,  0.99720391,
                 -0.57873591, -0.57380915],
                [ 1.7315312 , -0.11935577,  1.25683526, ..., -1.00280393,
                  1.72790383, -0.57380915],
                [ 1.73187761, -0.87055909,  1.46377078, ...,  0.99720391,
                 -0.57873591, -0.57380915]])
```



10. Split the data into training and testing

```
In [63]: from sklearn.model_selection import train_test_split

In [64]: x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state=0)

In [65]: x_train.shape

Out[65]: (8000, 15)

In [66]: x_test.shape

Out[66]: (2000, 15)

In [67]: y_train.shape

Out[67]: (8000,)

In [98]: y_test.shape

Out[98]: (2000,)
```