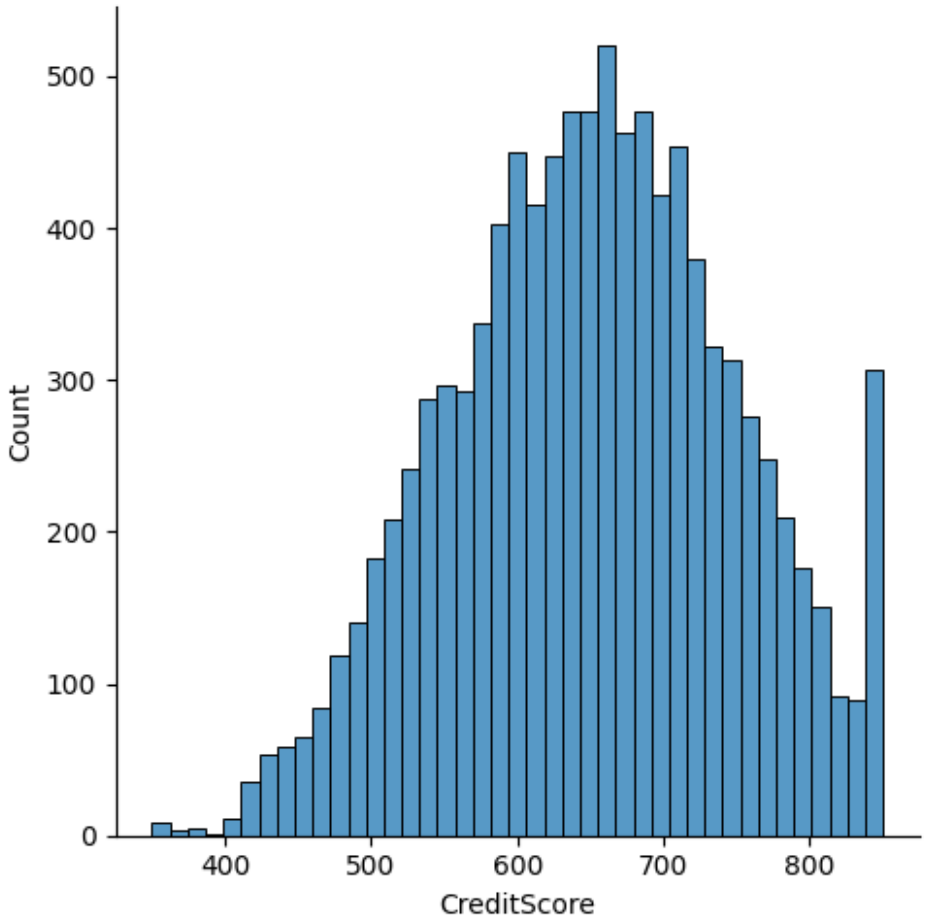
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib import rcParams

df=pd.read_csv('Churn_Modelling.csv')
df.head()
```

```
   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


   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
```

Univariate analysis

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

```
<seaborn.axisgrid.FacetGrid at 0x26e7513e1a0>
```

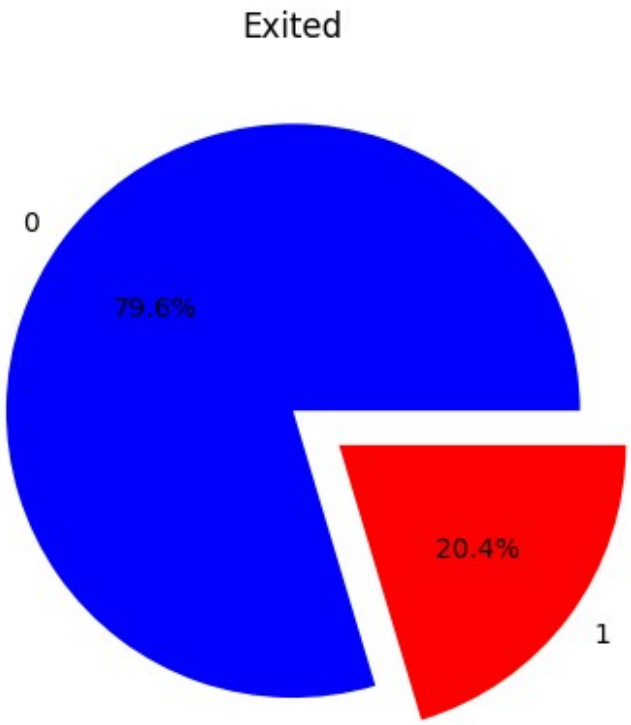



```
df.Exited.value_counts()
```

```
0    7963
1    2037
Name: Exited, dtype: int64
```

```
plt.pie(df.Exited.value_counts(),
[0,0.2],labels=['0','1'],autopct="%1.1f%%",colors=['blue','red'])
plt.title('Exited')
```

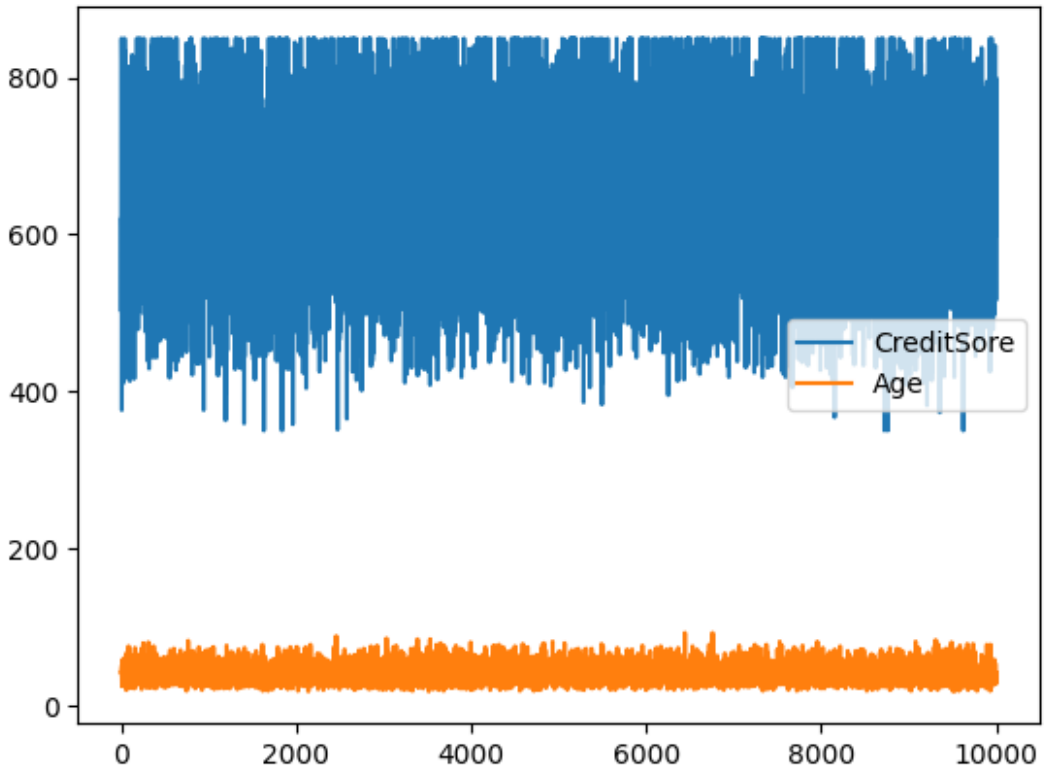
```
Text(0.5, 1.0, 'Exited')
```



Bi-variate analysis

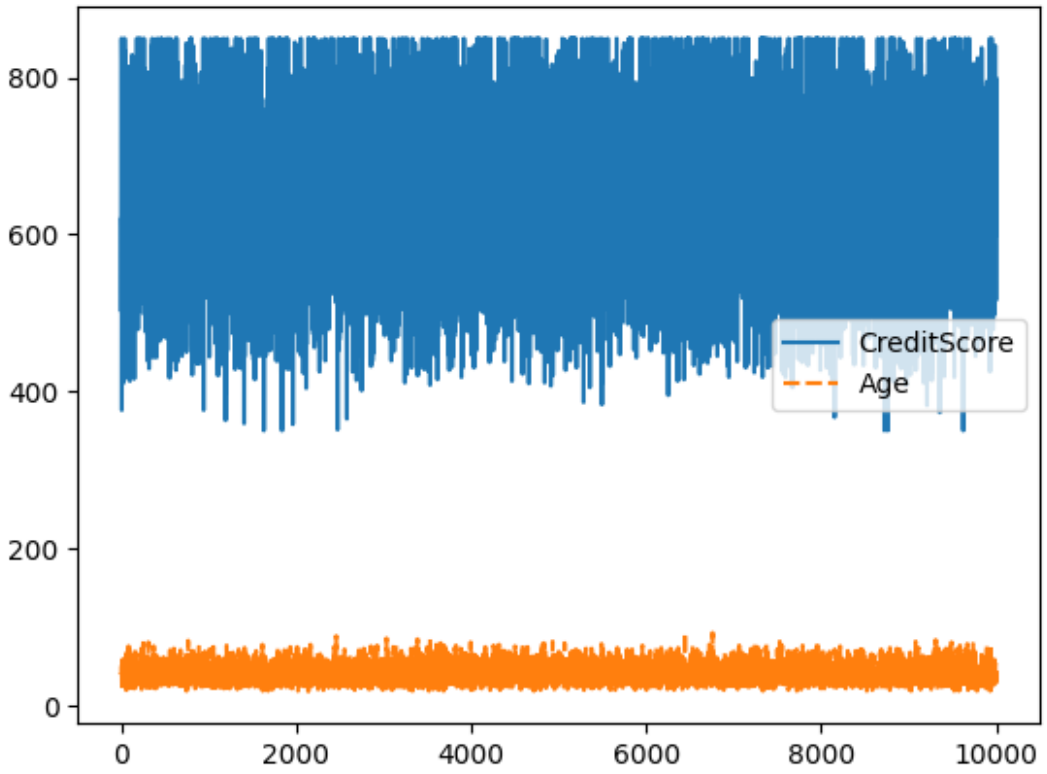
```
df.CreditScore.plot()
df.Age.plot()
plt.legend(['CreditSore','Age'])
```

```
<matplotlib.legend.Legend at 0x26e75285210>
```




```
sns.lineplot([df.CreditScore,df.Age])
```

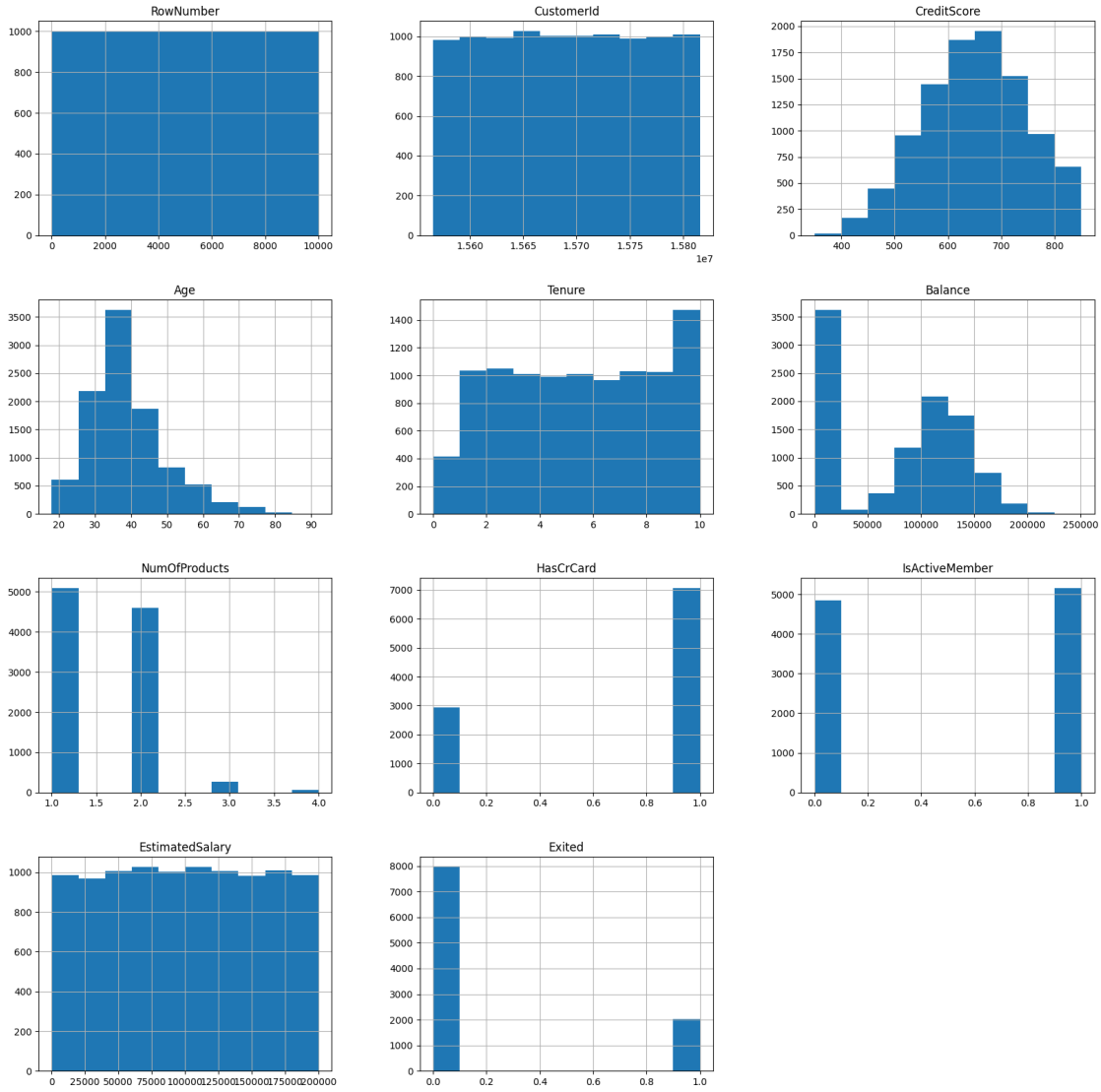
```
<AxesSubplot:>
```



Multivariate Analysis

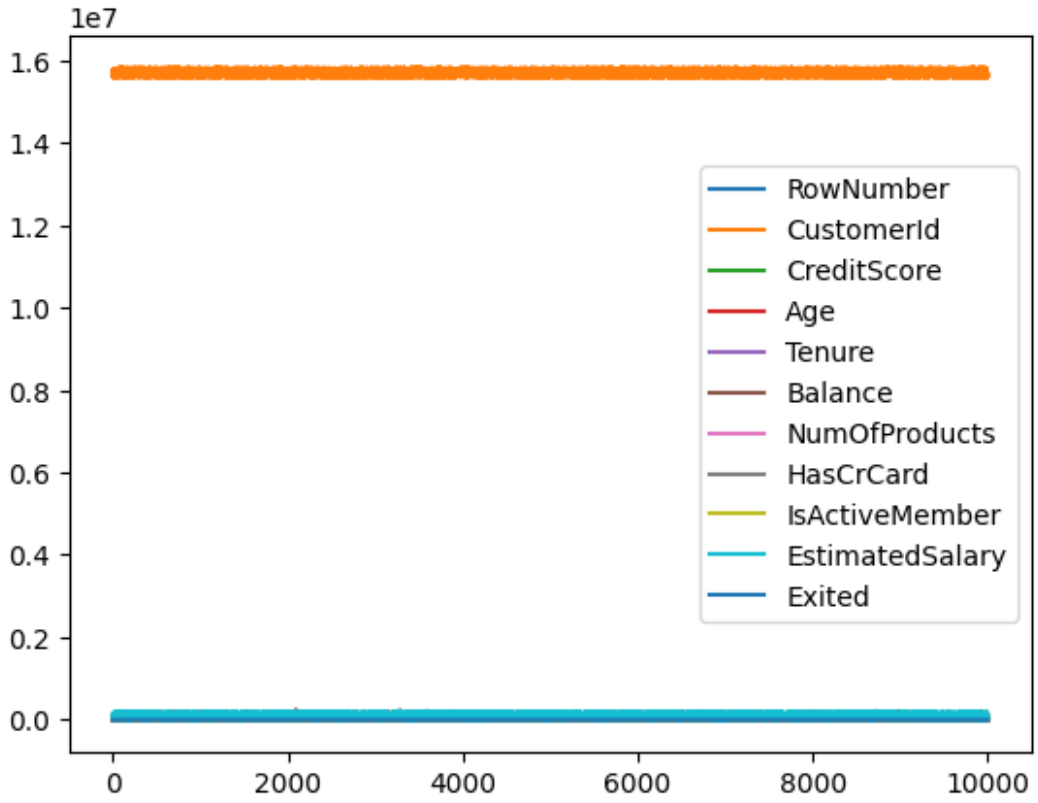
```
df.hist(figsize=(20,20))
```

```
array([[<AxesSubplot:title={'center':'RowNumber'}>,
        <AxesSubplot:title={'center':'CustomerId'}>,
        <AxesSubplot:title={'center':'CreditScore'}>],
       [<AxesSubplot:title={'center':'Age'}>,
        <AxesSubplot:title={'center':'Tenure'}>,
        <AxesSubplot:title={'center':'Balance'}>],
       [<AxesSubplot:title={'center':'NumOfProducts'}>,
        <AxesSubplot:title={'center':'HasCrCard'}>,
        <AxesSubplot:title={'center':'IsActiveMember'}>],
       [<AxesSubplot:title={'center':'EstimatedSalary'}>,
        <AxesSubplot:title={'center':'Exited'}>, <AxesSubplot:>]],
      dtype=object)
```




```
df.plot()
```

```
<AxesSubplot:>
```




```
sns.pairplot(df)
```

```
<seaborn.axisgrid.PairGrid at 0x26e7a407cd0>
```


```
df.isnull().any()
```

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

```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 14 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   RowNumber        10000 non-null  int64
 1   CustomerId       10000 non-null  int64
 2   Surname          10000 non-null  object
 3   CreditScore      10000 non-null  int64
 4   Geography        10000 non-null  object
 5   Gender           10000 non-null  object
 6   Age              10000 non-null  int64
 7   Tenure           10000 non-null  int64
 8   Balance          10000 non-null  float64
 9   NumOfProducts    10000 non-null  int64
 10  HasCrCard        10000 non-null  int64
 11  IsActiveMember   10000 non-null  int64
 12  EstimatedSalary  10000 non-null  float64
 13  Exited           10000 non-null  int64
dtypes: float64(2), int64(9), object(3)
memory usage: 1.1+ MB

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
```

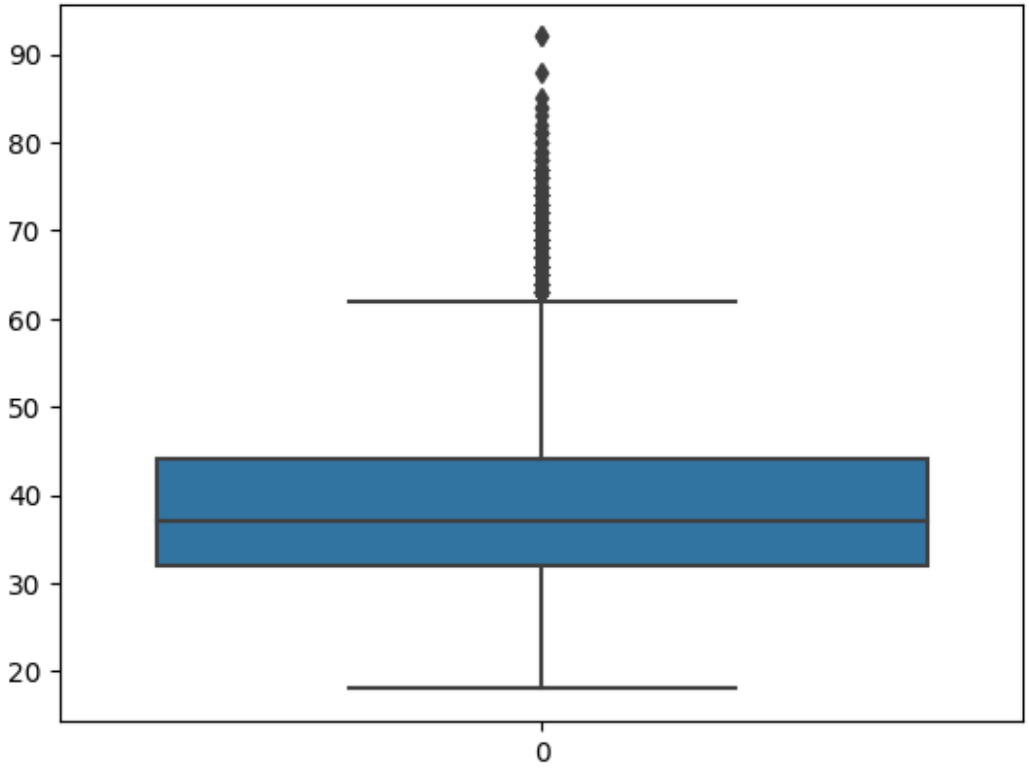
Outliear Detection

```
df.shape
```

```
(10000, 14)
```

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

```
<AxesSubplot:>
```




```
q1=df.Age.quantile(0.25)  #(Q1)
q3=df.Age.quantile(0.75)

IQR=q3-q1
```

```
upper_limit= q3 + 1.5*IQR

lower_limit= q1 - 1.5*IQR

upper_limit
```

```
62.0
```

```
lower_limit
```

```
14.0
```

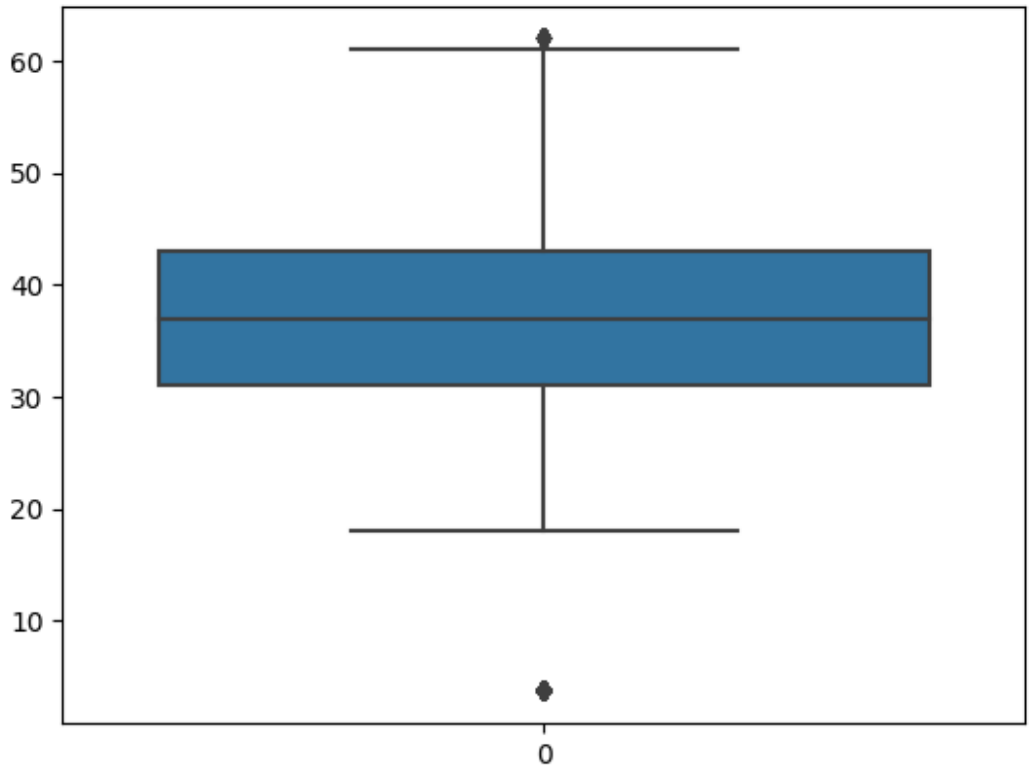
```
df.median()
```

```
C:\Users\nojma\AppData\Local\Temp\ipykernel_4040\530051474.py:1:
FutureWarning: Dropping of nuisance 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.
  df.median()
```

```
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
```

```
df['Age']= np.where(df['Age']>upper_limit,3.7,df['Age'])

sns.boxplot(df.Age)
```

```
<AxesSubplot:>
```




```
df.shape
```

```
(10000, 14)
```

The Categorical columns and perform encoding.

```
df.head()
```

```
   RowNumber  CustomerId   Surname  CreditScore Geography  Gender
Age  \
0          1    15634602  Hargrave          619    France  Female
42.0
1          2    15647311      Hill          608     Spain  Female
41.0
2          3    15619304      Onio          502    France  Female
42.0
3          4    15701354      Boni          699    France  Female
39.0
4          5    15737888  Mitchell          850     Spain  Female
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
```

```
df.Surname.value_counts()
```

```
Smith       32
Scott       29
Martin      29
Walker      28
Brown       26
            ..
Izmailov     1
Bold         1
Bonham       1
Poninski     1
Burbidge     1
Name: Surname, Length: 2932, dtype: int64
```

```
df.Gender.value_counts()
```

```
Male      5457
Female    4543
Name: Gender, dtype: int64
```

```
df.Geography.value_counts()
```

```
France     5014
Germany    2509
Spain      2477
Name: Geography, dtype: int64
```

```
from sklearn.preprocessing import LabelEncoder
```

```
le=LabelEncoder()
```

```
df.Gender=le.fit_transform(df.Gender)
df.Geography=le.fit_transform(df.Geography)
df.Surname=le.fit_transform(df.Surname)
```

```
df.head()
```

```
   RowNumber  CustomerId  Surname  CreditScore  Geography  Gender
Age  \
0          1    15634602     1115          619          0       0
42.0
1          2    15647311     1177          608          2       0
41.0
```

```
2          3    15619304     2040          502          0       0
42.0
3          4    15701354      289          699          0       0
39.0
4          5    15737888     1822          850          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
```

```
df.tail()
```

```
      RowNumber  CustomerId  Surname  CreditScore  Geography  Gender
Age  \
9995       9996    15606229     1999          771          0       1
39.0
9996       9997    15569892     1336          516          0       1
35.0
9997       9998    15584532     1570          709          0       0
36.0
9998       9999    15682355     2345          772          1       1
42.0
9999      10000    15628319     2751          792          0       0
28.0

      Tenure    Balance  NumOfProducts  HasCrCard  IsActiveMember  \
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
9995         96270.64       0
9996        101699.77       0
9997         42085.58       1
9998         92888.52       1
9999         38190.78       0
```

Split the Data into Dependent and Independent variables.

```
y=df['EstimatedSalary']
y
```

```
0       101348.88
1       112542.58
2       113931.57
3        93826.63
4        79084.10
          ...
9995     96270.64
9996    101699.77
9997     42085.58
9998     92888.52
9999     38190.78
Name: EstimatedSalary, Length: 10000, dtype: float64
```

```
x=df.drop(columns=['EstimatedSalary'],axis=1)
x
```

```
      RowNumber  CustomerId  Surname  CreditScore  Geography  Gender
Age  \
0             1    15634602     1115          619          0       0
42.0
1             2    15647311     1177          608          2       0
41.0
2             3    15619304     2040          502          0       0
42.0
3             4    15701354      289          699          0       0
39.0
4             5    15737888     1822          850          2       0
43.0
...         ...         ...      ...          ...        ...     ...
...
9995       9996    15606229     1999          771          0       1
39.0
9996       9997    15569892     1336          516          0       1
35.0
9997       9998    15584532     1570          709          0       0
36.0
9998       9999    15682355     2345          772          1       1
42.0
9999      10000    15628319     2751          792          0       0
28.0

      Tenure    Balance  NumOfProducts  HasCrCard  IsActiveMember
Exited
0          2       0.00              1          1               1
1
1          1   83807.86              1          0               1
```

```
0
2          8  159660.80              3          1               0
1
3          1       0.00              2          0               0
0
4          2  125510.82              1          1               1
0
...      ...        ...            ...        ...             ...
...
9995       5       0.00              2          1               0
0
9996      10   57369.61              1          1               1
0
9997       7       0.00              1          0               1
1
9998       3   75075.31              2          1               0
1
9999       4  130142.79              1          1               0
0

[10000 rows x 13 columns]
```

Scaling

```
from sklearn.preprocessing import scale

x_scaled=pd.DataFrame(scale(x),columns=x.columns)
x_scaled.head()
```

```
   RowNumber  CustomerId   Surname  CreditScore  Geography
Gender  \
0  -1.731878   -0.783213 -0.464183    -0.326221  -0.901886 -1.095988

1  -1.731531   -0.606534 -0.390911    -0.440036   1.515067 -1.095988

2  -1.731185   -0.995885  0.628988    -1.536794  -0.901886 -1.095988

3  -1.730838    0.144767 -1.440356     0.501521  -0.901886 -1.095988

4  -1.730492    0.652659  0.371354     2.063884   1.515067 -1.095988


        Age    Tenure   Balance  NumOfProducts  HasCrCard
IsActiveMember  \
0  0.506703 -1.041760 -1.225848      -0.911583   0.646092
0.970243
1  0.413424 -1.387538  0.117350      -0.911583  -1.547768
0.970243
2  0.506703  1.032908  1.333053       2.527057   0.646092       -
1.030670
3  0.226868 -1.387538 -1.225848       0.807737  -1.547768       -
```

```
1.030670
4  0.599981 -1.041760  0.785728      -0.911583   0.646092
0.970243
```

```
     Exited
0  1.977165
1 -0.505775
2  1.977165
3 -0.505775
4 -0.505775
```

Split the Train Test split

```
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,random_state=0)
```

```
X_train.shape
```

```
(7000, 13)
```

```
y_train.shape
```

```
(7000,)
```

```
X_test.shape
```

```
(3000, 13)
```

```
y_test.shape
```

```
(3000,)
```