```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
from sklearn.preprocessing import scale

##loading dataset

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

data.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
```

```
data.tail()
```

```
      RowNumber  CustomerId    Surname  CreditScore Geography  Gender
Age  \
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
```

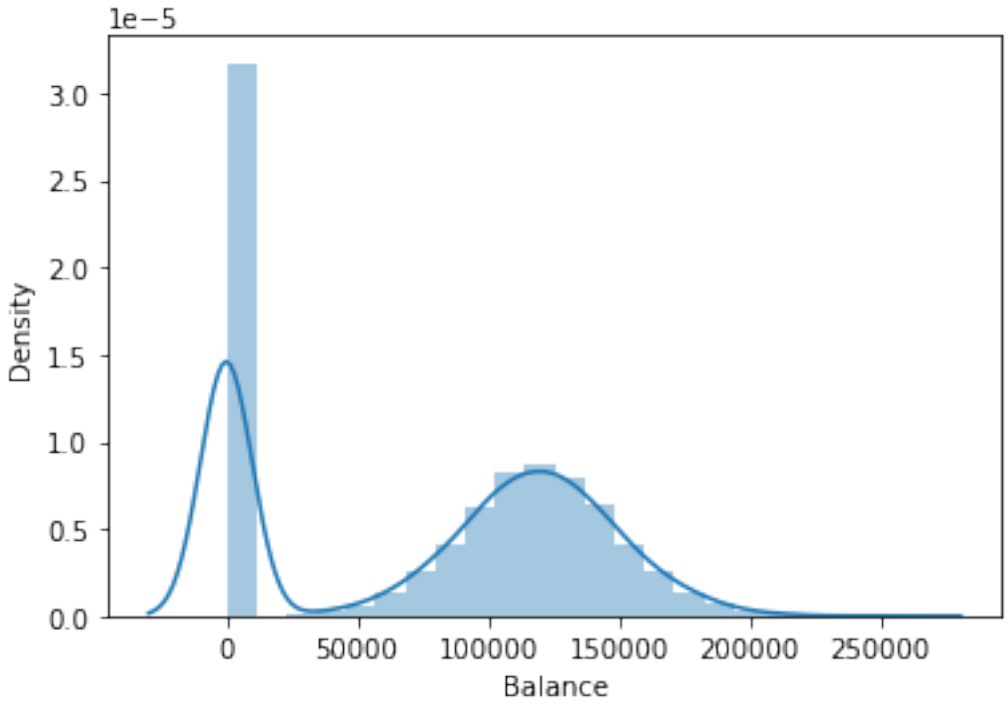
```
9999      10000    15628319     Walker          792    France  Female
28

      Tenure    Balance  NumOfProducts  HasCrCard  IsActiveMember  \
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
```

```
  #univariate analysis
```

```
sns.distplot(data.Balance)
```

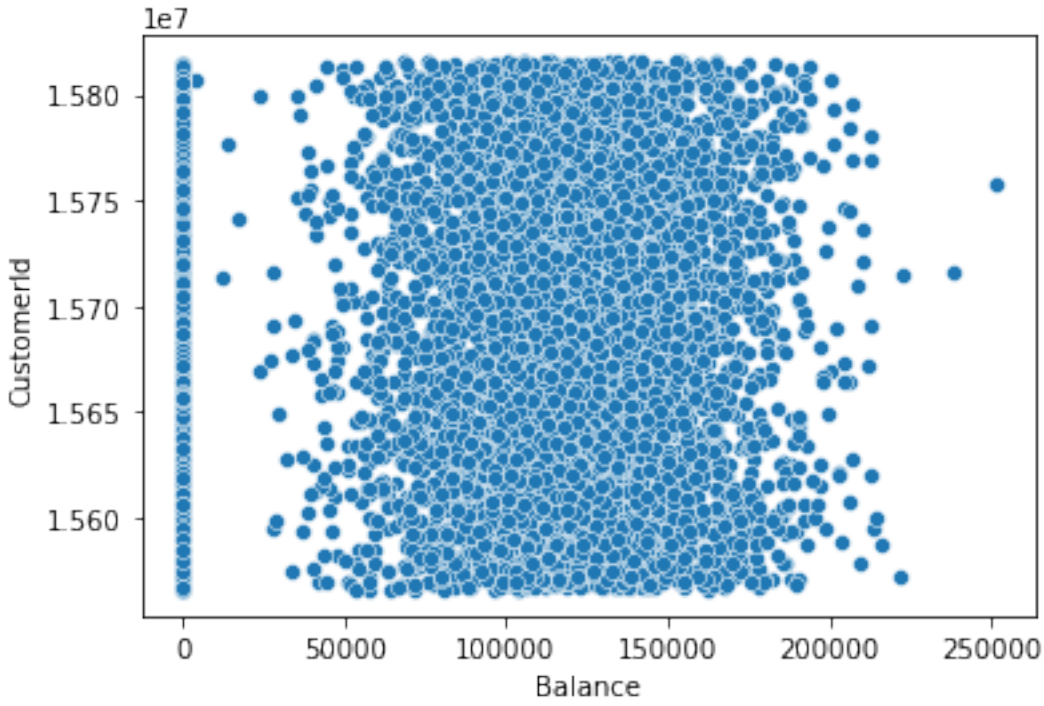
```
C:\Users\amarnath\anaconda3\lib\site-packages\seaborn\
distributions.py:2619: FutureWarning: `distplot` is a deprecated
function and will be removed in a future version. Please adapt your
code to use either `displot` (a figure-level function with similar
flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

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



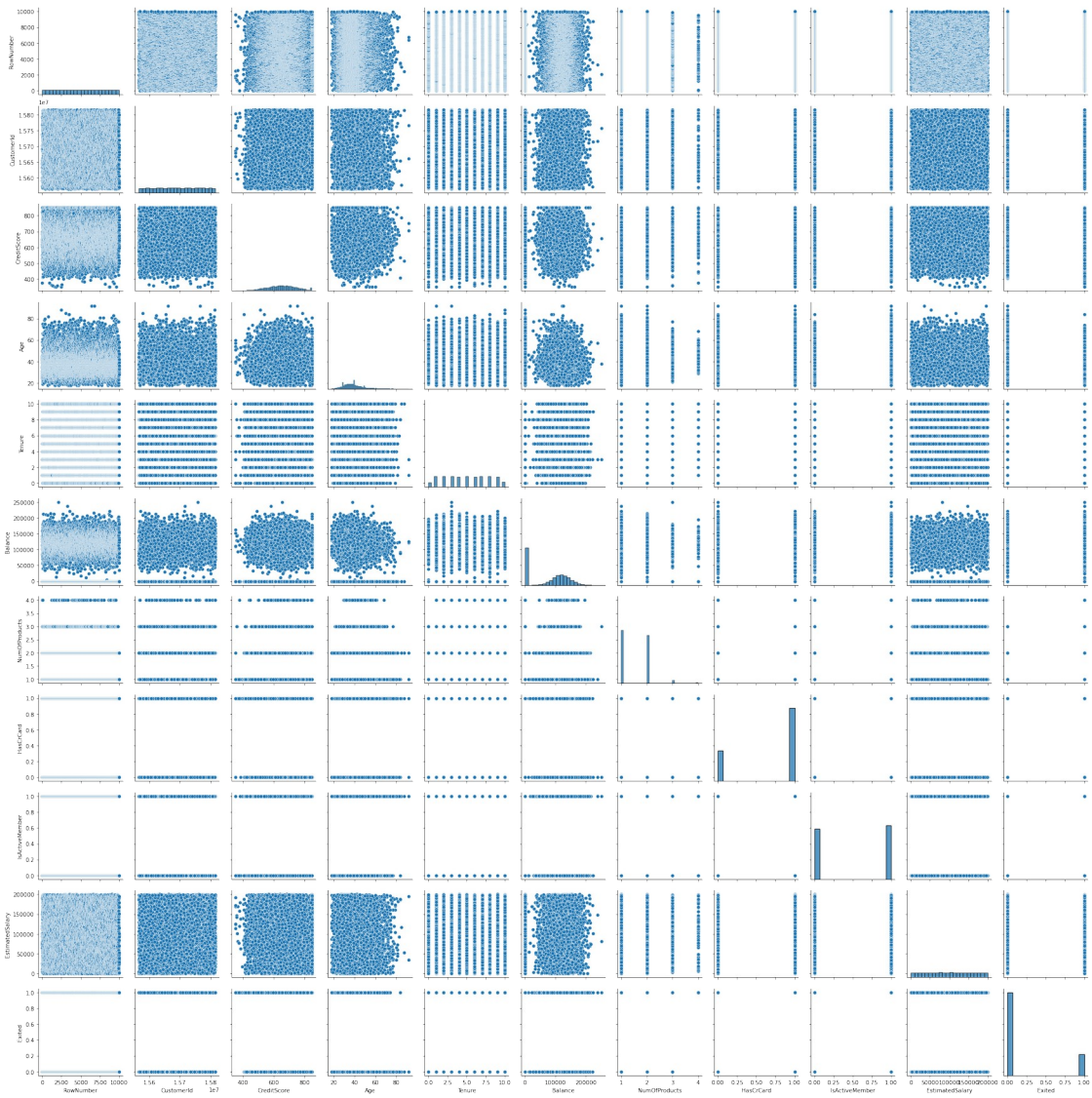

```
#bivariate analysis

sns.scatterplot(data.Balance,data.CustomerId)
```

```
C:\Users\amarnath\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='Balance', ylabel='CustomerId'>
```




```
#multivariate analysis
sns.pairplot(data)
```

```
<seaborn.axisgrid.PairGrid at 0x1e27ba06820>
```



```
#descriptive statics
```

```
data.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
```

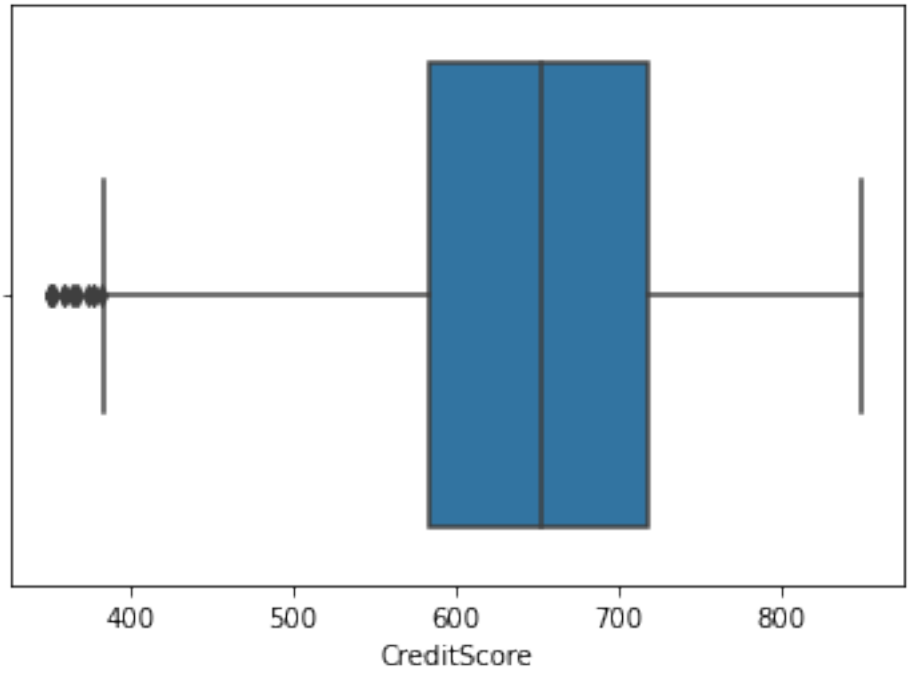
```
5.000000
75%     7500.25000  1.575323e+07    718.000000     44.000000
7.000000
max    10000.00000  1.581569e+07    850.000000     92.000000
10.000000

             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
```

```
#handling the missing values

data.isna().sum()
```

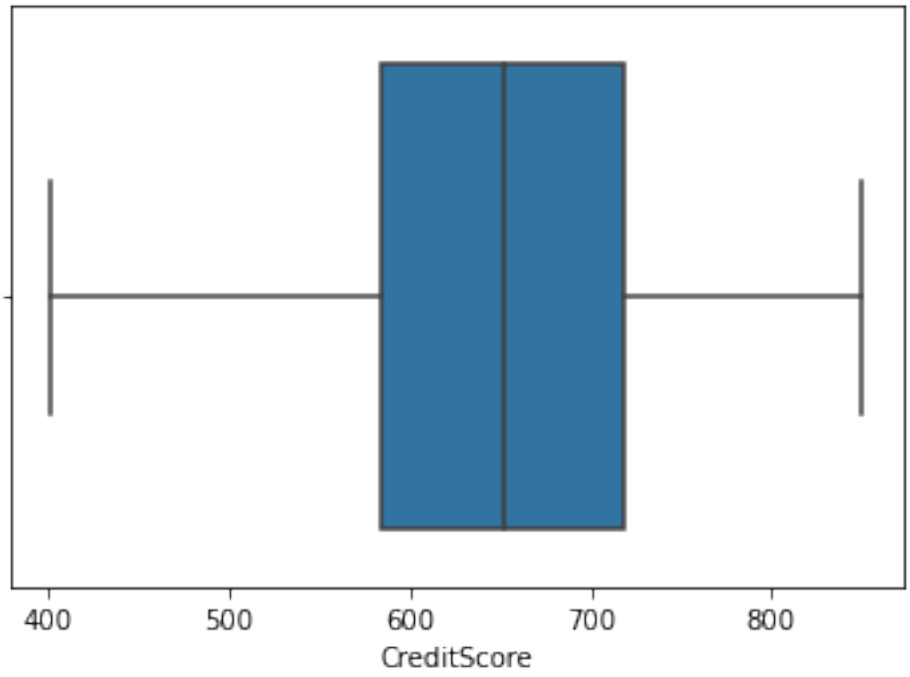
```
RowNumber          0
CustomerId         0
Surname            0
CreditScore        0
Geography          0
Gender             0
Age                0
Tenure             0
Balance            0
NumOfProducts      0
HasCrCard          0
IsActiveMember     0
EstimatedSalary    0
Exited             0
dtype: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

```
#handling outliers

sns.boxplot(data['CreditScore'])
```

```
C:\Users\amarnath\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'>
```




```
data['CreditScore']=np.where(data['CreditScore']<400,650,data['CreditS
core'])

sns.boxplot(data['CreditScore'])

C:\Users\amarnath\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'>
```




```
#encoding

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

data.tail()
```

```
      RowNumber  CustomerId    Surname  CreditScore Geography  Gender
Age  \
9995       9996    15606229   Obijiaku          771    France       1
39
9996       9997    15569892  Johnstone          516    France       1
35
9997       9998    15584532        Liu          709    France       0
36
9998       9999    15682355  Sabbatini          772   Germany       1
42
9999      10000    15628319     Walker          792    France       0
28

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

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

      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 the independent variables


x=data.drop(columns=['Geography'])
```

```
      RowNumber  CustomerId    Surname  CreditScore  Gender  Age
Tenure  \
0             1    15634602   Hargrave          619       0   42
2
1             2    15647311       Hill          608       0   41
1
2             3    15619304       Onio          502       0   42
8
3             4    15701354       Boni          699       0   39
1
4             5    15737888   Mitchell          850       0   43
2
...         ...         ...        ...          ...     ...  ...     .
..
9995       9996    15606229   Obijiaku          771       1   39
5
9996       9997    15569892  Johnstone          516       1   35
10
9997       9998    15584532        Liu          709       0   36
7
```

```
9998       9999    15682355  Sabbatini          772       1   42
3
9999      10000    15628319     Walker          792       0   28
4

        Balance  NumOfProducts  HasCrCard  IsActiveMember
EstimatedSalary  \
0          0.00              1          1               1
101348.88
1      83807.86              1          0               1
112542.58
2     159660.80              3          1               0
113931.57
3          0.00              2          0               0
93826.63
4     125510.82              1          1               1
79084.10
...         ...            ...        ...             ...
...
9995       0.00              2          1               0
96270.64
9996   57369.61              1          1               1
101699.77
9997       0.00              1          0               1
42085.58
9998   75075.31              2          1               0
92888.52
9999  130142.79              1          1               0
38190.78

      Exited
0          1
1          0
2          1
3          0
4          0
...      ...
9995       0
9996       0
9997       1
9998       1
9999       0

[10000 rows x 13 columns]
```

```
x=data.drop(columns=['Surname','Geography'])
```

```
x
```

```
      RowNumber  CustomerId  CreditScore  Gender  Age  Tenure
Balance  \
0             1    15634602          619       0   42       2
0.00
1             2    15647311          608       0   41       1
83807.86
2             3    15619304          502       0   42       8
159660.80
3             4    15701354          699       0   39       1
0.00
4             5    15737888          850       0   43       2
125510.82
...         ...         ...          ...     ...  ...     ...        .
..
9995       9996    15606229          771       1   39       5
0.00
9996       9997    15569892          516       1   35      10
57369.61
9997       9998    15584532          709       0   36       7
0.00
9998       9999    15682355          772       1   42       3
75075.31
9999      10000    15628319          792       0   28       4
130142.79

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

```
[10000 rows x 12 columns]
```

```
x=scale(x)
```

```
x
```

```
array([[-1.73187761, -0.78321342, -0.33452426, ...,  0.97024255,
         0.02188649,  1.97716468],
       [-1.7315312 , -0.60653412, -0.44928208, ...,  0.97024255,
         0.21653375, -0.50577476],
       [-1.73118479, -0.99588476, -1.55513017, ..., -1.03067011,
         0.2406869 ,  1.97716468],
       ...,
       [ 1.73118479, -1.47928179,  0.60440336, ...,  0.97024255,
        -1.00864308,  1.97716468],
       [ 1.7315312 , -0.11935577,  1.26165269, ..., -1.03067011,
        -0.12523071,  1.97716468],
       [ 1.73187761, -0.87055909,  1.47030328, ..., -1.03067011,
        -1.07636976, -0.50577476]])
```

```
# Splitting 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,y,test_size=0.2,rando
m_state=0)
```

```
x_train.shape
```

```
(8000, 12)
```

```
x_test.shape
```

```
(2000, 12)
```

```
y_test.shape
```

```
(2000,)
```