

ASSIGNMENT 2

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

import warnings
warnings.filterwarnings(action = 'ignore')

from google.colab import files
uploaded = files.upload()

<IPython.core.display.HTML object>

Saving Churn_Modelling.csv to Churn_Modelling.csv

import io
data= pd.read_csv(io.BytesIO(uploaded['Churn_Modelling.csv']))

data.dtypes

RowNumber          int64
CustomerId          int64
Surname            object
CreditScore         int64
Geography          object
Gender             object
Age                int64
Tenure             int64
Balance            float64
NumOfProducts      int64
HasCrCard           int64
IsActiveMember     int64
EstimatedSalary    float64
Exited             int64
dtype: object

data.select_dtypes(include=['int64','float64','Int64']).dtypes

RowNumber          int64
CustomerId          int64
CreditScore         int64
Age                int64
Tenure             int64
Balance            float64
NumOfProducts      int64
HasCrCard           int64
IsActiveMember     int64
EstimatedSalary    float64
```

```
Exited          int64
dtype: object
```

```
data.groupby(['Surname']).agg({'RowNumber':'count', 'Exited':'mean'})
                                ).reset_index().sort_values(by='RowNumber',
ascending=False).head()
```

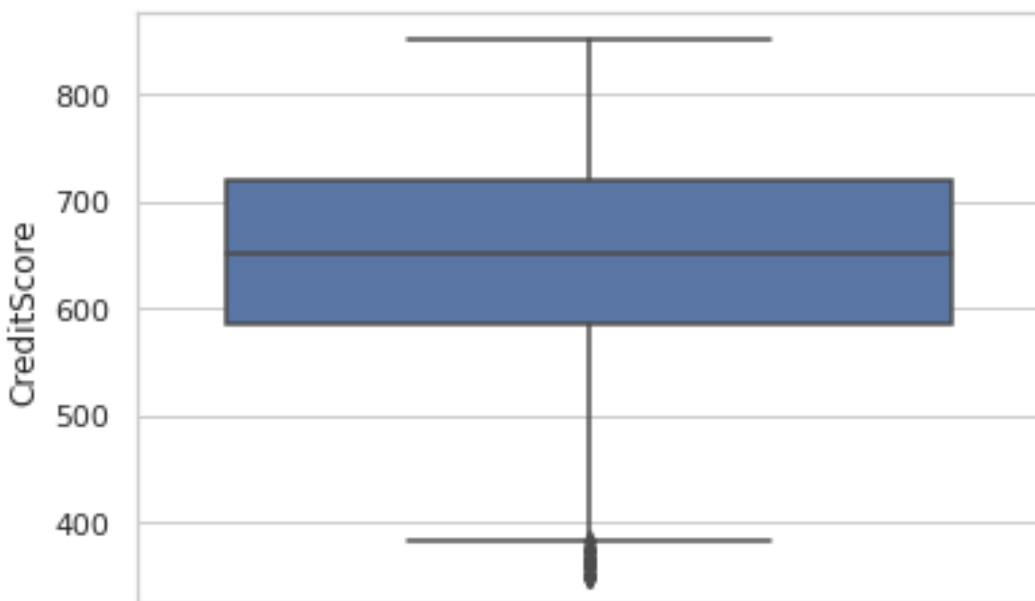
	Surname	RowNumber	Exited
2473	Smith	32	0.281250
1689	Martin	29	0.310345
2389	Scott	29	0.103448
2751	Walker	28	0.142857
336	Brown	26	0.192308

```
#univariate analysis
```

```
sns.set(style="whitegrid")
```

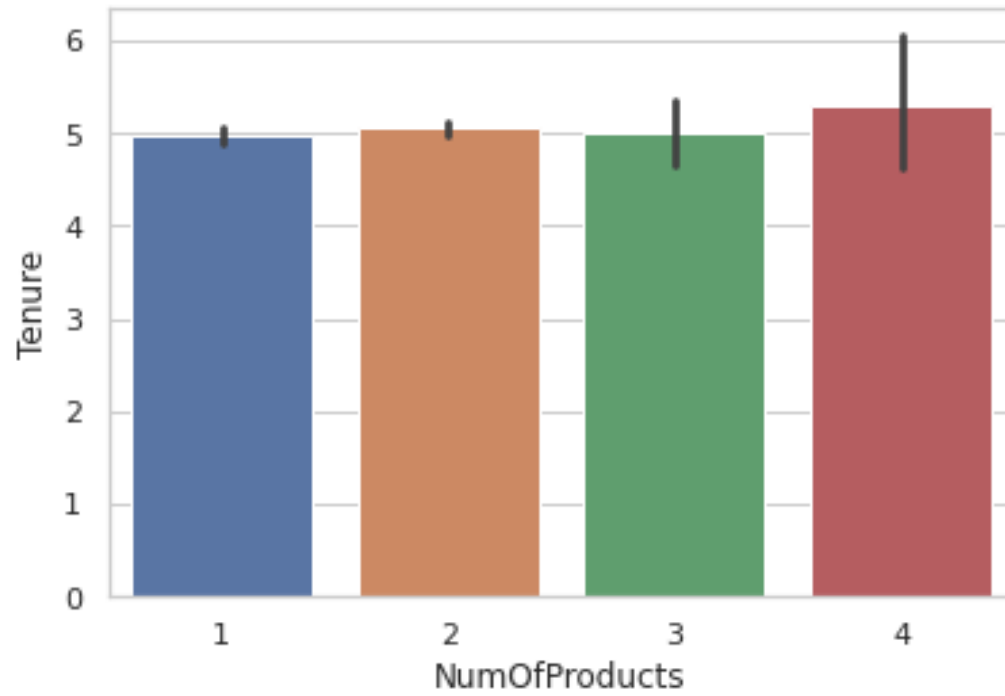
```
sns.boxplot(y=data['CreditScore'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f9b2a005750>
```

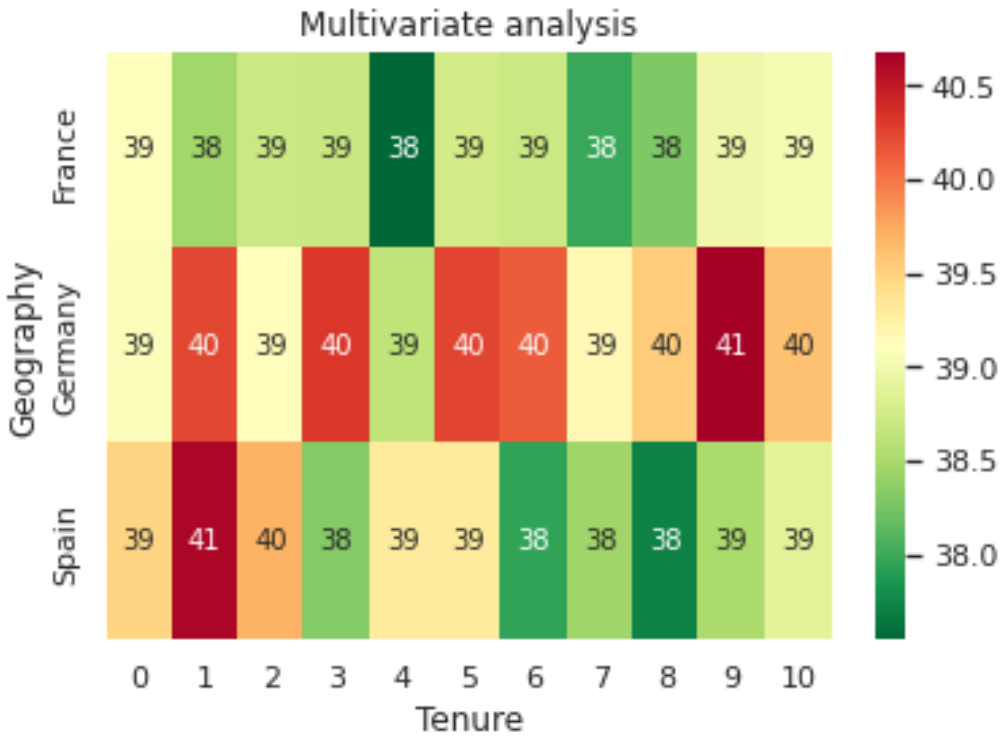


```
sns.barplot(x=data.NumOfProducts,y=data.Tenure)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f9b29b02390>
```



```
#multivariate  
result = pd.pivot_table(data=data, index='Geography',  
columns='Tenure', values='Age')  
sns.heatmap(result, annot=True, cmap = 'RdYlGn_r').set_title('Multivariate  
analysis')  
plt.show()
```



```
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	10000.000000	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

```
data.describe(include=['object'])
```

	Surname	Geography	Gender
count	10000	10000	10000
unique	2932	3	2
top	Smith	France	Male
freq	32	5014	5457

```
data['Age'].mode()
```

```
0    37
dtype: int64
```

```
data["Age"].mean()
```

```
38.9218
```

```
m=round(data["Age"].mean())
```

```
data["Age"].median()
```

```
37.0
```

```
s=round(data['Age'].std())
print(s)
```

```
10
```

```
#check 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
```

#Find the outliers and replace the outlier

```
CreditScores = data['CreditScore']
```

```
CreditScores
```

```
0      619
1      608
2      502
3      699
4      850
```

```
...
```

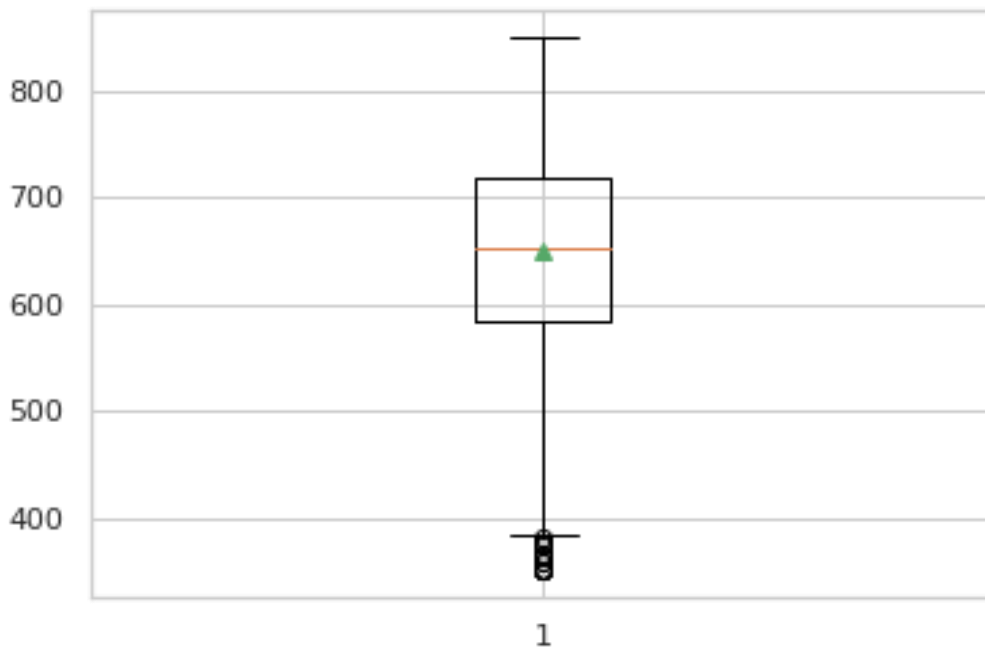
```
9995    771
9996    516
9997    709
9998    772
9999    792
```

```
Name: CreditScore, Length: 10000, dtype: int64
```

```
import matplotlib.pyplot as plt
```

```
plt.boxplot(data['CreditScore'],showmeans = True)
```

```
plt.show()
```



```
df = data[data['CreditScore'] >= 378]
```

```
for i in data['CreditScore']:
```

```
    if(i<378):
```

```
        print(i)
```

```
print(data['CreditScore'])
```

```
376
```

```
376
```

```
363
```

```

359
350
350
358
351
365
367
350
350
373
350
0      619
1      608
2      502
3      699
4      850
...
9995   771
9996   516
9997   709
9998   772
9999   792
Name: CreditScore, Length: 10000, dtype: int64

```

```

b = data['Balance']
b

```

```

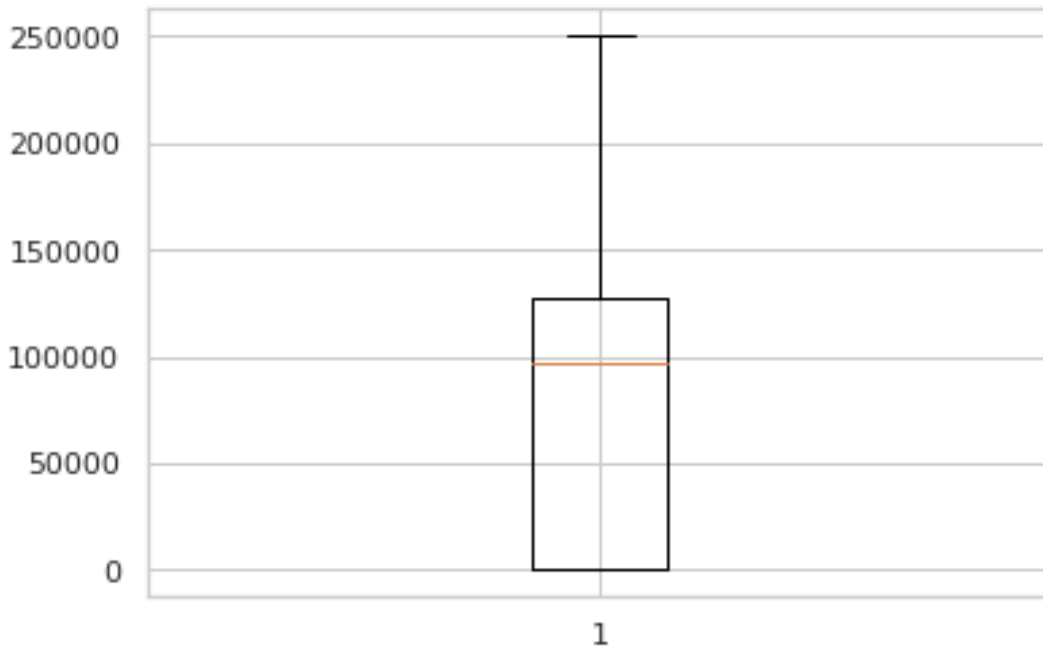
0      0.00
1    83807.86
2   159660.80
3      0.00
4   125510.82
...
9995   0.00
9996   57369.61
9997   0.00
9998   75075.31
9999  130142.79
Name: Balance, Length: 10000, dtype: float64

```

```

plt.boxplot(b)
plt.show()

```



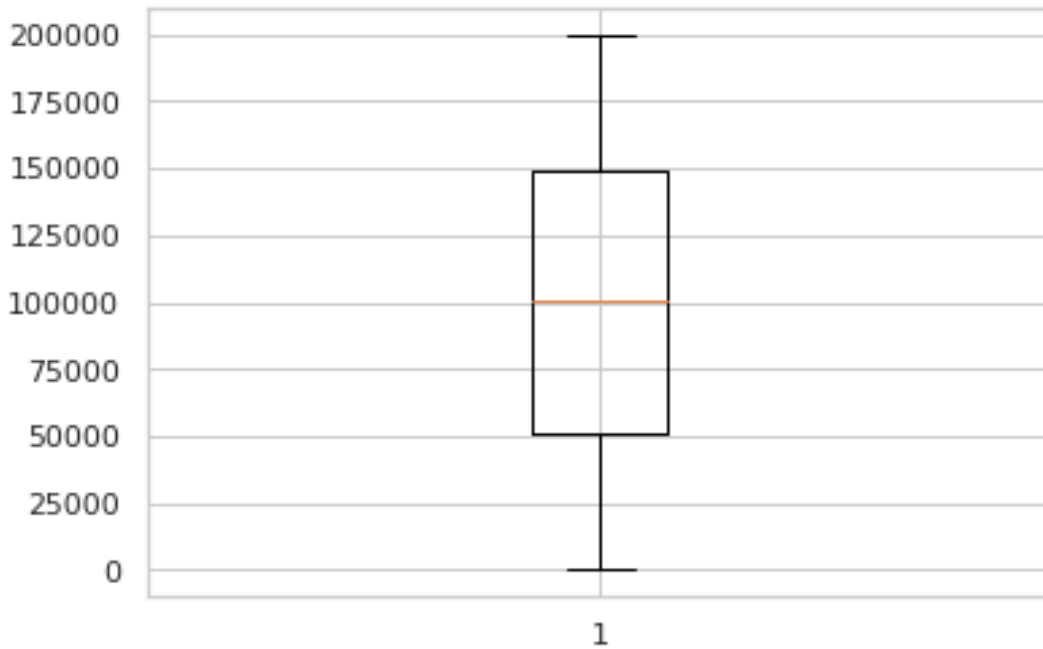
```
e= df['EstimatedSalary']  
e
```

```
0      101348.88  
1      112542.58  
2      113931.57  
3       93826.63  
4       79084.10
```

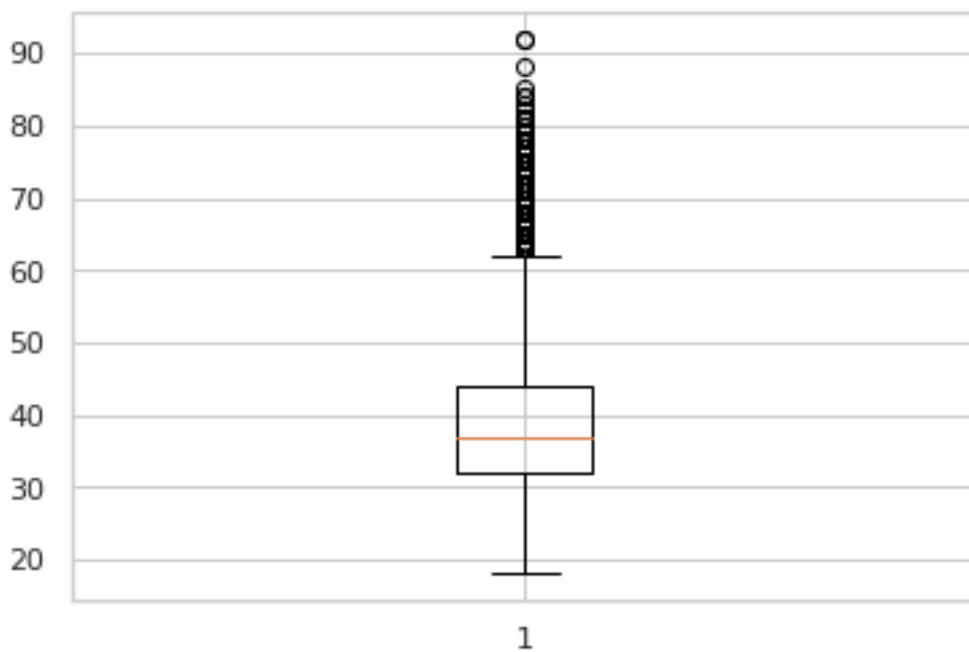
```
...  
9995    96270.64  
9996   101699.77  
9997    42085.58  
9998    92888.52  
9999    38190.78
```

```
Name: EstimatedSalary, Length: 9986, dtype: float64
```

```
plt.boxplot(e)  
plt.show()
```

```
a = data['Age']
plt.boxplot(a)
plt.show()
```



```
ageOutliers = np.where(df['Age'] > 60)
ageOutliers
```

```
(array([ 41,  43,  57,  84, 103, 157, 180, 229, 233, 242, 251,
        275, 309, 363, 370, 384, 386, 398, 415, 483, 537, 558,
```

560, 566, 601, 611, 616, 629, 657, 677, 695, 735, 765,
768, 806, 810, 822, 858, 883, 887, 920, 927, 946, 950,
955, 961, 967, 995, 1007, 1037, 1038, 1053, 1112, 1116, 1190,
1202, 1231, 1232, 1243, 1249, 1275, 1282, 1325, 1339, 1384, 1403,
1406, 1429, 1435, 1453, 1515, 1539, 1584, 1603, 1610, 1637, 1785,
1805, 1852, 1860, 1895, 1898, 1901, 1927, 1974, 1989, 1995, 2005,
2032, 2046, 2071, 2087, 2096, 2101, 2147, 2152, 2157, 2237, 2254,
2267, 2291, 2294, 2426, 2431, 2451, 2452, 2511, 2512, 2525, 2533,
2545, 2590, 2606, 2650, 2661, 2704, 2708, 2751, 2763, 2768, 2769,
2772, 2782, 2846, 2868, 2892, 2899, 2916, 2917, 2999, 3024, 3045,
3101, 3133, 3157, 3183, 3194, 3220, 3296, 3299, 3302, 3305, 3308,
3337, 3357, 3359, 3369, 3373, 3375, 3378, 3387, 3394, 3425, 3453,
3488, 3490, 3518, 3522, 3532, 3540, 3550, 3554, 3564, 3566, 3584,
3593, 3632, 3637, 3638, 3642, 3681, 3682, 3693, 3710, 3719, 3724,
3752, 3765, 3804, 3817, 3871, 3872, 3879, 3900, 3901, 3918, 3931,
3938, 3971, 3985, 4001, 4016, 4039, 4042, 4086, 4133, 4138, 4148,
4153, 4161, 4232, 4235, 4247, 4264, 4271, 4288, 4304, 4309, 4326,
4351, 4357, 4369, 4378, 4387, 4426, 4429, 4454, 4481, 4482, 4492,
4497, 4550, 4554, 4581, 4586, 4635, 4669, 4689, 4738, 4742, 4792,
4806, 4823, 4840, 4922, 4938, 4957, 4983, 4991, 5011, 5024, 5029,
5059, 5123, 5127, 5139, 5150, 5188, 5214, 5216, 5226, 5246, 5290,
5304, 5359, 5368, 5396, 5430, 5448, 5481, 5499, 5505, 5511, 5567,
5568, 5572, 5630, 5642, 5646, 5651, 5655, 5662, 5674, 5689, 5733,
5768, 5774, 5808, 5816, 5831, 5858, 5898, 5948, 5987, 6037, 6107,
6143, 6157, 6158, 6162, 6164, 6203, 6221, 6269, 6280, 6306, 6348,
6357, 6364, 6366, 6401, 6434, 6506, 6521, 6523, 6572, 6603, 6617,
6697, 6700, 6706, 6712, 6750, 6754, 6803, 6890, 6961, 6988, 6999,
7048, 7049, 7054, 7062, 7069, 7085, 7129, 7130, 7133, 7147, 7185,
7193, 7229, 7234, 7263, 7293, 7353, 7366, 7383, 7490, 7505, 7514,
7517, 7539, 7543, 7614, 7615, 7620, 7659, 7678, 7683, 7685, 7700,
7706, 7710, 7711, 7718, 7764, 7767, 7775, 7779, 7793, 7804, 7842,
7885, 7889, 7900, 7924, 7947, 7986, 8010, 8028, 8085, 8089, 8146,
8160, 8183, 8197, 8205, 8207, 8294, 8311, 8375, 8384, 8434, 8448,
8457, 8459, 8468, 8478, 8552, 8558, 8567, 8592, 8664, 8676, 8679,
8701, 8748, 8750, 8751, 8756, 8775, 8781, 8810, 8853, 8888, 8905,
8918, 8958, 9006, 9009, 9050, 9068, 9090, 9100, 9104, 9150, 9162,
9211, 9249, 9267, 9272, 9280, 9297, 9306, 9309, 9312, 9320, 9321,
9339, 9367, 9378, 9389, 9412, 9415, 9425, 9459, 9477, 9493, 9542,
9544, 9569, 9574, 9576, 9580, 9582, 9632, 9657, 9659, 9667, 9672,
9674, 9704, 9719, 9720, 9722, 9733, 9739, 9751, 9818, 9865, 9880,
9883, 9922]),)

```
da = data[data['Age'] <=60 ]
da
```

	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

[9536 rows x 14 columns]

#Check for Categorical columns and perform encoding.

data

	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]

```
from pandas.api.types import is_string_dtype
continuous=[]
categorical=[]
for data1 in data:
    if is_string_dtype(data[data1]):
        categorical.append(data1)
    else:
        continuous.append(data1)
categorical
```

['Surname', 'Geography', 'Gender']

#Split the data into dependent and independent variables.

```
x = data.iloc[:, 0:1].values
y = data.iloc[:, 1]
print(x)
print(y)
```

```
[[ 1]
 [ 2]
```

```
[ 3]
...
[ 9998]
[ 9999]
[10000]]
0      15634602
1      15647311
2      15619304
3      15701354
4      15737888
...
9995    15606229
9996    15569892
9997    15584532
9998    15682355
9999    15628319
Name: CustomerId, Length: 10000, dtype: int64
```

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(x)
X_test = sc.transform(x)
```

#Split 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(data,y,test_size=0.2)

print(X_train)
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	\
4909	4910	15787258	Ross	596	Spain	Female	29	
8461	8462	15758769	Coffey	625	France	Female	44	
19	20	15568982	Hao	726	France	Female	24	
1952	1953	15781884	Knox	624	Germany	Male	27	
4986	4987	15582090	Iroawuchi	684	Spain	Female	36	
...	
7812	7813	15695280	Hung	532	Germany	Male	24	
7208	7209	15596165	Degtyarev	547	Germany	Male	25	
7220	7221	15706637	Chang	718	Spain	Male	40	
8586	8587	15694039	Jen	650	Germany	Female	46	
5166	5167	15694644	Wood	455	Spain	Female	43	

	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	\
4909	6	0.00	2	1	0	
8461	7	0.00	1	1	0	
19	6	0.00	2	1	1	
1952	9	94667.29	2	0	1	
4986	4	0.00	1	1	0	
...	
7812	8	142755.25	1	0	0	
7208	4	98141.57	2	1	1	

7220	9	0.00	2	0	0
8586	9	149003.76	2	1	0
5166	6	0.00	1	1	1

	EstimatedSalary	Exited
4909	116696.77	0
8461	4791.80	0
19	54724.03	0
1952	4470.52	0
4986	117038.96	0
...
7812	34231.48	0
7208	52309.80	0
7220	121537.91	0
8586	176902.83	0
5166	81250.79	0

[8000 rows x 14 columns]

print(X_test)

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	\
6547	6548	15608760	Cox	656	France	Female	30	
1047	1048	15793949	Cheng	726	France	Female	48	
6031	6032	15743153	Singh	740	Germany	Female	40	
2553	2554	15732270	Hung	727	Spain	Male	71	
5752	5753	15697948	Henderson	752	Spain	Female	36	
...	
207	208	15679531	Collins	618	France	Male	34	
3838	3839	15778154	Kung	628	Germany	Male	50	
8583	8584	15715888	Allardyce	591	France	Female	38	
7570	7571	15791944	Harker	697	France	Male	32	
1780	1781	15601008	Stevenson	802	France	Male	33	

	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	\
6547	4	74323.20	1	1	1	
1047	4	0.00	1	1	0	
6031	2	122295.17	2	1	1	
2553	8	0.00	1	1	1	
5752	3	0.00	2	1	1	
...	
207	5	134954.53	1	1	1	
3838	4	122227.71	1	0	1	
8583	2	142289.28	1	0	1	
7570	7	175464.85	3	1	0	
1780	8	0.00	2	1	0	

	EstimatedSalary	Exited
6547	22929.08	0
1047	114020.06	1

6031	30812.84	0
2553	198446.91	1
5752	48505.10	0
...
207	151954.39	0
3838	14217.77	1
8583	119638.85	0
7570	116442.42	1
1780	143706.18	0

[2000 rows x 14 columns]

print(y_train)

4909	15787258
8461	15758769
19	15568982
1952	15781884
4986	15582090

...	
7812	15695280
7208	15596165
7220	15706637
8586	15694039
5166	15694644

Name: CustomerId, Length: 8000, dtype: int64

print(y_test)

6547	15608760
1047	15793949
6031	15743153
2553	15732270
5752	15697948

...	
207	15679531
3838	15778154
8583	15715888
7570	15791944
1780	15601008

Name: CustomerId, Length: 2000, dtype: int64