

Assignment – 2

Assignment Date	05 October 2022
Student Name	Abirami R
Student Roll Number	811519104005
Maximum Marks	2 Marks

```
import pandas as pd
import numpy as np
from scipy import stats
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
from sklearn.model_selection import train_test_split
```

```
churn=pd.read_csv("E:/Churn_Modelling.csv")
```

```
churn.head(10)
```

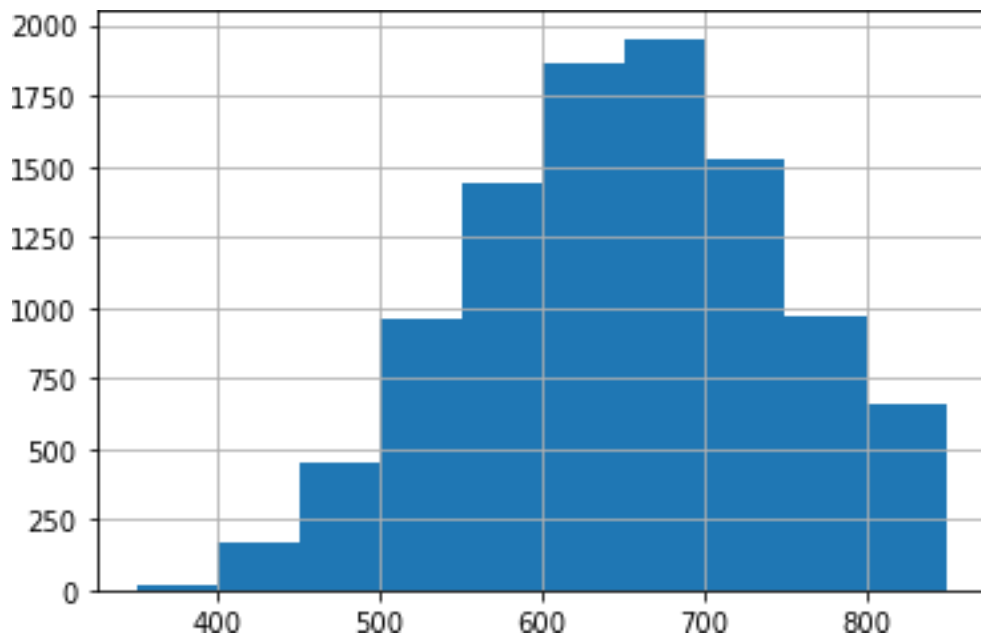
	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age
0	1	15634602	Hargrave	619	France	Female	42
1	2	15647311	Hill	608	Spain	Female	41
2	3	15619304	Onio	502	France	Female	42
3	4	15701354	Boni	699	France	Female	39
4	5	15737888	Mitchell	850	Spain	Female	43
5	6	15574012	Chu	645	Spain	Male	44
6	7	15592531	Bartlett	822	France	Male	50
7	8	15656148	Obinna	376	Germany	Female	29
8	9	15792365	He	501	France	Male	44
9	10	15592389	H?	684	France	Male	27

	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	
5	8	113755.78	2	1	0	
6	7	0.00	2	1	1	
7	4	115046.74	4	1	0	
8	4	142051.07	2	0	1	
9	2	134603.88	1	1	1	

2	113931.57	1
3	93826.63	0
4	79084.10	0
5	149756.71	1
6	10062.80	0
7	119346.88	1
8	74940.50	0
9	71725.73	0

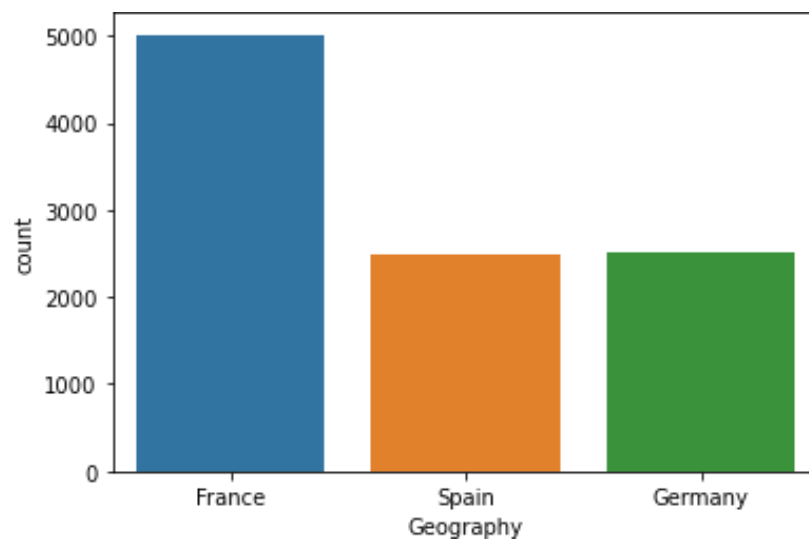
```
churn.CreditScore.hist()
```

```
<AxesSubplot:>
```



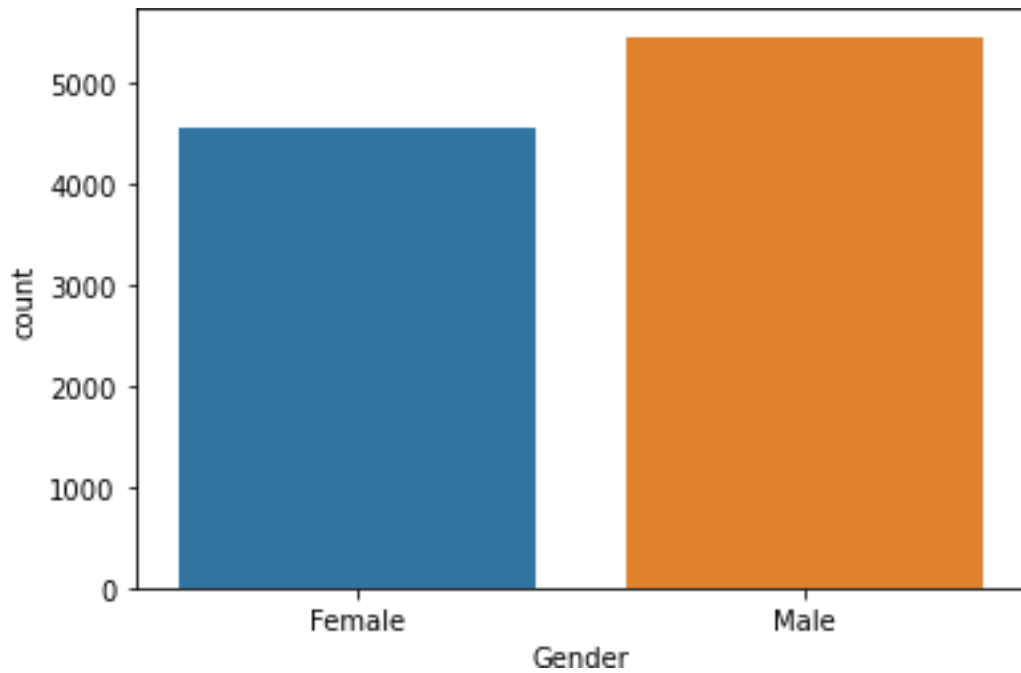
```
sns.countplot(x="Geography", data=churn)
```

```
<AxesSubplot:xlabel='Geography', ylabel='count'>
```



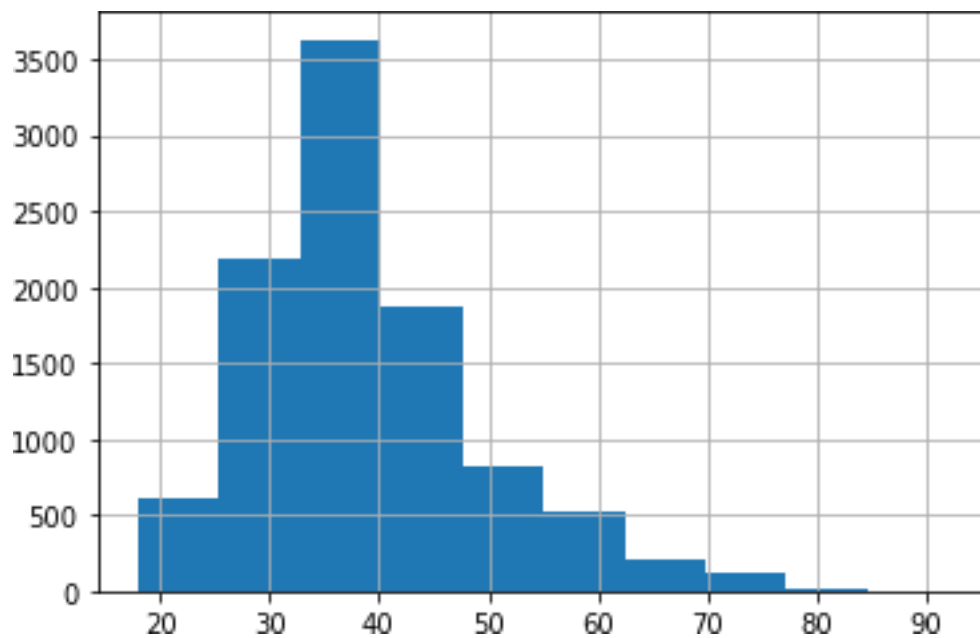
```
sns.countplot(x="Gender",data=churn)
```

```
<AxesSubplot:xlabel='Gender', ylabel='count'>
```



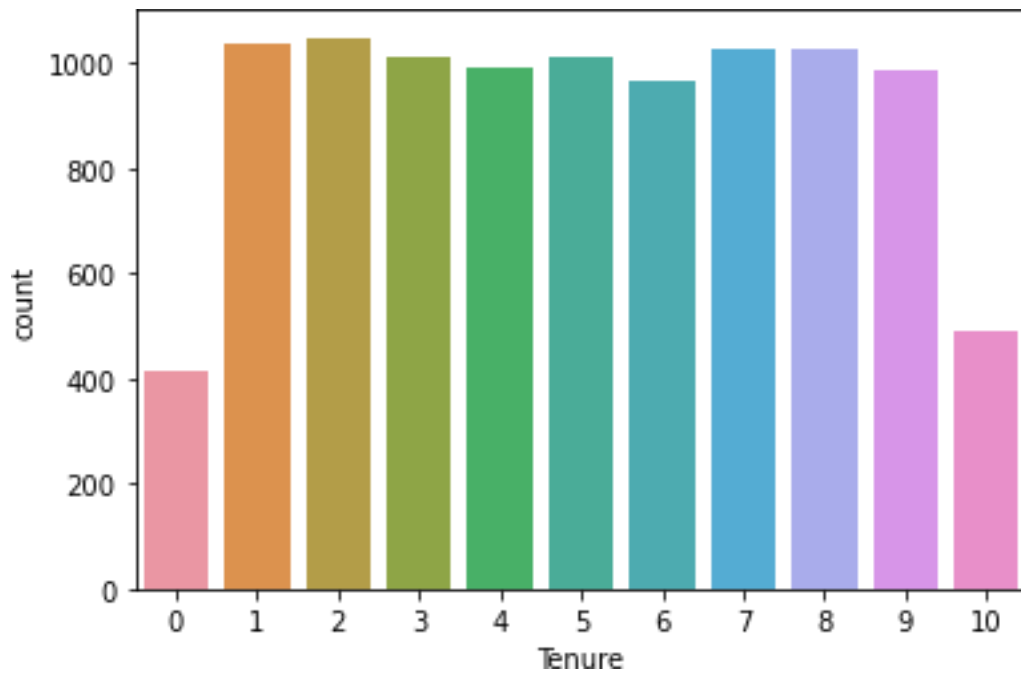
```
churn.Age.hist()
```

```
<AxesSubplot:>
```



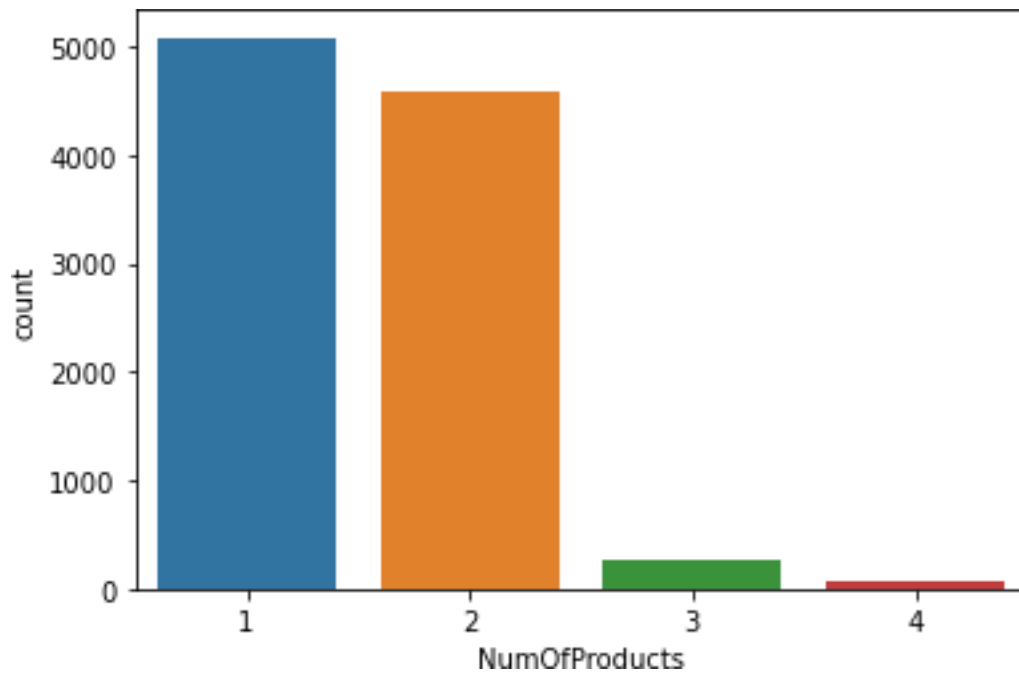
```
sns.countplot(x="Tenure",data=churn)
```

```
<AxesSubplot:xlabel='Tenure', ylabel='count'>
```



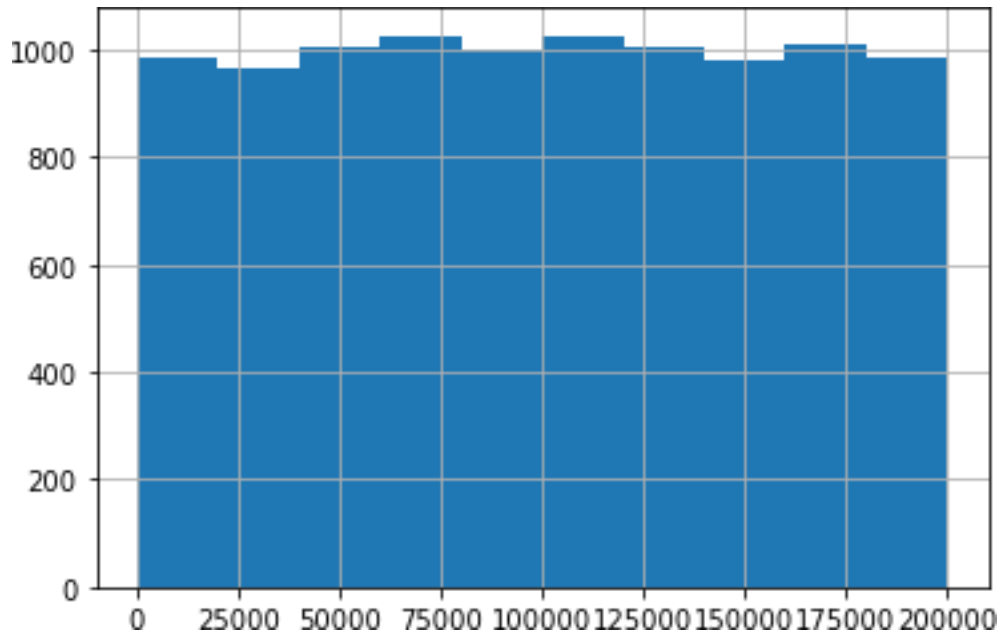
```
sns.countplot(x="NumOfProducts",data=churn)
```

```
<AxesSubplot:xlabel='NumOfProducts', ylabel='count'>
```

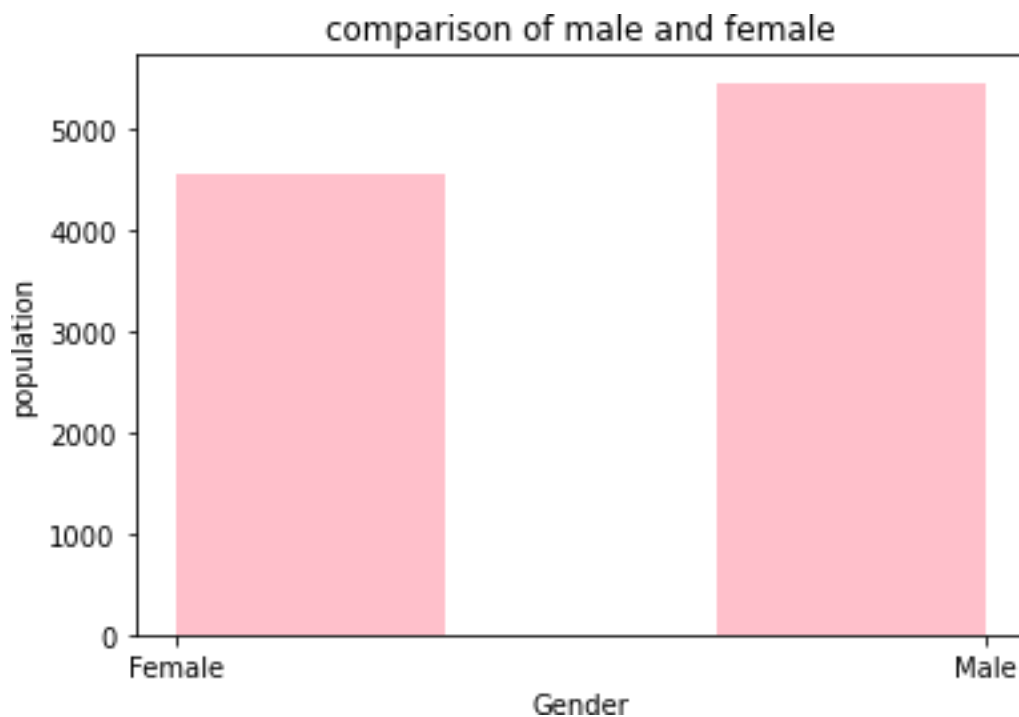


```
churn.EstimatedSalary.hist()
```

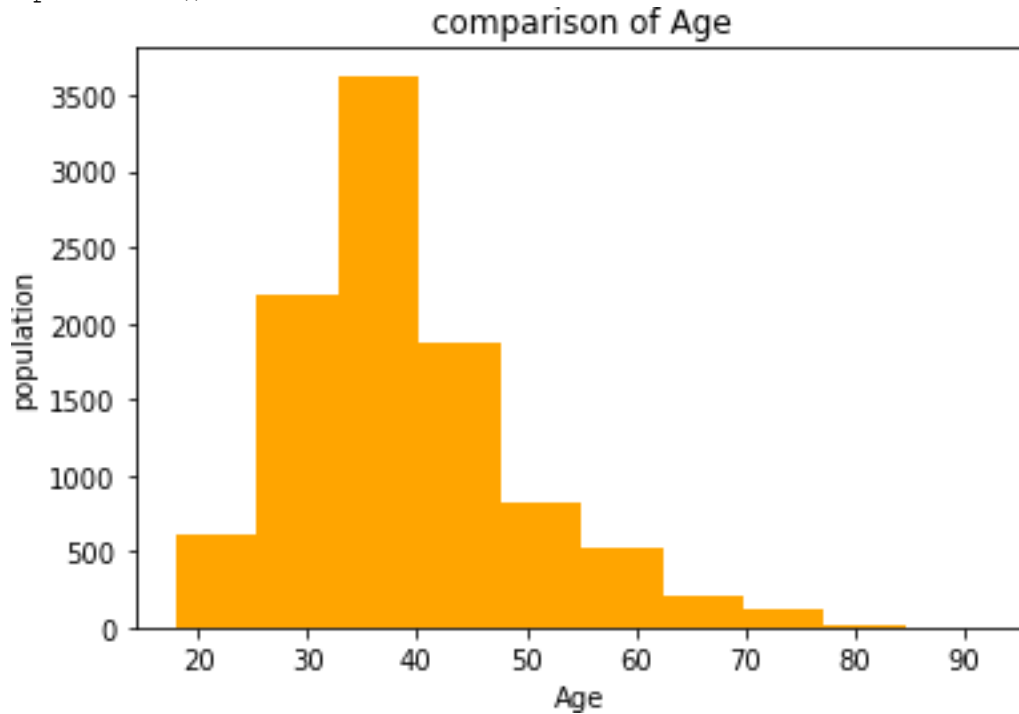
```
<AxesSubplot:>
```



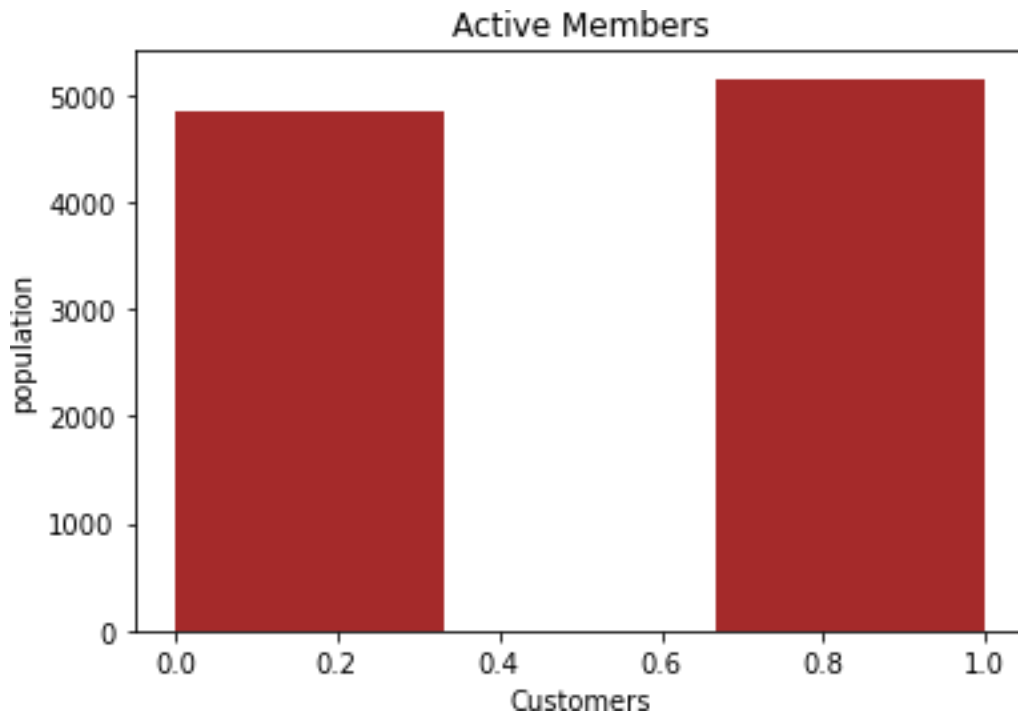
```
plt.hist(x = churn.Gender, bins = 3, color = 'pink')  
plt.title('comparison of male and female')  
plt.xlabel('Gender')  
plt.ylabel('population')  
plt.show()
```



```
plt.hist(x = churn.Age, bins = 10, color = 'orange')
plt.title('comparison of Age')
plt.xlabel('Age')
plt.ylabel('population')
plt.show()
```



```
plt.hist(x = churn.IsActiveMember, bins = 3, color = 'brown')
plt.title('Active Members')
plt.xlabel('Customers')
plt.ylabel('population')
plt.show()
```



```
churn.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
```

```
churn.CreditScore.describe()
```

```
count      10000.000000
mean         650.528800
std           96.653299
min          350.000000
25%          584.000000
50%          652.000000
75%          718.000000
max          850.000000
Name: CreditScore, dtype: float64
```

```
churn.Geography.describe()
```

```
count      10000
unique         3
top      France
freq       5014
Name: Geography, dtype: object
```

```
churn.Gender.describe()
```

```
count      10000
unique         2
top      Male
freq       5457
Name: Gender, dtype: object
```

```
churn.Age.describe()
```

```
count      10000.000000
```

```
mean      38.921800
std       10.487806
min       18.000000
25%      32.000000
50%      37.000000
75%      44.000000
max       92.000000
Name: Age, dtype: float64
```

```
churn.Tenure.describe()
```

```
count      10000.000000
mean        5.012800
std         1.892174
min         0.000000
Name: Tenure, dtype: float64
```

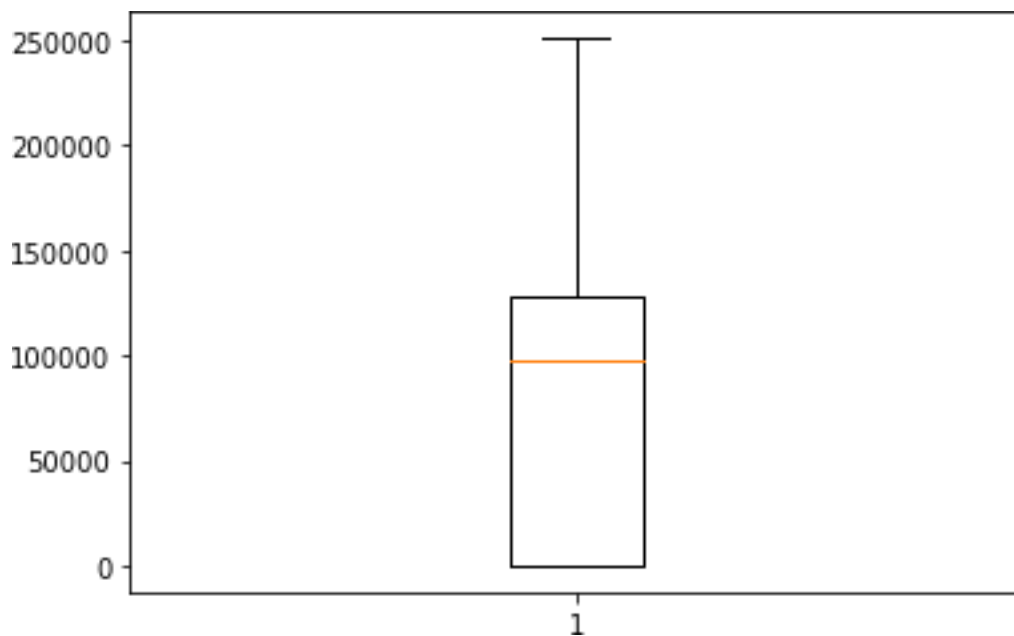
```
churn.EstimatedSalary.describe()
count      10000.000000
mean     100090.239881
std      57510.492818
min       11.580000
25%      51002.110000
50%     100193.915000
75%     149388.247500
max     199992.480000
Name: EstimatedSalary, dtype: float64
```

```
churn.isnull().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
```

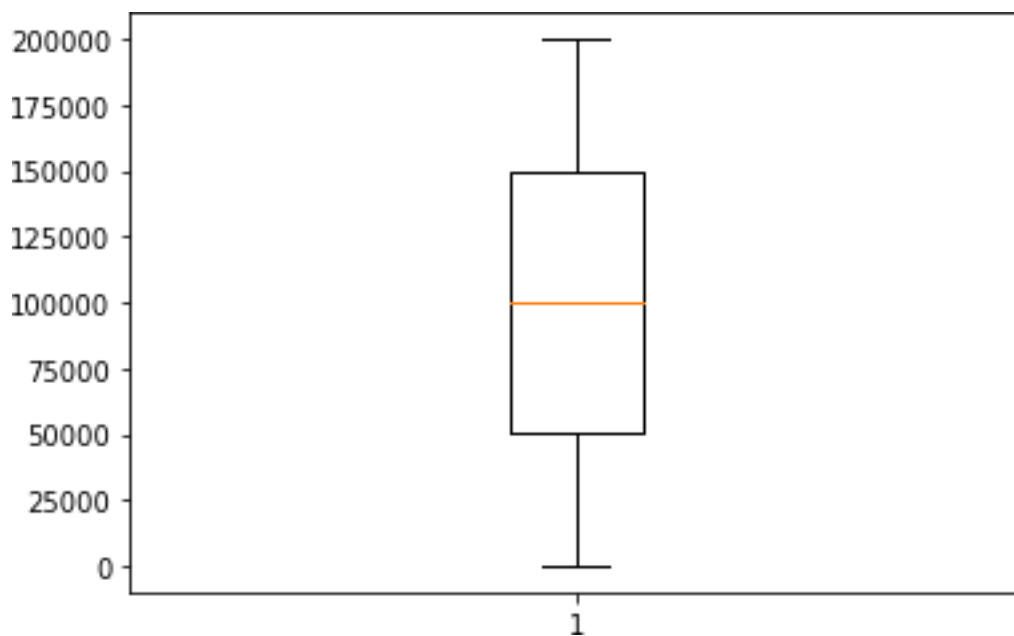
```
plt.boxplot(churn.Balance)
```

```
{'whiskers': [<matplotlib.lines.Line2D at 0x1c6e4427340>,
<matplotlib.lines.Line2D at 0x1c6e4427610>],
'caps': [<matplotlib.lines.Line2D at 0x1c6e44279a0>,
<matplotlib.lines.Line2D at 0x1c6e4427bb0>],
'boxes': [<matplotlib.lines.Line2D at 0x1c6e4427070>],
'medians': [<matplotlib.lines.Line2D at 0x1c6e4427e80>],
'fliers': [<matplotlib.lines.Line2D at 0x1c6e4439190>],
'means': []}
```

```
plt.boxplot(churn.EstimatedSalary)
```

```
{'whiskers': [<matplotlib.lines.Line2D at 0x1c6e4491670>,  
<matplotlib.lines.Line2D at 0x1c6e4491940>],  
'caps': [<matplotlib.lines.Line2D at 0x1c6e4491c10>,  
<matplotlib.lines.Line2D at 0x1c6e4491ee0>],  
'boxes': [<matplotlib.lines.Line2D at 0x1c6e4491370>],  
'medians': [<matplotlib.lines.Line2D at 0x1c6e449b1f0>],  
'fliers': [<matplotlib.lines.Line2D at 0x1c6e449b4c0>],  
'means': []}
```



```

from sklearn import preprocessing
label_encoder = preprocessing.LabelEncoder()
churn['Geography']= label_encoder.fit_transform(churn['Geography'])
churn['Gender']= label_encoder.fit_transform(churn['Gender'])

churn = churn.drop(['CustomerId', 'Surname', 'RowNumber'], axis = 1)

y=churn.Exited
churn.drop(['Exited'], axis = 1,inplace=True)
x=churn

from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler().fit(x)
scaled_data=scaler.transform(x)

x= pd.DataFrame(scaled_data)

x_train,x_test,y_train,y_test= train_test_split(x,y,train_size = 0.8,
test_size = 0.2,random_state =42)

x_train.head(5)

```

	0	1	2	3	4	5	6	7	8
9254	0.672	0.0	1.0	0.189189	0.6	0.000000	0.333333	1.0	1.0
1561	0.564	0.5	1.0	0.324324	0.4	0.476786	0.333333	1.0	1.0
1670	0.418	1.0	1.0	0.081081	0.3	0.457317	0.000000	1.0	0.0
6087	0.422	0.0	0.0	0.121622	0.9	0.540606	0.000000	1.0	0.0
6669	0.334	0.0	1.0	0.513514	0.9	0.566554	0.000000	0.0	0.0

```

y_train.head(5)

```

9254	0
1561	0
1670	1
6087	1
6669	1

```

Name: Exited, dtype: int64

x_test.head(5)

```

	0	1	2	3	4	5	6	7	8
6252	0.492	0.5	1.0	0.189189	0.3	0.385452	0.333333	0.0	0.0
4684	0.546	0.0	1.0	0.337838	0.1	0.000000	0.333333	1.0	1.0
1731	0.502	1.0	0.0	0.351351	0.4	0.000000	0.333333	1.0	0.0

```
0.292777
4742  0.312  0.5  1.0  0.554054  0.8  0.474902  0.333333  1.0  1.0
0.853422
4521  0.420  1.0  0.0  0.121622  0.7  0.498194  0.000000  1.0  1.0
0.573346
```

```
y_test.head(5)
```

```
6252    0
4684    0
1731    0
4742    0
4521    0
```

```
Name: Exited, dtype: int64
```