LOADING THE DATASET

file=("Churn_Modelling.csv")

import pandas as pd

dataset=pd.read_csv(file)

dataset

۸۵۵	RowNumbe	er Custome	rId	Surname	CreditScore	Geography	Gender
Age 0	\	1 15634	602	Hargrave	619	France	Female
42 1		2 15647	311	Hill	608	Spain	Female
41 2		3 15619	304	Onio	502	France	Female
42 3		4 15701	354	Boni	699	France	Female
39 4 43		5 15737	888	Mitchell	856	Spain	Female
9995	999	96 15606	229	0bijiaku	771	France	Male
39 9996	999	97 15569	892	Johnstone	516	France	Male
35 9997	999	98 15584	532	Liu	709	France	Female
36 9998	999	99 15682	355	Sabbatini	772	Germany	Male
42 9999 28	1000	00 15628	319	Walker	792	France	Female
0 1 2 3 4	Tenure 2 1 8 1 2	Balance 0.00 83807.86 159660.80 0.00 125510.82	Num	OfProducts 1 1 3 2 1	HasCrCard 1 0 1 0 1	IsActiveMem	ber \ 1 1 0 0 1
9995 9996 9997 9998 9999	 5 10 7 3 4	0.00 57369.61 0.00 75075.31 130142.79		2 1 1 2 1	1 1 0 1		 0 1 1 0 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]

dataset.dtypes

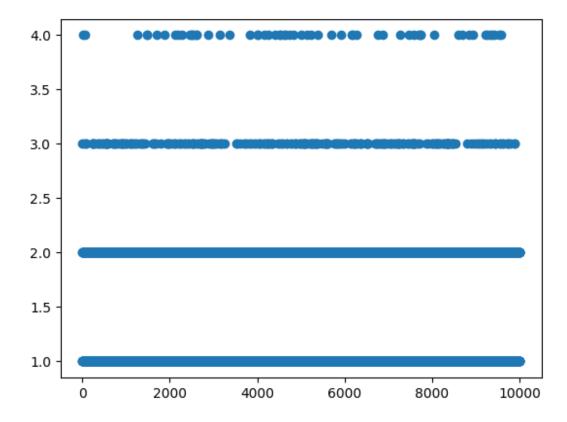
RowNumber	int64
CustomerId	int64
Surname	object
CreditScore	int64
Geography	object
Gender	object
Age	int64
Tenure	int64
Balance	float64
NumOfProducts	int64
HasCrCard	int64
IsActiveMember	int64
EstimatedSalary	float64
Exited	int64

dtype: object

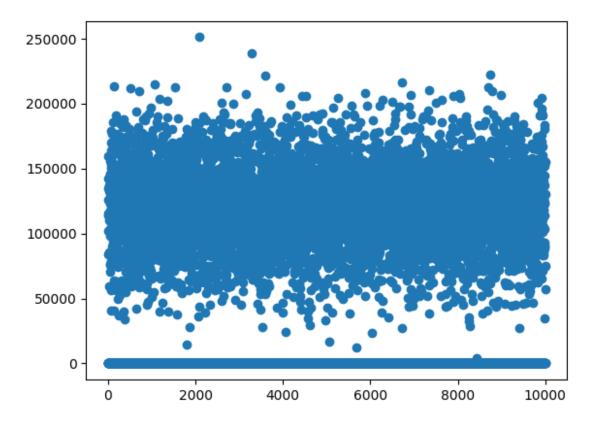
import matplotlib.pyplot as plt
import seaborn as sns

plt.scatter(dataset.index,dataset['NumOfProducts'])

<matplotlib.collections.PathCollection at 0x13a2831f0>

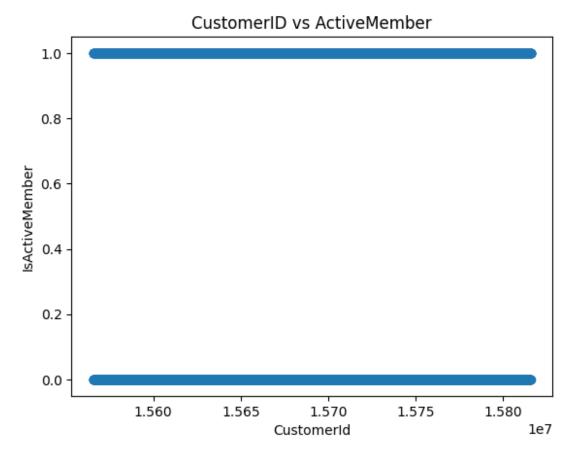


plt.scatter(dataset.index,dataset['Balance'])
<matplotlib.collections.PathCollection at 0x13a3a8550>



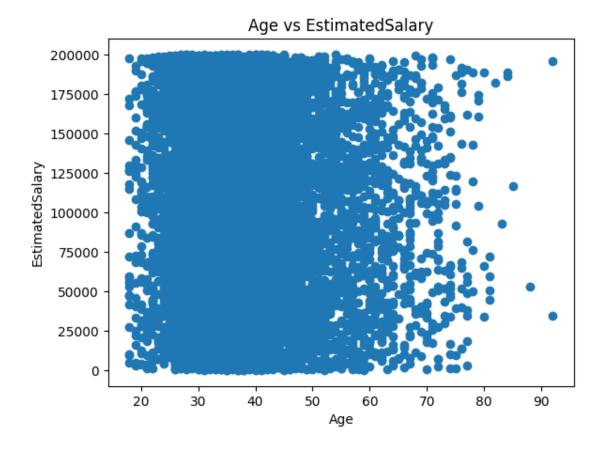
```
plt.scatter(dataset.CustomerId,dataset.IsActiveMember)
plt.title("CustomerID vs ActiveMember")
plt.xlabel("CustomerId")
plt.ylabel("IsActiveMember")
```

Text(0, 0.5, 'IsActiveMember')



```
plt.scatter(dataset.Age,dataset.EstimatedSalary)
plt.title("Age vs EstimatedSalary")
plt.xlabel("Age")
plt.ylabel("EstimatedSalary")

Text(0, 0.5, 'EstimatedSalary')
```



PERFORMING DESCRIPTIVE ANALYSIS ON THE DATASET

dataset.describe()

RowNumber	CustomerId	CreditScore	Age
Tenure \ count 10000.00000 10000.000000	1.000000e+04	10000.000000	10000.000000
mean 5000.50000 5.012800	1.569094e+07	650.528800	38.921800
std 2886.89568 2.892174	7.193619e+04	96.653299	10.487806
min 1.00000 0.000000	1.556570e+07	350.000000	18.000000
25% 2500.75000 3.000000	1.562853e+07	584.000000	32.000000
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 10.000000	1.581569e+07	850.000000	92.000000

Balance NumOfProducts HasCrCard IsActiveMember \

count mean	10000.000000 76485.889288	10000.000000 1.530200	10000.00000 0.70550	10000.000000 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

dataset.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

dataset[dataset.isnull().any(axis=1)]

Empty DataFrame

Columns: [RowNumber, CustomerId, Surname, CreditScore, Geography,

Gender, Age, Tenure, Balance, NumOfProducts, HasCrCard,

IsActiveMember, EstimatedSalary, Exited]

Index: []

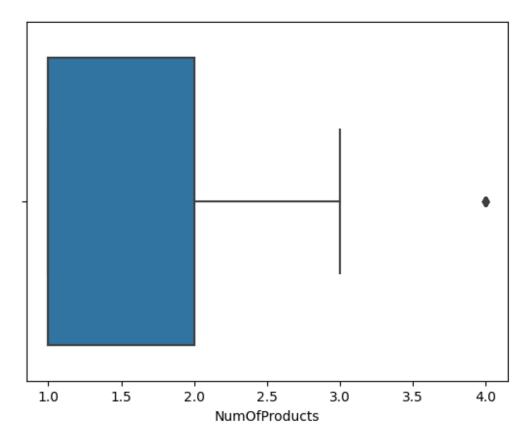
FIND THE OUTLIERS AND REPLACE IT

sns.boxplot(dataset['NumOfProducts'],data=dataset)

/Library/Frameworks/Python.framework/Versions/3.10/lib/python3.10/ 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='NumOfProducts'>



```
Q1=dataset['NumOfProducts'].quantile(0.25)
Q3=dataset['NumOfProducts'].quantile(0.75)
IQR=Q3-Q1
IQR

1.0

P_mean = dataset['NumOfProducts'].mean()
p_std = dataset['NumOfProducts'].std()
low= P_mean - (3 * p_std)
high= P_mean + (3 * p_std)
p_outliers = dataset[(dataset['NumOfProducts'] < low) |
(dataset['NumOfProducts'] > high)]
p_outliers.head()
```

A	RowNumb	er Cust	omerId	Surname	CreditScor	e Geography	Gender
Age 7	\	8 15	656148	0binna	37	6 Germany	Female
29 70		71 15	703793	Konovalova	73	8 Germany	Male
58 1254	12	55 15	610383	Dumetolisa	62	8 France	Female
46 1469	14	70 15	670374	Wright	81	9 Germany	Female
49 1488 30	148	89 15	625824	Kornilova	59	6 Spain	Male
7 70 1254 1469 1488		Balan 115046. 133745. 46870. 120656. 121345.	74 44 43 86 88 Exite		HasCrCard 1 1 1 0 1	IsActiveMemb	oer \ 0 0 0 0
7 70 1254 1469 1488	10	19346.88 28373.86 31272.14 66164.30 41921.75		1 1 1 1			

Label encoding

```
dataset["Gender"] = dataset["Gender"].astype('category')
dataset.dtypes
```

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

dtype: object

```
dataset["gender_cat"] = dataset["Gender"].cat.codes
dataset
```

RowNumbe	r Custome	rId	Surname	CreditS	core	Geography	Gender
\	1 15634	602	Hargrave		619	France	Female
	2 15647	311	_		608	Spain	Female
						•	Female
							Female
							Female
	3 13737	000	HITCHETT			Spain	T Cilia CC
		• • •					
999	5 15606	229	0bijiaku		771	France	Male
999	7 15569	892 .	Johnstone		516	France	Male
999	8 15584	532	Liu		709	France	Female
999	9 15682	355	Sabbatini		772	Germany	Male
1000	9 15628	319	Walker		792	France	Female
1	0.00	Num0	fProducts 1 1 3 2 1	HasCrCa	rd I 1 0 1 0	IsActiveMem	ber \ 1 0 0 1
 5 10 7 3	0.00 57369.61 0.00 75075.31		2 1 1 2 1		1 1 0 1		 0 1 1 0 0
10 11 11 9 7 9 10 4	1348.88 2542.58 3931.57 3826.63 9084.10 6270.64 1699.77 2085.58 2888.52	1 0 1 0 0 0 1	gender_c	0 0 0 0 1 1 0			
	9999 9999 10000 Tenure 2 1 8 1 2 5 10 7 3 4 Estimate 10 11 11 99 79	1 15634 2 15647 3 15619 4 15701 5 15737 9996 15606 9997 15569 9998 15584 9999 15682 10000 15628 Tenure Balance 2 0.00 1 83807.86 8 159660.80 1 0.00 2 125510.82 5 0.00 10 57369.61 7 0.00 3 75075.31 4 130142.79	1 15634602 2 15647311 3 15619304 4 15701354 5 15737888 9996 15606229 9997 15569892 9998 15584532 9999 15682355 10000 15628319 Tenure Balance NumO 2 0.00 1 83807.86 8 159660.80 1 0.00 2 125510.82 5 0.00 10 57369.61 7 0.00 3 75075.31 4 130142.79 EstimatedSalary Exited 101348.88 112542.58 013931.57 93826.63 79084.10 0 96270.64 101699.77 42085.58 1 92888.52	1 15634602 Hargrave 2 15647311 Hill 3 15619304 Onio 4 15701354 Boni 5 15737888 Mitchell 9996 15606229 Obijiaku 9997 15569892 Johnstone 9998 15584532 Liu 9999 15682355 Sabbatini 10000 15628319 Walker Tenure Balance NumOfProducts 2 0.00 1 1 83807.86 1 8 159660.80 3 1 0.00 2 2 125510.82 1 5 0.00 2 10 57369.61 1 7 0.00 3 3 75075.31 2 4 130142.79 1 EstimatedSalary Exited gender_collings of the collings of	1 15634602 Hargrave 2 15647311 Hill 3 15619304 Onio 4 15701354 Boni 5 15737888 Mitchell 9996 15606229 Obijiaku 9997 15569892 Johnstone 9998 15584532 Liu 9999 15682355 Sabbatini 10000 15628319 Walker Tenure Balance NumOfProducts HasCrCa 2 0.00 1 1 83807.86 1 8 159660.80 3 1 0.00 2 2 125510.82 1 5 0.00 2 2 125510.82 1 5 0.00 1 3 75075.31 2 4 130142.79 1 EstimatedSalary Exited gender_cat 101348.88 1 112542.58 0 0 113931.57 1 0 93826.63 0 0 79084.10 0 0 96270.64 0 1 101699.77 0 1 42085.58 1 0 92888.52 1	1 15634602 Hargrave 619 2 15647311 Hill 608 3 15619304 Onio 502 4 15701354 Boni 699 5 15737888 Mitchell 850 9996 15606229 Obijiaku 771 9997 15569892 Johnstone 516 9998 15584532 Liu 709 9999 15682355 Sabbatini 772 10000 15628319 Walker 792 Tenure Balance NumOfProducts HasCrCard 1 2 0.00 1 1 1 83807.86 1 0 8 159660.80 3 1 1 0.00 2 0 2 125510.82 1 1 5 0.00 2 1 10 57369.61 1 1 7 0.00 2 0 3 75075.31 2 1 4 130142.79 1 1 EstimatedSalary Exited gender_cat 101348.88 1 112542.58 0 0 113931.57 1 0 93826.63 0 0 79084.10 0 0 0 96270.64 0 1 101699.77 0 1 42085.58 1 0 92888.52 1 1	1 15634602 Hargrave 619 France 2 15647311 Hill 608 Spain 3 15619304 Onio 502 France 4 15701354 Boni 699 France 5 15737888 Mitchell 850 Spain 9996 15606229 Obijiaku 771 France 9997 15569892 Johnstone 516 France 9998 15584532 Liu 709 France 9999 15682355 Sabbatini 772 Germany 10000 15628319 Walker 792 France Tenure Balance NumOfProducts HasCrCard IsActiveMem 2 0.00 1 1 1 83807.86 1 0 8 159660.80 3 1 1 0.00 2 0 2 125510.82 1 1 5 0.00 2 1 10 57369.61 1 1 7 0.00 1 0 3 75075.31 2 1 4 130142.79 1 1 EstimatedSalary Exited gender_cat 101348.88 1 112542.58 0 0 13931.57 1 0 93826.63 0 0 79084.10 0 0 93826.63 0 0 79084.10 0 0 92888.52 1 0

```
Split the data into dependent and independent variables.
X = dataset.iloc[:, :-1].values
array([[1, 15634602, 'Hargrave', ..., 1, 101348.88, 1],
       [2, 15647311, 'Hill', ..., 1, 112542.58, 0],
       [3, 15619304, 'Onio', ..., 0, 113931.57, 1],
       [9998, 15584532, 'Liu', ..., 1, 42085.58, 1],
       [9999, 15682355, 'Sabbatini', ..., 0, 92888.52, 1],
       [10000, 15628319, 'Walker', ..., 0, 38190.78, 0]],
dtype=object)
Y = dataset.iloc[:, -2].values
array([1, 0, 1, ..., 1, 1, 0])
Scale the independent variables
from sklearn import preprocessing
x=dataset.iloc[:,12:14]
Χ
      EstimatedSalary Exited
0
            101348.88
                             1
1
            112542.58
                             0
2
            113931.57
                             1
3
             93826.63
                             0
4
             79084.10
                             0
             96270.64
                             0
9995
9996
            101699.77
                             0
9997
             42085.58
                             1
                             1
             92888.52
9998
9999
             38190.78
[10000 \text{ rows } \times 2 \text{ columns}]
min max scaler = preprocessing.MinMaxScaler(feature range = (0, 1))
new x= min max scaler.fit transform(x)
new x
array([[0.50673489, 1.
                               ],
       [0.56270874, 0.
                               ],
       [0.56965435, 1.
```

```
[0.21039009, 1. ],
[0.46442905, 1. ],
[0.19091423, 0. ]])
```

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(X, Y, test_size=0.33, random_state=1)

dataset.shape
(10000, 15)

X_train.shape
(6700, 14)

X_test.shape
(3300, 14)
```