Loading Dataset

In [4]:

```
#importing the libraries
import numpy as np
import pandas as pd
```

In [5]:

```
#Loading the Dataset
pdDf = pd.read_csv('/cor
pdDf.head()
```

Out[5]:

	RowNumber	CustomerId	Sur
0	1	15634602	Har
1	2	15647311	
2	3	15619304	
3	4	15701354	
4	5	15737888	Mi

In [6]:

```
pdDf.dtypes
```

Out[6]:

```
RowNumber          int
64
CustomerId         int
64
Surname            obje
ct
CreditScore        int
64
```

In [6]:
```
pdDf.dtypes
```

Out[6]:
```
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
```

1. Univariate Analysis

In [7]:

```
import matplotlib.pyplot
import seaborn as sns
#Univariate Analysis
pdDf[['Age','CreditScore
```

Out[7]:

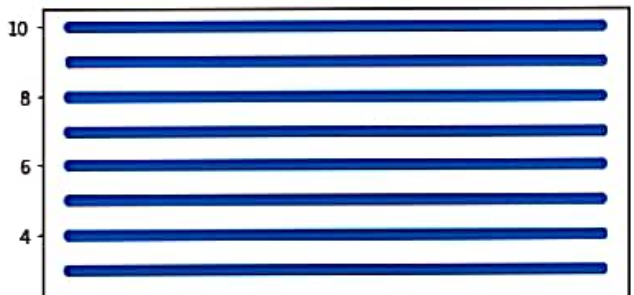
	Age	CreditSc
count	10000.000000	10000.000
mean	38.921800	650.528
std	10.487806	96.653
min	18.000000	350.000
25%	32.000000	584.000
50%	37.000000	652.000
75%	44.000000	718.000
max	92.000000	850.000

In [8]:

```
#Univariate Scatter Plot
plt.scatter(pdDf.index,p
```

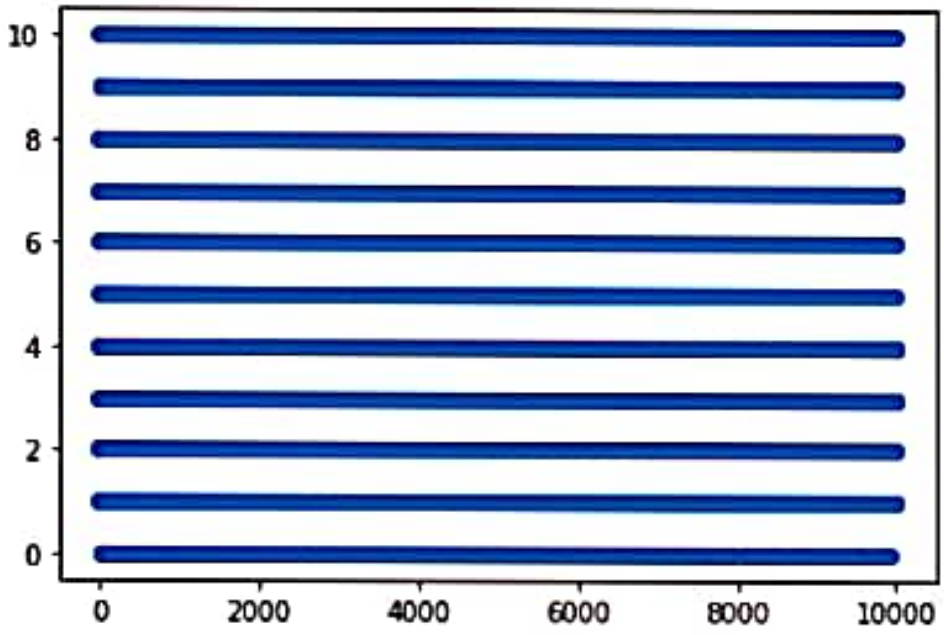
Out[8]:

```
<matplotlib.collections.
PathCollection at 0x7fdf
9d9e8d10>
```



Out[8]:

```
<matplotlib.collections.
PathCollection at 0x7fdf
9d9e8d10>
```

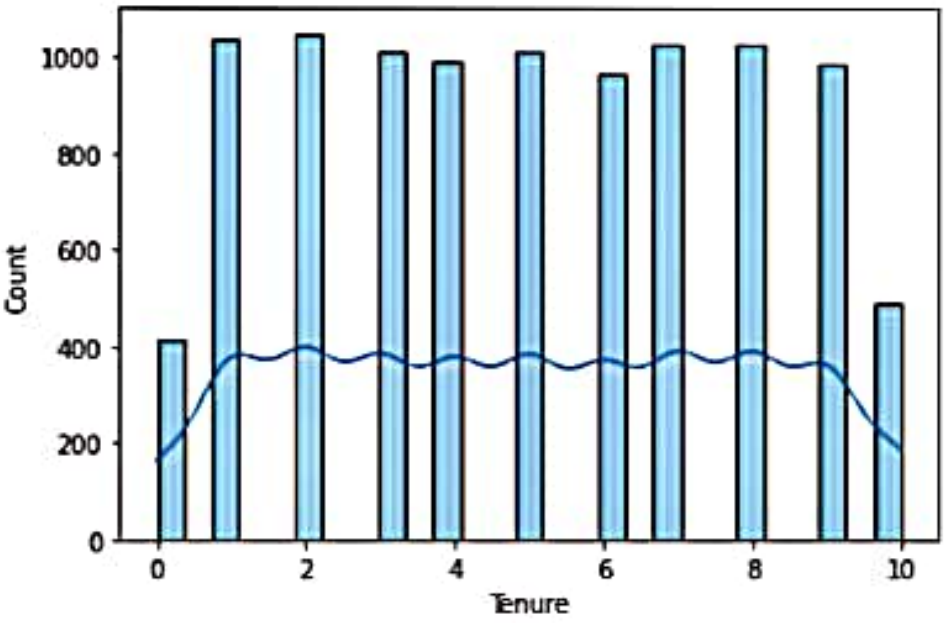


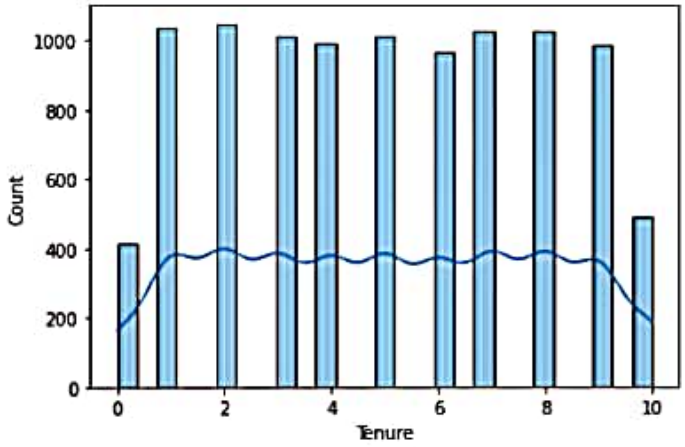
In [9]:

```
#Histogram for the Tenur
sns.histplot(pdDf.Tenure
```

Out[9]:

```
<matplotlib.axes._subplo
ts.AxesSubplot at 0x7fdf
9d516350>
```





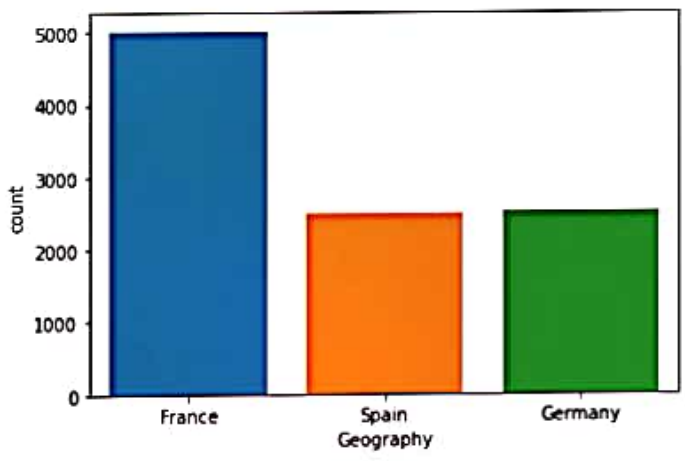
In [10]:

```
#Univariate Analysis of
sns.countplot(pdDf.Geogr
```

```
/usr/local/lib/python3.
7/dist-packages/seaborn/
_decorators.py:43: Futur
eWarning: Pass the follo
wing variable as a keywo
rd arg: x. From version
0.12, the only valid pos
itional argument will be
`data`, and passing othe
r arguments without an e
xplicit keyword will res
ult in an error or misin
terpretation.
  FutureWarning
```

Out[10]:

```
<matplotlib.axes._subplo
ts.AxesSubplot at 0x7fdf
9d41bfd0>
```



2) Bi - Variate Analysis

In [11]:
```
pdDf[['Age','CreditScore
```

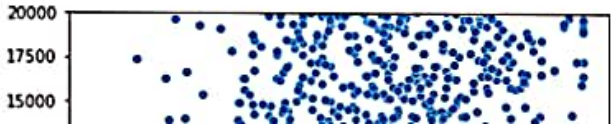
Out[11]:

	Age	Cred
Age	1.000000	-0.
CreditScore	-0.003965	1.
EstimatedSalary	-0.007201	-0.

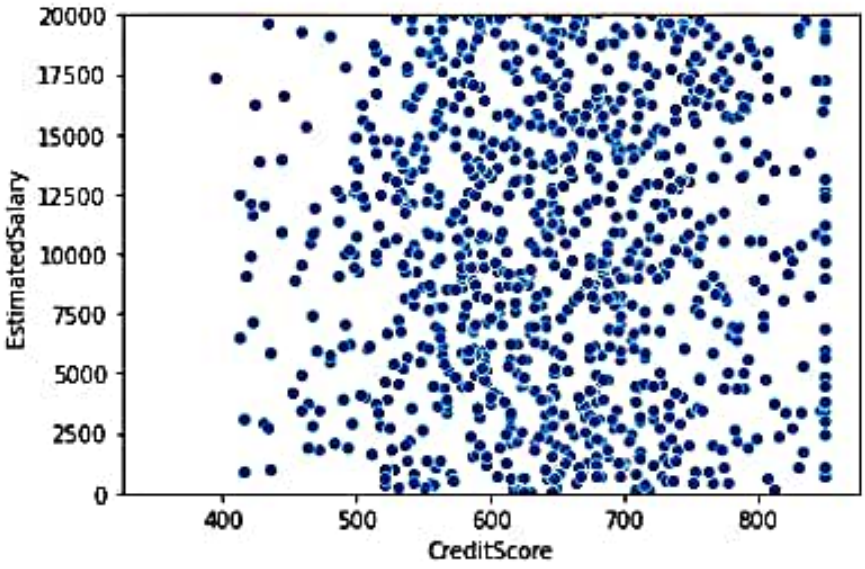
In [12]:
```
sns.scatterplot(pdDf.Cre
plt.ylim(0,20000)
```

```
/usr/local/lib/python3.
7/dist-packages/seaborn/
_decorators.py:43: Futur
eWarning: Pass the follo
wing variables as keywor
d args: x, y. From versi
on 0.12, the only valid
positional argument will
be `data`, and passing o
ther arguments without a
n explicit keyword will
result in an error or mi
sinterpretation.
  FutureWarning
```

Out[12]:
```
(0.0, 20000.0)
```

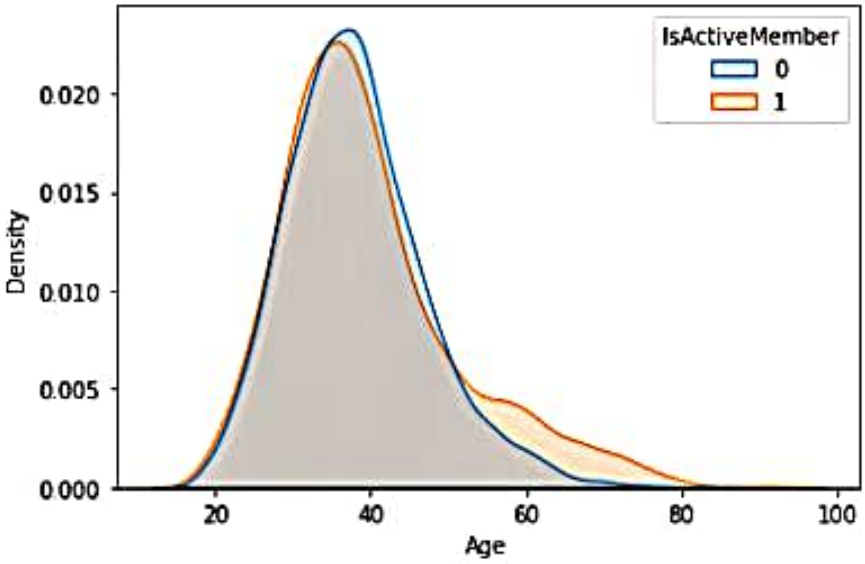


Out[12]: (0.0, 20000.0)



In [13]:
```
sns.kdeplot(data=pdDf,x=
```

Out[13]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf9d31a6d0>

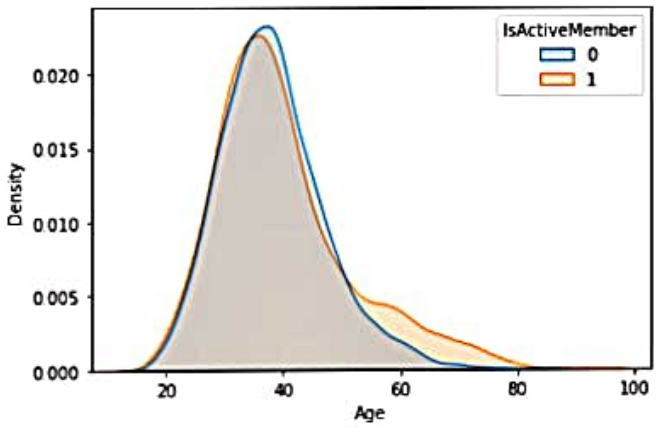


In [14]:
```
sns.countplot(data=pdDf,
```

Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf

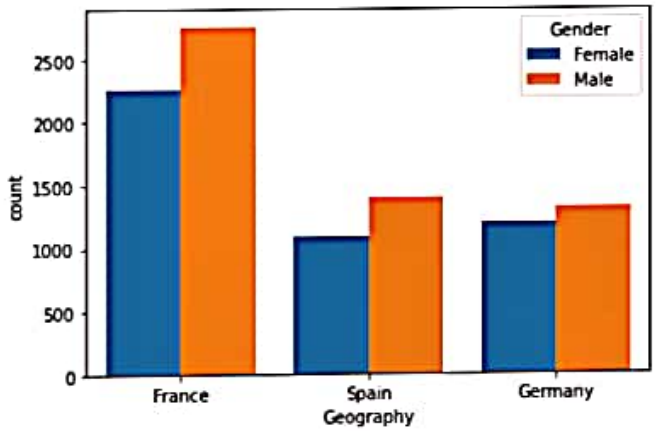
In [13]:
```
sns.kdeplot(data=pdDf,x=
```

Out[13]:
```
<matplotlib.axes._subplo
ts.AxesSubplot at 0x7fdf
9d31a6d0>
```



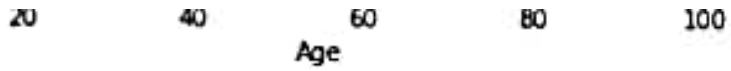
In [14]:
```
sns.countplot(data=pdDf,
```

Out[14]:
```
<matplotlib.axes._subplo
ts.AxesSubplot at 0x7fdf
9d29d750>
```



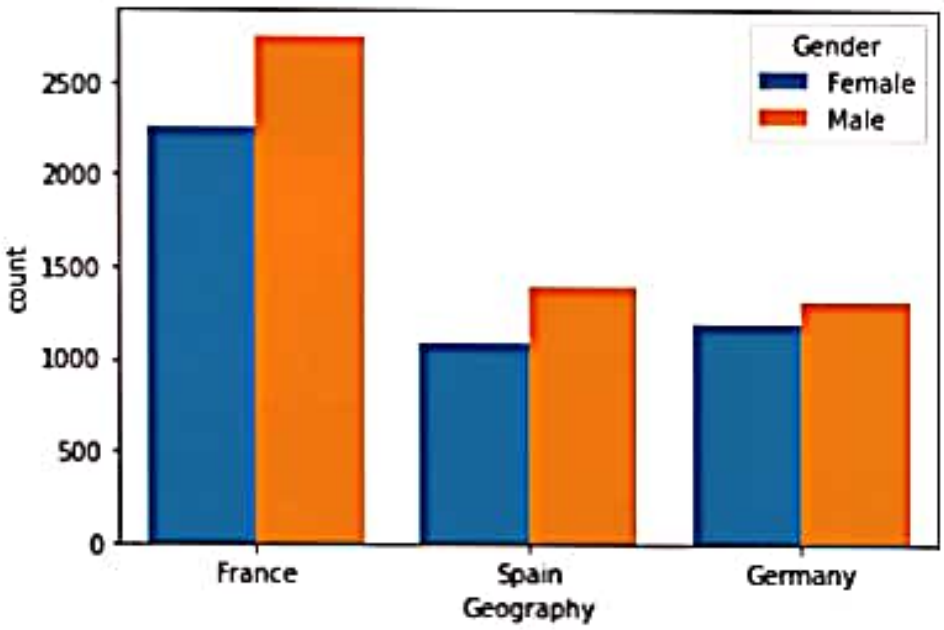
In [15]:
```
pd.crosstab(pdDf.Gender,
```

Out[15]:
```
Gender
Female      4543
Male        5457
All        10000
Name: All, dtype: int64
```



In [14]:
```
sns.countplot(data=pdDf,
```

Out[14]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fdf9d29d750>
```



In [15]:
```
pd.crosstab(pdDf.Gender,
```

Out[15]:
```
Gender
Female      4543
Male        5457
All        10000
Name: All, dtype: int64
```

3) Multivariate Analysis

In [16]:
```
sns.pairplot(data=pdDf,h
```

Out[16]:
```
<seaborn.axisgrid.PairGrid at 0x7fdf9d270f90>
```

4. Perform descriptive statistics on the dataset

In [17]:
```
pdDf.describe()
```

Out[17]:

	RowNumber	Customer
count	10000.00000	1.000000e+
mean	5000.50000	1.569094e+
std	2886.89568	7.193619e+
min	1.00000	1.556570e+
25%	2500.75000	1.562853e+
50%	5000.50000	1.569074e+
75%	7500.25000	1.575323e+
max	10000.00000	1.581569e+

5. Handle the Missing values

In [18]:
```
pdDf.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              10
000 non-null  int64
 7   Tenure           10
000 non-null  int64
 8   Balance          10
000 non-null  float64
 9   NumOfProducts    10
000 non-null  int64
 10  HasCrCard        10
000 non-null  int64
 11  IsActiveMember   10
000 non-null  int64
 12  EstimatedSalary  10
000 non-null  float64
 13  Exited           10
000 non-null  int64
dtypes: float64(2), int6
4(9), object(3)
memory usage: 1.1+ MB
```

In [19]:

```
print(pdDf.isnull().sum(
##No Null Values Presen
```

```
000 non-null  int64
 11  IsActiveMember   10
000 non-null  int64
 12  EstimatedSalary  10
000 non-null  float64
 13  Exited           10
000 non-null  int64
dtypes: float64(2), int6
4(9), object(3)
memory usage: 1.1+ MB
```

In [19]:

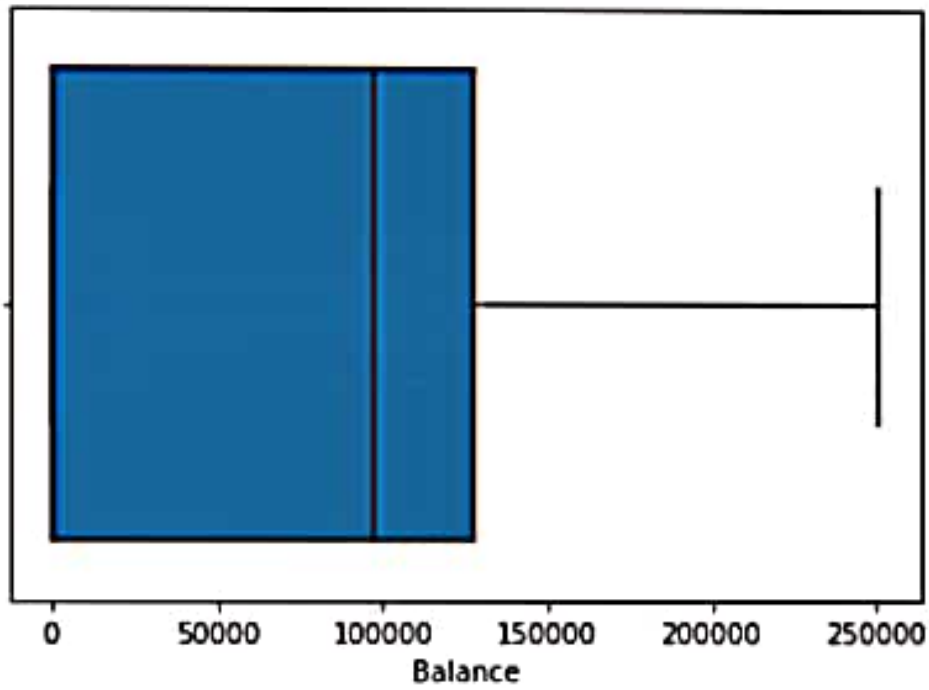
```
print(pdDf.isnull().sum(
##No Null Values Presen
```

```
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
```

6. Find the outliers and replace the outliers

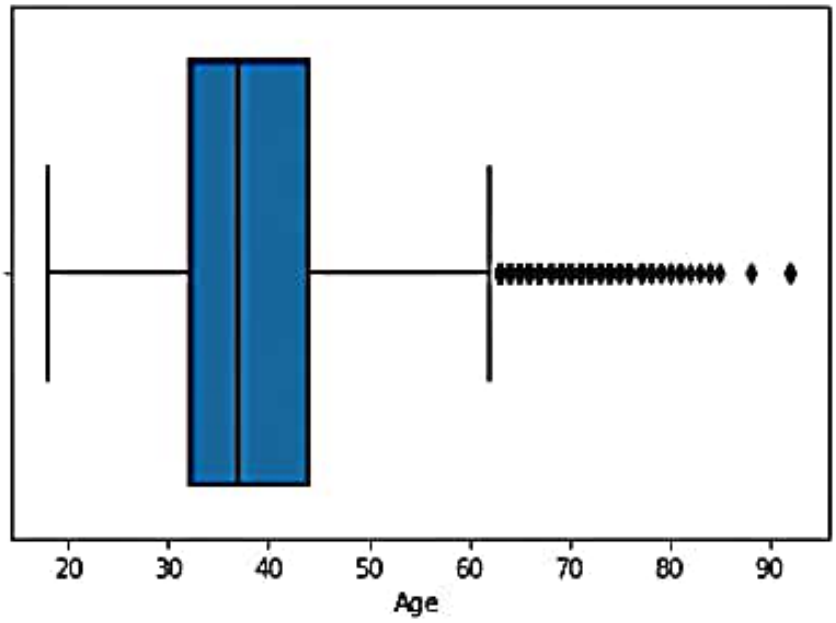
In [20]:
```
sns.boxplot(x=pdDf["Bala
```

Out[20]:
```
<matplotlib.axes._subplo
ts.AxesSubplot at 0x7fdf
97ddf350>
```



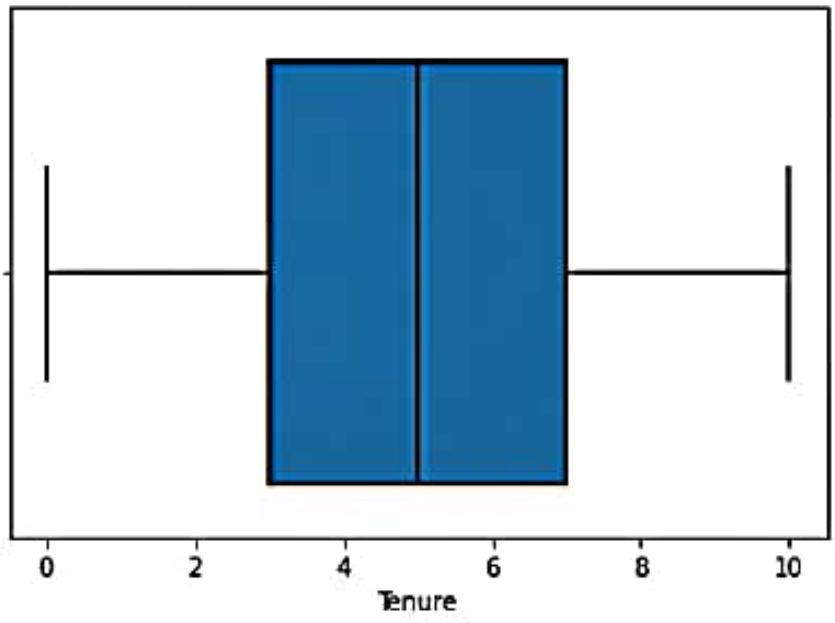
In [21]:
```
sns.boxplot(x=pdDf['Age'
```

Out[21]:
```
<matplotlib.axes._subplo
ts.AxesSubplot at 0x7fdf
979c9390>
```



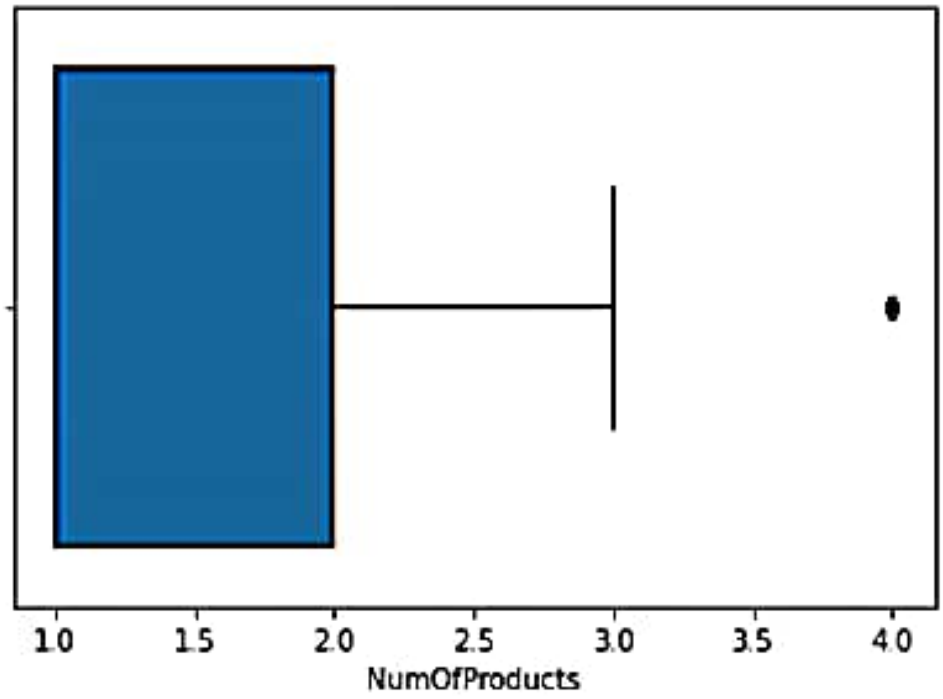
In [22]:
```
sns.boxplot(x=pdDf['Tenu
```

Out[22]:
```
<matplotlib.axes._subplo
ts.AxesSubplot at 0x7fdf
97a10610>
```



In [23]:
```
sns.boxplot(x=pdDf['NumC
```

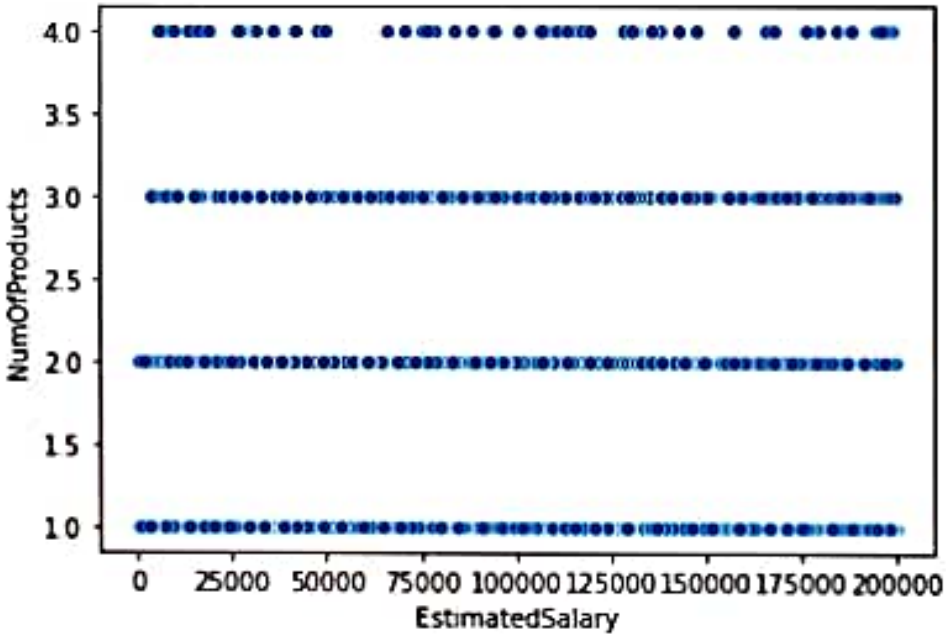
Out[23]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fdf960d13d0>
```



In [24]:
```
sns.scatterplot(y=pdDf['
```

Out[24]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fdf9609a8d0>
```

Out[24]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf9609a8d0>



Identifying Outliers with Interquartile Range (IQR)

Identifying Outliers with Interquartile Range (IQR)

In [25]:
```
Q1 = pdDf.quantile(0.25)
Q3 = pdDf.quantile(0.75)
IQR = Q3 - Q1
print(IQR)
```

```
RowNumber              499
9.5000
CustomerId           12470
5.5000
CreditScore             13
4.0000
Age                      1
2.0000
Tenure
4.0000
Balance              12764
4.2400
NumOfProducts
1.0000
HasCrCard
1.0000
IsActiveMember
1.0000
EstimatedSalary       9838
6.1375
Exited
0.0000
dtype: float64
```

In [26]:
```
categorical = pdDf.drop(

rows = int(np.ceil(categ
```

In [26]:

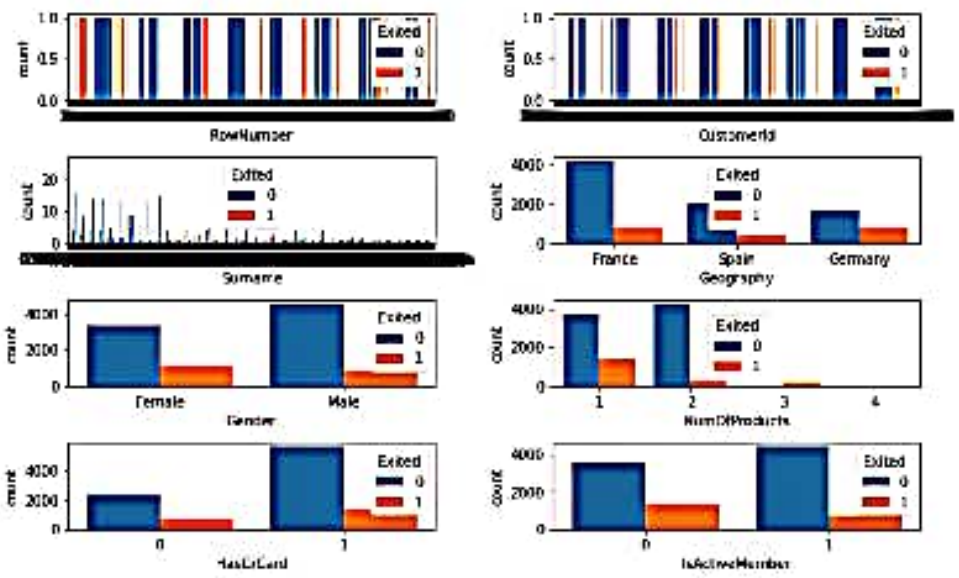
```
categorical = pdDf.drop(

rows = int(np.ceil(categ

# create sub-plots anf
fig, axes = plt.subplots
axes = axes.flatten()

for row in range(rows):
    cols = min(2, categ
    for col in range(col
        col_name = categ
        ax = axes[row*2
        sns.countplot(da

plt.tight_layout()
```

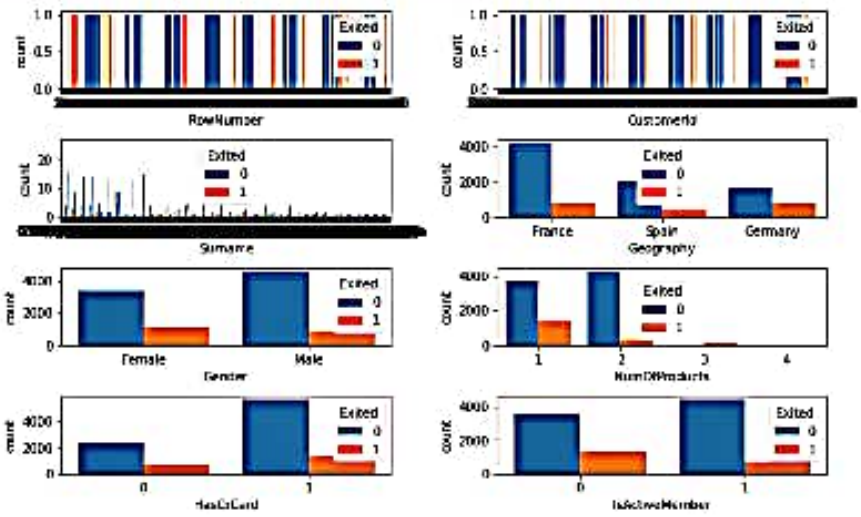


```
rows = int(np.ceil(categ

# create sub-plots anf
fig, axes = plt.subplots
axes = axes.flatten()

for row in range(rows):
    cols = min(2, categ
    for col in range(col
        col_name = categ
        ax = axes[row*2
        sns.countplot(da

plt.tight_layout()
```



In [27]:

```
#We can see outliers in
pdDf = pdDf.loc[pdDf['Cr
pdDf = pdDf.loc[pdDf['Ag
pdDf = pdDf.loc[pdDf['Ba
pdDf = pdDf.loc[pdDf['Nu
```

7. Check for Categorical columns and perform encoding

In [28]:
```
pdDf['Surname'].value_cc
```

Out[28]:
```
Smith          30
Scott          29
Martin         28
Walker         26
Brown          26
               ..
Fishbourne      1
McIver          1
Valentin        1
Izmailov        1
Burbidge        1
Name: Surname, Length: 2
882, dtype: int64
```

In [29]:
```
pdDf['Geography'].value_
```

Out[29]:
```
France     4872
Spain      2396
Germany    2384
Name: Geography, dtype:
int64
```

In [30]:
```
pdDf['Gender'].value_cou
```

Out[30]:
```
Male       5305
Female     4347
Name: Gender, dtype: int
64
```

In [31]:

```
gen_n = {"Male":0,"Fema]
pdDf = pdDf.replace(gen_
pdDf.head()
```

Out[31]:

	RowNumber	CustomerId	Sur
0	1	15634602	Har
1	2	15647311	
3	4	15701354	
4	5	15737888	Mi
5	6	15574012	

In [32]:

```
geo_n = {"France":0,"Spa
pdDf = pdDf.replace(geo_
pdDf.head()
```

Out[32]:

	RowNumber	CustomerId	Sur
0	1	15634602	Har
1	2	15647311	
3	4	15701354	
4	5	15737888	Mi
5	6	15574012	

In [33]:

```
temp = pdDf['Surname'].u
di = {}
i=0
for x in temp:
  di[x] = i
  i=i+1
```

In [34]:

```
pdDf = pdDf.replace(di)
pdDf.head()
```

Out[34]:

	RowNumber	CustomerId	Sur
0	1	15634602	
1	2	15647311	
3	4	15701354	
4	5	15737888	
5	6	15574012	

8. Split the data into dependent and independent variables

In [35]:

```
#X = INDEPENDENT VARIABL
#Y = DEPENDENT VARIABLE
X = pdDf.drop(columns=['
y = pdDf['Exited']
```

9. Scale the independent variables

In [36]:
```
#BEFORE SCALING LET'S DR
X = X.drop(columns=['Row
```

In [37]:
```
from sklearn import prep
minmax = preprocessing.M
temp = minmax.fit_transf
cols = X.columns
X = pd.DataFrame(data=te
```

In [38]:
```
X.head()
```

Out[38]:

	CreditScore	Geography	Genc
0	0.485523	0.0	
1	0.461024	0.5	
2	0.663697	0.0	
3	1.000000	0.5	
4	0.543430	0.5	

10. Split the data into training and testing

In [39]:
```
from sklearn.model_selec
X_train, X_test, y_trair
```

In [40]:
```
#EXTRA WORK
from sklearn import tree
from sklearn.metrics imp
dt_clf = tree.DecisionTr
dt_clf.fit(X_train, y_tr
pred = dt_clf.predict(X_
accuracy_score(y_train,
```

Out[40]:
```
0.8629366489046774
```