Assignment 2 - authored by, Kishore Akash YS

1. Download the dataset from the source [here (https://drive.google.com/file/d/1_HcM0K8wt4b7FMLkc1V1dv0y6l_9ULzy/view?usp=sharing)](https://drive.google.com/file/d/1_HcM0K8wt4b7FMLkc1V1dv0y6l_9ULzy/view?usp=sharing).

About the dataset:

This dataset is all about churn modelling of a credit company. It has the details about the end user who are using credit card and also it has some variables to depicit the churn of the customer.

RowNumber - Serial number of the rows
CustomerId - Unique identification of customer
Surname - Name of the customer
CreditScore - Cipil score of the customer
Geography - Location of the bank
Gender - Sex of the customer
Age - Age of the customer
Tenure - Repayment period for the credit amount
Balance - Current balance in thier creidt card
NumOfProducts - Products owned by the customer from the company
HasCrCard - Has credit card or not (0 - no , 1 - yes)
IsactiveMember - Is a active member or not
EstimatedSalary - Salary of the customer
Exited - Churn of the customer

In [1]:
```
import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

2. Load the dataset

In [3]:
```
df = pd.read_csv("Churn_Modelling.csv")
df.head()
```

Out[3]:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance
0	1	15634602	Hargrave	619	France	Female	42	2	0.00
1	2	15647311	Hill	608	Spain	Female	41	1	83807.86
2	3	15619304	Onio	502	France	Female	42	8	159660.80
3	4	15701354	Boni	699	France	Female	39	1	0.00
4	5	15737888	Mitchell	850	Spain	Female	43	2	125510.82

In [4]:
```
df.tail()
```

Out[4]:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Bala
9995	9996	15606229	Obijiaku	771	France	Male	39	5	
9996	9997	15569892	Johnstone	516	France	Male	35	10	5736
9997	9998	15584532	Liu	709	France	Female	36	7	
9998	9999	15682355	Sabbatini	772	Germany	Male	42	3	7507
9999	10000	15628319	Walker	792	France	Female	28	4	13014

3 a). Univariate analysis

In [5]:
```
#checking for categorical variables
category = df.select_dtypes(include=[np.object])
print("Categorical Variables: ",category.shape[1])

#checking for numerical variables
numerical = df.select_dtypes(include=[np.int64,np.float64])
print("Numerical Variables: ",numerical.shape[1])
```

```
Categorical Variables:  3
Numerical Variables:  11
```

In [6]:
```
df.columns
```

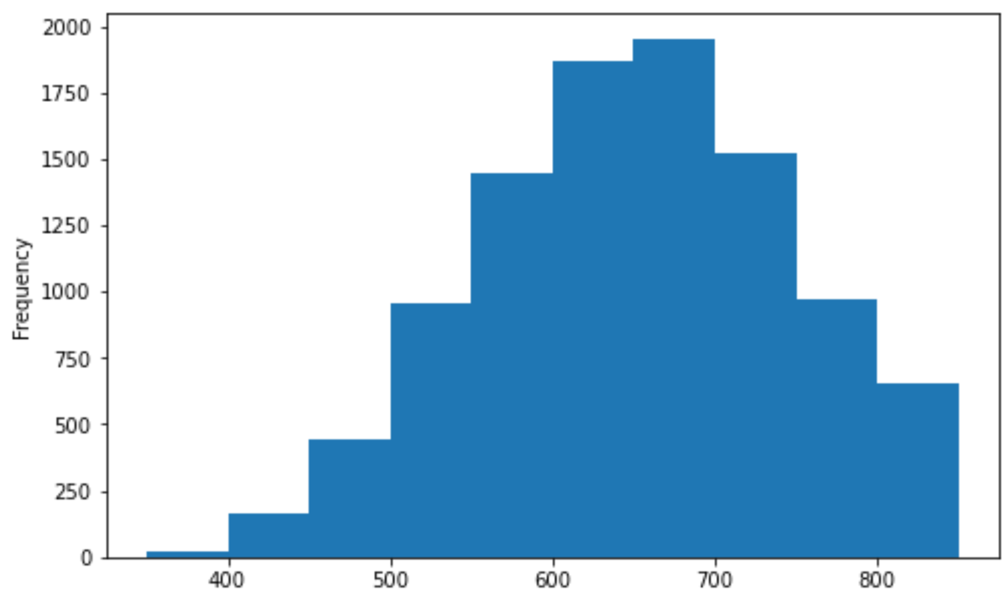
Out[6]:
```
Index(['RowNumber', 'CustomerId', 'Surname', 'CreditScore', 'Geography',
       'Gender', 'Age', 'Tenure', 'Balance', 'NumOfProducts', 'HasCrCard',
       'IsActiveMember', 'EstimatedSalary', 'Exited'],
      dtype='object')
```

In [7]:
```
df.shape
```

Out[7]:
```
(10000, 14)
```

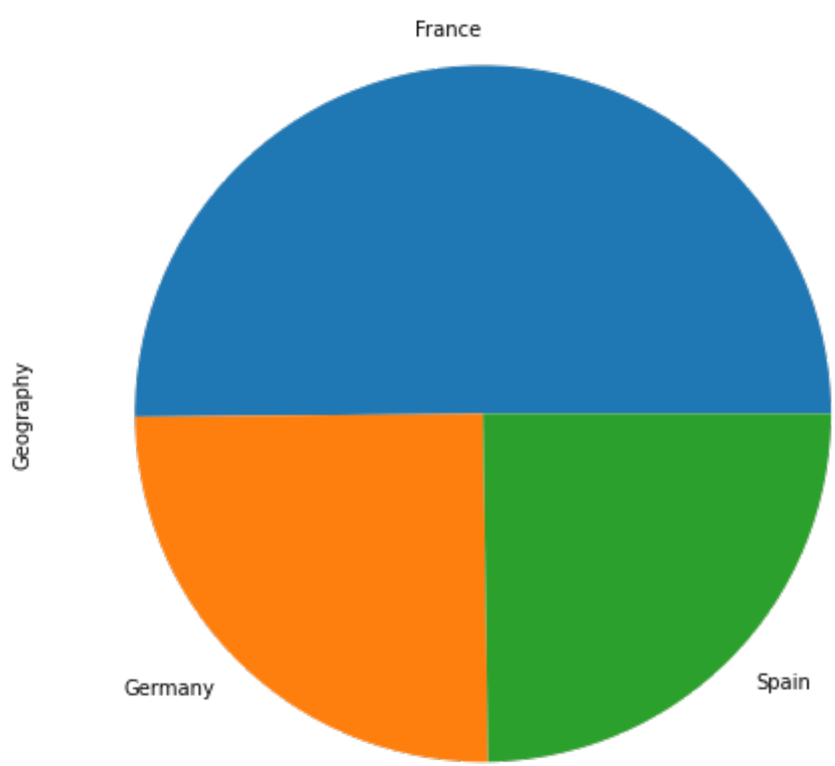
In [8]:
```
credit = df['CreditScore']
credit.plot(kind="hist",figsize=(8,5))
```

Out[8]: <AxesSubplot:ylabel='Frequency'>



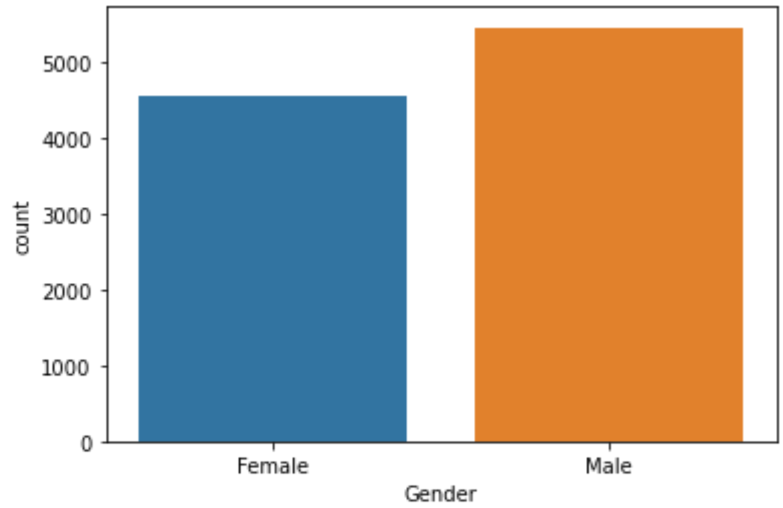
In [9]:
```
geo = df['Geography'].value_counts()
geo.plot(kind="pie",figsize=(10,8))
```

Out[9]: <AxesSubplot:ylabel='Geography'>



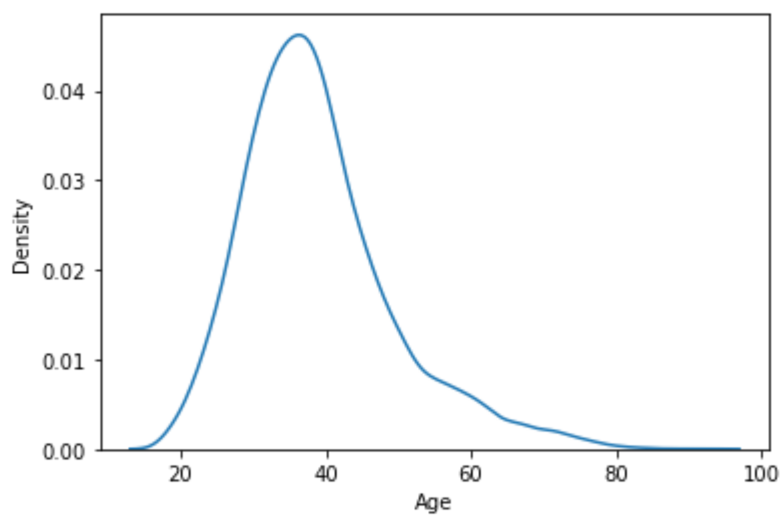
In [10]:
```
sns.countplot(df['Gender'])
```

Out[10]: <AxesSubplot:xlabel='Gender', ylabel='count'>



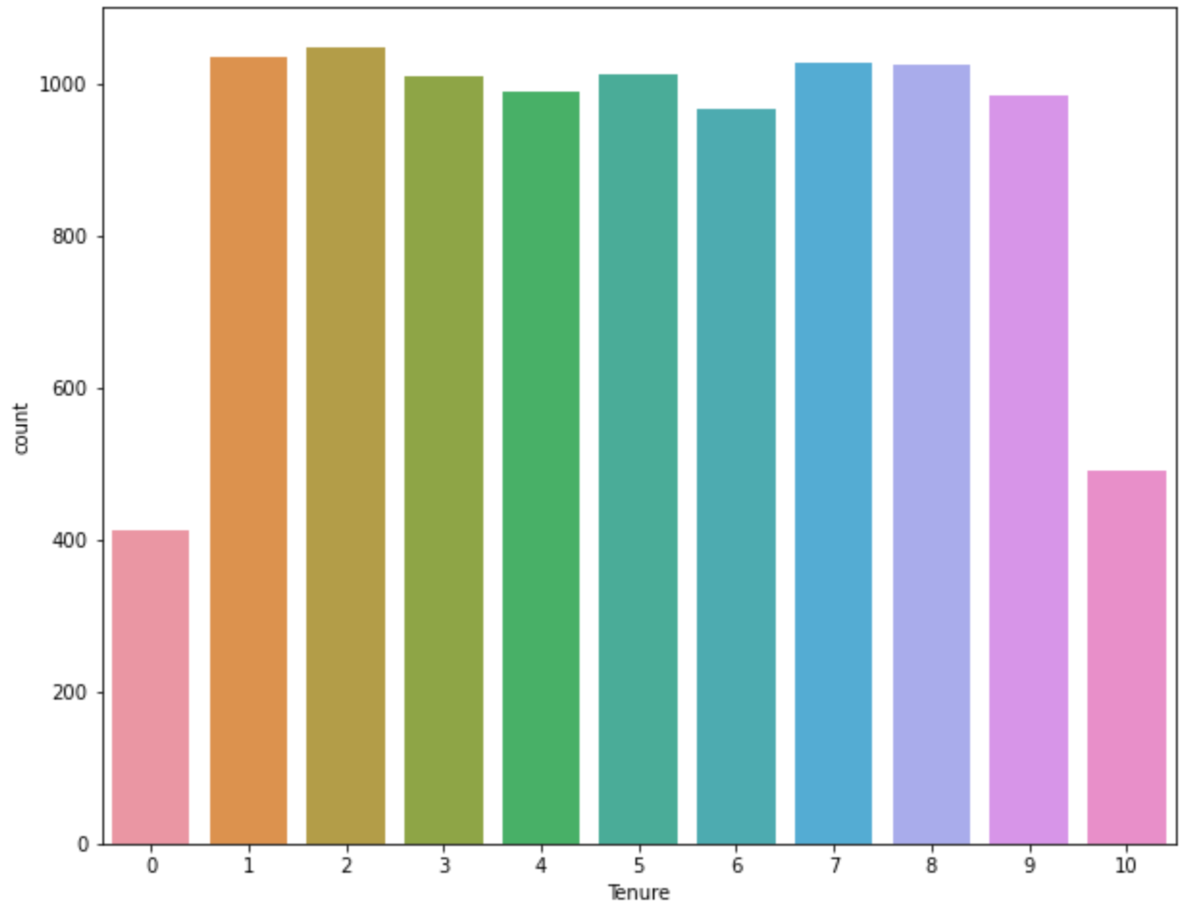
In [11]:
```
sns.distplot(df['Age'],hist=False)
```

Out[11]: <AxesSubplot:xlabel='Age', ylabel='Density'>



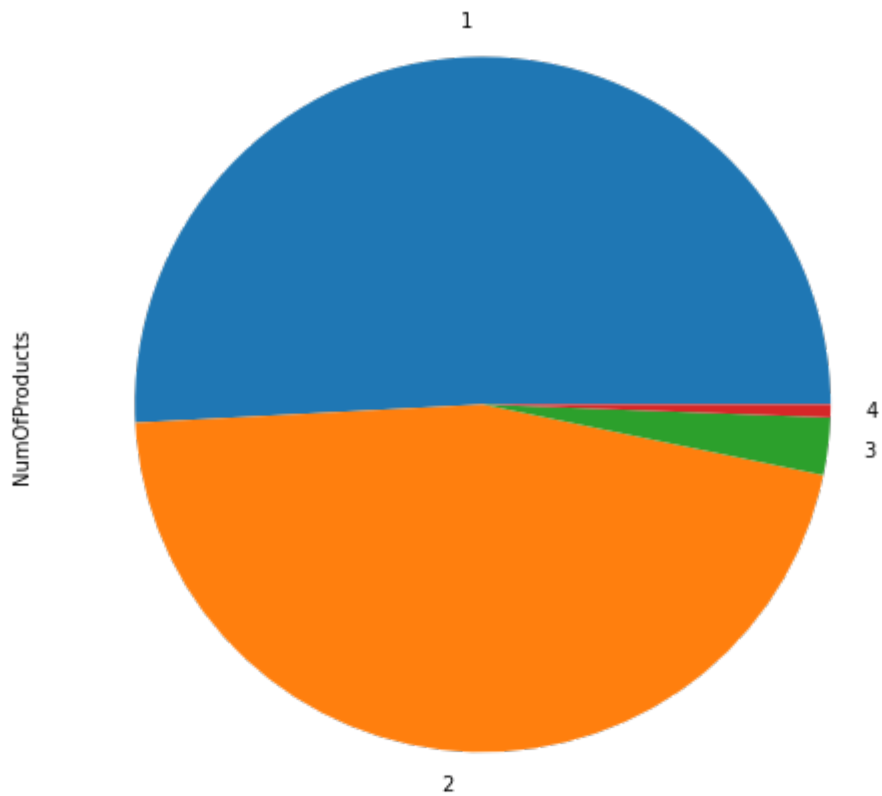
In [12]:
```
plt.figure(figsize=(10,8))
sns.countplot(df['Tenure'])
```

Out[12]: <AxesSubplot:xlabel='Tenure', ylabel='count'>



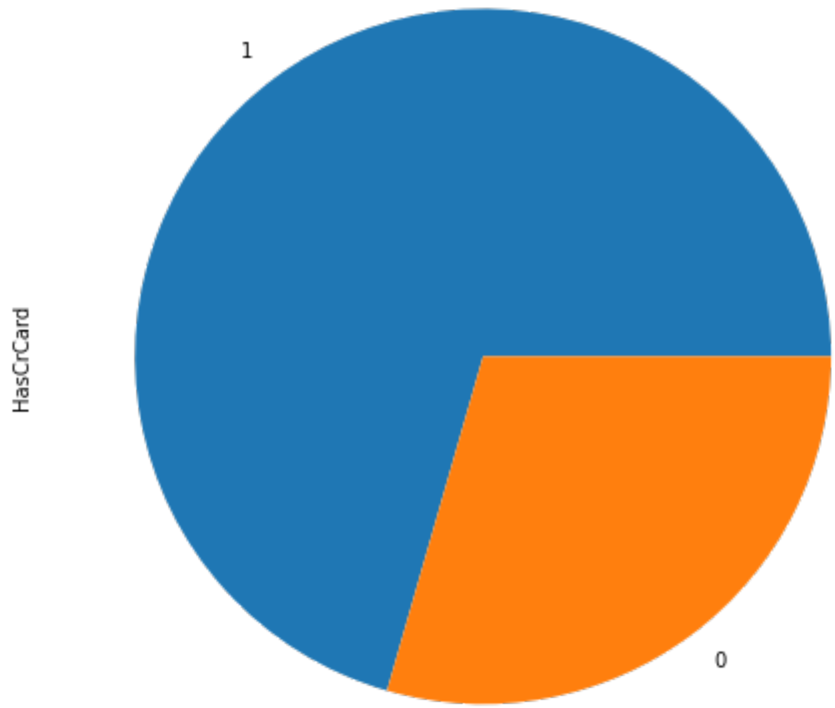
In [13]:
```
product = df['NumOfProducts'].value_counts()
product.plot(kind="pie",figsize=(10,8))
```

Out[13]:
```
<AxesSubplot:ylabel='NumOfProducts'>
```



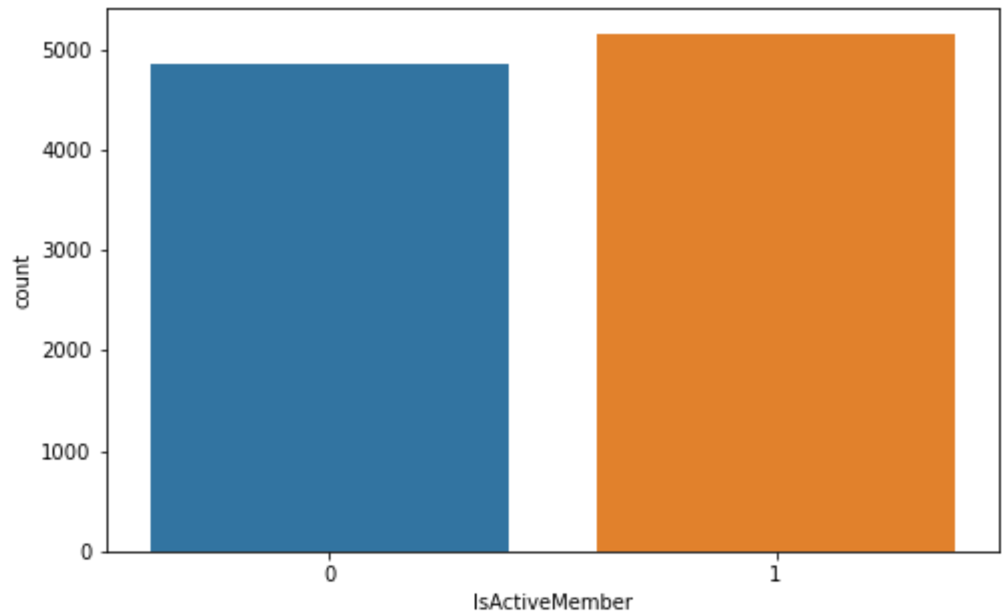
In [14]:
```
cr = df['HasCrCard'].value_counts()
cr.plot(kind="pie",figsize=(10,8))
```

Out[14]: <AxesSubplot:ylabel='HasCrCard'>



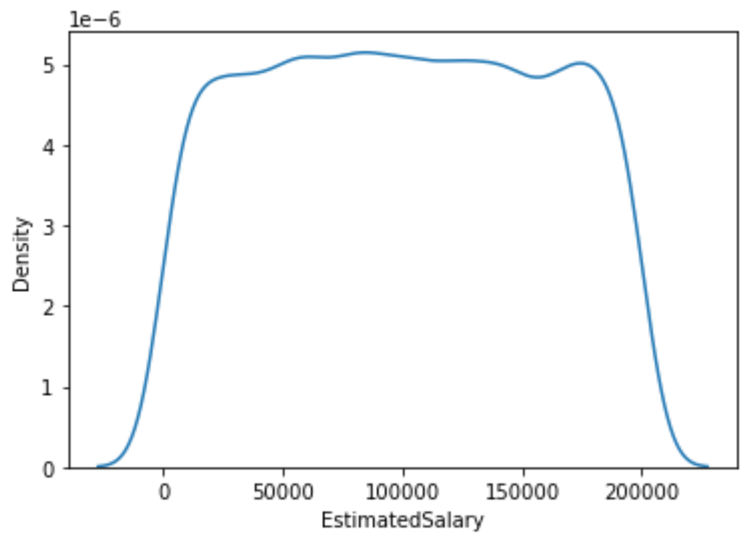
In [15]:
```
plt.figure(figsize=(8,5))
sns.countplot(df['IsActiveMember'])
```

Out[15]: <AxesSubplot:xlabel='IsActiveMember', ylabel='count'>



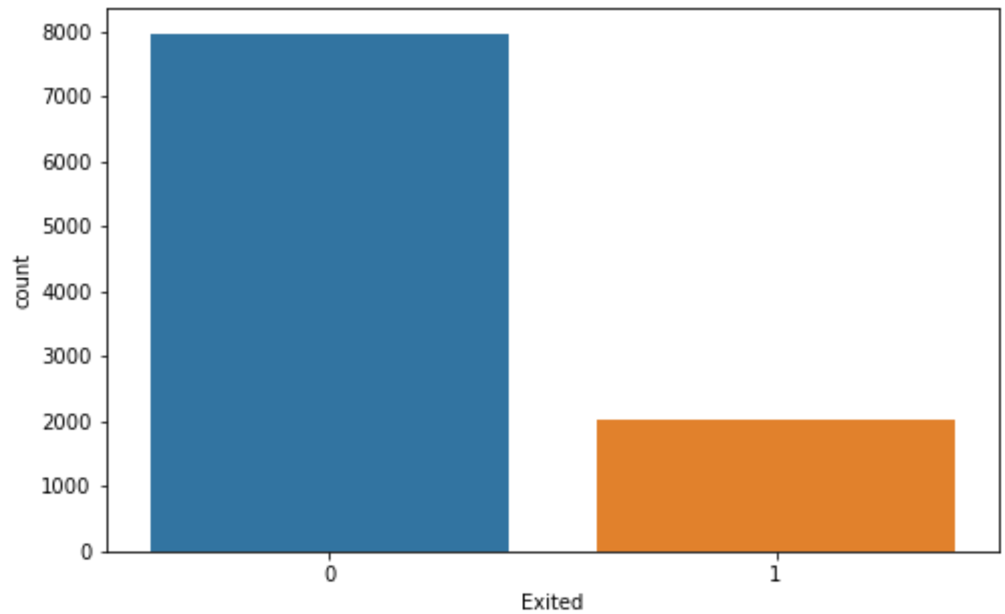
In [16]:
```
sns.distplot(df['EstimatedSalary'],hist=False)
```

Out[16]: <AxesSubplot:xlabel='EstimatedSalary', ylabel='Density'>



In [17]:
```
plt.figure(figsize=(8,5))
sns.countplot(df['Exited'])
```

Out[17]: <AxesSubplot:xlabel='Exited', ylabel='count'>



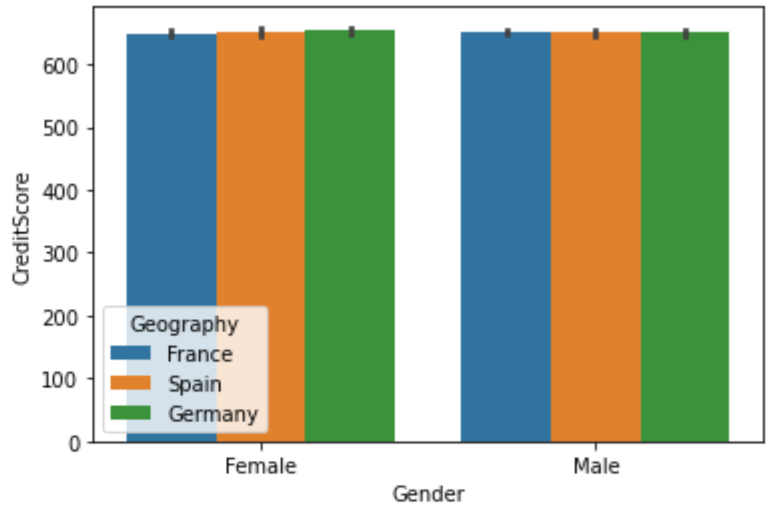
Inference:

1. The data has 11 numerical variables and 3 categorical variables.
2. It has 10000 rows and 14 columns
3. The normalized credit score is around 700, More than 500 people have credit score greater than 800.
4. France occupies 50% of customers, where as Germany and Spain shared equal.
5. Dataset is dominated by Male Customers.
6. Median age is around 40 to 45.
7. Highest number of customer has thier tenure period for 2 years.
8. Credit company has maximum customers, who uses single product.
9. Most of the customer has credit card.
10. More than 40% of the population is not an active member.
11. The Churn is less compared to the satisfaction. **Dataset is imbalanced.**

3 b). Bivariate analysis

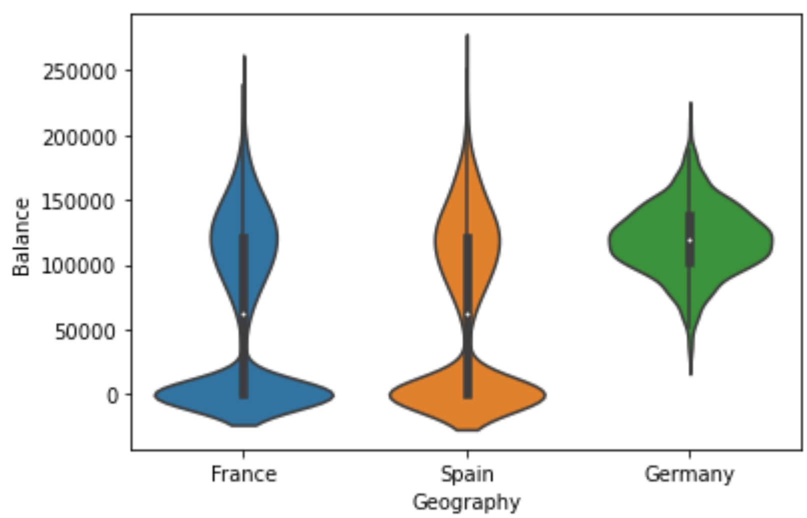
In [18]:
```
sns.barplot(x='Gender',y='CreditScore',hue='Geography',data=df)
```

Out[18]: <AxesSubplot:xlabel='Gender', ylabel='CreditScore'>



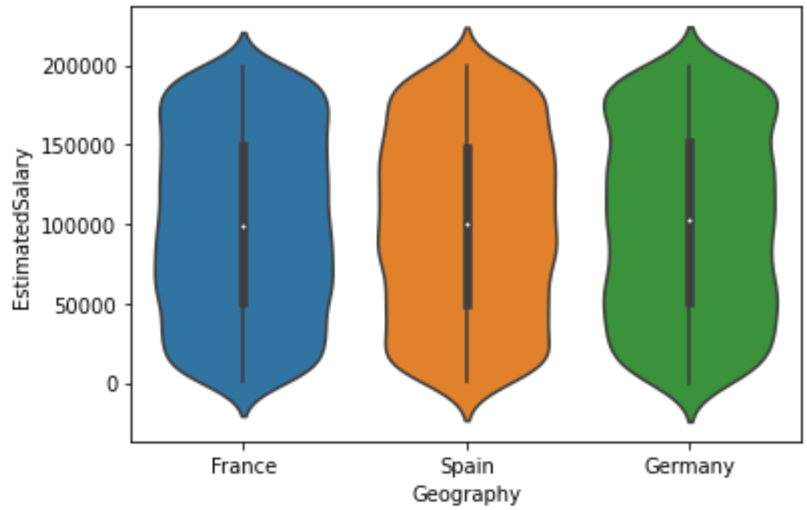
In [19]:
```
sns.violinplot(x='Geography',y='Balance',data=df)
```

Out[19]: <AxesSubplot:xlabel='Geography', ylabel='Balance'>



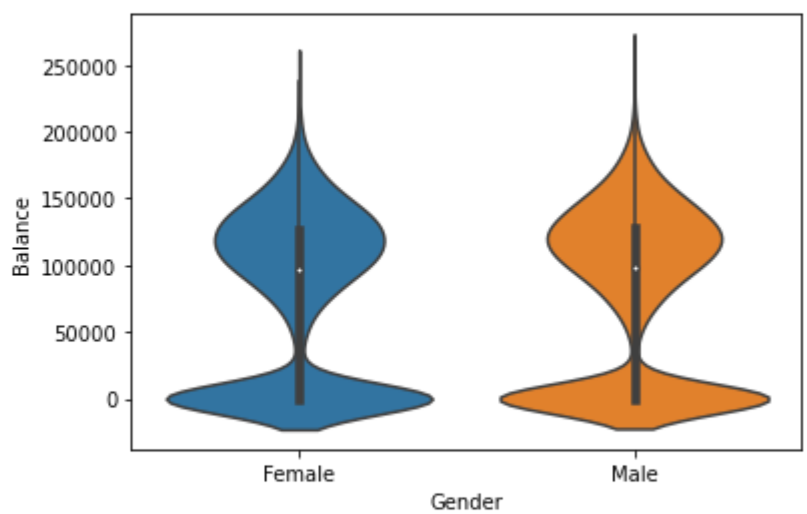
In [20]:
```
sns.violinplot(x='Geography',y='EstimatedSalary',data=df)
```

Out[20]: <AxesSubplot:xlabel='Geography', ylabel='EstimatedSalary'>



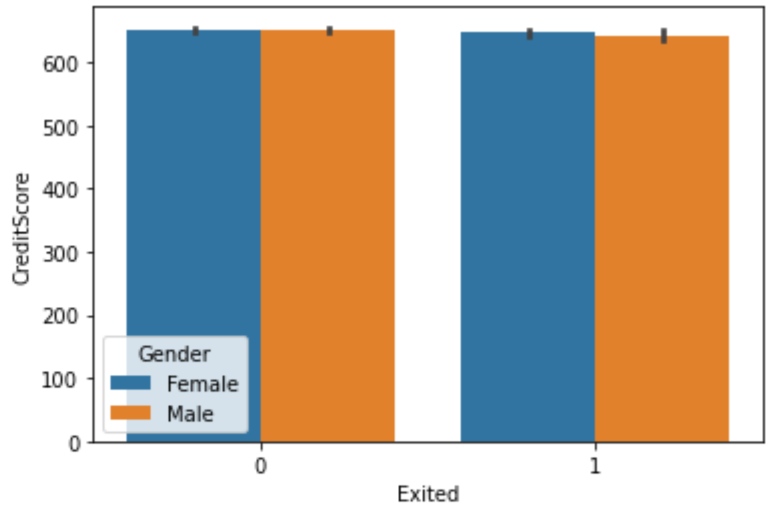
In [21]:
```
sns.violinplot(x='Gender',y='Balance',data=df)
```

Out[21]: <AxesSubplot:xlabel='Gender', ylabel='Balance'>



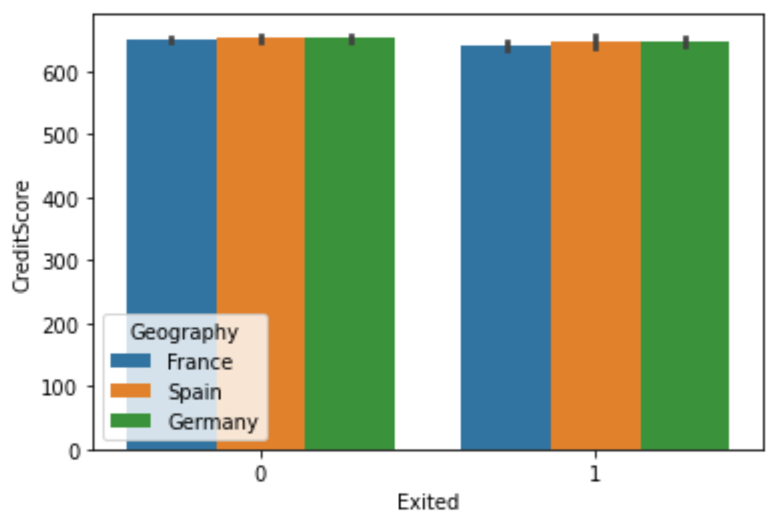
In [22]:
```
sns.barplot(x='Exited',y='CreditScore',hue='Gender',data=df)
```

Out[22]: <AxesSubplot:xlabel='Exited', ylabel='CreditScore'>



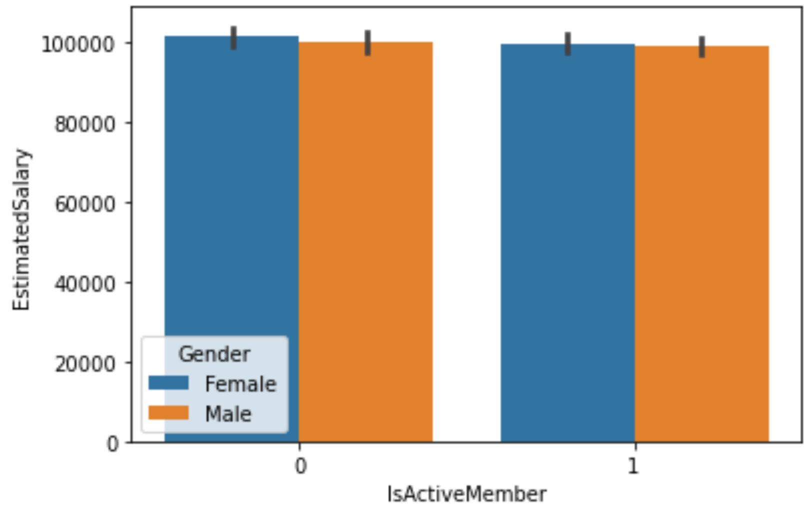
In [23]:
```
sns.barplot(x='Exited',y='CreditScore',hue='Geography',data=df)
```

Out[23]: <AxesSubplot:xlabel='Exited', ylabel='CreditScore'>



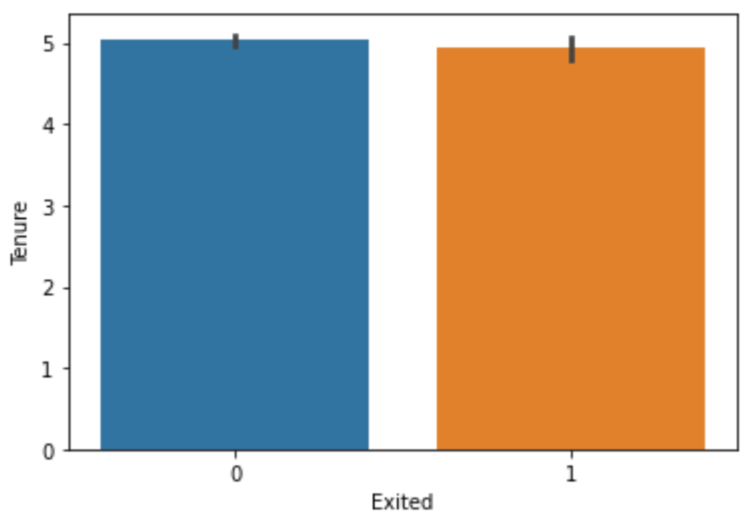
In [24]:
```
sns.barplot(x='IsActiveMember',y='EstimatedSalary',hue='Gender',data=df)
```

Out[24]: <AxesSubplot:xlabel='IsActiveMember', ylabel='EstimatedSalary'>



In [25]:
```
sns.barplot(x='Exited',y='Tenure',data=df)
```

Out[25]: <AxesSubplot:xlabel='Exited', ylabel='Tenure'>



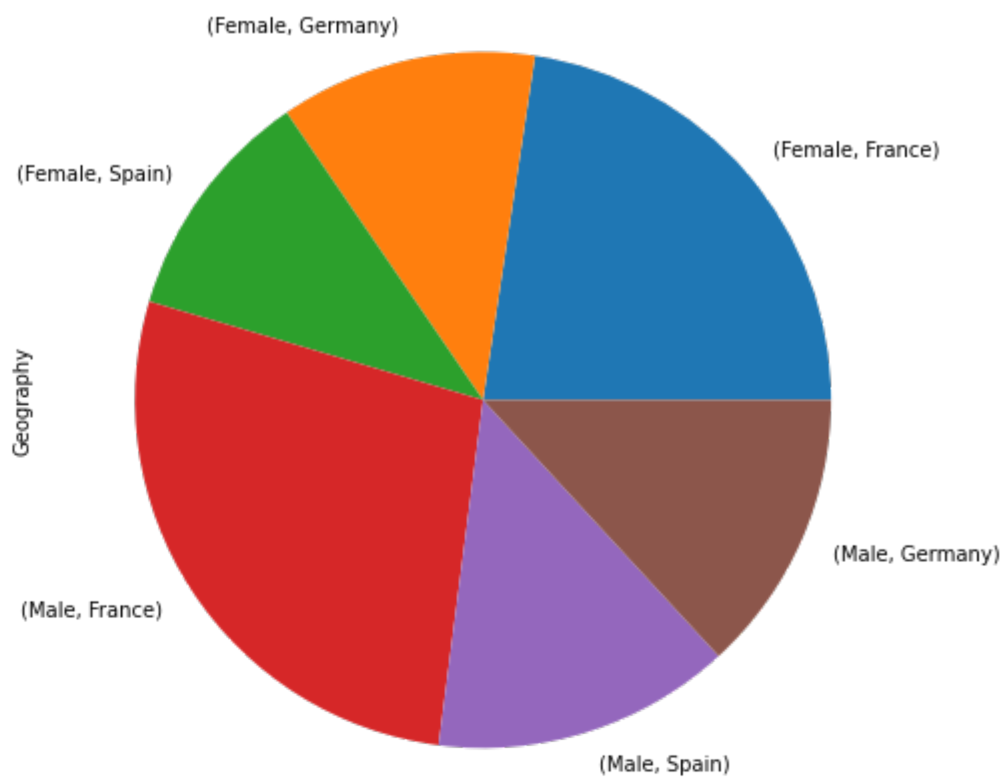
Inference:

1. Credit score for Male is higher in Spain.
2. Average bank salary lies in the range of 100k to 150k.
3. Estimated salary is normalized and same for all country.
4. Credit score for churn is low.
5. Churn in Germany is higher compared to other countries.
6. Exited people tenure period is around 6 years.

3 c). Multivariate analysis

In [26]:
```
gp1 = df.groupby('Gender')['Geography'].value_counts()
gp1.plot(kind='pie',figsize=(10,8))
print(gp1)
```

```
Gender  Geography
Female  France       2261
        Germany      1193
        Spain        1089
Male    France       2753
        Spain        1388
        Germany      1316
Name: Geography, dtype: int64
```



In [27]:
```
gp2 = df.groupby('Gender')['Age'].mean()
print(gp2)
```

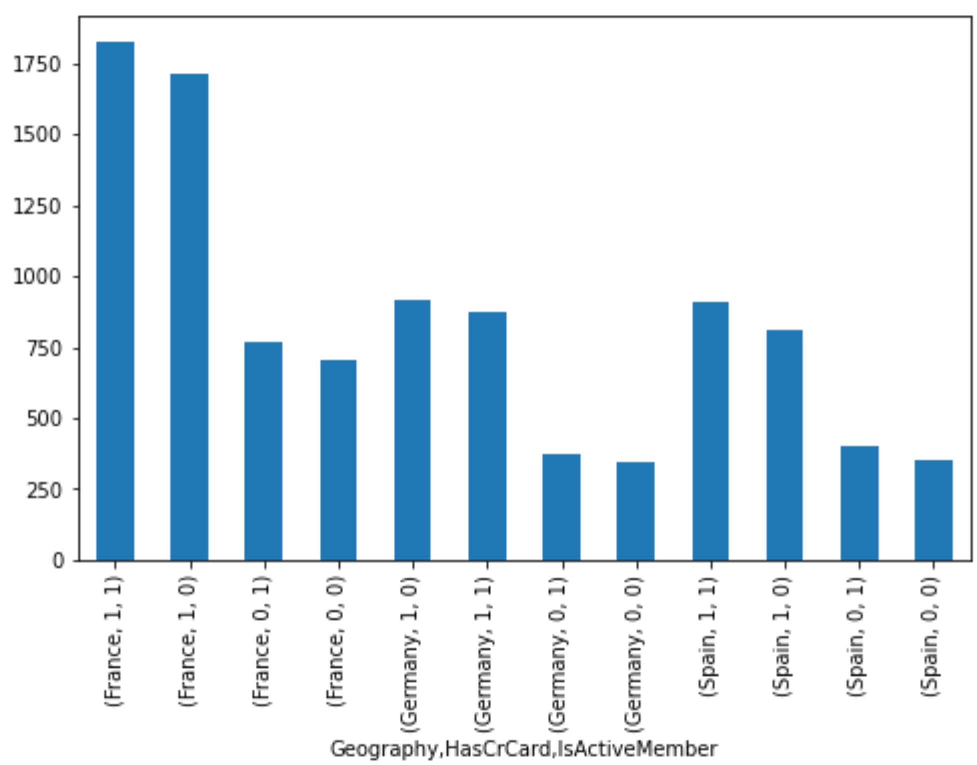
```
Gender
Female    39.238389
Male      38.658237
Name: Age, dtype: float64
```

In [28]:
```
gp3 = df.groupby(['Gender','Geography'])['Tenure'].mean()
print(gp3)
```

```
Gender  Geography
Female  France       4.950022
        Germany      4.965633
        Spain        5.000000
Male    France       5.049401
        Germany      5.050152
        Spain        5.057637
Name: Tenure, dtype: float64
```

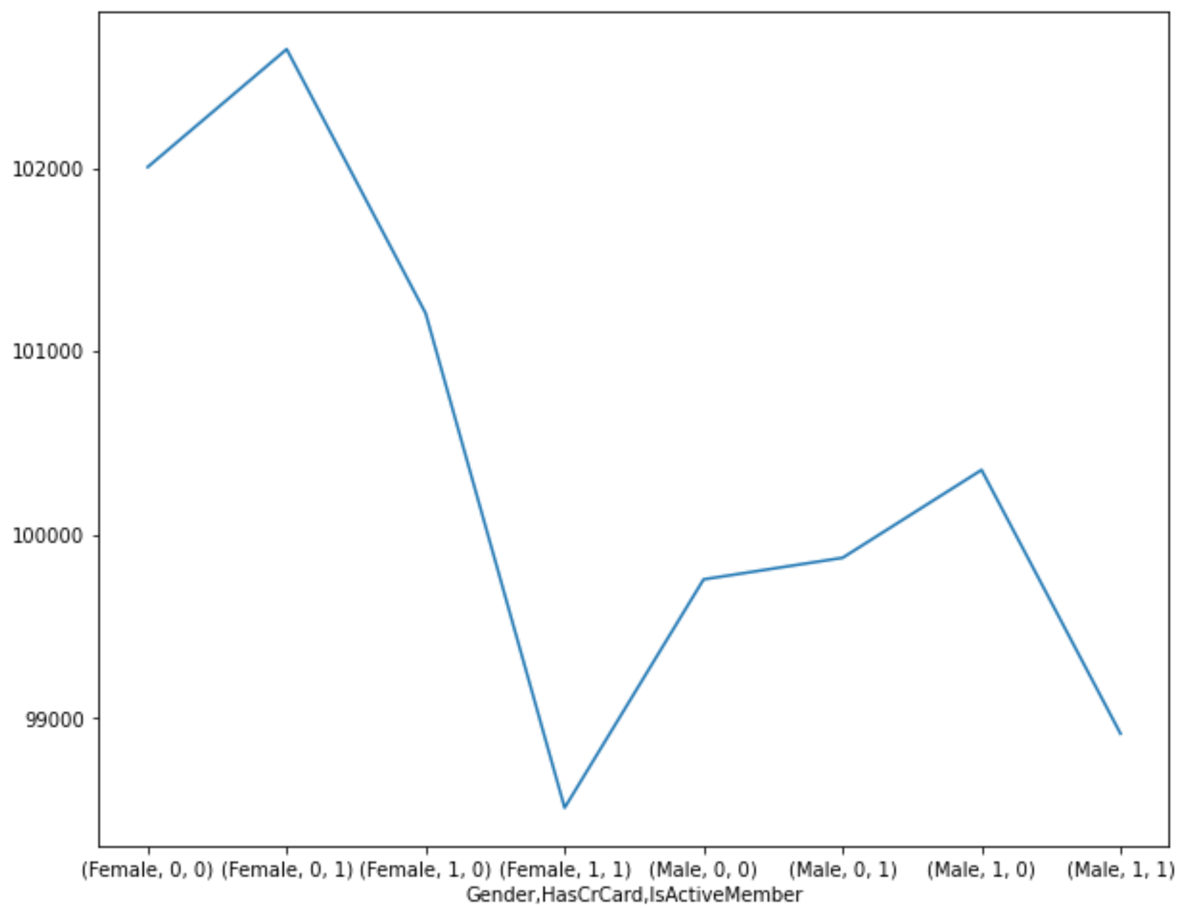
In [29]:
```
gp4 = df.groupby('Geography')['HasCrCard','IsActiveMember'].value_counts()
gp4.plot(kind="bar",figsize=(8,5))
print(gp4)
```

```
Geography  HasCrCard  IsActiveMember
France     1          1                 1826
                      0                 1717
           0          1                  765
                      0                  706
Germany    1          0                  918
                      1                  873
           0          1                  375
                      0                  343
Spain      1          1                  908
                      0                  813
           0          1                  404
                      0                  352
dtype: int64
```



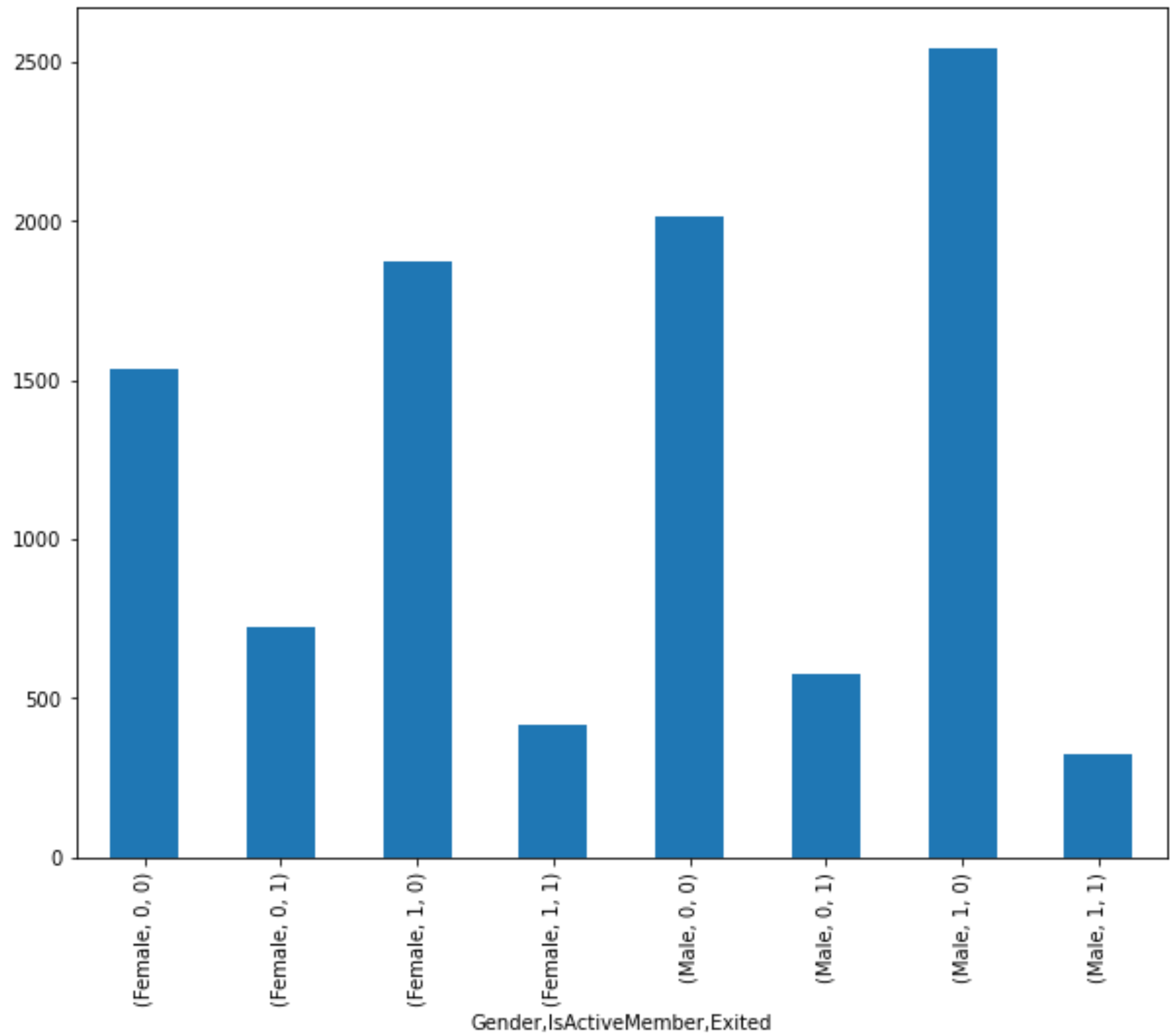
In [30]:
```
gp5 = df.groupby(['Gender','HasCrCard','IsActiveMember'])['EstimatedSalary'].m
gp5.plot(kind="line",figsize=(10,8))
print(gp5)
```

```
Gender  HasCrCard  IsActiveMember
Female  0          0                 102006.080352
                   1                 102648.996944
        1          0                 101208.014567
                   1                  98510.152300
Male    0          0                  99756.431151
                   1                  99873.931251
        1          0                 100353.378996
                   1                  98914.378703
Name: EstimatedSalary, dtype: float64
```



In [31]:
```
gp6 = df.groupby(['Gender','IsActiveMember'])['Exited'].value_counts()
gp6.plot(kind='bar',figsize=(10,8))
print(gp6)
```

```
Gender  IsActiveMember  Exited
Female  0               0         1534
                        1          725
        1               0         1870
                        1          414
Male    0               0         2013
                        1          577
        1               0         2546
                        1          321
Name: Exited, dtype: int64
```

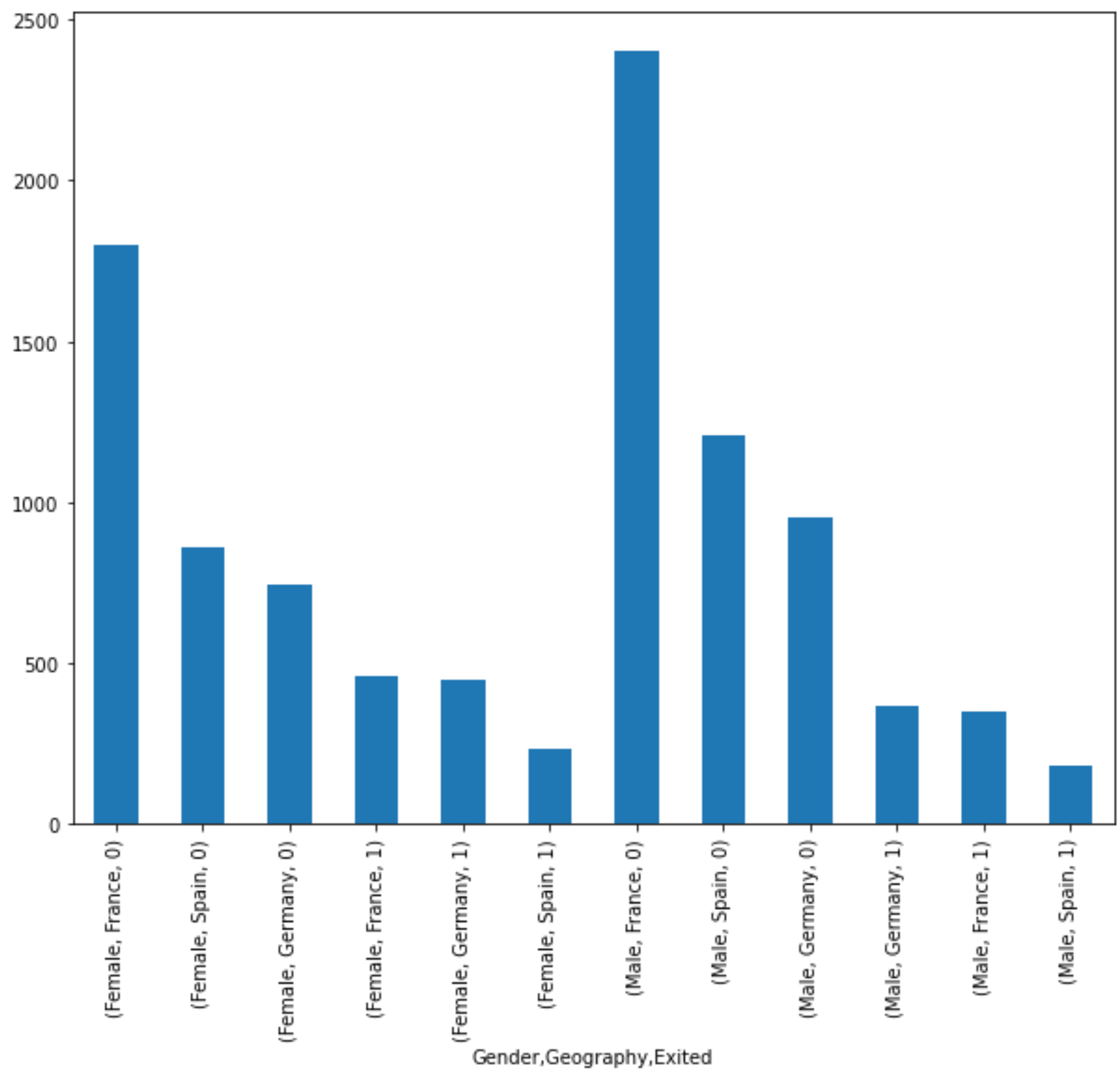


In [32]:
```
gp7 = df.groupby('Exited')['Balance','EstimatedSalary'].mean()
print(gp7)
```

```
             Balance  EstimatedSalary
Exited
0       72745.296779     99738.391772
1       91108.539337    101465.677531
```

In [33]:
```
gp8 = df.groupby('Gender')['Geography','Exited'].value_counts()
gp8.plot(kind='bar',figsize=(10,8))
print (gp8)
```

```
Gender  Geography  Exited
Female  France     0         1801
        Spain      0          858
        Germany    0          745
        France     1          460
        Germany    1          448
        Spain      1          231
Male    France     0         2403
        Spain      0         1206
        Germany    0          950
                   1          366
        France     1          350
        Spain      1          182
dtype: int64
```



Inference:

1. Germany has more female customers compared to male customers.
2. Average age of Male is 38, whereas average age of Female is 39.
3. Tenure period for both male and female is high in Spain.
4. It is observed that, those who have credit card are very active member in the company.
5. The estimated salary for a person who is not having credit card is high when compared to those having them.
6. Churn for inactive member is high compared to active member.
7. Those who churn has thier estimated salary very low.
8. France has the more churn rate.

4. Descriptive statistics

In [34]:
```
df.describe().T
```

Out[34]:

	count	mean	std	min	25%	50%
RowNumber	10000.0	5.000500e+03	2886.895680	1.00	2500.75	5.000500e+03
CustomerId	10000.0	1.569094e+07	71936.186123	15565701.00	15628528.25	1.569074e+07
CreditScore	10000.0	6.505288e+02	96.653299	350.00	584.00	6.520000e+02
Age	10000.0	3.892180e+01	10.487806	18.00	32.00	3.700000e+01
Tenure	10000.0	5.012800e+00	2.892174	0.00	3.00	5.000000e+00
Balance	10000.0	7.648589e+04	62397.405202	0.00	0.00	9.719854e+04
NumOfProducts	10000.0	1.530200e+00	0.581654	1.00	1.00	1.000000e+00
HasCrCard	10000.0	7.055000e-01	0.455840	0.00	0.00	1.000000e+00
IsActiveMember	10000.0	5.151000e-01	0.499797	0.00	0.00	1.000000e+00
EstimatedSalary	10000.0	1.000902e+05	57510.492818	11.58	51002.11	1.001939e+05
Exited	10000.0	2.037000e-01	0.402769	0.00	0.00	0.000000e+00

5. Handling the missing values

In [35]:
```
df.isnull().sum()
```

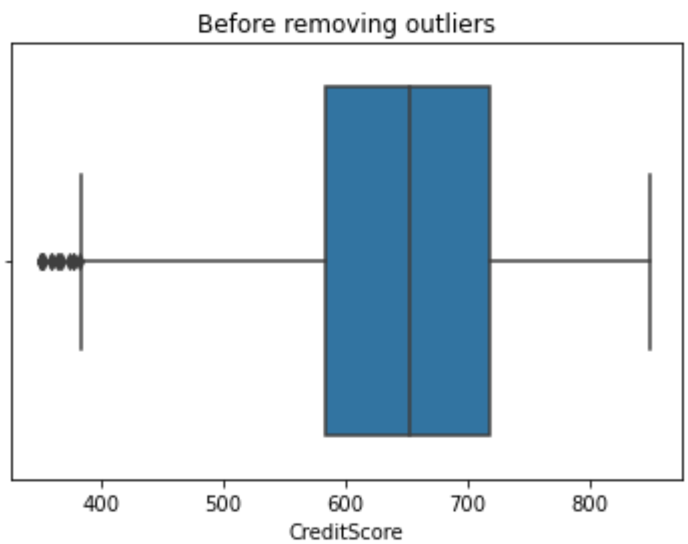
Out[35]:
```
RowNumber          0
CustomerId         0
Surname            0
CreditScore        0
Geography          0
Gender             0
Age                0
Tenure             0
Balance            0
NumOfProducts      0
HasCrCard          0
IsActiveMember     0
EstimatedSalary    0
Exited             0
dtype: int64
```

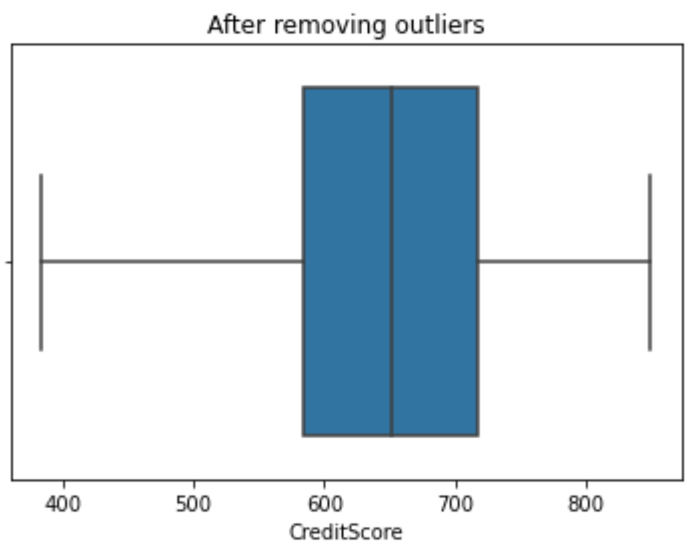
There is no missing value in the dataset

6. Finding outliers

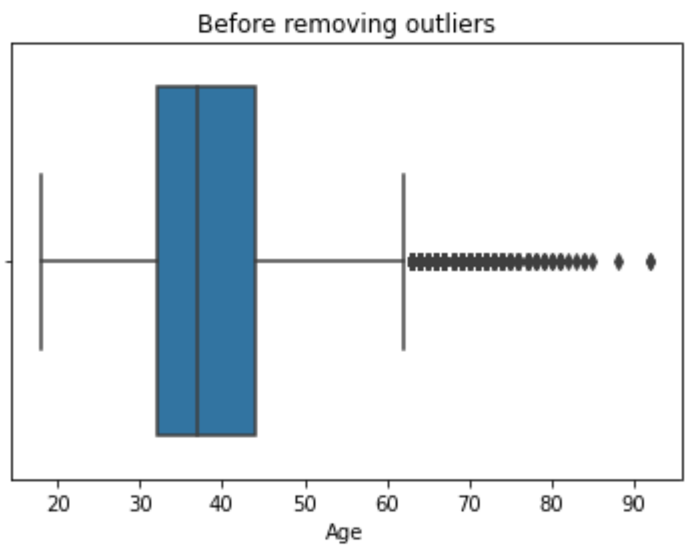
In [36]:
```
def replace_outliers(df, field_name):
    Q1 = np.percentile(df[field_name],25,interpolation='midpoint')
    Q3 = np.percentile(df[field_name],75,interpolation='midpoint')
    IQR = Q3-Q1
    maxi = Q3+1.5*IQR
    mini = Q1-1.5*IQR
    df[field_name]=df[field_name].mask(df[field_name]>maxi,maxi)
    df[field_name]=df[field_name].mask(df[field_name]<mini,mini)
```

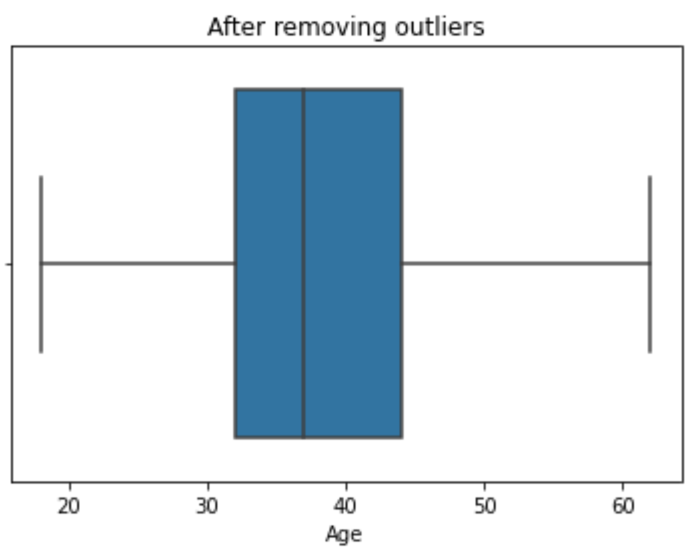
In [37]:
```
plt.title("Before removing outliers")
sns.boxplot(df['CreditScore'])
plt.show()
plt.title("After removing outliers")
replace_outliers(df, 'CreditScore')
sns.boxplot(df['CreditScore'])
plt.show()
```





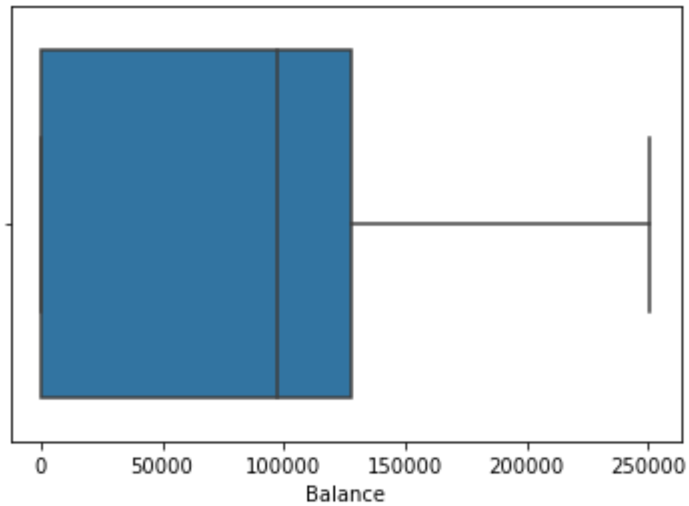
In [38]:
```
plt.title("Before removing outliers")
sns.boxplot(df['Age'])
plt.show()
plt.title("After removing outliers")
replace_outliers(df, 'Age')
sns.boxplot(df['Age'])
plt.show()
```





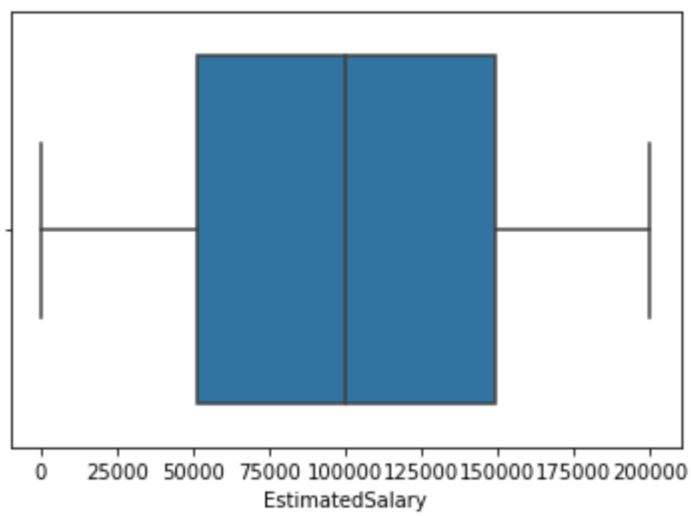
In [39]:
```
sns.boxplot(df['Balance'])
```

Out[39]: <AxesSubplot:xlabel='Balance'>



In [40]:
```
sns.boxplot(df['EstimatedSalary'])
```

Out[40]: <AxesSubplot:xlabel='EstimatedSalary'>



Outliers from Age and Credit Score columns are removed

7. Check for categorical column and perform encoding.

In [41]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

In [42]:
```
df['Gender'] = le.fit_transform(df['Gender'])
df['Geography'] = le.fit_transform(df['Geography'])
```

In [43]:
```
df.head()
```

Out[43]:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance
0	1	15634602	Hargrave	619.0	0	0	42.0	2	0.00
1	2	15647311	Hill	608.0	2	0	41.0	1	83807.86
2	3	15619304	Onio	502.0	0	0	42.0	8	159660.80
3	4	15701354	Boni	699.0	0	0	39.0	1	0.00
4	5	15737888	Mitchell	850.0	2	0	43.0	2	125510.82

Only two columns(Gender and Geography) is label encoded

Removing unwanted columns and checking for feature importance

In [44]:
```
df = df.drop(['RowNumber','CustomerId','Surname'],axis=1)
```

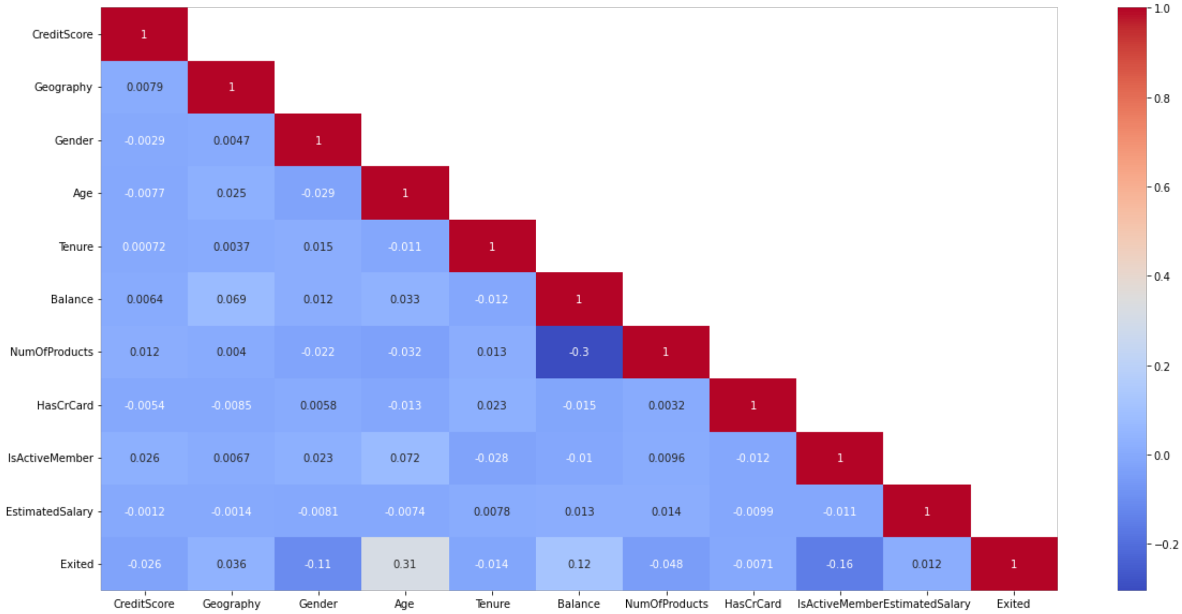
In [45]:
```
df.head()
```

Out[45]:

	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActi
0	619.0	0	0	42.0	2	0.00	1	1	
1	608.0	2	0	41.0	1	83807.86	1	0	
2	502.0	0	0	42.0	8	159660.80	3	1	
3	699.0	0	0	39.0	1	0.00	2	0	
4	850.0	2	0	43.0	2	125510.82	1	1	

In [46]:
```
plt.figure(figsize=(20,10))
df_lt = df.corr(method = "pearson")
df_lt1 = df_lt.where(np.tril(np.ones(df_lt.shape)).astype(np.bool))
sns.heatmap(df_lt1,annot=True,cmap="coolwarm")
```

Out[46]: <AxesSubplot:>



1. The Removed columns are nothing to do with model building.

2. Feature importance also checked using pearson correlation.

8. Data Splitting

In [47]:
```
target = df['Exited']
data = df.drop(['Exited'],axis=1)
```

In [48]:
```
print(data.shape)
print(target.shape)
```

```
(10000, 10)
(10000,)
```

9. Scaling the independent values

In [49]:
```
from sklearn.preprocessing import StandardScaler
se = StandardScaler()
```

In [50]:
```
data['CreditScore'] = se.fit_transform(pd.DataFrame(data['CreditScore']))
data['Age'] = se.fit_transform(pd.DataFrame(data['Age']))
data['Balance'] = se.fit_transform(pd.DataFrame(data['Balance']))
data['EstimatedSalary'] = se.fit_transform(pd.DataFrame(data['EstimatedSalary'
```

In [51]:
```
data.head()
```

Out[51]:

	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	Is
0	-0.326878	0	0	0.342615	2	-1.225848	1	1	
1	-0.440804	2	0	0.240011	1	0.117350	1	0	
2	-1.538636	0	0	0.342615	8	1.333053	3	1	
3	0.501675	0	0	0.034803	1	-1.225848	2	0	
4	2.065569	2	0	0.445219	2	0.785728	1	1	

10. Train test split

In [52]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(data,target,test_size=0.25,ra
```

In [53]:
```
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(7500, 10)
(2500, 10)
(7500,)
(2500,)
```

Conclusion:

1. The model is scaled using StandarScaler method.
2. The train and test split ratio is 15:5.
3. As it is a classification problem, basic algorithms can be used to build ML models.