# Data Visualization and Pre-processing

## Import libraries

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

## Load dataset

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
data = pd.read_csv('drive/My Drive/Churn_Modelling.csv')
```

```
data.head()
```

| | RowNumber | CustomerId | Surname | CreditScore | Geography | Gender | Age | Tenure | Ba |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 15634602 | Hargrave | 619 | France | Female | 42 | 2 | |
| **1** | 2 | 15647311 | Hill | 608 | Spain | Female | 41 | 1 | 838 |
| **2** | 3 | 15619304 | Onio | 502 | France | Female | 42 | 8 | 1596 |
| **3** | 4 | 15701354 | Boni | 699 | France | Female | 39 | 1 | |
| **4** | 5 | 15737888 | Mitchell | 850 | Spain | Female | 43 | 2 | 1255 |

```
data.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
```

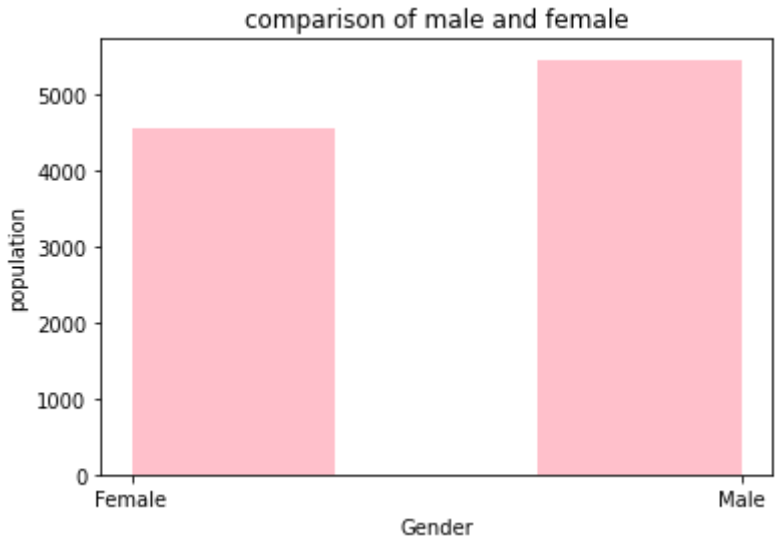
# Visualisations

### 1. Univariate Analysis

```
data['Gender'].value_counts()
```

```
Male      5457
Female    4543
Name: Gender, dtype: int64
```

```
# Plotting the features of the dataset to see the correlation between them

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

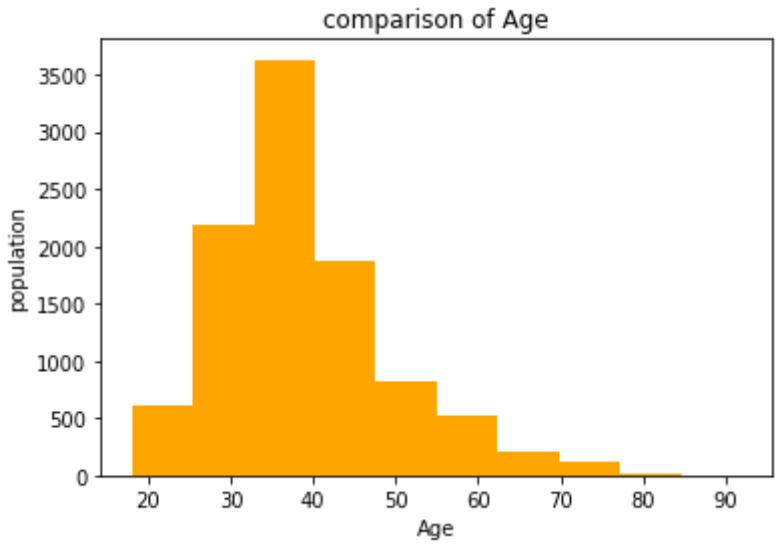



```
data['Age'].value_counts()
```

```
37    478
38    477
35    474
36    456
34    447
```

```
...
92      2
82      1
88      1
85      1
83      1
Name: Age, Length: 70, dtype: int64
```

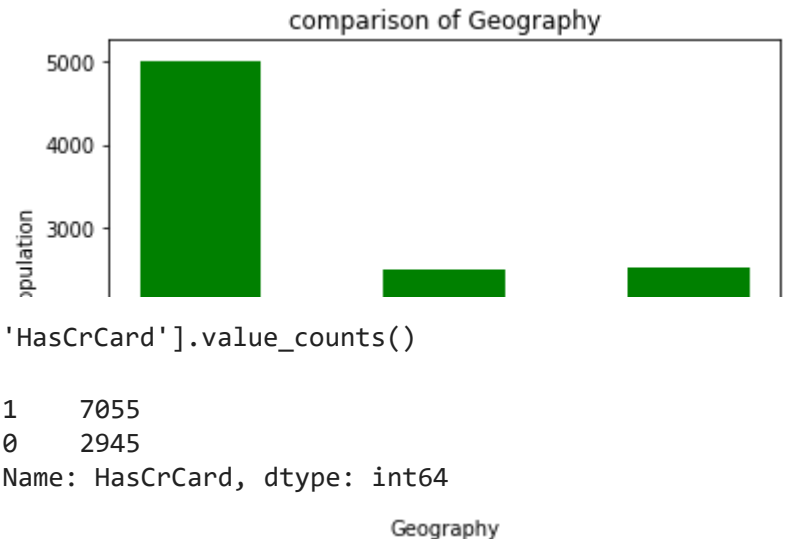
```
# comparison of age in the dataset

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




```
data['Geography'].value_counts()
```

```
France     5014
Germany    2509
Spain      2477
Name: Geography, dtype: int64
```

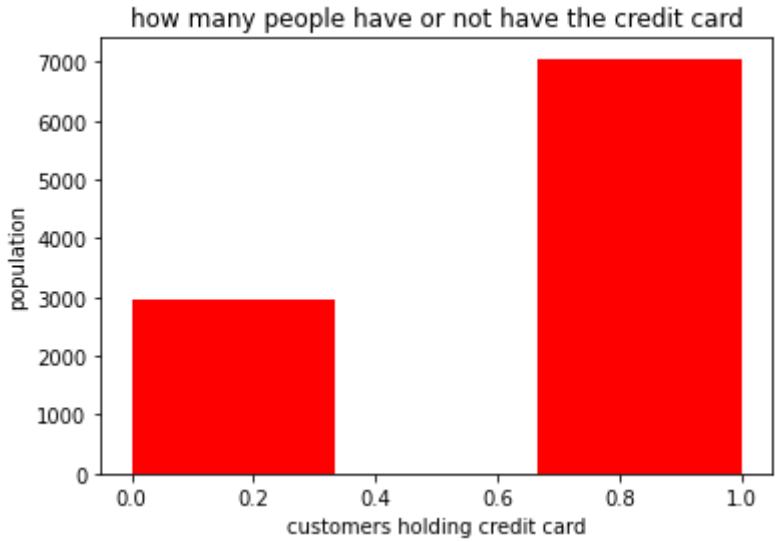
```
# comparison of geography

plt.hist(x = data.Geography, bins = 5, color = 'green')
plt.title('comparison of Geography')
plt.xlabel('Geography')
plt.ylabel('population')
plt.show()
```




```
data['HasCrCard'].value_counts()
```

```
1    7055
0    2945
Name: HasCrCard, dtype: int64
```

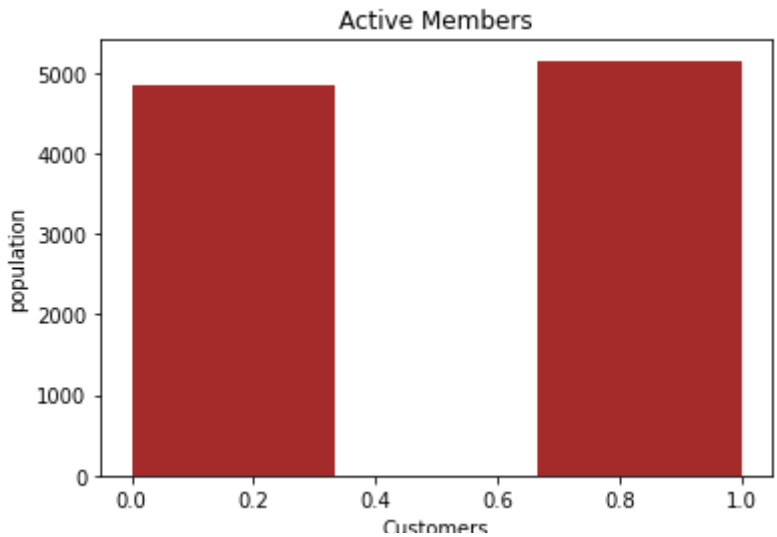
```
# comparision of how many customers hold the credit card

plt.hist(x = data.HasCrCard, bins = 3, color = 'red')
plt.title('how many people have or not have the credit card')
plt.xlabel('customers holding credit card')
plt.ylabel('population')
plt.show()
```




```
data['IsActiveMember'].value_counts()
```

```
1    5151
0    4849
Name: IsActiveMember, dtype: int64
```

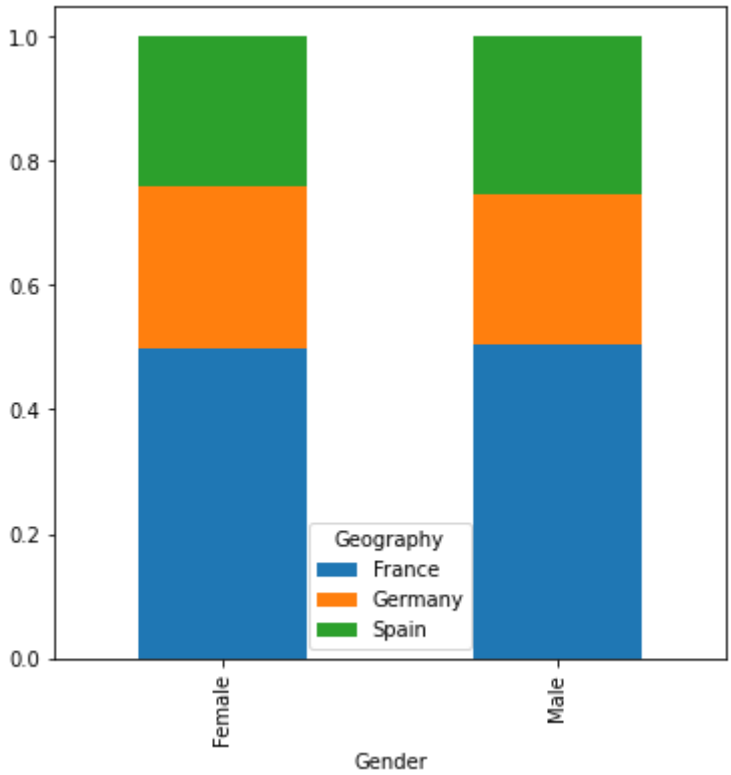
```
# How many active member does the bank have ?

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




## 2. Bi - Variate Analysis

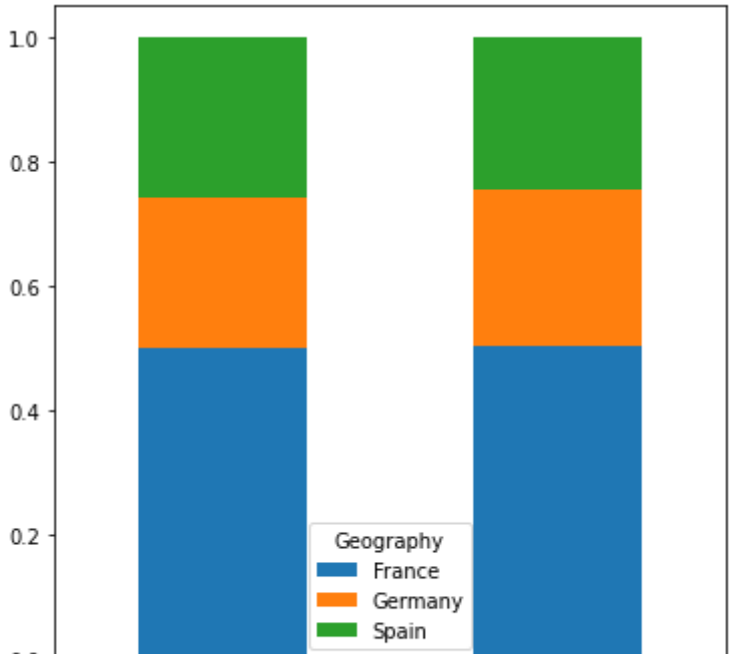
```
# comparison between Geography and Gender

Gender = pd.crosstab(data['Gender'],data['Geography'])
Gender.div(Gender.sum(1).astype(float), axis=0).plot(kind="bar", stacked=True, figsize=(6,
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6a93dbbfd0>
```



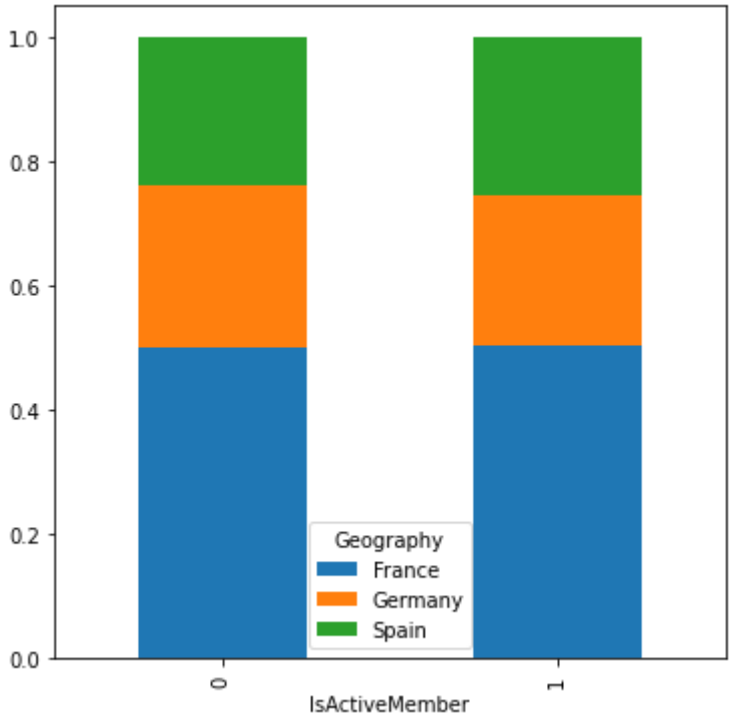

```
# comparison between geography and card holders

HasCrCard = pd.crosstab(data['HasCrCard'], data['Geography'])
HasCrCard.div(HasCrCard.sum(1).astype(float), axis = 0).plot(kind = 'bar',
                                                stacked = True,figsize = (6, 6))
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6a93ced590>
```



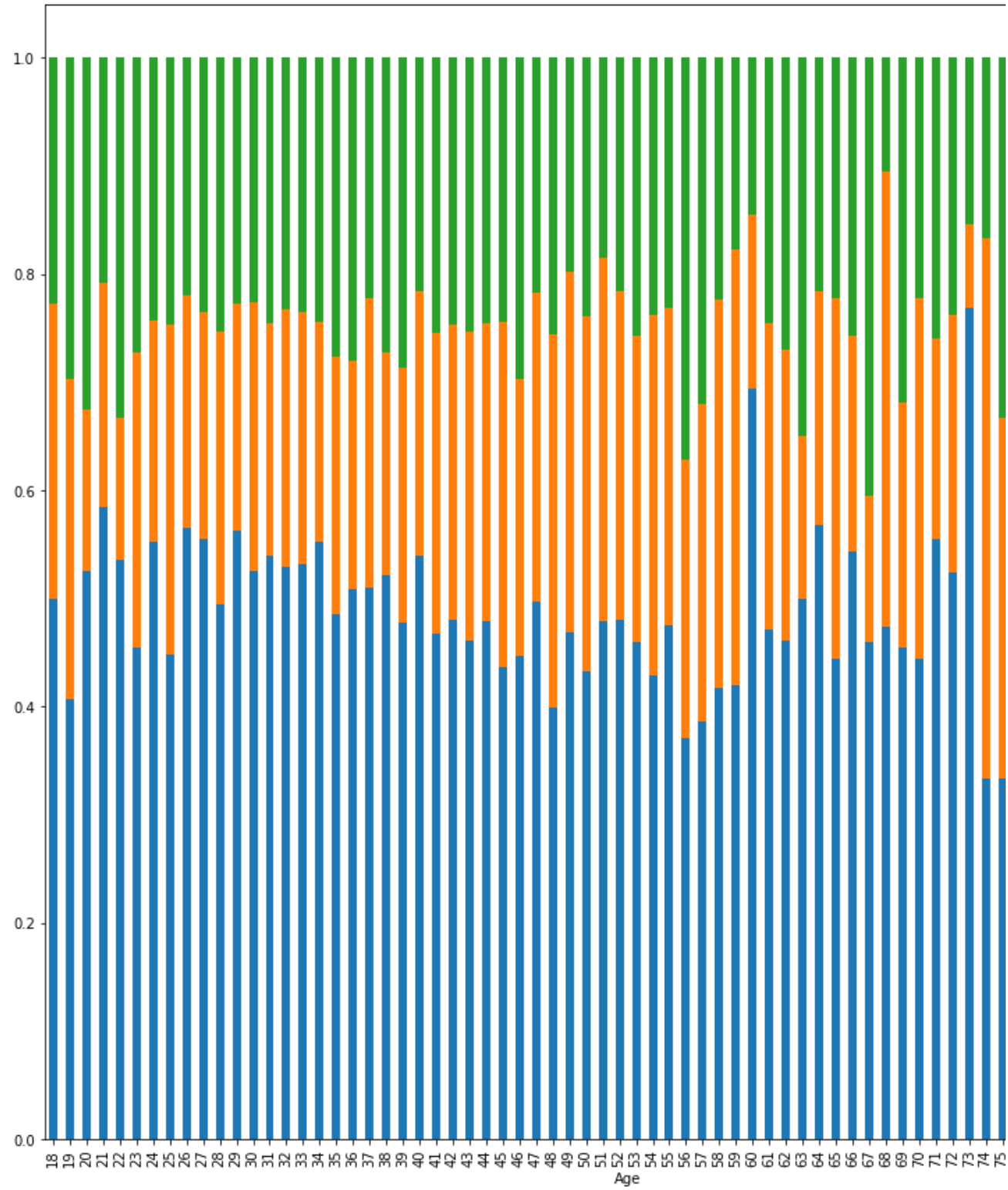

```
# comparison of active member in differnt geographies

IsActiveMember = pd.crosstab(data['IsActiveMember'], data['Geography'])
IsActiveMember.div(IsActiveMember.sum(1).astype(float), axis = 0).plot(kind = 'bar',
                                                stacked = True, figsize= (6, 6))
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6a93c7c950>
```




```
# comparing ages in different geographies

Age = pd.crosstab(data['Age'], data['Geography'])
Age.div(Age.sum(1).astype(float), axis = 0).plot(kind = 'bar',
                                                stacked = True, figsize = (15,15))
```

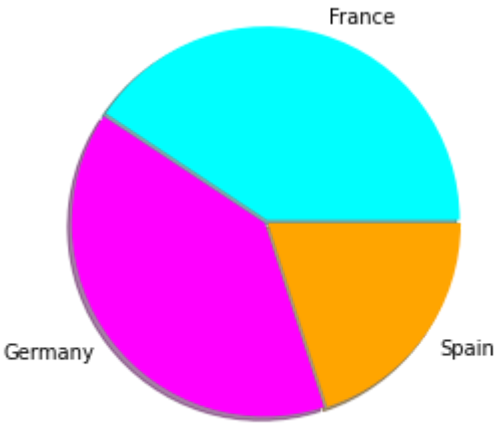
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6a93bfea10>
```




```
# calculating total balance in france, germany and spain

total_france = data.Balance[data.Geography == 'France'].sum()
total_germany = data.Balance[data.Geography == 'Germany'].sum()
total_spain = data.Balance[data.Geography == 'Spain'].sum()

print("Total Balance in France :",total_france)
print("Total Balance in Germany :",total_germany)
print("Total Balance in Spain :",total_spain)
```

```
Total Balance in France : 311332479.49
Total Balance in Germany : 300402861.38
Total Balance in Spain : 153123552.01
```

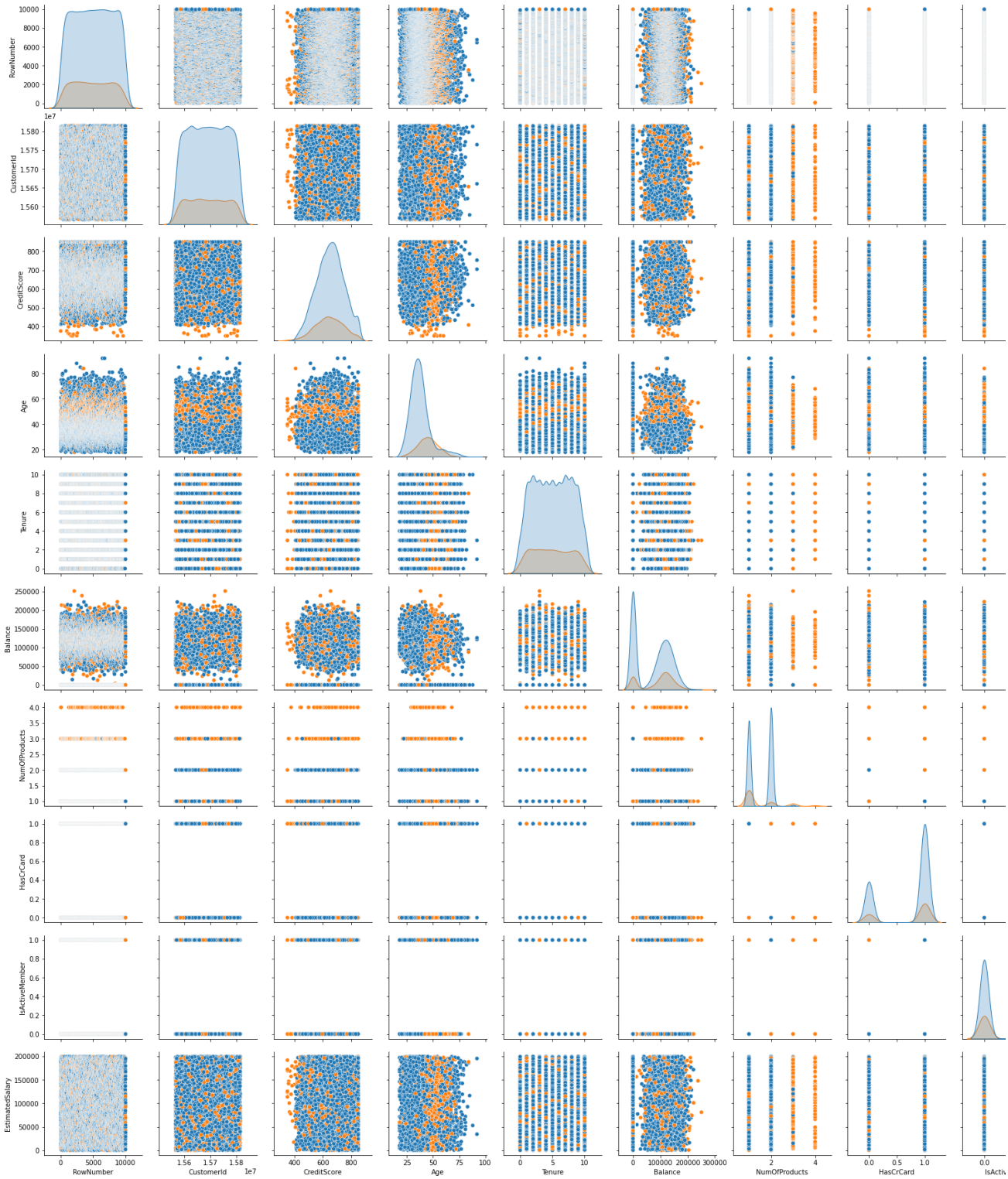
```
# plotting a pie chart

labels = 'France', 'Germany', 'Spain'
colors = ['cyan', 'magenta', 'orange']
sizes =  [311, 300, 153]
explode = [ 0.01, 0.01, 0.01]

plt.pie(sizes, colors = colors, labels = labels, explode = explode, shadow = True)

plt.axis('equal')
plt.show()
```




### 3. Multi - Variate Analysis

```
sns.pairplot(data=data, hue='Exited')
```

```
<seaborn.axisgrid.PairGrid at 0x7f6a93ddd510>
```



# Descriptive statistics

[ ] ↳ *1 cell hidden*

# Handle the Missing values

[ ] ↳ *2 cells hidden*

# Find the outliers and replace the outliers

[ ] ↳ *6 cells hidden*

# Preprocessing

[ ] ↳ *2 cells hidden*

# Split the data into dependent and independent variables

[ ] ↳ *1 cell hidden*

# Check for Categorical columns and perform encoding

[ ] ↳ *1 cell hidden*

# Split the data into training and testing

[ ] ↳ *1 cell hidden*

# Scale the independent variables

[ ] ↳ *1 cell hidden*