Load dataset and importing required library

```
import numpy as np
import pandas as pd
```

Double-click (or enter) to edit

```
df = pd.read_csv('/content/sample_data/Churn_Modelling.csv')
```

```
df.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

```
df.shape
```

```
(10000, 14)
```

Statistical analysis

```
df.info
```

```
<bound method DataFrame.info of       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   
...         ...         ...        ...          ...       ...     ...  ...   
9995       9996    15606229   Obijiaku          771    France    Male   39   
9996       9997    15569892  Johnstone          516    France    Male   35   
9997       9998    15584532        Liu          709    France  Female   36   
9998       9999    15682355  Sabbatini          772   Germany    Male   42   
9999      10000    15628319     Walker          792    France  Female   28   

      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
     ...      ...        ...            ...        ...             ...
     9995       5       0.00              2          1               0
     9996      10   57369.61              1          1               1
     9997       7       0.00              1          0               1
     9998       3   75075.31              2          1               0
     9999       4  130142.79              1          1               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]>
```

```
df.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
```

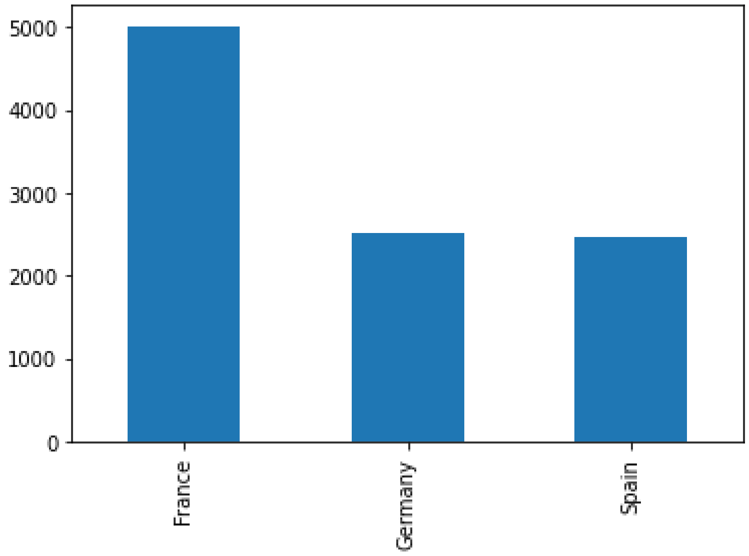
```
df.drop(['RowNumber','CustomerId','Surname'],axis=1, inplace=True)
```

Data Visualization

```
import matplotlib.pyplot as plt
import seaborn as sns
```

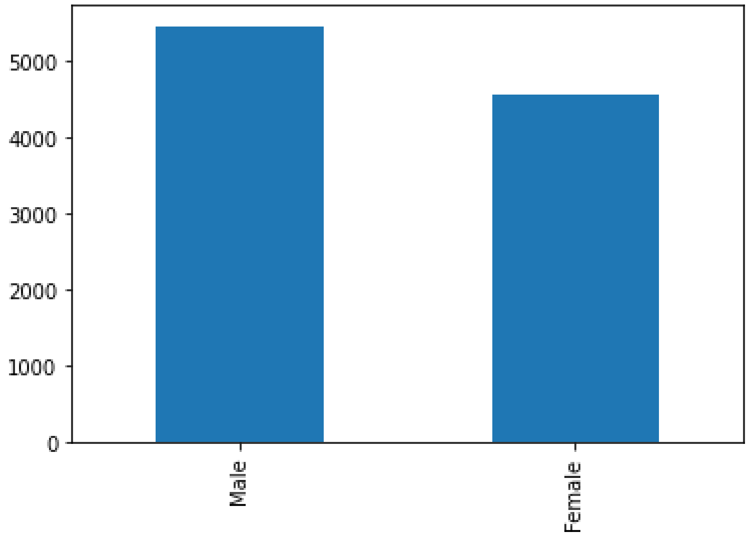
```
df.Geography.value_counts().plot(kind='bar')
df.Geography.value_counts()
```

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



```
df.Gender.value_counts().plot(kind='bar')
df.Gender.value_counts()
```

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

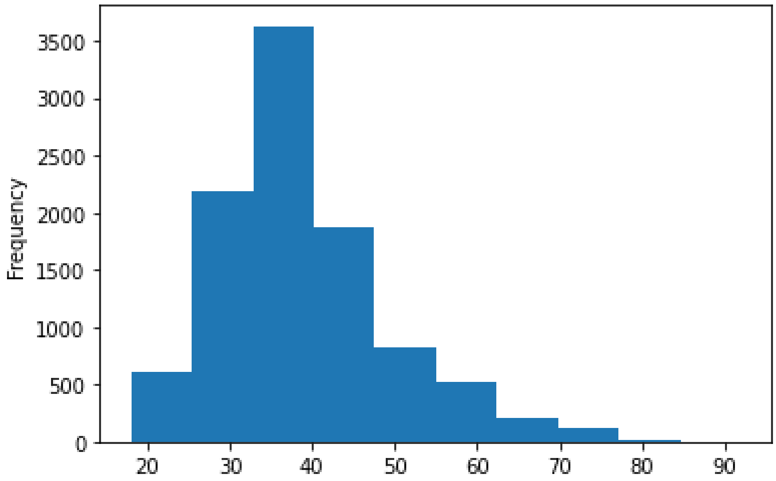


```
df.Age.describe()
```

```
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
```

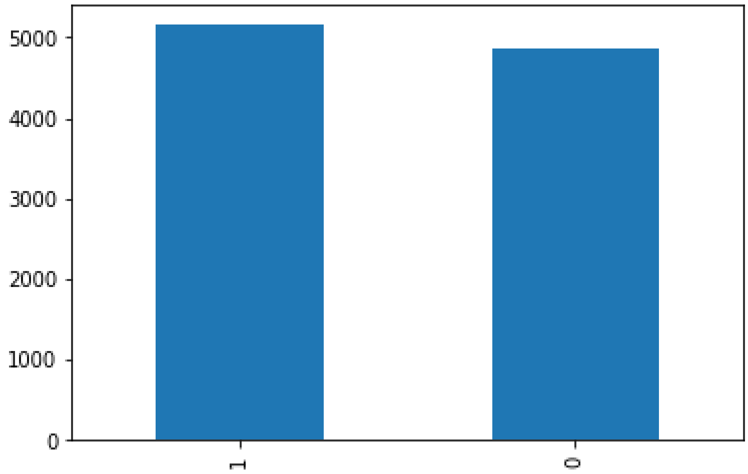
```
df.Age.plot(kind='hist')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8482367e90>
```



```
df.IsActiveMember.value_counts().plot(kind='bar')
df.IsActiveMember.value_counts()
```

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

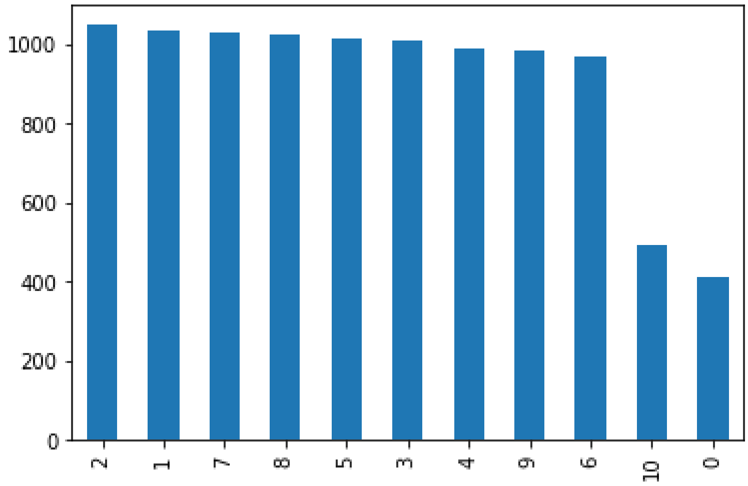


```
df.HasCrCard.value_counts().plot(kind='bar')
df.HasCrCard.value_counts()
```

```
1    7055
0    2945
```

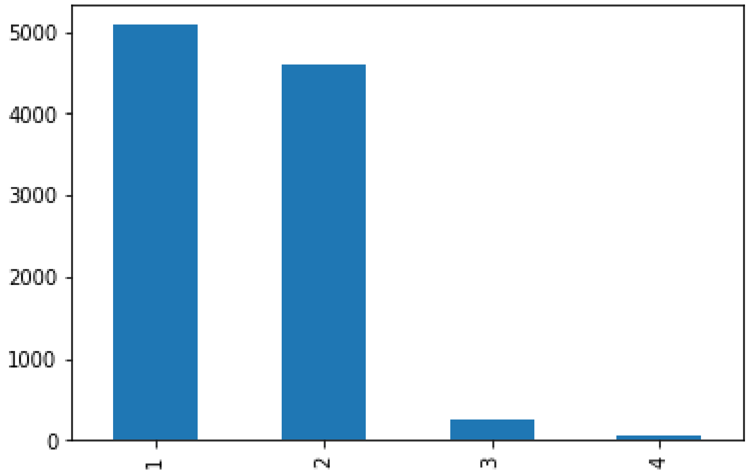
```
df.Tenure.value_counts().plot(kind='bar');
df.Tenure.value_counts()
```

```
2     1048
1     1035
7     1028
8     1025
5     1012
3     1009
4      989
9      984
6      967
10     490
0      413
Name: Tenure, dtype: int64
```



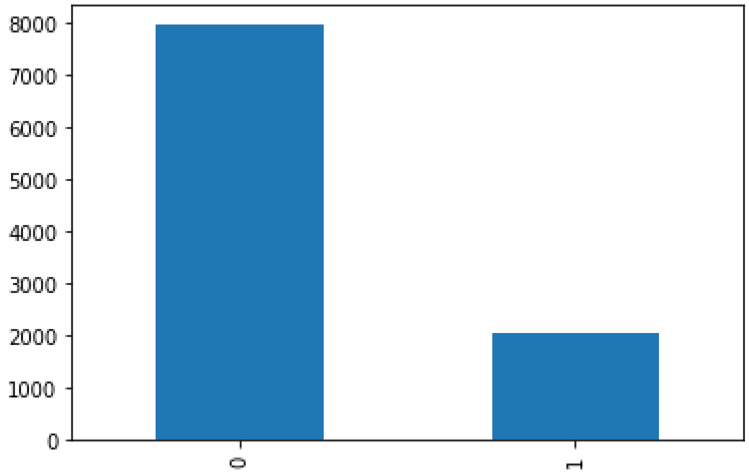
```
df.NumOfProducts.value_counts().plot(kind='bar');
df.NumOfProducts.value_counts()
```

```
1    5084
2    4590
3     266
4      60
Name: NumOfProducts, dtype: int64
```



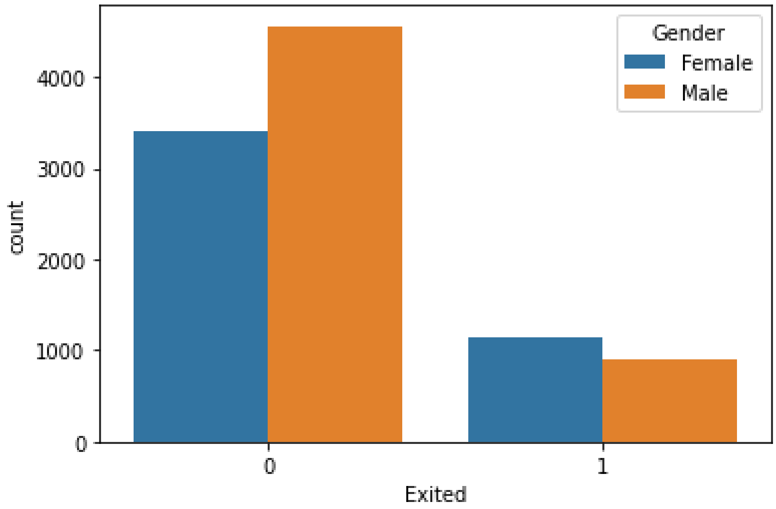
```
df.Exited.value_counts().plot(kind='bar');
df.Exited.value_counts()
```

```
0    7963
1    2037
Name: Exited, dtype: int64
```



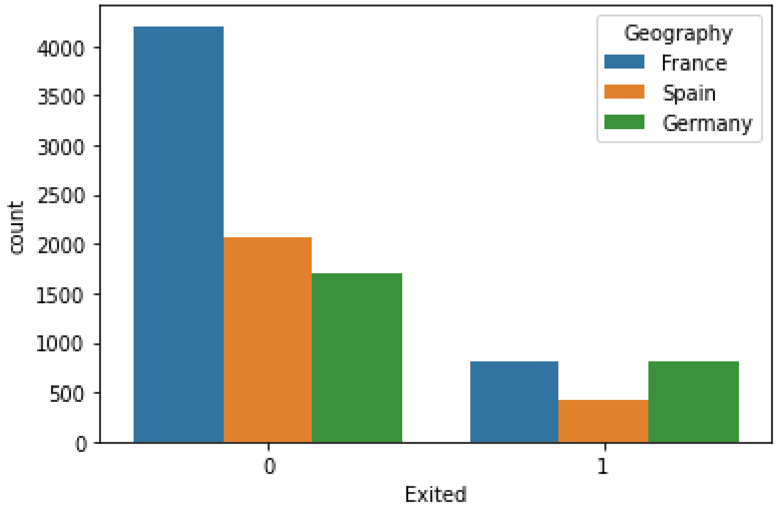
```
sns.countplot(x=df.Exited,hue=df.Gender)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8481bb1410>
```



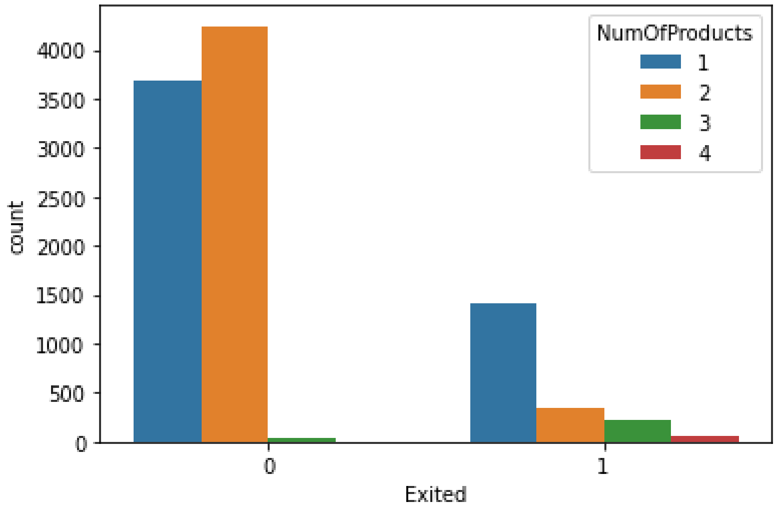
```
sns.countplot(x=df.Exited,hue=df.Geography)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8481b387d0>
```



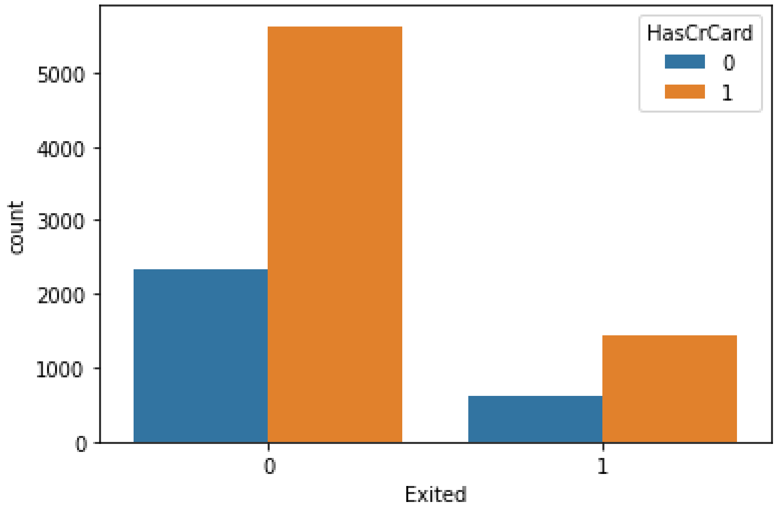
```
sns.countplot(x=df.Exited,hue=df.NumOfProducts)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8481ab1e50>
```



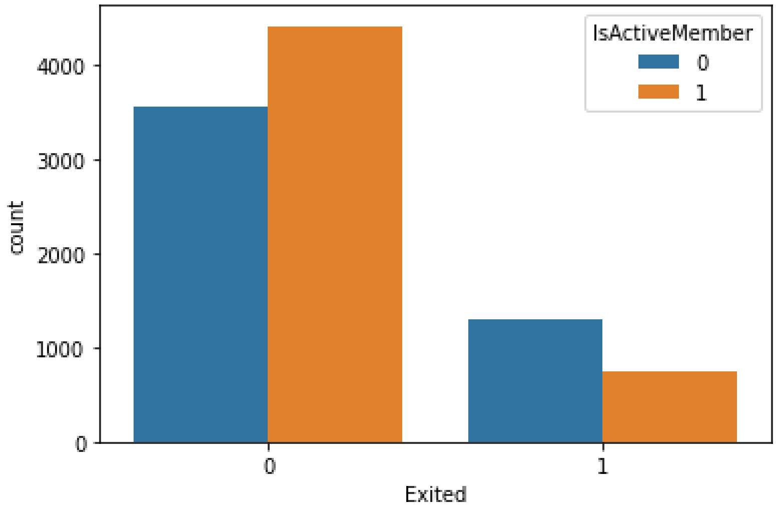
```
sns.countplot(x=df.Exited,hue=df.HasCrCard)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8481a43a50>
```



```
sns.countplot(x=df.Exited,hue=df.IsActiveMember)
```

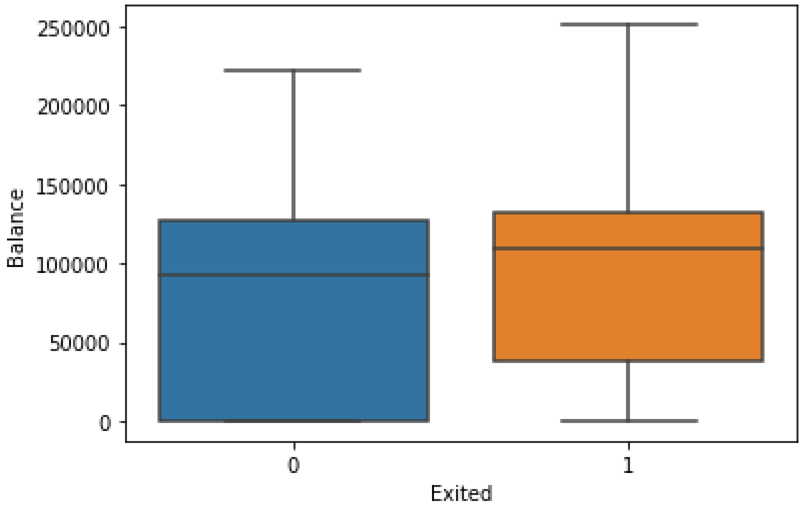
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f84819ade90>
```



Finding Outliers

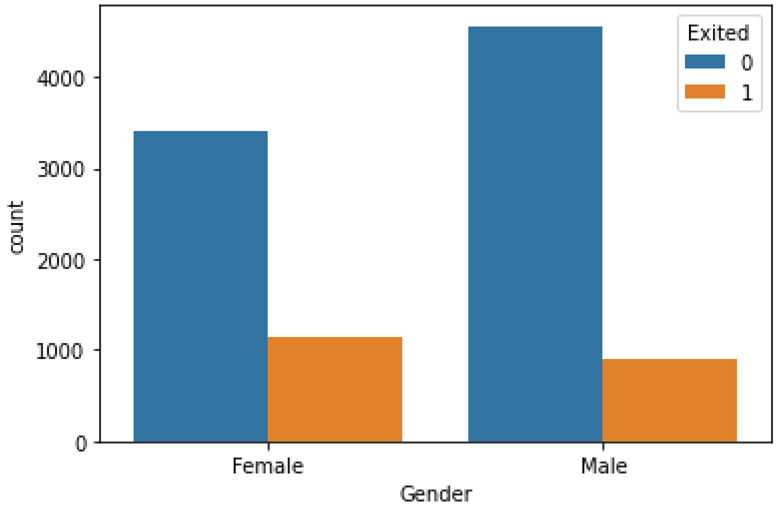
```
sns.boxplot(x=df.Exited,y=df.Balance)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8481a2fd90>
```



```
sns.countplot(x="Gender",hue="Exited",data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8481c00050>
```



categorizing with LabelEncoding

```
df['Geography']=df['Geography'].map({'France':0,'Spain':1,'Germany':2})
```

```
X=df.iloc[:,:-1].values
y=df.iloc[:,-1].values
```

```
X.shape
```

```
     (10000, 10)
```

Feature Scaling

```
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import MinMaxScaler
```

```
le=LabelEncoder()
X[:,2]=le.fit_transform(X[:,2])
```

```
print(X)
```

```
     [[619 0 0 ... 1 1 101348.88]
      [608 1 0 ... 0 1 112542.58]
      [502 0 0 ... 1 0 113931.57]
      ...
      [709 0 0 ... 0 1 42085.58]
      [772 2 1 ... 1 0 92888.52]
      [792 0 0 ... 1 0 38190.78]]
```

```
MnScaler = MinMaxScaler()
```

```
X = MnScaler.fit_transform(X)
```

Train Test Split

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test =train_test_split(X,y,test_size=0.2, random_state=0)
```

```
from sklearn.preprocessing import StandardScaler
```

```
stdscaler = StandardScaler()
X_train = stdscaler.fit_transform(X_train)
X_test = stdscaler.transform(X_test)
```