Assignment

In [ ]:

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

Load csv file

In [ ]:

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

Out[ ]:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasC
0	1	15634602	Hargrave	619	France	Female	42	2	0.00	1	
1	2	15647311	Hill	608	Spain	Female	41	1	83807.86	1	
2	3	15619304	Onio	502	France	Female	42	8	159660.80	3	
3	4	15701354	Boni	699	France	Female	39	1	0.00	2	
4	5	15737888	Mitchell	850	Spain	Female	43	2	125510.82	1	
...	...	...	...	...	...	...	...	...	...	...	
9995	9996	15606229	Obijiaku	771	France	Male	39	5	0.00	2	
9996	9997	15569892	Johnstone	516	France	Male	35	10	57369.61	1	
9997	9998	15584532	Liu	709	France	Female	36	7	0.00	1	
9998	9999	15682355	Sabbatini	772	Germany	Male	42	3	75075.31	2	
9999	10000	15628319	Walker	792	France	Female	28	4	130142.79	1	

10000 rows × 14 columns

In [ ]:

```

```

Data virtualization

In [ ]:

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

In [ ]:

```
df1=df.head(10)
df1
```

Out[ ]:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCar
0	1	15634602	Hargrave	619	France	Female	42	2	0.00	1	
1	2	15647311	Hill	608	Spain	Female	41	1	83807.86	1	

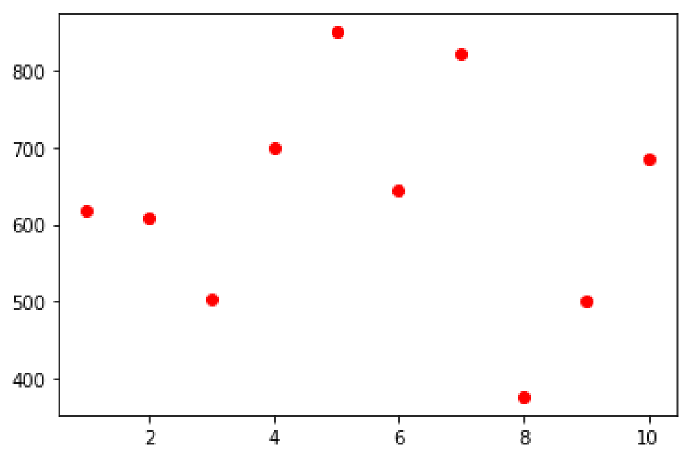
	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard
2	3	15619304	Onio	502	France	Female	42	8	159660.80	3	
3	4	15701354	Boni	699	France	Female	39	1	0.00	2	
4	5	15737888	Mitchell	850	Spain	Female	43	2	125510.82	1	
5	6	15574012	Chu	645	Spain	Male	44	8	113755.78	2	
6	7	15592531	Bartlett	822	France	Male	50	7	0.00	2	
7	8	15656148	Obinna	376	Germany	Female	29	4	115046.74	4	
8	9	15792365	He	501	France	Male	44	4	142051.07	2	
9	10	15592389	H?	684	France	Male	27	2	134603.88	1	

In [ ]:

```
plt.scatter(df1['RowNumber'],df1['CreditScore'],color='r')
```

Out[ ]:

```
<matplotlib.collections.PathCollection at 0x7f938fed23d0>
```



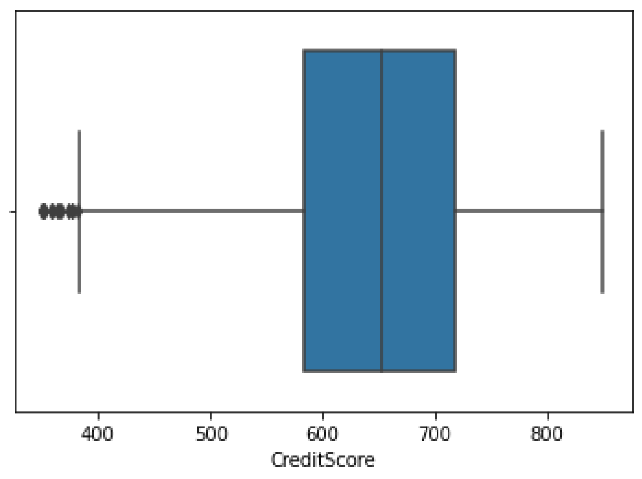
In [ ]:

```
sns.boxplot(df['CreditScore'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: 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.
  FutureWarning
```

Out[ ]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f938fe3f8d0>
```



In [ ]:

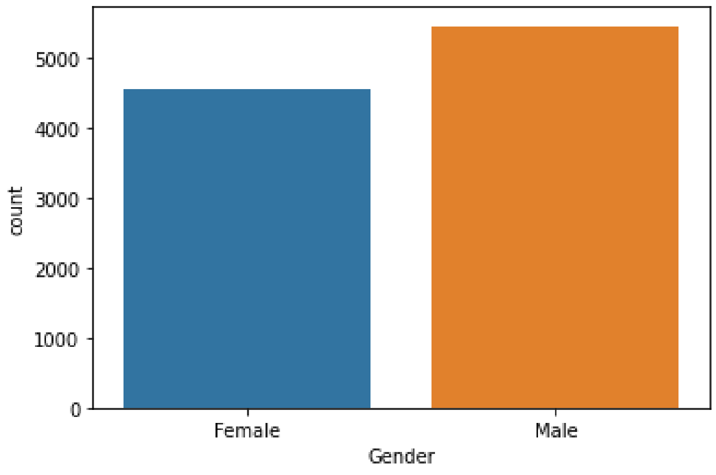
```
sns.countplot(df['Gender'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the
```

```
following variable as a keyword arg: x. From version 0.12, the only valid positional argu
ment will be `data`, and passing other arguments without an explicit keyword will result
in an error or misinterpretation.
  FutureWarning
```

Out[ ]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f938f985c50>
```



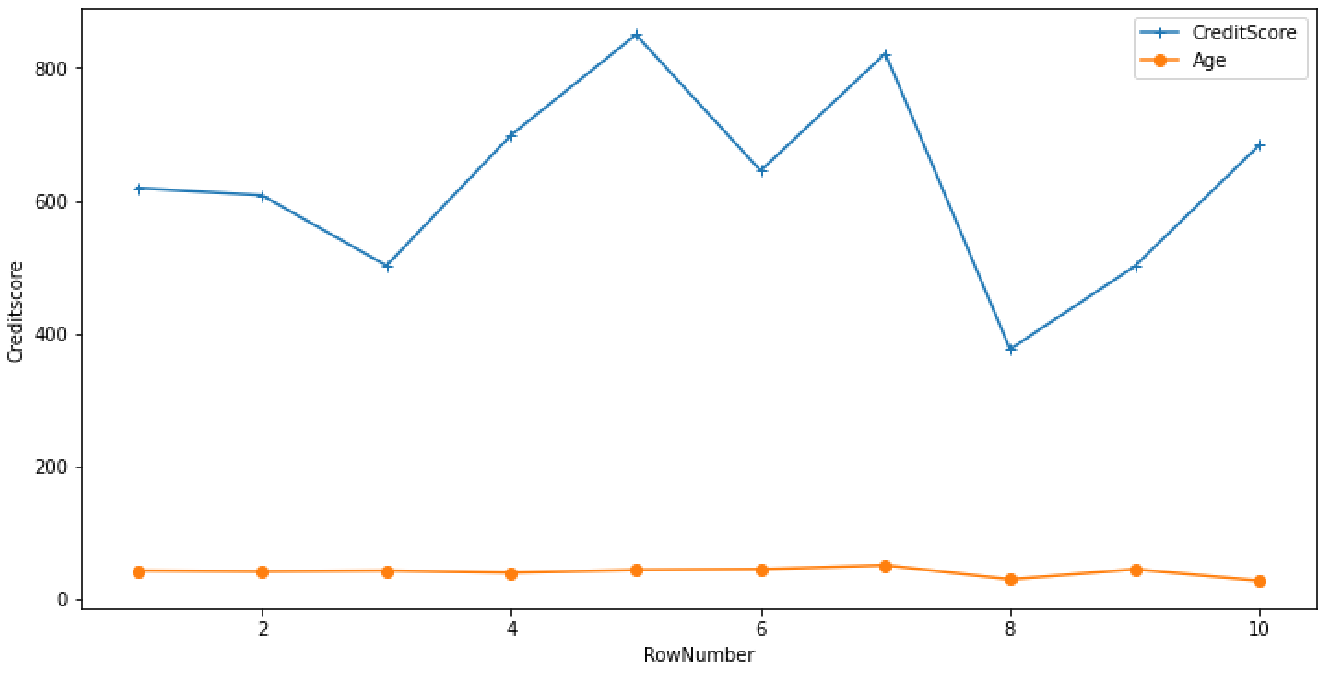
In [ ]:

```
x=df1['RowNumber']
y1=df1['CreditScore']
y2=df1['Age']

plt.figure(figsize=(12,6))
plt.plot(x,y1,label='CreditScore',marker="+")
plt.plot(x,y2,label='Age',marker='o')
plt.xlabel('RowNumber')
plt.ylabel('Creditscore')
plt.legend()
```

Out[ ]:

```
<matplotlib.legend.Legend at 0x7f9390730850>
```



Descriptive statistics

In [ ]:

```
df.describe(include='all')
```

Out[ ]:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance
count	10000.00000	1.000000e+04	10000	10000.000000	10000	10000	10000.000000	10000.000000	10000.000000
unique	NaN	NaN	2932	NaN	3	2	NaN	NaN	NaN
top	NaN	NaN	Smith	NaN	France	Male	NaN	NaN	NaN
freq	NaN	NaN	32	NaN	5014	5457	NaN	NaN	NaN
mean	5000.50000	1.569094e+07	NaN	650.528800	NaN	NaN	38.921800	5.012800	76485.889288
std	2886.89568	7.193619e+04	NaN	96.653299	NaN	NaN	10.487806	2.892174	62397.405202
min	1.00000	1.556570e+07	NaN	350.000000	NaN	NaN	18.000000	0.000000	0.000000
25%	2500.75000	1.562853e+07	NaN	584.000000	NaN	NaN	32.000000	3.000000	0.000000
50%	5000.50000	1.569074e+07	NaN	652.000000	NaN	NaN	37.000000	5.000000	97198.540000
75%	7500.25000	1.575323e+07	NaN	718.000000	NaN	NaN	44.000000	7.000000	127644.240000
max	10000.00000	1.581569e+07	NaN	850.000000	NaN	NaN	92.000000	10.000000	250898.090000

In [ ]:

```
df.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
```

Missing values

In [ ]:

```
df.isnull().sum()
```

Out[ ]:

```
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
```

Outlier and replacing

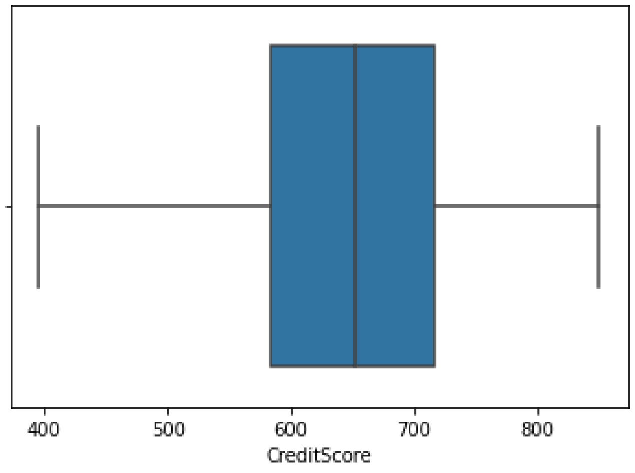
In [ ]:

```
sns.boxplot(df['CreditScore'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the
following variable as a keyword arg: x. From version 0.12, the only valid positional argu
ment will be `data`, and passing other arguments without an explicit keyword will result
in an error or misinterpretation.
  FutureWarning
```

Out[ ]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f939c2442d0>
```



In [ ]:

```
df[df['CreditScore']<390]=652
```

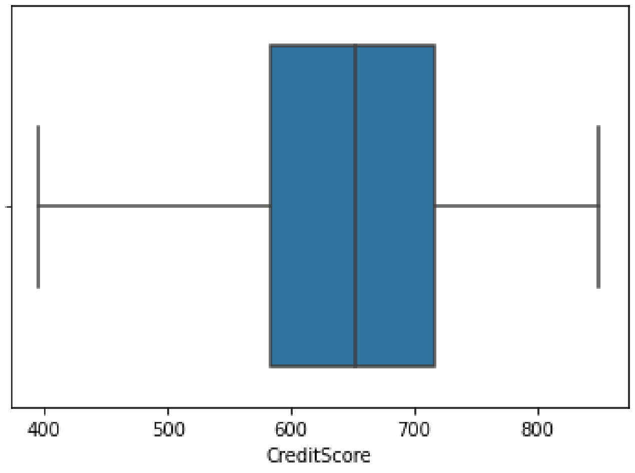
In [ ]:

```
sns.boxplot(df['CreditScore'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the
following variable as a keyword arg: x. From version 0.12, the only valid positional argu
ment will be `data`, and passing other arguments without an explicit keyword will result
in an error or misinterpretation.
  FutureWarning
```

Out[ ]:

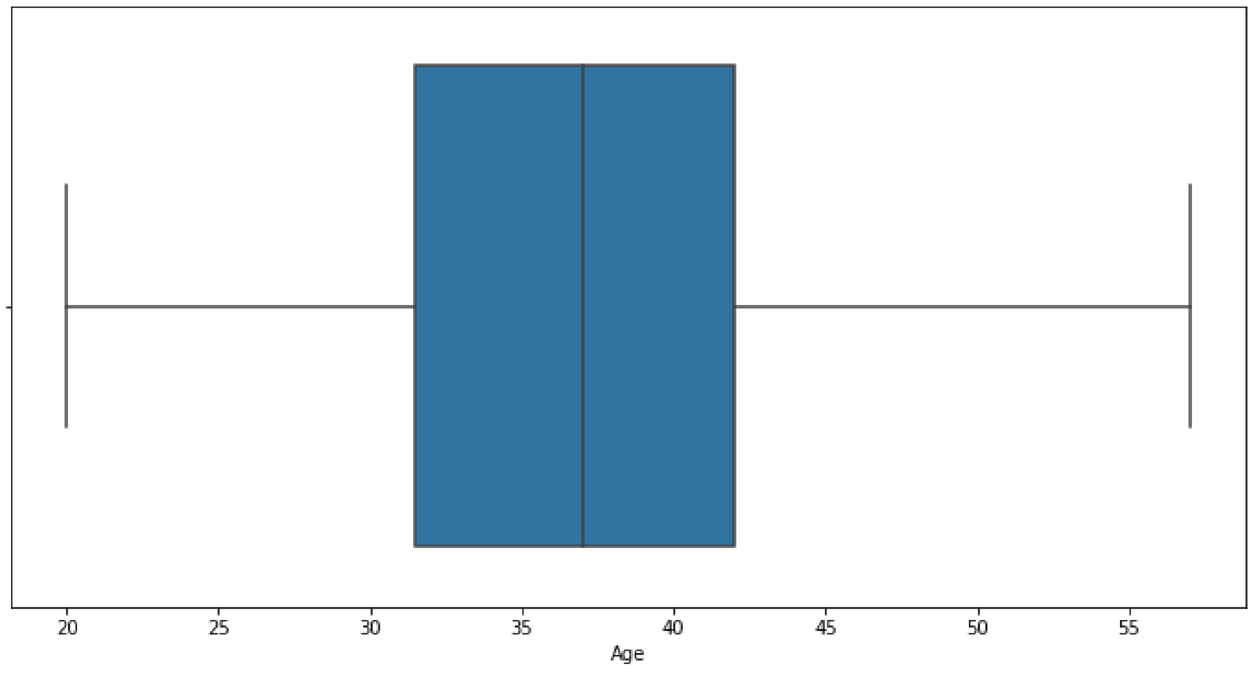
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f938ec40e10>
```



```
plt.figure(figsize=(12,6))
sns.boxplot(df['Age'])

plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the
following variable as a keyword arg: x. From version 0.12, the only valid positional argu
ment will be `data`, and passing other arguments without an explicit keyword will result
in an error or misinterpretation.
  FutureWarning
```



In [ ]:

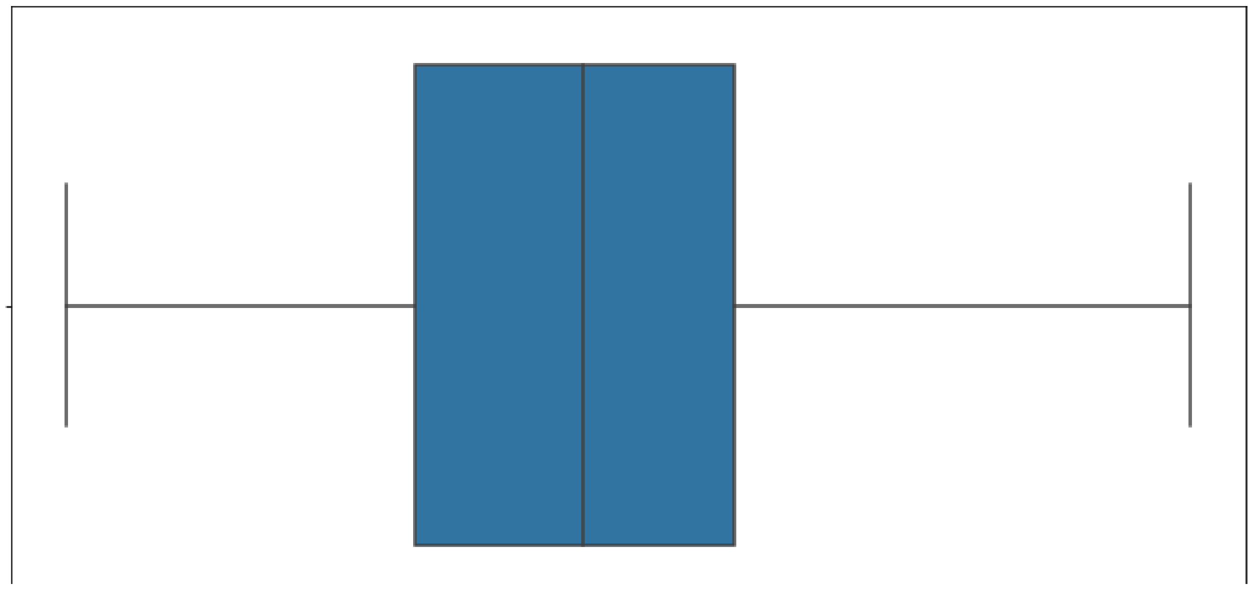
```
df[df['Age']>57]=37
df[df['Age']<20]=37
```

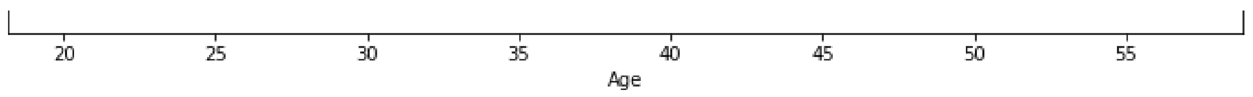
In [ ]:

```
plt.figure(figsize=(12,6))
sns.boxplot(df['Age'])

plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the
following variable as a keyword arg: x. From version 0.12, the only valid positional argu
ment will be `data`, and passing other arguments without an explicit keyword will result
in an error or misinterpretation.
  FutureWarning
```





In [ ]:

```
df=df.drop_duplicates()
df
```

Out[ ]:

	RowNumber	CustomerId	Surname	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMemb
0	1	15634602	Hargrave	619	42	2	0.00	1	1	
1	2	15647311	Hill	608	41	1	83807.86	1	0	
2	3	15619304	Onio	502	42	8	159660.80	3	1	
3	4	15701354	Boni	699	39	1	0.00	2	0	
4	5	15737888	Mitchell	850	43	2	125510.82	1	1	
...	...	...	...	...	...	...	...	...	...	
9274	9996	15606229	Obijiaku	771	39	5	0.00	2	1	
9275	9997	15569892	Johnstone	516	35	10	57369.61	1	1	
9276	9998	15584532	Liu	709	36	7	0.00	1	0	
9277	9999	15682355	Sabbatini	772	42	3	75075.31	2	1	
9278	10000	15628319	Walker	792	28	4	130142.79	1	1	

9279 rows × 17 columns

In [ ]:

```
df=df.reset_index()
df
```

Out[ ]:

	index	RowNumber	CustomerId	Surname	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActive
0	0	1	15634602	Hargrave	619	42	2	0.00	1	1	
1	1	2	15647311	Hill	608	41	1	83807.86	1	0	
2	2	3	15619304	Onio	502	42	8	159660.80	3	1	
3	3	4	15701354	Boni	699	39	1	0.00	2	0	
4	4	5	15737888	Mitchell	850	43	2	125510.82	1	1	
...	...	...	...	...	...	...	...	...	...	...	
9274	9274	9996	15606229	Obijiaku	771	39	5	0.00	2	1	
9275	9275	9997	15569892	Johnstone	516	35	10	57369.61	1	1	
9276	9276	9998	15584532	Liu	709	36	7	0.00	1	0	
9277	9277	9999	15682355	Sabbatini	772	42	3	75075.31	2	1	
9278	9278	10000	15628319	Walker	792	28	4	130142.79	1	1	

9279 rows × 18 columns

Categorical Column

In [ ]:

```
country = pd.get_dummies(df['Geography'])
```

```
country
```

Out[ ]:

	37	France	Germany	Spain
0	0	1	0	0
1	0	0	0	1
2	0	1	0	0
3	0	1	0	0
4	0	0	0	1
...	...	...	...	...
9276	0	1	0	0
9277	0	1	0	0
9278	0	1	0	0
9279	0	0	1	0
9280	0	1	0	0

9281 rows × 4 columns

In [ ]:

```
df=df.join(country)
df
```

Out[ ]:

	index	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts
0	0	1	15634602	Hargrave	619	France	Female	42	2	0.00	1
1	1	2	15647311	Hill	608	Spain	Female	41	1	83807.86	1
2	2	3	15619304	Onio	502	France	Female	42	8	159660.80	3
3	3	4	15701354	Boni	699	France	Female	39	1	0.00	2
4	4	5	15737888	Mitchell	850	Spain	Female	43	2	125510.82	1
...	...	...	...	...	...	...	...	...	...	...	...
9276	9995	9996	15606229	Obijiaku	771	France	Male	39	5	0.00	2
9277	9996	9997	15569892	Johnstone	516	France	Male	35	10	57369.61	1
9278	9997	9998	15584532	Liu	709	France	Female	36	7	0.00	1
9279	9998	9999	15682355	Sabbatini	772	Germany	Male	42	3	75075.31	2
9280	9999	10000	15628319	Walker	792	France	Female	28	4	130142.79	1

9281 rows × 19 columns

In [ ]:

```
df=df.drop('Geography',axis=1)
df
```

Out[ ]:

	index	RowNumber	CustomerId	Surname	CreditScore	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard
0	0	1	15634602	Hargrave	619	Female	42	2	0.00	1	1
1	1	2	15647311	Hill	608	Female	41	1	83807.86	1	0
2	2	3	15619304	Onio	502	Female	42	8	159660.80	3	1
3	3	4	15701354	Boni	699	Female	39	1	0.00	2	0

	index	RowNumber	CustomerId	Surname	CreditScore	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard
4	4	5	15737888	Mitchell	850	Female	43	2	125510.82	1	1
...	...	...	...	...	...	...	...	...	...	...	...
9276	9995	9996	15606229	Obijiaku	771	Male	39	5	0.00	2	1
9277	9996	9997	15569892	Johnstone	516	Male	35	10	57369.61	1	1
9278	9997	9998	15584532	Liu	709	Female	36	7	0.00	1	0
9279	9998	9999	15682355	Sabbatini	772	Male	42	3	75075.31	2	1
9280	9999	10000	15628319	Walker	792	Female	28	4	130142.79	1	1

9281 rows × 18 columns

In [ ]:

```
df=df.drop(37,axis=1)
df
```

Out[ ]:

	index	RowNumber	CustomerId	Surname	CreditScore	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard
0	0	1	15634602	Hargrave	619	Female	42	2	0.00	1	1
1	1	2	15647311	Hill	608	Female	41	1	83807.86	1	0
2	2	3	15619304	Onio	502	Female	42	8	159660.80	3	1
3	3	4	15701354	Boni	699	Female	39	1	0.00	2	0
4	4	5	15737888	Mitchell	850	Female	43	2	125510.82	1	1
...	...	...	...	...	...	...	...	...	...	...	...
9276	9995	9996	15606229	Obijiaku	771	Male	39	5	0.00	2	1
9277	9996	9997	15569892	Johnstone	516	Male	35	10	57369.61	1	1
9278	9997	9998	15584532	Liu	709	Female	36	7	0.00	1	0
9279	9998	9999	15682355	Sabbatini	772	Male	42	3	75075.31	2	1
9280	9999	10000	15628319	Walker	792	Female	28	4	130142.79	1	1

9281 rows × 17 columns

In [ ]:

```

```

In [ ]:

```
from sklearn.preprocessing import LabelEncoder
from collections import Counter as count
```

In [ ]:

```
df.iloc[7:8,:]
```

Out[ ]:

	index	RowNumber	CustomerId	Surname	CreditScore	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsAct
7	7	37	37	37	37	37	37	37	37.0	37	37	

In [ ]:

```
df=df.drop([7,8],axis=0)
df
```

Out[ ]:

	index	RowNumber	CustomerId	Surname	CreditScore	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard
0	0	1	15634602	Hargrave	619	Female	42	2	0.00	1	1
1	1	2	15647311	Hill	608	Female	41	1	83807.86	1	0
2	2	3	15619304	Onio	502	Female	42	8	159660.80	3	1
3	3	4	15701354	Boni	699	Female	39	1	0.00	2	0
4	4	5	15737888	Mitchell	850	Female	43	2	125510.82	1	1
...	...	...	...	...	...	...	...	...	...	...	...
9276	9995	9996	15606229	Obijiaku	771	Male	39	5	0.00	2	1
9277	9996	9997	15569892	Johnstone	516	Male	35	10	57369.61	1	1
9278	9997	9998	15584532	Liu	709	Female	36	7	0.00	1	0
9279	9998	9999	15682355	Sabbatini	772	Male	42	3	75075.31	2	1
9280	9999	10000	15628319	Walker	792	Female	28	4	130142.79	1	1

9279 rows × 17 columns

In [ ]:

```
df=df.reset_index()
df
```

Out[ ]:

	level_0	index	RowNumber	CustomerId	Surname	CreditScore	Gender	Age	Tenure	Balance	NumOfProducts	Ha
0	0	0	1	15634602	Hargrave	619	Female	42	2	0.00	1	
1	1	1	2	15647311	Hill	608	Female	41	1	83807.86	1	
2	2	2	3	15619304	Onio	502	Female	42	8	159660.80	3	
3	3	3	4	15701354	Boni	699	Female	39	1	0.00	2	
4	4	4	5	15737888	Mitchell	850	Female	43	2	125510.82	1	
...	...	...	...	...	...	...	...	...	...	...	...	
9274	9276	9995	9996	15606229	Obijiaku	771	Male	39	5	0.00	2	
9275	9277	9996	9997	15569892	Johnstone	516	Male	35	10	57369.61	1	
9276	9278	9997	9998	15584532	Liu	709	Female	36	7	0.00	1	
9277	9279	9998	9999	15682355	Sabbatini	772	Male	42	3	75075.31	2	
9278	9280	9999	10000	15628319	Walker	792	Female	28	4	130142.79	1	

9279 rows × 18 columns

In [ ]:

```
gender = pd.get_dummies(df['Gender'])
gender
```

Out[ ]:

	Female	Male
0	1	0
1	1	0
2	1	0
3	1	0
4	1	0

	Female	Male
...	...	...
9274	0	1
9275	0	1
9276	1	0
9277	0	1
9278	1	0

9279 rows × 2 columns

In [ ]:

```
df=df.join(gender)
df
```

Out[ ]:

	level_0	index	RowNumber	CustomerId	Surname	CreditScore	Gender	Age	Tenure	Balance	NumOfProducts	Ha
0	0	0	1	15634602	Hargrave	619	Female	42	2	0.00	1	
1	1	1	2	15647311	Hill	608	Female	41	1	83807.86	1	
2	2	2	3	15619304	Onio	502	Female	42	8	159660.80	3	
3	3	3	4	15701354	Boni	699	Female	39	1	0.00	2	
4	4	4	5	15737888	Mitchell	850	Female	43	2	125510.82	1	
...	...	...	...	...	...	...	...	...	...	...	...	
9274	9276	9995	9996	15606229	Obijiaku	771	Male	39	5	0.00	2	
9275	9277	9996	9997	15569892	Johnstone	516	Male	35	10	57369.61	1	
9276	9278	9997	9998	15584532	Liu	709	Female	36	7	0.00	1	
9277	9279	9998	9999	15682355	Sabbatini	772	Male	42	3	75075.31	2	
9278	9280	9999	10000	15628319	Walker	792	Female	28	4	130142.79	1	

9279 rows × 20 columns

In [ ]:

```
df=df.drop('Gender',axis=1)
df
```

Out[ ]:

	level_0	index	RowNumber	CustomerId	Surname	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard
0	0	0	1	15634602	Hargrave	619	42	2	0.00	1	1
1	1	1	2	15647311	Hill	608	41	1	83807.86	1	0
2	2	2	3	15619304	Onio	502	42	8	159660.80	3	1
3	3	3	4	15701354	Boni	699	39	1	0.00	2	0
4	4	4	5	15737888	Mitchell	850	43	2	125510.82	1	1
...	...	...	...	...	...	...	...	...	...	...	...
9274	9276	9995	9996	15606229	Obijiaku	771	39	5	0.00	2	1
9275	9277	9996	9997	15569892	Johnstone	516	35	10	57369.61	1	1
9276	9278	9997	9998	15584532	Liu	709	36	7	0.00	1	0
9277	9279	9998	9999	15682355	Sabbatini	772	42	3	75075.31	2	1
9278	9280	9999	10000	15628319	Walker	792	28	4	130142.79	1	1

9279 rows × 19 columns

In [ ]:

```
df=df.drop('index',axis=1)
df
```

Out[ ]:

	level_0	RowNumber	CustomerId	Surname	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActi
0	0	1	15634602	Hargrave	619	42	2	0.00	1	1	
1	1	2	15647311	Hill	608	41	1	83807.86	1	0	
2	2	3	15619304	Onio	502	42	8	159660.80	3	1	
3	3	4	15701354	Boni	699	39	1	0.00	2	0	
4	4	5	15737888	Mitchell	850	43	2	125510.82	1	1	
...	...	...	...	...	...	...	...	...	...	...	
9274	9276	9996	15606229	Obijiaku	771	39	5	0.00	2	1	
9275	9277	9997	15569892	Johnstone	516	35	10	57369.61	1	1	
9276	9278	9998	15584532	Liu	709	36	7	0.00	1	0	
9277	9279	9999	15682355	Sabbatini	772	42	3	75075.31	2	1	
9278	9280	10000	15628319	Walker	792	28	4	130142.79	1	1	

9279 rows × 18 columns

In [ ]:

```
df=df.drop('level_0',axis=1)
df
```

Out[ ]:

	RowNumber	CustomerId	Surname	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMemb
0	1	15634602	Hargrave	619	42	2	0.00	1	1	
1	2	15647311	Hill	608	41	1	83807.86	1	0	
2	3	15619304	Onio	502	42	8	159660.80	3	1	
3	4	15701354	Boni	699	39	1	0.00	2	0	
4	5	15737888	Mitchell	850	43	2	125510.82	1	1	
...	...	...	...	...	...	...	...	...	...	
9274	9996	15606229	Obijiaku	771	39	5	0.00	2	1	
9275	9997	15569892	Johnstone	516	35	10	57369.61	1	1	
9276	9998	15584532	Liu	709	36	7	0.00	1	0	
9277	9999	15682355	Sabbatini	772	42	3	75075.31	2	1	
9278	10000	15628319	Walker	792	28	4	130142.79	1	1	

9279 rows × 17 columns

Dependent and independent variable

In [ ]:

```
x1=df.iloc[:,0:11]
x1
```

Out[ ]:

	RowNumber	CustomerId	Surname	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMemb
0	1	15634602	Hargrave	619	42	2	0.00	1	1	
1	2	15647311	Hill	608	41	1	83807.86	1	0	
2	3	15619304	Onio	502	42	8	159660.80	3	1	
3	4	15701354	Boni	699	39	1	0.00	2	0	
4	5	15737888	Mitchell	850	43	2	125510.82	1	1	
...	...	...	...	...	...	...	...	...	...	
9274	9996	15606229	Obijiaku	771	39	5	0.00	2	1	
9275	9997	15569892	Johnstone	516	35	10	57369.61	1	1	
9276	9998	15584532	Liu	709	36	7	0.00	1	0	
9277	9999	15682355	Sabbatini	772	42	3	75075.31	2	1	
9278	10000	15628319	Walker	792	28	4	130142.79	1	1	

9279 rows × 11 columns

In [ ]:

```
x2=df.iloc[:,12:17]
x2
```

Out[ ]:

	France	Germany	Spain	Female	Male
0	1	0	0	1	0
1	0	0	1	1	0
2	1	0	0	1	0
3	1	0	0	1	0
4	0	0	1	1	0
...	...	...	...	...	...
9274	1	0	0	0	1
9275	1	0	0	0	1
9276	1	0	0	1	0
9277	0	1	0	0	1
9278	1	0	0	1	0

9279 rows × 5 columns

In [ ]:

```
x1=x1.join(x2)
x1
```

Out[ ]:

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

...	RowNumber	CustomerId	Surname	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMemb
9274	9996	15606229	Obijiaku	771	39	5	0.00	2	1	
9275	9997	15569892	Johnstone	516	35	10	57369.61	1	1	
9276	9998	15584532	Liu	709	36	7	0.00	1	0	
9277	9999	15682355	Sabbatini	772	42	3	75075.31	2	1	
9278	10000	15628319	Walker	792	28	4	130142.79	1	1	

9279 rows × 16 columns

In [ ]:

```
x1=x1.drop('Surname',axis=1)
x1
```

Out[ ]:

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	Estimate
0	1	15634602	619	42	2	0.00	1	1	1	10
1	2	15647311	608	41	1	83807.86	1	0	1	11
2	3	15619304	502	42	8	159660.80	3	1	0	11
3	4	15701354	699	39	1	0.00	2	0	0	9
4	5	15737888	850	43	2	125510.82	1	1	1	7
...	...	...	...	...	...	...	...	...	...	
9274	9996	15606229	771	39	5	0.00	2	1	0	9
9275	9997	15569892	516	35	10	57369.61	1	1	1	10
9276	9998	15584532	709	36	7	0.00	1	0	1	4
9277	9999	15682355	772	42	3	75075.31	2	1	0	9
9278	10000	15628319	792	28	4	130142.79	1	1	0	3

9279 rows × 15 columns

In [ ]:

```
y=df.iloc[:,11:12]
y
```

Out[ ]:

	Exited
0	1
1	0
2	1
3	0
4	0
...	...
9274	0
9275	0
9276	1
9277	1
9278	0

9279 rows × 1 columns

Training and testing

In [ ]:

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

In [ ]:

```
x_train, x_test, y_train, y_test = train_test_split(x1, y, test_size=0.33, random_state=
1)
```

In [ ]:

```
x_train
```

Out[ ]:

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	Estimate
5336	5769	15729083	674	36	2	154525.70	1	0	1	2
2897	3110	15735878	850	47	10	134381.52	1	0	0	2
7110	7648	15674583	768	25	0	78396.08	1	1	1	
188	201	15604482	850	30	2	141040.01	1	1	1	
8549	9204	15774401	773	51	4	0.00	2	0	0	12
...	...	...	...	...	...	...	...	...	...	
2895	3108	15697424	597	30	2	119370.11	1	1	1	18
7813	8408	15675626	726	28	2	0.00	1	0	0	9
905	979	15799515	652	48	8	133297.24	1	1	0	7
5192	5612	15721207	625	42	6	100047.33	1	1	0	9
235	251	15628112	771	36	5	77846.90	1	0	0	9

6216 rows × 15 columns

In [ ]:

```
x_test
```

Out[ ]:

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	Estimate
5430	5870	15734461	562	31	2	112708.20	1	0	1	18
2495	2679	15767793	819	38	10	0.00	2	1	0	3
4816	5211	15738954	551	35	7	129717.30	2	0	0	8
6588	7088	15615832	675	35	8	155621.08	1	0	1	3
2517	2702	15797010	649	31	2	0.00	2	1	0	1
...	...	...	...	...	...	...	...	...	...	
4789	5182	15711287	661	35	5	128415.45	1	1	0	14
5064	5474	15596863	787	38	3	158373.23	1	1	1	2
2959	3176	15764604	586	35	7	164769.02	3	1	0	11
2537	2724	15681550	614	41	8	121558.46	1	1	1	
166	178	15790355	606	36	5	190479.48	2	0	0	17

3063 rows × 15 columns

In [ ]:

```

```

Scaling

In [ ]:

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

In [ ]:

```
nm = MinMaxScaler()
```

In [ ]:

```
s_xtrain=nm.fit_transform(x_train)
```

In [ ]:

```
s_xtrain
```

Out[ ]:

```
array([[0.57685769, 0.65355676, 0.6097561 , ..., 0.        , 0.        ,
        1.        ],
       [0.31093109, 0.68073795, 1.        , ..., 0.        , 1.        ,
        0.        ],
       [0.76477648, 0.43554716, 0.81818182, ..., 0.        , 0.        ,
        1.        ],
       ...,
       [0.09780978, 0.93529715, 0.56097561, ..., 0.        , 1.        ,
        0.        ],
       [0.56115612, 0.62205137, 0.50110865, ..., 0.        , 0.        ,
        1.        ],
       [0.0250025 , 0.24965498, 0.8248337 , ..., 0.        , 1.        ,
        0.        ]])
```

In [ ]:

```
s_xtest=nm.transform(x_test)
```

In [ ]:

```
s_xtest
```

Out[ ]:

```
array([[0.5869587 , 0.6750697 , 0.36141907, ..., 0.        , 0.        ,
        1.        ],
       [0.26782678, 0.80840357, 0.93126386, ..., 0.        , 1.        ,
        0.        ],
       [0.52105211, 0.69304249, 0.33702882, ..., 0.        , 0.        ,
        1.        ],
       ...,
       [0.31753175, 0.79564701, 0.41463415, ..., 0.        , 1.        ,
        0.        ],
       [0.27232723, 0.46341639, 0.4767184 , ..., 0.        , 1.        ,
        0.        ],
       [0.01770177, 0.89865554, 0.45898004, ..., 0.        , 0.        ,
        1.        ]])
```