1.Download the dataset 2.Load the dataset

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

```
data=pd.read_csv("Churn_Modelling.csv",encoding='ISO-8859-1')
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

Saved successfully!

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balar
count	10000.00000	1.000000e+04	10000.000000	10000.000000	10000.000000	10000.000
mean	5000.50000	1.569094e+07	650.528800	38.921800	5.012800	76485.889
std	2886.89568	7.193619e+04	96.653299	10.487806	2.892174	62397.405
min	1.00000	1.556570e+07	350.000000	18.000000	0.000000	0.000
25%	2500.75000	1.562853e+07	584.000000	32.000000	3.000000	0.000
50%	5000.50000	1.569074e+07	652.000000	37.000000	5.000000	97198.540
75%	7500.25000	1.575323e+07	718.000000	44.000000	7.000000	127644.240
max	10000.00000	1.581569e+07	850.000000	92.000000	10.000000	250898.090

```
data.dtypes
```

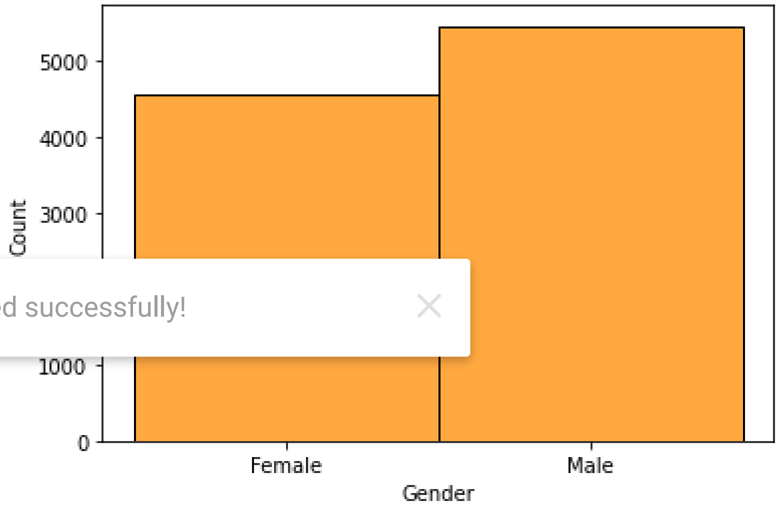
```
RowNumber           int64
CustomerId          int64
Surname            object
CreditScore         int64
```

```
Geography          object
Gender             object
Age                 int64
Tenure              int64
Balance           float64
NumOfProducts       int64
HasCrCard           int64
IsActiveMember      int64
EstimatedSalary   float64
Exited              int64
dtype: object
```

3.Perform Below Visualizations Univariate Analysis ,Bi - Variate Analysis,Multi - Variate Analysis **

```
sns.histplot(data["Gender"],color='darkorange')
```

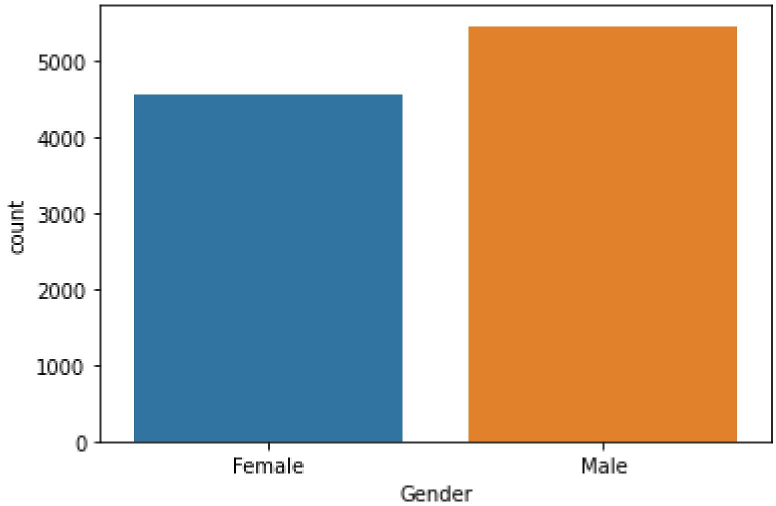
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f3f675c9c50>
```



Saved successfully!

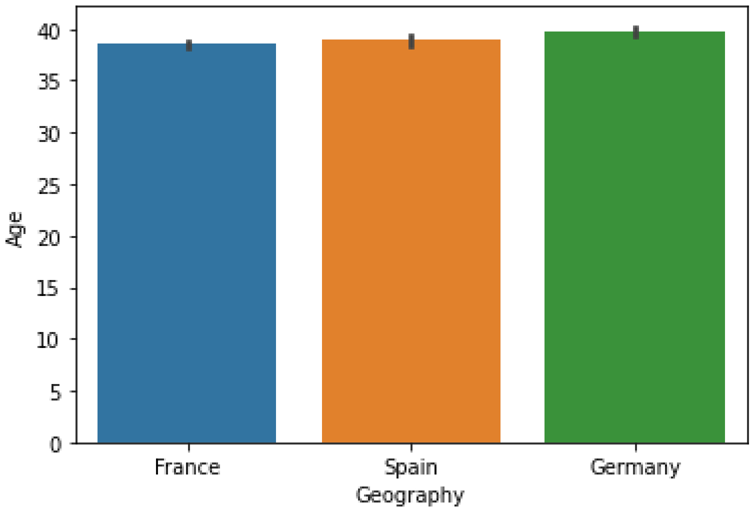
```
sns.countplot(data['Gender'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pas
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f3f6752bf50>
```



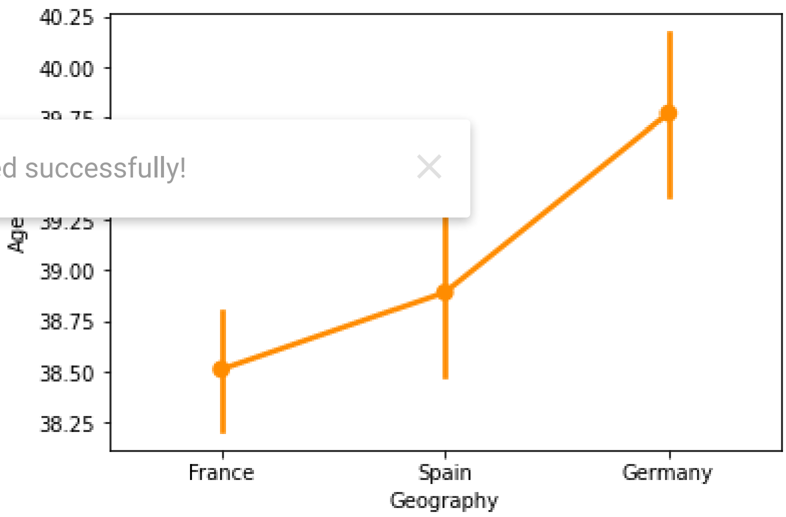
```
sns.barplot(x='Geography',y='Age',data=data)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f3f66ff5310>
```




```
sns.pointplot(x='Geography',y='Age',data=data,color='darkorange')
```

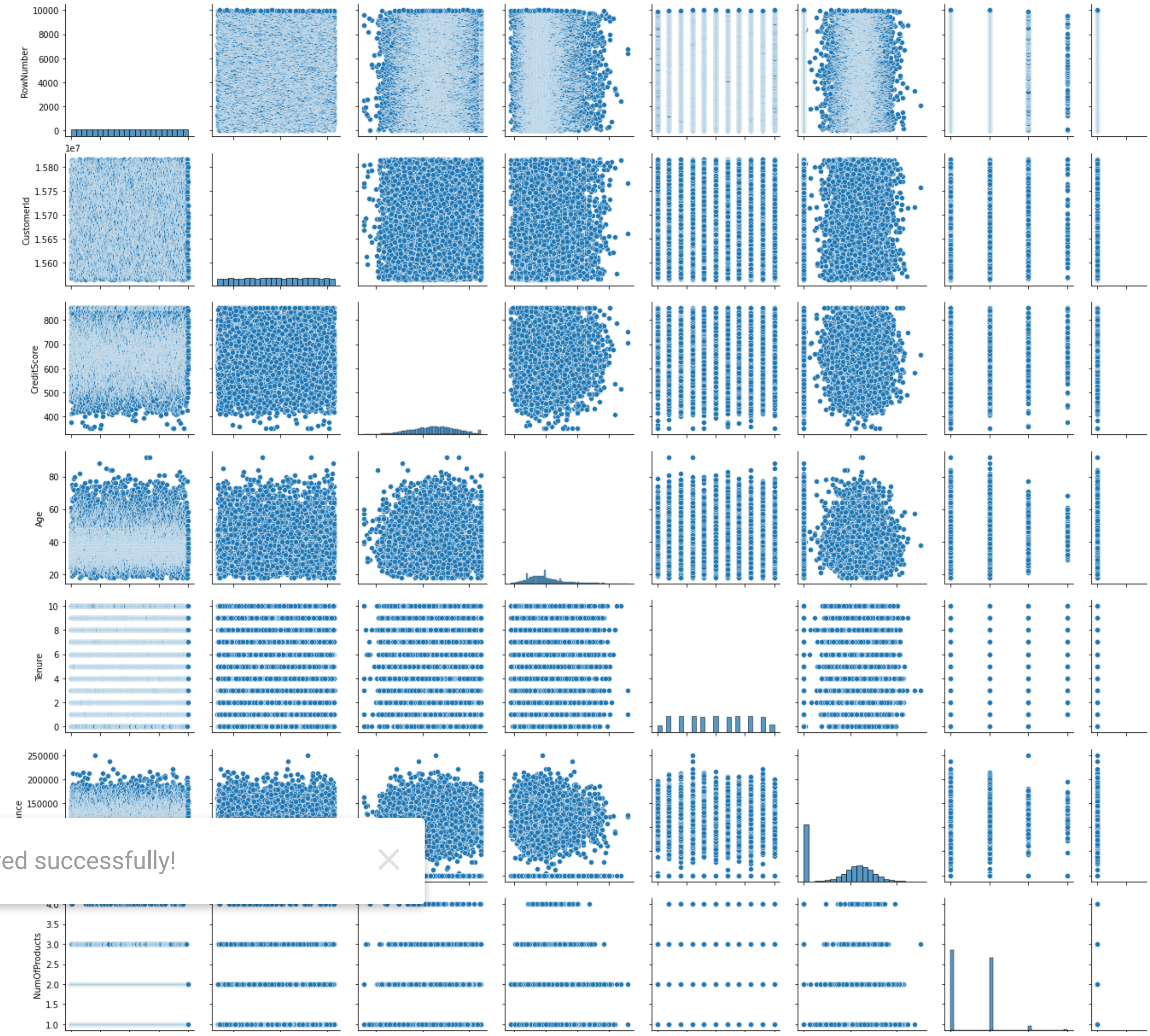
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f3f66f68810>
```



Saved successfully!

```
sns.pairplot(data)
```

<seaborn.axisgrid.PairGrid at 0x7f3f66ee65d0>



Saved successfully!

4. Perform descriptive statistics on the dataset

```
data.describe().T
```

	count	mean	std	min	25%	
RowNumber	10000.0	5.000500e+03	2886.895680	1.00	2500.75	5.00050(
CustomerId	10000.0	1.569094e+07	71936.186123	15565701.00	15628528.25	1.56907‹
CreditScore	10000.0	6.505288e+02	96.653299	350.00	584.00	6.52000(
EstimatedSalary	10000.0	1.000902e+05	57510.492818	11.58	51002.11	1.00193‹

5.Handle the Missing values.

```
data.isnull().sum().sum()
```

```
0
```

There is no missing values in this dataset

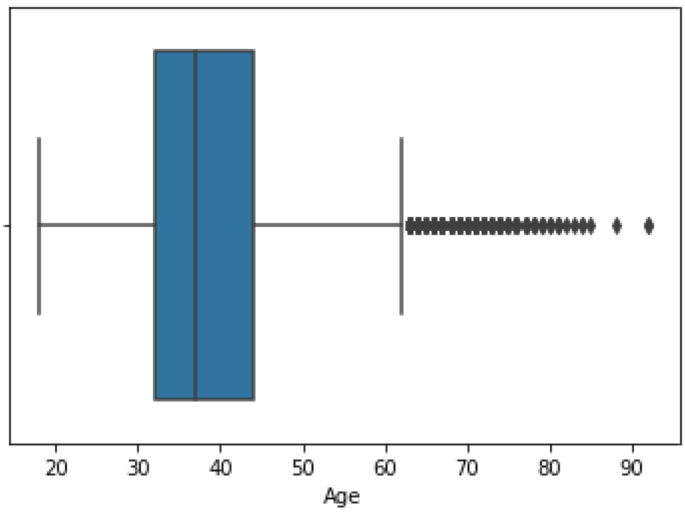
```
missing_values=data.isnull().sum()
missing_values[missing_values>0]/len(data)*100
```

```
Series([], dtype: float64)
```

6.Find the outliers and replace the outliers

Saved successfully!

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pas
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f3f617ca550>
```



7.Check for Categorical columns and perform encoding.

```
print(data['Gender'].unique())
print(data['Age'].unique())
```

```
['Female' 'Male']
```

```
[42 41 39 43 44 50 29 27 31 24 34 25 35 45 58 32 38 46 36 33 40 51 61 49
 37 19 66 56 26 21 55 75 22 30 28 65 48 52 57 73 47 54 72 20 67 79 62 53
 80 59 68 23 60 70 63 64 18 82 69 74 71 76 77 88 85 84 78 81 92 83]
```

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

```
one_hot_encoded_data = pd.get_dummies(data, columns = ['Age', 'Gender'])
print(one_hot_encoded_data)
```

Saved successfully!

```
      RowNumber  CustomerId    Surname  CreditScore Geography  Tenure  \
0             1    15634602   Hargrave          619    France       2
1             2    15647311       Hill          608     Spain       1
                                  Onio          502    France       8
                                  Boni          699    France       1
                              itchell           850     Spain       2
...         ...         ...        ...          ...       ...     ...
9995       9996    15606229   Obijiaku          771    France       5
9996       9997    15569892  Johnstone          516    France      10
9997       9998    15584532        Liu          709    France       7
9998       9999    15682355  Sabbatini          772   Germany       3
9999      10000    15628319     Walker          792    France       4

        Balance  NumOfProducts  HasCrCard  IsActiveMember  ...  Age_80  \
0          0.00              1          1               1  ...       0
1      83807.86              1          0               1  ...       0
2     159660.80              3          1               0  ...       0
3          0.00              2          0               0  ...       0
4     125510.82              1          1               1  ...       0
...         ...            ...        ...             ...  ...     ...
9995       0.00              2          1               0  ...       0
9996   57369.61              1          1               1  ...       0
9997       0.00              1          0               1  ...       0
9998   75075.31              2          1               0  ...       0
9999  130142.79              1          1               0  ...       0

      Age_81  Age_82  Age_83  Age_84  Age_85  Age_88  Age_92  Gender_Female  \
0          0       0       0       0       0       0       0              1
1          0       0       0       0       0       0       0              1
2          0       0       0       0       0       0       0              1
3          0       0       0       0       0       0       0              1
4          0       0       0       0       0       0       0              1
...      ...     ...     ...     ...     ...     ...     ...            ...
9995       0       0       0       0       0       0       0              0
9996       0       0       0       0       0       0       0              0
```

```
9997      0      0      0      0      0      0      0           1
9998      0      0      0      0      0      0      0           0
9999      0      0      0      0      0      0      0           1

      Gender_Male
0               0
1               0
2               0
3               0
4               0
...           ...
9995            1
9996            1
9997            0
9998            1
9999            0

[10000 rows x 84 columns]
```

8.Split the data into dependent and independent variables.

```
from sklearn.datasets import load_iris
```

```
from sklearn import preprocessing
data = load_iris()
```

Saved successfully!

```
print("Dependent variable")
print(X_data)
print("Independent variable")
print(target)
```

```
Dependent variable
[[5.1 3.5 1.4 0.2]
 [4.9 3.  1.4 0.2]
 [4.7 3.2 1.3 0.2]
 [4.6 3.1 1.5 0.2]
 [5.  3.6 1.4 0.2]
 [5.4 3.9 1.7 0.4]
 [4.6 3.4 1.4 0.3]
 [5.  3.4 1.5 0.2]
 [4.4 2.9 1.4 0.2]
 [4.9 3.1 1.5 0.1]
 [5.4 3.7 1.5 0.2]
 [4.8 3.4 1.6 0.2]
 [4.8 3.  1.4 0.1]
 [4.3 3.  1.1 0.1]
 [5.8 4.  1.2 0.2]
 [5.7 4.4 1.5 0.4]
 [5.4 3.9 1.3 0.4]
 [5.1 3.5 1.4 0.3]
 [5.7 3.8 1.7 0.3]
 [5.1 3.8 1.5 0.3]
 [5.4 3.4 1.7 0.2]
 [5.1 3.7 1.5 0.4]
```

```
[4.6 3.6 1.  0.2]
[5.1 3.3 1.7 0.5]
[4.8 3.4 1.9 0.2]
[5.  3.  1.6 0.2]
[5.  3.4 1.6 0.4]
[5.2 3.5 1.5 0.2]
[5.2 3.4 1.4 0.2]
[4.7 3.2 1.6 0.2]
[4.8 3.1 1.6 0.2]
[5.4 3.4 1.5 0.4]
[5.2 4.1 1.5 0.1]
[5.5 4.2 1.4 0.2]
[4.9 3.1 1.5 0.2]
[5.  3.2 1.2 0.2]
[5.5 3.5 1.3 0.2]
[4.9 3.6 1.4 0.1]
[4.4 3.  1.3 0.2]
[5.1 3.4 1.5 0.2]
[5.  3.5 1.3 0.3]
[4.5 2.3 1.3 0.3]
[4.4 3.2 1.3 0.2]
[5.  3.5 1.6 0.6]
[5.1 3.8 1.9 0.4]
[4.8 3.  1.4 0.3]
[5.1 3.8 1.6 0.2]
[4.6 3.2 1.4 0.2]
[5.3 3.7 1.5 0.2]
[5.  3.3 1.4 0.2]
```

Saved successfully!

```
[5.5 2.3 4.  1.3]
[6.5 2.8 4.6 1.5]
[5.7 2.8 4.5 1.3]
[6.3 3.3 4.7 1.6]
```

9.Scale the independent variable

```
standard = preprocessing.scale(target)
print(standard)
```

```
[-1.22474487 -1.22474487 -1.22474487 -1.22474487 -1.22474487 -1.22474487
 -1.22474487 -1.22474487 -1.22474487 -1.22474487 -1.22474487 -1.22474487
 -1.22474487 -1.22474487 -1.22474487 -1.22474487 -1.22474487 -1.22474487
 -1.22474487 -1.22474487 -1.22474487 -1.22474487 -1.22474487 -1.22474487
 -1.22474487 -1.22474487 -1.22474487 -1.22474487 -1.22474487 -1.22474487
 -1.22474487 -1.22474487 -1.22474487 -1.22474487 -1.22474487 -1.22474487
 -1.22474487 -1.22474487 -1.22474487 -1.22474487 -1.22474487 -1.22474487
 -1.22474487 -1.22474487 -1.22474487 -1.22474487 -1.22474487 -1.22474487
 -1.22474487 -1.22474487  0.          0.          0.          0.
  0.          0.          0.          0.          0.          0.
  0.          0.          0.          0.          0.          0.
  0.          0.          0.          0.          0.          0.
  0.          0.          0.          0.          0.          0.
  0.          0.          0.          0.          0.          0.
  0.          0.          0.          0.          0.          0.
  0.          0.          0.          0.          0.          0.
  0.          0.          0.          0.          1.22474487  1.22474487
```

```
 1.22474487  1.22474487  1.22474487  1.22474487  1.22474487  1.22474487
 1.22474487  1.22474487  1.22474487  1.22474487  1.22474487  1.22474487
 1.22474487  1.22474487  1.22474487  1.22474487  1.22474487  1.22474487
 1.22474487  1.22474487  1.22474487  1.22474487  1.22474487  1.22474487
 1.22474487  1.22474487  1.22474487  1.22474487  1.22474487  1.22474487
 1.22474487  1.22474487  1.22474487  1.22474487  1.22474487  1.22474487
 1.22474487  1.22474487  1.22474487  1.22474487  1.22474487  1.22474487
 1.22474487  1.22474487  1.22474487  1.22474487  1.22474487  1.22474487]
```

10.Split the data into training and testing

```
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split


df = pd.read_csv('Churn_Modelling.csv')


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


X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.05, random_state=0)
```

Saved successfully!

```
st)
```

```
      RowNumber  CustomerId         Surname  CreditScore Geography  Gender  \
799         800    15567367             Tao          601   Germany  Female
1069       1070    15628674         Iadanza          844    France    Male
8410       8411    15609913           Clark          743    France  Female
9436       9437    15771000          Powell          684    France    Male
5099       5100    15731555       Ross-Watt          595   Germany  Female
...         ...         ...             ...          ...       ...     ...
9225       9226    15584928  Ugochukwutubelum        594   Germany  Female
4859       4860    15647111           White          794     Spain  Female
3264       3265    15574372          Hoolan          738    France    Male
9845       9846    15664035         Parsons          590     Spain  Female
2732       2733    15592816        Udokamma          623   Germany  Female

      Age  Tenure    Balance  NumOfProducts  HasCrCard  IsActiveMember  \
799    42       9  133636.16              1          0               1
1069   40       7  113348.14              1          1               0
8410   46       9       0.00              1          1               0
9436   38       4       0.00              3          1               0
5099   45       9  106000.12              1          0               0
...   ...     ...        ...            ...        ...             ...
9225   32       4  120074.97              2          1               1
4859   22       4  114440.24              1          1               1
3264   35       5  161274.05              2          1               0
9845   38       9       0.00              2          1               1
2732   48       1  108076.33              1          1               0

      EstimatedSalary
799         103315.74
```

```
1069      31904.31
8410     113436.08
9436      75609.84
5099     191448.96
...            ...
9225     162961.79
4859     107753.07
3264     181429.87
9845     148750.16
2732     118855.26

[9500 rows x 13 columns]       RowNumber  CustomerId    Surname  CreditScore Geog
9394       9395    15615753   Upchurch          597   Germany  Female   35
898         899    15654700    Fallaci          523    France  Female   40
2398       2399    15633877   Morrison          706     Spain  Female   42
5906       5907    15745623    Worsnop          788    France    Male   32
2343       2344    15765902     Gibson          706   Germany    Male   38
...         ...         ...        ...          ...       ...     ...  ...
8938       8939    15722409    Ritchie          693     Spain    Male   47
9291       9292    15679804   Esquivel          636    France    Male   36
491         492    15699005     Martin          710    France  Female   41
2021       2022    15795519   Vasiliev          716   Germany  Female   18
4299       4300    15711991  Chiawuotu          615    France    Male   30

      Tenure    Balance  NumOfProducts  HasCrCard  IsActiveMember  \
9394       8  131101.04              1          1               1
898        2  102967.41              1          1               0
2398       8   95386.82              1          1               1
```

Saved successfully!

Colab paid products - Cancel contracts here