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

Read the Dataset

```
data = pd.read_csv("/content/Churn_Modelling.csv")
```

```
data
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure
0	1	15634602	Hargrave	619	France	Female	42.0	2
1	2	15647311	Hill	608	Spain	Female	41.0	1
2	3	15619304	Onio	502	France	Female	42.0	8
3	4	15701354	Boni	699	France	Female	39.0	1
4	5	15737888	Mitchell	850	Spain	Female	43.0	2
...	...	...	...	...	...	...	...	...
9995	9996	15606229	Obijiaku	771	France	Male	39.0	5
9996	9997	15569892	Johnstone	516	France	Male	35.0	10
9997	9998	15584532	Liu	709	France	Female	36.0	7
9998	9999	15682355	Sabbatini	772	Germany	Male	42.0	3
9999	10000	15628319	Walker	792	France	Female	28.0	4

Saved successfully!

```
data.dtypes
```

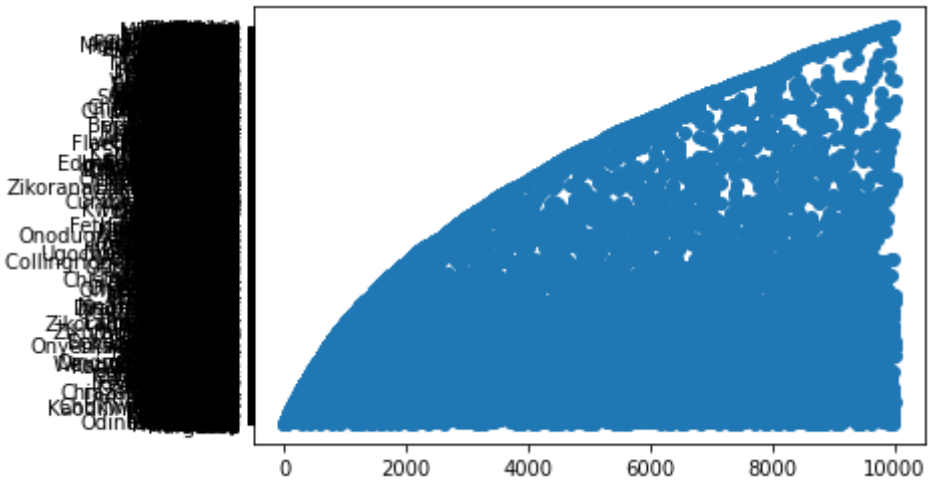
```
RowNumber            int64
CustomerId           int64
Surname             object
CreditScore          int64
Geography           object
Gender              object
Age                float64
Tenure               int64
Balance            float64
NumOfProducts        int64
HasCrCard            int64
IsActiveMember       int64
```

```
EstimatedSalary    float64
Exited               int64
dtype: object
```

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

Univariate Analysis

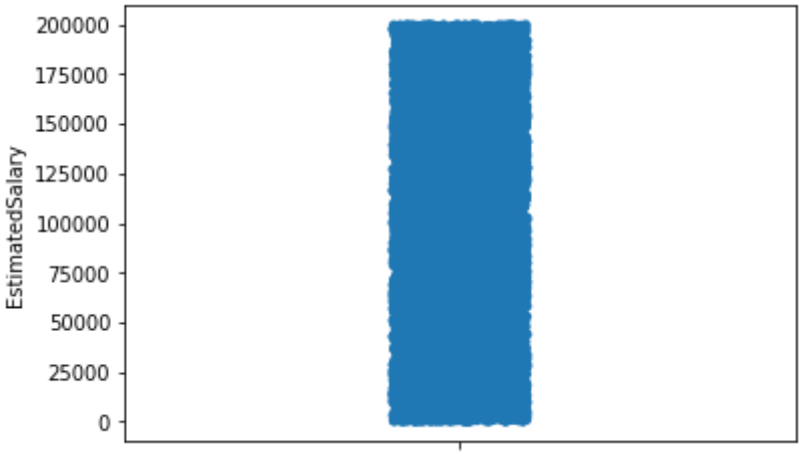
```
plt.scatter(data.index,data['Surname'])
plt.show()
```




```
sns.stripplot(y=data['EstimatedSalary'])
```

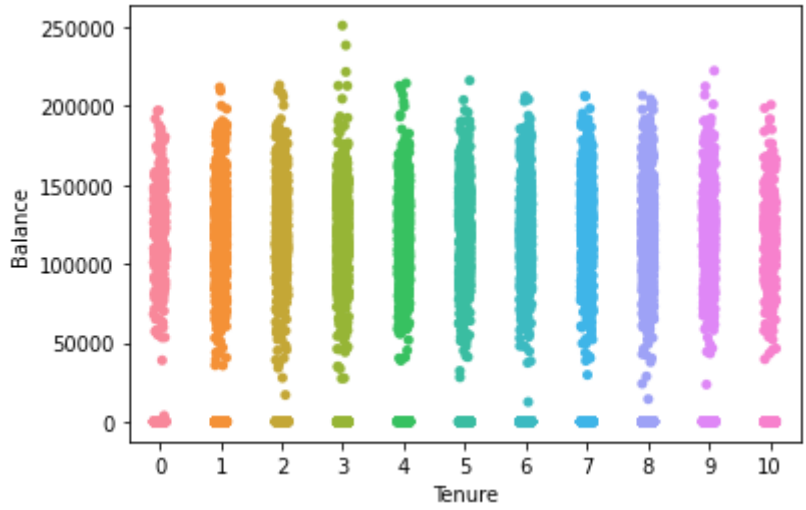
Saved successfully!

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5a2af88250>
```




```
sns.stripplot(x=data['Tenure'],y=data['Balance'])
```

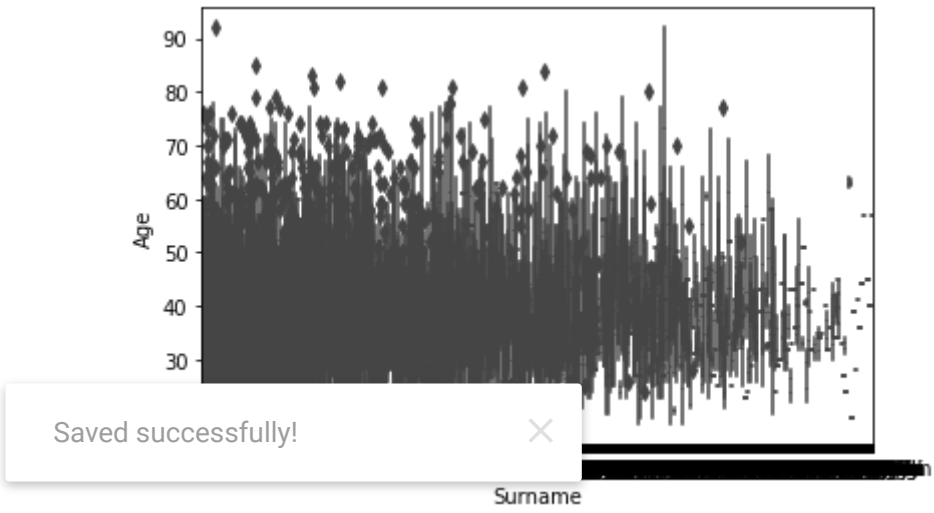
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5a2ae810d0>
```



Double-click (or enter) to edit

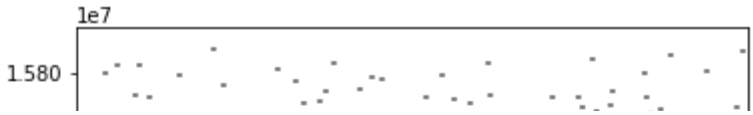
Bivariate Analysis

```
sns.boxplot(x='Surname',y='Age',data=data)
plt.show()
```



Saved successfully!

```
sns.violinplot(x='RowNumber',y='CustomerId',data=data,size=4)
plt.show()
```

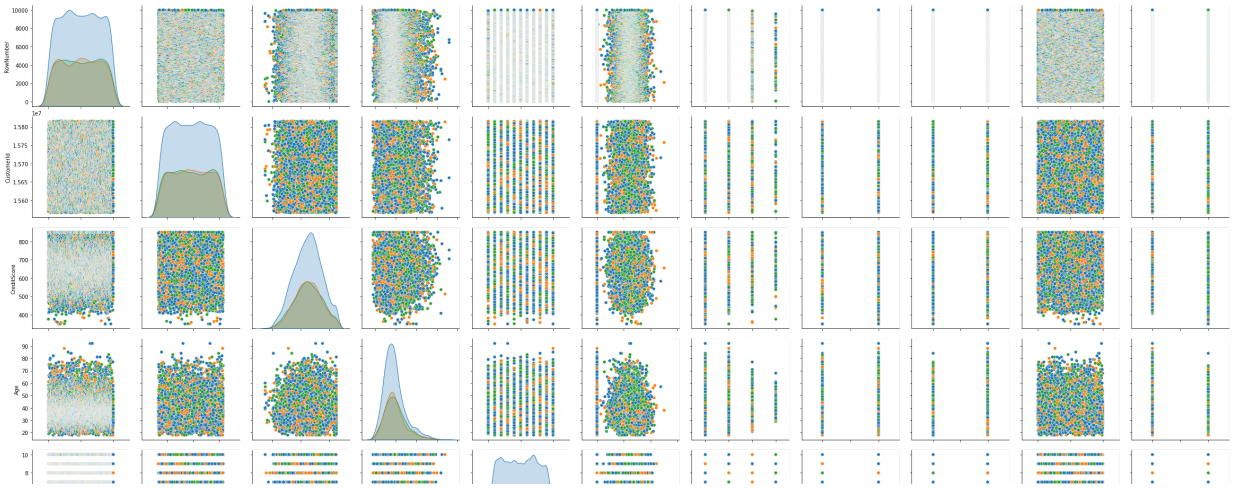


Multivariate Analysis

```
sns.pairplot(data, hue="Geography", size=3)
plt.show()
```

Saved successfully!

```
/usr/local/lib/python3.7/dist-packages/seaborn/axisgrid.py:2076: UserWarning: The `s
  warnings.warn(msg, UserWarning)
```



Descriptive statistics

```
data["Balance"].mean()
```

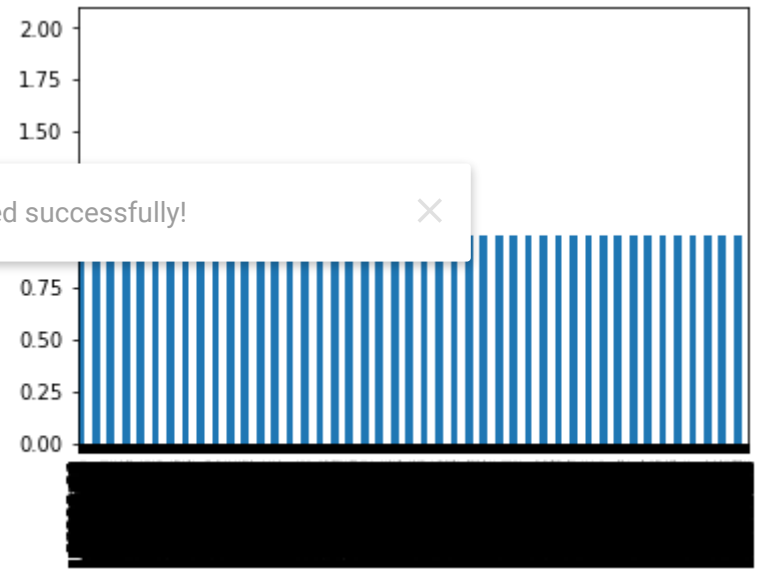
```
76485.889288
```

```
data["Balance"].median()
```

```
97198.54000000001
```

```
data["EstimatedSalary"].value_counts().plot(kind="bar")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5a19115350>
```



Saved successfully!

Handling Missing Values

```
new_data = data.fillna(0)
new_data
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure
0	1	15634602	Hargrave	619	France	Female	42.0	2
1	2	15647311	Hill	608	Spain	Female	41.0	1
2	3	15619304	Onio	502	France	Female	42.0	8
3	4	15701354	Boni	699	France	Female	39.0	1
4	5	15737888	Mitchell	850	Spain	Female	43.0	2
...	...	...	...	...	...	...	...	...
9995	9996	15606229	Obijiaku	771	France	Male	39.0	5
9996	9997	15569892	Johnstone	516	France	Male	35.0	10
9997	9998	15584532	Liu	709	France	Female	36.0	7
9998	9999	15682355	Sabbatini	772	Germany	Male	42.0	3
9999	10000	15628319	Walker	792	France	Female	28.0	4

10000 rows × 14 columns

```
new_data = data.fillna(method="ffill")
new_data
```

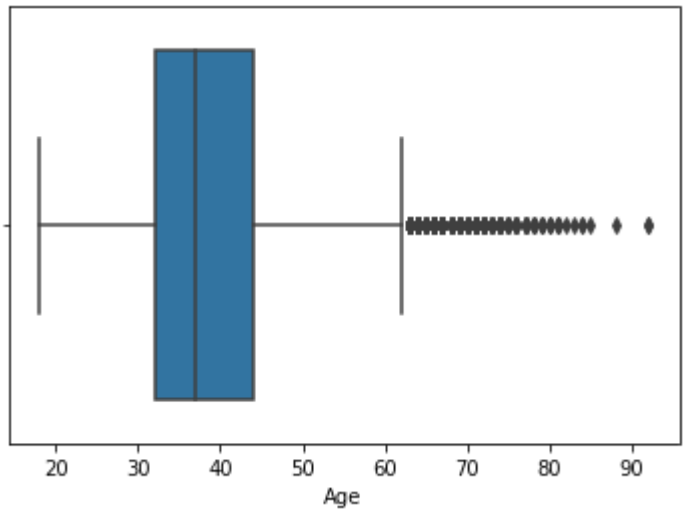
	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure
0	1	15634602	Hargrave	619	France	Female	42.0	2
1	2	15647311	Hill	608	Spain	Female	41.0	1
2	3	15619304	Onio	502	France	Female	42.0	8
					Boni	699	France	Female	39.0	1
4	5	15737888	Mitchell	850	Spain	Female	43.0	2
...	...	...	...	...	...	...	...	...
9995	9996	15606229	Obijiaku	771	France	Male	39.0	5
9996	9997	15569892	Johnstone	516	France	Male	35.0	10
9997	9998	15584532	Liu	709	France	Female	36.0	7
9998	9999	15682355	Sabbatini	772	Germany	Male	42.0	3
9999	10000	15628319	Walker	792	France	Female	28.0	4

10000 rows × 14 columns

Find outlier and replace outlier

```
sns.boxplot(data['Age'],data=data)
```

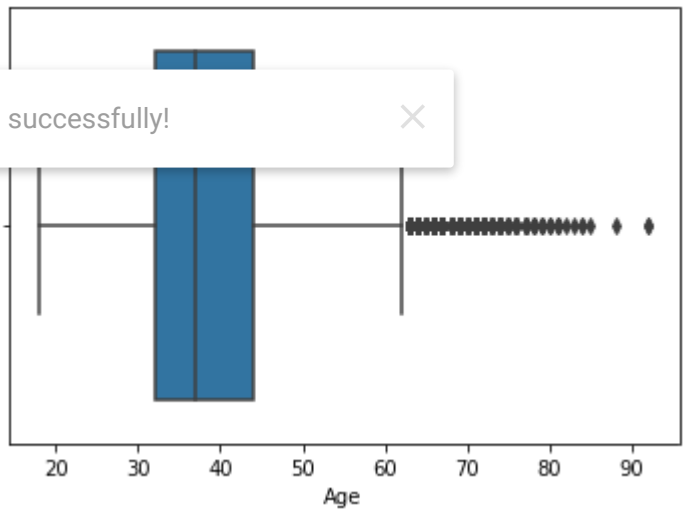
```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pas
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f5a1ee954d0>
```



Replace Outlier

```
sns.boxplot(data['Age'],data=data)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pas
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f5a19f60390>
```



Saved successfully!

```
def drop_outliers_IQR(df):

   q1=df.quantile(0.25)

   q3=df.quantile(0.75)
```

```
    IQR=q3-q1

    not_outliers = df[~((df<(q1-1.5*IQR)) | (df>(q3+1.5*IQR)))]

    outliers_dropped = outliers.dropna().reset_Age()

    return outliers_dropped


def remove_outlier_IQR(data):
    Q1=data.quantile(0.25)
    Q3=data.quantile(0.75)
    IQR=Q3-Q1
    data_final=df[~((data<(Q1-1.5*IQR)) | (data>(Q3+1.5*IQR)))]
    return data_final
```

Check categorical column perform encoding

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

```
     37.0    478
     38.0    477
     35.0    473
     36.0    456
     34.0    447
            ...
     92.0      2
     82.0      1
     88.0      1
     85.0      1
     83.0      1
     Name: Age, Length: 70, dtype: int64
```

```
data_sklearn = data.copy()
```

Saved successfully!

```
abelEncoder

lb_make = LabelEncoder()
data_sklearn['Tenure_code'] = lb_make.fit_transform(data['Tenure'])

data_sklearn.head()
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	B
0	1	15634602	Hargrave	619	France	Female	42.0	2	
1	2	15647311	Hill	608	Spain	Female	41.0	1	83

Split the data into dependent and independent variables.

```
x=data.iloc[:,1:4]
y=data.iloc[:,4]
```

Independent Variable

x

	CustomerId	Surname	CreditScore
0	15634602	Hargrave	619
1	15647311	Hill	608
2	15619304	Onio	502
3	15701354	Boni	699
4	15737888	Mitchell	850
...	...	...	...
9995	15606229	Obijiaku	771
9996	15569892	Johnstone	516
9997	15584532	Liu	709
			772
			792

10000 rows × 3 columns

Saved successfully!

Dependent Variable

y

```
0        France
1         Spain
2        France
3        France
4         Spain
         ...
9995     France
9996     France
9997     France
```

```
9998    Germany
9999     France
Name: Geography, Length: 10000, dtype: object
```

scale the independent variables

```
data = data.filter(["CustomerId", "Surname", "Creditscore"], axis = 1)
data.head()
```

	CustomerId	Surname
331	15601274	Hsieh
3222	15575247	Cartwright
4302	15791867	Hicks
3404	15576928	Walsh
9400	15584897	Kuo

split the data into testing and training

```
data = data.sample(frac = 1)
```

```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
```

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

Saved successfully!

```
train_test_split(
    X, y, test_size=0.05, random_state=0)
```

```
y = np.array(data["Surname"])
print("Shape of y:",y.shape)
print(y)
```

```
Shape of y: (10000,)
['Hsieh' 'Cartwright' 'Hicks' ... 'Ugorji' 'Grubb' 'Y?an']
```

```
print("Enter the splitting factor (i.e) ratio between train and test")
s_f = float(input())
```

```
Enter the splitting factor (i.e) ratio between train and test
8
```

```
n_train = math.floor(s_f * X.shape[0])
```

```
n_test = math.ceil((1-s_f) * X.shape[0])
X_train = X[:n_train]
y_train = y[:n_train]
X_test = X[n_train:]
y_test = y[n_train:]
print("Total Number of rows in train:",X_train.shape[0])
print("Total Number of rows in test:",X_test.shape[0])
```

```
Total Number of rows in train: 10000
Total Number of rows in test: 0
```

```
print("X:")
print(X)
print("y:")
print(y)
```

```
X:
      CustomerId
331     15601274
3222    15575247
4302    15791867
3404    15576928
9400    15584897
...          ...
8283    15754569
7766    15647259
6801    15776947
659     15603065
8388    15806570

[10000 rows x 1 columns]
y:
['Hsieh' 'Cartwright' 'Hicks' ... 'Ugorji' 'Grubb' 'Y?an']
```

```
print("X_train:")
```

Saved successfully!

```
print("\nX_test")
print(X_test)
print("\ny_test")
print(y_test)
```

Colab paid products - Cancel contracts here

✓ 0s completed at 2:42 PM

Saved successfully!