ASSIGNMENT 2

Assignment Date	21 /10/2022
Student Name	ANUVITHA G
Student Roll Number	61771921002
Maximum Marks	2 Marks

Data Visualization and Pre-processing

Perform Below Tasks to complete the assignment: -

Tasks: -

1. Download the dataset: Dataset

2. Load the dataset.

3. Perform Below Visualizations.

● Univariate Analysis

● Bi - Variate Analysis

● Multi - Variate Analysis

4. Perform descriptive statistics on the dataset.

5. Handle the Missing values.

6. Find the outliers and replace the outliers

7. Check for Categorical columns and perform encoding.

8. Split the data into dependent and independent variables.

9. Scale the independent variables

10. Split the data into training and testing

1.

Downloading dataset

```
[ ] from google.colab import files
    uploaded = files.upload()
```

Choose Files No file chosen   Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving Churn_Modelling.csv to Churn_Modelling.csv
```

2.

Loading dataset

```
import seaborn as sb
import pandas as pd
import matplotlib.pyplot as plt
churn=pd.read_csv("Churn_Modelling.csv")
churn.head()
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
0	1	15634602	Hargrave	619	France	Female	42	2	0.00	1	1	1	101348.88	1
1	2	15647311	Hill	608	Spain	Female	41	1	83807.86	1	0	1	112542.58	0
2	3	15619304	Onio	502	France	Female	42	8	159660.80	3	1	0	113931.57	1
3	4	15701354	Boni	699	France	Female	39	1	0.00	2	0	0	93826.63	0
4	5	15737888	Mitchell	850	Spain	Female	43	2	125510.82	1	1	1	79084.10	0

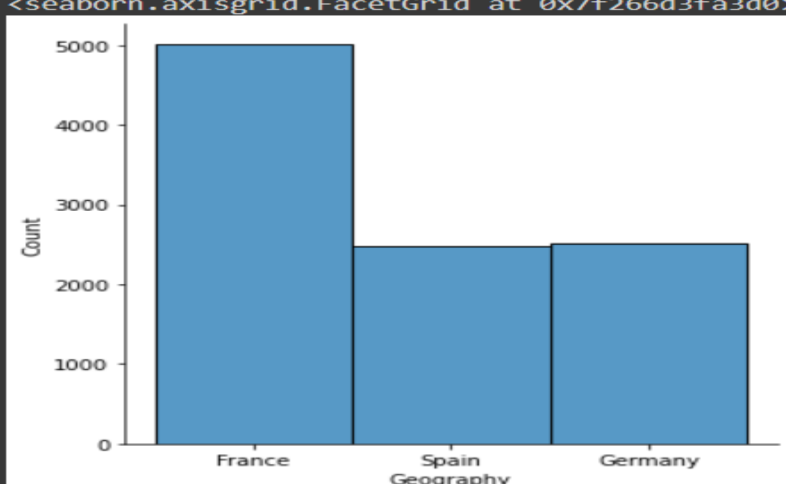
3.

Visualization

Univariate analysis

```
sb.displot(churn, x="Geography")
```

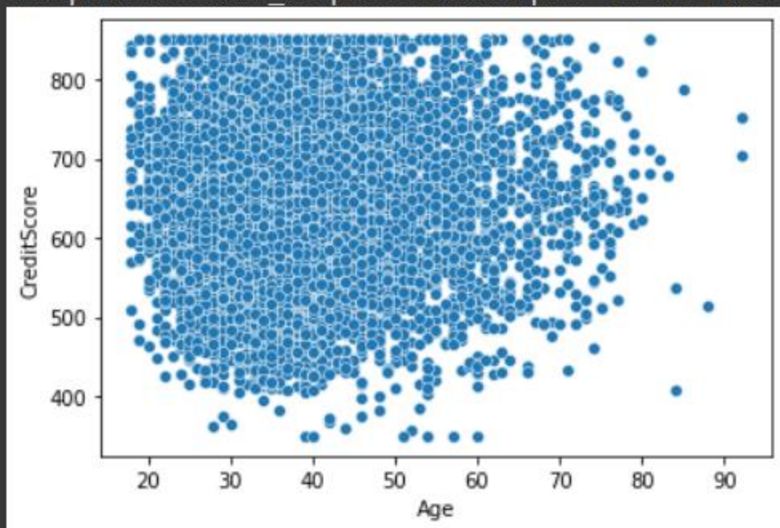
```
<seaborn.axisgrid.FacetGrid at 0x7f266d3fa3d0>
```



Bivariate analysis

```
[ ] sb.scatterplot(data=churn, x="Age", y="CreditScore")
```

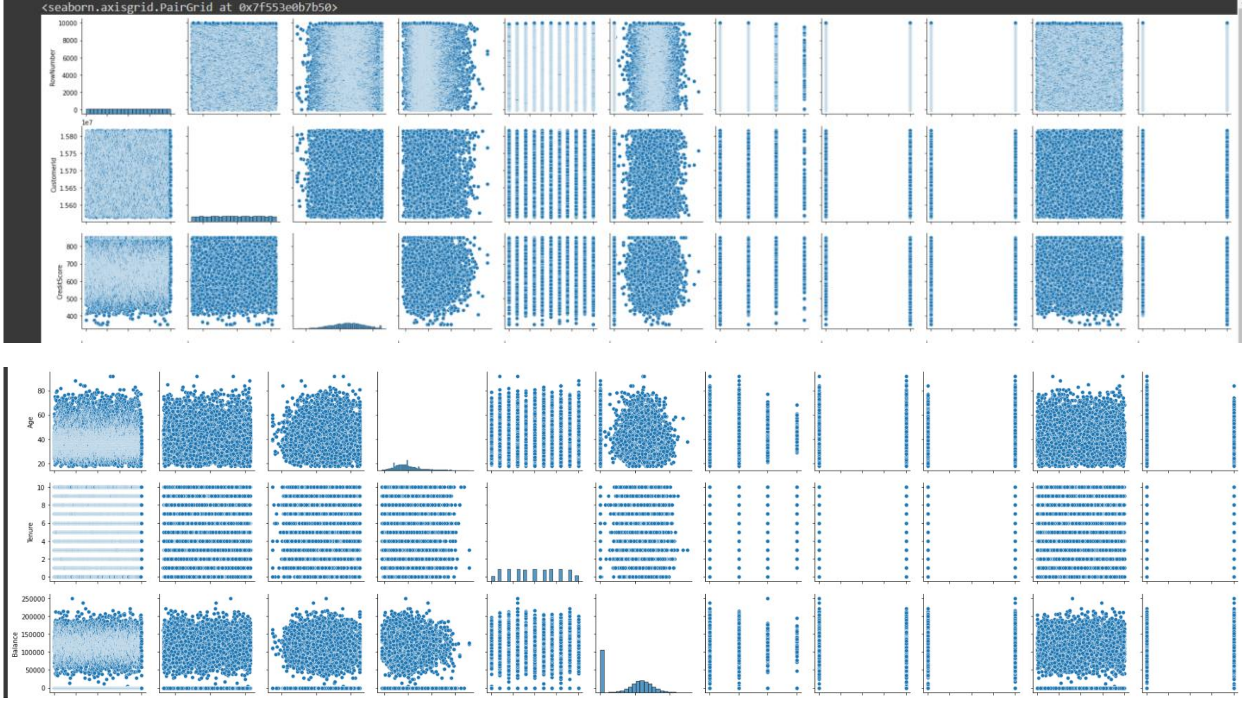
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f553e5f2150>
```

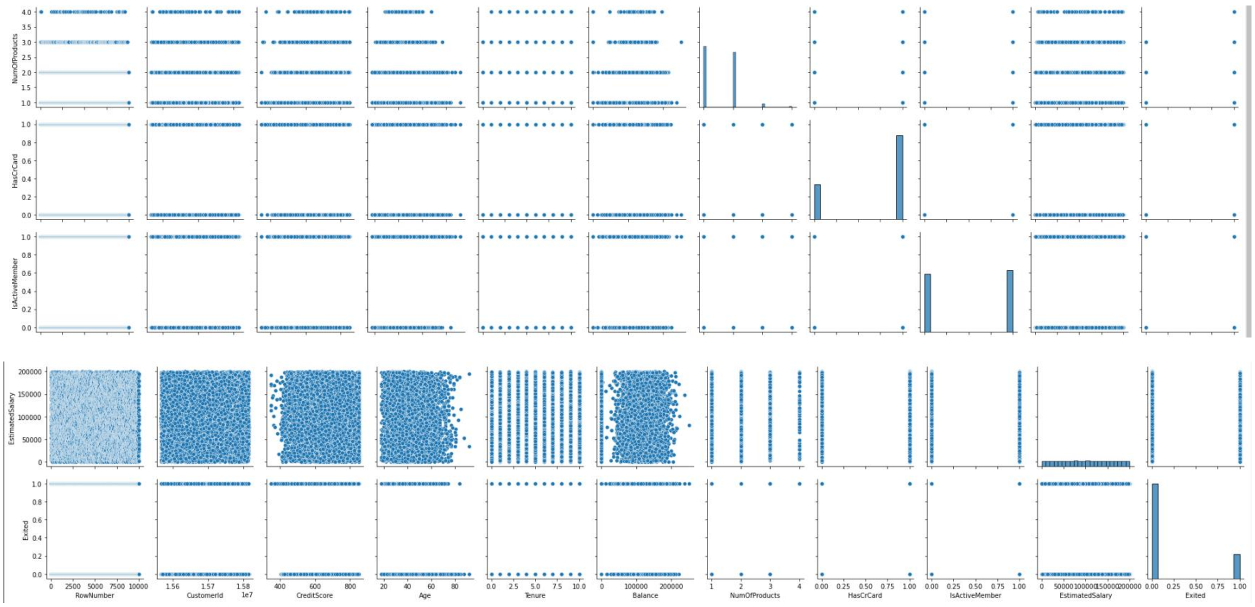


Multivariate analysis

```
[ ] sb.pairplot(churn)
```

```
<seaborn.axisgrid.PairGrid at 0x7f553e0b7b50>
```





4.

Descriptive statistics

```
churn.describe()
```

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
count	10000.00000	1.000000e+04	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.00000	10000.000000	10000.000000	10000.000000
mean	5000.50000	1.569094e+07	650.528800	38.921800	5.012800	76485.889288	1.530200	0.70550	0.515100	100090.239881	0.203700
std	2886.89568	7.193619e+04	96.653299	10.487806	2.892174	62397.405202	0.581654	0.45584	0.499797	57510.492818	0.402769
min	1.00000	1.556570e+07	350.000000	18.000000	0.000000	0.000000	1.000000	0.00000	0.000000	11.580000	0.000000
25%	2500.75000	1.562853e+07	584.000000	32.000000	3.000000	0.000000	1.000000	0.00000	0.000000	51002.110000	0.000000
50%	5000.50000	1.569074e+07	652.000000	37.000000	5.000000	97198.540000	1.000000	1.00000	1.000000	100193.915000	0.000000
75%	7500.25000	1.575323e+07	718.000000	44.000000	7.000000	127644.240000	2.000000	1.00000	1.000000	149388.247500	0.000000
max	10000.00000	1.581569e+07	850.000000	92.000000	10.000000	250898.090000	4.000000	1.00000	1.000000	199992.480000	1.000000

5.

Handling missing value

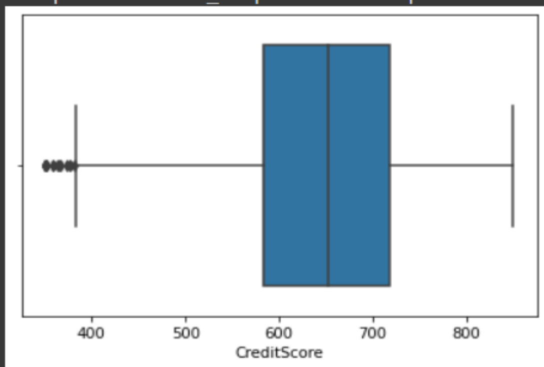
```
churn.isnull().any()
```

```
RowNumber          False
CustomerId         False
Surname            False
CreditScore        False
Geography          False
Gender             False
Age                False
Tenure             False
Balance            False
NumOfProducts      False
HasCrCard          False
IsActiveMember     False
EstimatedSalary    False
Exited             False
dtype: bool
```

6.

```
sb.boxplot(churn.CreditScore)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following v
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f26691dde50>
```




```
q1=churn.CreditScore.quantile(0.25)
q3=churn.CreditScore.quantile(0.75)
IQR = q3-q1
upper=q3 + 1.5 * IQR
print(upper)
```

```
919.0
```

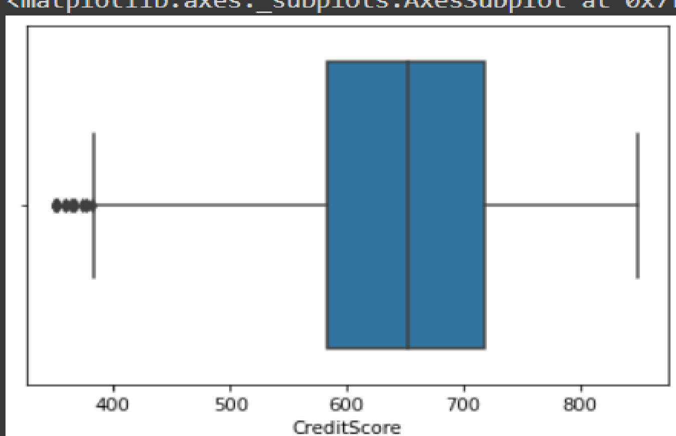
```
import statistics
statistics.median(churn.CreditScore)
```

```
652.0
```

```
import numpy as np
churn.CreditScore=np.where(churn.CreditScore>upper,652,churn.CreditScore)
```

```
sb.boxplot(churn.CreditScore)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning:
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f2668f35350>
```



7.

Check for Categorical columns and perform encoding.

Checking for categorical column

```
[ ] churn.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
```

```
obj = churn.select_dtypes(include=['object']).copy()
obj.head()
```

	Surname	Geography	Gender
0	Hargrave	France	Female
1	Hill	Spain	Female
2	Onio	France	Female
3	Boni	France	Female
4	Mitchell	Spain	Female

one hot encoding

```
df=pd.get_dummies(churn,columns=['Gender'])
df.head()
df2=pd.get_dummies(df,columns=['Geography'])
df2.head()
```

	RowNumber	CustomerId	Surname	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited	Gender_Female	Gender_Male
0	1	15634602	Hargrave	619	42	2	0.00	1	1	1	101348.88	1	1	0
1	2	15647311	Hill	608	41	1	83807.86	1	0	1	112542.58	0	1	0
2	3	15619304	Onio	502	42	8	159660.80	3	1	0	113931.57	1	1	0
3	4	15701354	Boni	699	39	1	0.00	2	0	0	93826.63	0	1	0
4	5	15737888	Mitchell	850	43	2	125510.82	1	1	1	79084.10	0	1	0

8.

Spliting the data into dependent and independent variables

```
y=df2['Surname']
y
```

```
0        Hargrave
1            Hill
2            Onio
3            Boni
4        Mitchell
           ...
9995     Obijiaku
9996    Johnstone
9997          Liu
9998    Sabbatini
9999       Walker
Name: Surname, Length: 10000, dtype: object
```

```
X=df2.drop(columns=['Surname'],axis=1)
X.head()
```

RowNumber	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited	Gender_Female	Gender_Male	Geography_France	Geography_
1	15634602	619	42	2	0.00	1	1	1	101348.88	1	1	0	1	
2	15647311	608	41	1	83807.86	1	0	1	112542.58	0	1	0	0	
3	15619304	502	42	8	159660.80	3	1	0	113931.57	1	1	0	1	
4	15701354	699	39	1	0.00	2	0	0	93826.63	0	1	0	1	
5	15737888	850	43	2	125510.82	1	1	1	79084.10	0	1	0	0	

+ Code + Text

9.

Scaling the independent variables

```
[ ] from sklearn.preprocessing import scale
```

```
[ ] x_scaled=pd.DataFrame(scale(X),columns=X.columns)
    x_scaled.head()
```

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited	Gender_Female	Gender_Male	Geography_France
0	-1.731878	-0.783213	-0.326221	0.293517	-1.041760	-1.225848	-0.911583	0.646092	0.970243	0.021886	1.977165	1.095988	-1.095988	0.997204
1	-1.731531	-0.606534	-0.440036	0.198164	-1.387538	0.117350	-0.911583	-1.547768	0.970243	0.216534	-0.505775	1.095988	-1.095988	-1.002804
2	-1.731185	-0.995885	-1.536794	0.293517	1.032908	1.333053	2.527057	0.646092	-1.030670	0.240687	1.977165	1.095988	-1.095988	0.997204
3	-1.730838	0.144767	0.501521	0.007457	-1.387538	-1.225848	0.807737	-1.547768	-1.030670	-0.108918	-0.505775	1.095988	-1.095988	0.997204
4	-1.730492	0.652659	2.063884	0.388871	-1.041760	0.785728	-0.911583	0.646092	0.970243	-0.365276	-0.505775	1.095988	-1.095988	-1.002804

10.

Spliting the data into training and testing

```
[ ] from sklearn.model_selection import train_test_split
    X_train,X_test,y_train,y_test=train_test_split(x_scaled,y,test_size=0.2,random_state=0)
```

```
[ ] X_train.shape

(8000, 16)
```

```
[ ] X_test.shape

(2000, 16)
```

```
y_train.shape

(8000,)
```

```
[ ] y_test.shape

(2000,)
```