Assignment -2

Data Visualization and Pre-processing

Assignment Date	28 September 2022
Student Name	K. Aafreen Benazir
Student Roll Number	913119104001
Maximum Marks	2 Marks

Question-1:

Download the dataset

Question-2:

Load the dataset:

Solution:

```
[23] import pandas as pd  #importing necessary libraries
     import numpy as np
     import matplotlib.pyplot as plt
     import seaborn as sns
```

```
[24] df=pd.read_csv("Churn_Modelling.csv")  #loading the data
```

```
[25] df
```

	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
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9995	9996	15606229	Obijiaku	771	France	Male	39	5	0.00	2	1	0	96270.64	0

Question-3:

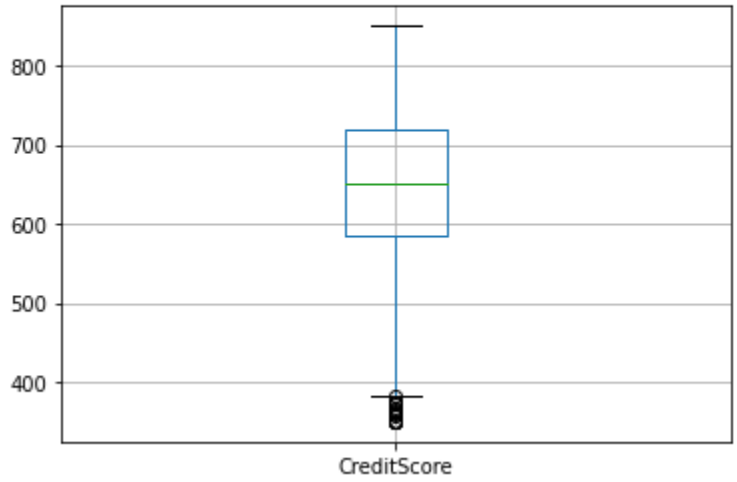
Perform Below Visualizations.

- Univariate Analysis
- Bi - Variate Analysis
- Multi - Variate Analysis

Solution:

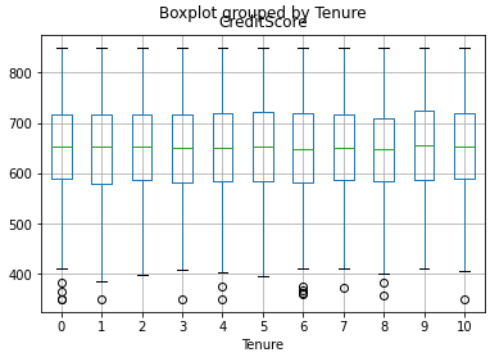
```
[4]  df.boxplot("CreditScore")      #Univariate
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f480bce8b50>
```



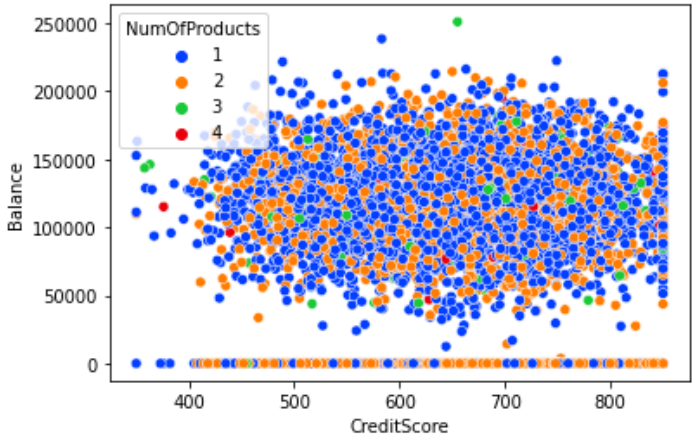

```
[5]  df.boxplot("CreditScore","Tenure")  #Bivariate
```

```
/usr/local/lib/python3.7/dist-packages/matplotlib/cbook/__init__.py:1376: VisibleDeprecationWarning: Creating an ndarray
  X = np.atleast_1d(X.T if isinstance(X, np.ndarray) else np.asarray(X))
<matplotlib.axes._subplots.AxesSubplot at 0x7f480bbdd6d0>
```



```
[6]  sns.scatterplot(x='CreditScore',y='Balance',data=df,palette='bright',
         hue='NumOfProducts')                                          #Multivariate
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f480b5d2910>
```



Question-4:

Perform descriptive statistics on the dataset.

Solution:

```
[7]  df.describe()   #descriptive statistics
```

	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

Question-5:

Handle the Missing values

Solution:

```
[10] df.fillna(5)
```

	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
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9995	9996	15606229	Obijiaku	771	France	Male	39	5	0.00	2	1	0	96270.64	0
9996	9997	15569892	Johnstone	516	France	Male	35	10	57369.61	1	1	1	101699.77	0
9997	9998	15584532	Liu	709	France	Female	36	7	0.00	1	0	1	42085.58	1
9998	9999	15682355	Sabbatini	772	Germany	Male	42	3	75075.31	2	1	0	92888.52	1
9999	10000	15628319	Walker	792	France	Female	28	4	130142.79	1	1	0	38190.78	0

Question-6:

Find the outliers and replace the outliers

Solution:

```
[16] out = df.drop(columns=['Gender', 'Tenure', 'HasCrCard', 'IsActiveMember', 'NumOfProducts', 'Exited']).quantile(q=[0.25, 0.50]) #OUTLIERS
     out
```

	RowNumber	CustomerId	CreditScore	Age	Balance	EstimatedSalary
0.25	2500.75	15628528.25	584.0	32.0	0.00	51002.110
0.50	5000.50	15690738.00	652.0	37.0	97198.54	100193.915

```
[17] Q1 = out.iloc[0]
     Q3 = out.iloc[1]
     iqr = Q3 - Q1
     iqr
```

```
RowNumber          2499.750
CustomerId        62209.750
CreditScore          68.000
Age                   5.000
Balance           97198.540
EstimatedSalary   49191.805
dtype: float64
```

```
[18] upper = out.iloc[1] + 1.5*iqr
     upper
```

```
RowNumber          8.750125e+03
CustomerId         1.578405e+07
CreditScore        7.540000e+02
Age                4.450000e+01
Balance            2.429964e+05
EstimatedSalary    1.739816e+05
dtype: float64
```

```
[19] lower = out.iloc[0] - 1.5*iqr
     lower
```

```
RowNumber         -1.248875e+03
CustomerId         1.553521e+07
CreditScore        4.820000e+02
Age                2.450000e+01
Balance           -1.457978e+05
EstimatedSalary   -2.278560e+04
dtype: float64
```

```
[20] df['CreditScore'] = np.where(df['CreditScore']>756, 650.5288, df['CreditScore'])
     df['Age'] = np.where(df['Age']>62, 38.9218, df['Age'])
```

Question-7:

Check for Categorical columns and perform encoding

Solution:

```
[11] from sklearn.preprocessing import OneHotEncoder

     encoder = OneHotEncoder(sparse=False)
     encoding = encoder.fit_transform(df)      #encoding
     encoding
```

```
array([[1., 0., 0., ..., 0., 0., 1.],
       [0., 1., 0., ..., 0., 1., 0.],
       [0., 0., 1., ..., 0., 0., 1.],
       ...,
       [0., 0., 0., ..., 0., 0., 1.],
       [0., 0., 0., ..., 0., 0., 1.],
       [0., 0., 0., ..., 0., 1., 0.]])
```

Question-8:

Split the data into dependent and independent variables.

Solution:

```
[12] x=df.iloc[:, :-1].values  #independent var
     x
```

```
array([[1, 15634602, 'Hargrave', ..., 1, 1, 101348.88],
       [2, 15647311, 'Hill', ..., 0, 1, 112542.58],
       [3, 15619304, 'Onio', ..., 1, 0, 113931.57],
       ...,
       [9998, 15584532, 'Liu', ..., 0, 1, 42085.58],
       [9999, 15682355, 'Sabbatini', ..., 1, 0, 92888.52],
       [10000, 15628319, 'Walker', ..., 1, 0, 38190.78]], dtype=object)
```

```
[13] y = df.iloc[:, -1].values  #dependent var
     y
```

```
array([1, 0, 1, ..., 1, 1, 0])
```

Question-9:

Scale the independent variables

Solution:

```
[26] from sklearn.preprocessing import StandardScaler
```

```
[29] df.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

```
sc = StandardScaler()
x = sc.fit_transform(x)
x = sc.transform(x)
```

Question-10:

Split the data into training and testing

Solution:

```
[14] from sklearn.model_selection import train_test_split       #spliting data into training and testing
     x_train, x_test, y_train, y_test = train_test_split(x, y, random_state=0, train_size = .75)
```

```
[15] print(x_train.shape)
     print(x_test.shape)
     print(y_train.shape)
     print(y_test.shape)
```

```
(7500, 13)
(2500, 13)
(7500,)
(2500,)
```