Assignment – 2

Data Visualization and Pre-processing

Assignment Date	22 September 2022
Student Name	Mr. T. Logavarthan
Student Roll Number	142219106048
Maximum Marks	2 Marks

TASKS:

1. Download the dataset

2. Load the dataset

Import dataset

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

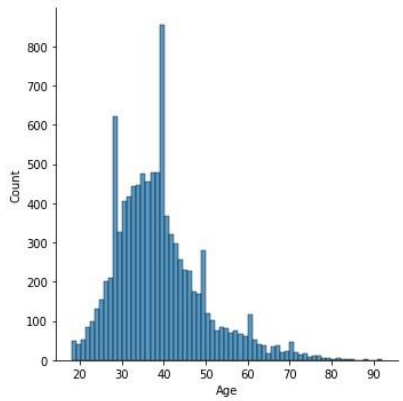
```
[3] data.head(8)
```

	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
5	6	15574012	Chu	645	Spain	Male	44	8	113755.78	2	1	0	149756.71	1
6	7	15592531	Bartlett	822	France	Male	50	7	0.00	2	1	1	10062.80	0
7	8	15656148	Obinna	376	Germany	Female	29	4	115046.74	4	1	0	119346.88	1

3. Perform Below Visualizations.
   - Univariate Analysis

```
sns.displot(data['Age'])
```

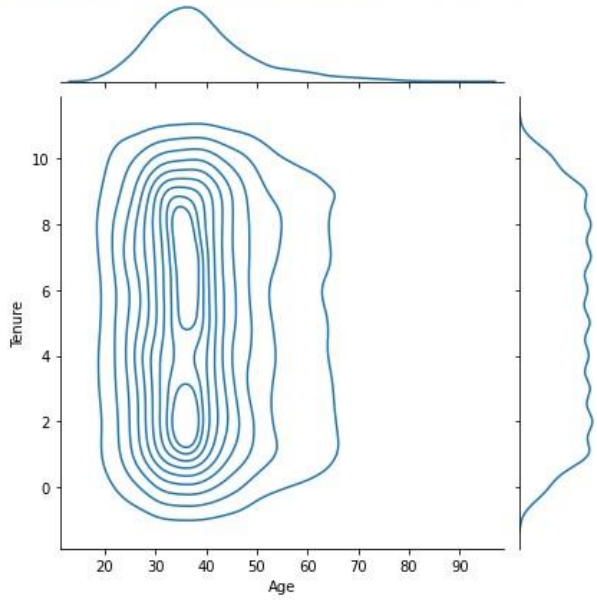
```
<seaborn.axisgrid.FacetGrid at 0x7ff5e0fd03d0>
```



   - Bi-Variate Analysis

```
sns.jointplot(data['Age'],data['Tenure'],kind="kde")
```

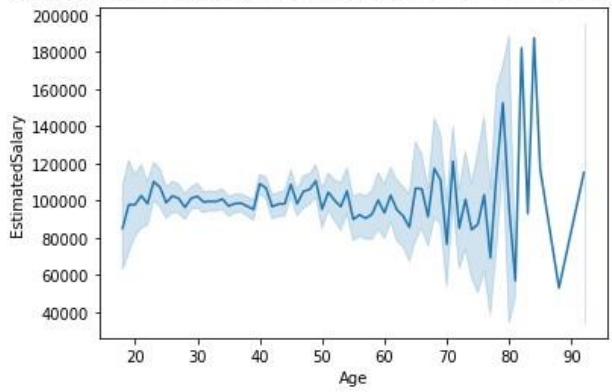
```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following
  FutureWarning
<seaborn.axisgrid.JointGrid at 0x7ff5c7c1b310>
```



- Multivariate Analysis

```
[6] sns.lineplot(data['Age'],data['EstimatedSalary'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the follo
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7ff5caa69690>
```



4. Perform descriptive statistics on the dataset

```
data
```

	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

10000 rows × 14 columns

```
[23] print("Mode = " ,data["Age"].mode())
     print("Mean = " ,data["Age"].mean())
     print("Mean after roundoff = " ,round(data["Age"].mean(), 2))
     print("Median = " ,data["Age"].median())
```

```
Mode =  0    37
dtype: int64
Mean =  38.9218
Mean after roundoff =  38.92
Median =  37.0
```

```
[24] print("Range = " ,data['Age'].max() - data['Age'].min())
```

```
Range =  74
```

```
[25] print("Standard Deviation = " ,round(data['Age'].std(),2))
```

```
Standard Deviation =  10.49
```

```
#To find statistics of all numerical Datas
round(data.describe(),2)
```

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
count	10000.00	10000.00	10000.00	10000.00	10000.00	10000.00	10000.00	10000.00	10000.00	10000.00	10000.0
mean	5000.50	15690940.57	650.53	38.92	5.01	76485.89	1.53	0.71	0.52	100090.24	0.2
std	2886.90	71936.19	96.65	10.49	2.89	62397.41	0.58	0.46	0.50	57510.49	0.4
min	1.00	15565701.00	350.00	18.00	0.00	0.00	1.00	0.00	0.00	11.58	0.0
25%	2500.75	15628528.25	584.00	32.00	3.00	0.00	1.00	0.00	0.00	51002.11	0.0
50%	5000.50	15690738.00	652.00	37.00	5.00	97198.54	1.00	1.00	1.00	100193.92	0.0
75%	7500.25	15753233.75	718.00	44.00	7.00	127644.24	2.00	1.00	1.00	149388.25	0.0
max	10000.00	15815690.00	850.00	92.00	10.00	250898.09	4.00	1.00	1.00	199992.48	1.0

```
[32] data.loc[data['EstimatedSalary']>60000]
```

	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
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9992	9993	15657105	Chukwualuka	726	Spain	Male	36	2	0.00	1	1	0	195192.40	0
9994	9995	15719294	Wood	800	France	Female	29	2	0.00	2	0	0	167773.55	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
9998	9999	15682355	Sabbatini	772	Germany	Male	42	3	75075.31	2	1	0	92888.52	1

7039 rows × 14 columns

```
[34] data.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
```

5. Handle the Missing values.

```
[36] data.isnull().sum()
```

```
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
```

6. Find the outliers and replace the outliers

```
[39] out = data.quantile(q =(0.25,0.75))
     out
```

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
0.25	2500.75	15628528.25	584.0	32.0	3.0	0.00	1.0	0.0	0.0	51002.1100	0.0
0.75	7500.25	15753233.75	718.0	44.0	7.0	127644.24	2.0	1.0	1.0	149388.2475	0.0

```
[41] iq = out.loc[0.75]-out.loc[0.25]
     iq
```

```
RowNumber            4999.5000
CustomerId         124705.5000
CreditScore           134.0000
Age                    12.0000
Tenure                  4.0000
Balance            127644.2400
NumOfProducts           1.0000
HasCrCard               1.0000
IsActiveMember          1.0000
EstimatedSalary     98386.1375
Exited                  0.0000
dtype: float64
```

```
[42] lower = out.loc[0.25]-1.5*iq
     lower
```

```
RowNumber         -4.998500e+03
CustomerId         1.544147e+07
CreditScore        3.830000e+02
Age                1.400000e+01
Tenure            -3.000000e+00
Balance           -1.914664e+05
NumOfProducts     -5.000000e-01
HasCrCard         -1.500000e+00
IsActiveMember    -1.500000e+00
EstimatedSalary   -9.657710e+04
Exited             0.000000e+00
dtype: float64
```

```
[43] upper = out.loc[0.75]+1.5*iq
     upper
```

```
RowNumber         1.499950e+04
CustomerId        1.594029e+07
CreditScore       9.190000e+02
Age               6.200000e+01
Tenure            1.300000e+01
Balance           3.191106e+05
NumOfProducts     3.500000e+00
HasCrCard         2.500000e+00
IsActiveMember    2.500000e+00
EstimatedSalary   2.969675e+05
Exited            0.000000e+00
dtype: float64
```

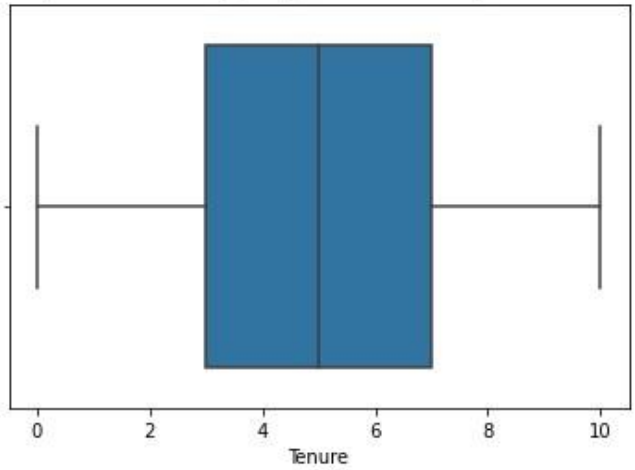
```
data.mean()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: FutureWa
  """Entry point for launching an IPython kernel.
RowNumber          5.000500e+03
CustomerId         1.569094e+07
CreditScore        6.505288e+02
Age                3.892180e+01
Tenure             5.012800e+00
Balance            7.648589e+04
NumOfProducts      1.530200e+00
HasCrCard          7.055000e-01
IsActiveMember     5.151000e-01
EstimatedSalary    1.000902e+05
Exited             2.037000e-01
dtype: float64
```

```
[45] data['Age'] = np.where(data['Age']>87,40,data['Age'])
     data['Tenure'] = np.where(data['Tenure']>87,31,data['Tenure'])
```

```
sns.boxplot(data['Tenure'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarni
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7ff5c603a050>
```



7. Check for Categorical columns and perform encoding.

```
[47] data.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

```
[48] data['Gender'].replace({"Female":0,"Male":1},inplace=True)
     data['Exited'].replace({"yes":1,"no":0},inplace=True)
```

```
[49] data_main = pd.get_dummies(data,columns=['Geography'])
```

```
data_main.head()
```

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

```
[51] y = data_main['Balance']
```

```
[52] x= data_main.drop(columns=['Balance'],axis=1)
```

```
[53] x.head()
```

8. Split the data into dependent and independent variables.

```
# Independent
x = data.iloc[:,0:3]
x
```

	RowNumber	CustomerId	Surname
0	1	15634602	Hargrave
1	2	15647311	Hill
2	3	15619304	Onio
3	4	15701354	Boni
4	5	15737888	Mitchell
...	...	...	...
9995	9996	15606229	Obijiaku
9996	9997	15569892	Johnstone
9997	9998	15584532	Liu
9998	9999	15682355	Sabbatini
9999	10000	15628319	Walker

10000 rows × 3 columns

```
[56] # Independent
     y = data['Exited']
     y
```

```
0       1
1       0
2       1
3       0
4       0
       ..
9995    0
9996    0
9997    1
9998    1
9999    0
Name: Exited, Length: 10000, dtype: int64
```

9. Scale the independent variables

```
[58] x= data_main.drop(columns=['Surname'],axis=1)
```

```
[59] names = x.columns
```

```
[60] names
```

```
Index(['RowNumber', 'CustomerId', 'CreditScore', 'Gender', 'Age', 'Tenure',
       'Balance', 'NumOfProducts', 'HasCrCard', 'IsActiveMember',
       'EstimatedSalary', 'Exited', 'Geography_France', 'Geography_Germany',
       'Geography_Spain'],
      dtype='object')
```

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

```
x = scale (x)
x
```

```
array([[-1.73187761, -0.78321342, -0.32622142, ...,  0.99720391,
        -0.57873591, -0.57380915],
       [-1.7315312 , -0.60653412, -0.44003595, ..., -1.00280393,
        -0.57873591,  1.74273971],
       [-1.73118479, -0.99588476, -1.53679418, ...,  0.99720391,
        -0.57873591, -0.57380915],
       ...,
       [ 1.73118479, -1.47928179,  0.60498839, ...,  0.99720391,
        -0.57873591, -0.57380915],
       [ 1.7315312 , -0.11935577,  1.25683526, ..., -1.00280393,
         1.72790383, -0.57380915],
       [ 1.73187761, -0.87055909,  1.46377078, ...,  0.99720391,
        -0.57873591, -0.57380915]])
```

```
[63] x = pd.DataFrame(x,columns=names)
```

```
[64] x.head()
```

	RowNumber	CustomerId	CreditScore	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited	Geography_Franc
0	-1.731878	-0.783213	-0.326221	-1.095988	0.296051	-1.041760	-1.225848	-0.911583	0.646092	0.970243	0.021886	1.977165	0.9972C
1	-1.731531	-0.606534	-0.440036	-1.095988	0.200347	-1.387538	0.117350	-0.911583	-1.547768	0.970243	0.216534	-0.505775	-1.0028C
2	-1.731185	-0.995885	-1.536794	-1.095988	0.296051	1.032908	1.333053	2.527057	0.646092	-1.030670	0.240687	1.977165	0.9972C
3	-1.730838	0.144767	0.501521	-1.095988	0.008939	-1.387538	-1.225848	0.807737	-1.547768	-1.030670	-0.108918	-0.505775	0.9972C
4	-1.730492	0.652659	2.063884	-1.095988	0.391755	-1.041760	0.785728	-0.911583	0.646092	0.970243	-0.365276	-0.505775	-1.0028C

10. Split the data into training and testing

```
[65] from sklearn.model_selection import train_test_split
```

```
[66] x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state=0)
```

```
x_train.head()
```

	RowNumber	CustomerId	CreditScore	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited	Geography_F
7389	0.827747	-0.195066	0.170424	-1.095988	-0.469582	-0.004426	-1.225848	0.807737	0.646092	-1.030670	1.108382	-0.505775	-1.0
9275	1.481077	0.810821	-2.312802	0.912419	0.296051	-1.387538	-0.012892	-0.911583	0.646092	0.970243	-0.747592	-0.505775	-1.0
2995	-0.694379	-1.507642	-1.195351	-1.095988	-0.948103	-1.041760	0.575076	-0.911583	0.646092	-1.030670	1.487464	-0.505775	0.9
5316	0.109639	1.243462	0.035916	0.912419	0.104643	-0.004426	0.467955	-0.911583	0.646092	-1.030670	1.278558	-0.505775	-1.0
356	-1.608556	-1.100775	2.063884	-1.095988	1.731613	1.032908	0.806010	0.807737	0.646092	0.970243	0.560069	-0.505775	-1.0

```
[68] x_test.head()
```

	RowNumber	CustomerId	CreditScore	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited	Geography_Fr
9394	1.522299	-1.045250	-0.553850	-1.095988	-0.373878	1.032908	0.875323	-0.911583	0.646092	0.970243	1.613046	-0.505775	-1.0C
898	-1.420801	-0.503813	-1.319512	-1.095988	0.104643	-1.041760	0.424422	-0.911583	0.646092	-1.030670	0.497532	1.977165	0.99
2398	-0.901186	-0.793292	0.573948	-1.095988	0.296051	1.032908	0.302927	-0.911583	0.646092	0.970243	-0.423561	-0.505775	-1.0C
5906	0.314021	0.760190	1.422384	0.912419	-0.660990	-0.350204	0.570464	-0.911583	-1.547768	-1.030670	-0.186439	-0.505775	0.99
2343	-0.920239	1.042107	0.573948	0.912419	-0.086765	-0.004426	1.387129	0.807737	0.646092	0.970243	0.618560	-0.505775	-1.0C

```
y_train,y_test
```

```
(7389    0
 9275    0
 2995    0
 5316    0
 356     0
        ..
 9225    0
 4859    0
 3264    0
 9845    0
 2732    1
 Name: Exited, Length: 8000, dtype: int64, 9394    0
 898     1
 2398    0
 5906    0
 2343    0
        ..
 1037    0
 2899    0
 9549    0
 2740    0
 6690    0
 Name: Exited, Length: 2000, dtype: int64)
```

COLAB LINK :
https://colab.research.google.com/drive/1v5GSQB_R4UiRDTSUiZZweKCYpPN2nblT#scrollTo=eyzUNIQQvCza