

Assignment -2
Data Visualization and Pre-Processing

Assignment Date	21 September 2022
Student Name	SANTHANALAKSHMI.R
Student Roll Number	812019106026
Maximum Marks	2 Marks

1.Download Dataset

importing package

```
import pandas as pd
import seaborn as sns
import numpy as np
from matplotlib import pyplot as plt
%matplotlib inline
```

2.Loading dataset

```
df = pd.read_csv("Churn_Modelling.csv")
```

In []:

```
df
```

Out[]:

	Row Number	Customer Id	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	Micheli	850	Spain	Female	43	2	125510.82	1	1	1	79084.10	0

	Row Num ber	Cust omer Id	Sur na me	Cred itSco re	Geo grap hy	Ge nd er	A g e	Te nu re	Bal anc e	NumO fProdu cts	Has CrC ard	IsActiv eMem ber	Estima tedSal ary	Ex ite d
...
9995	9996	15606229	Obijaku	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	Sabatin	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

1. 10000 rows × 14 columns

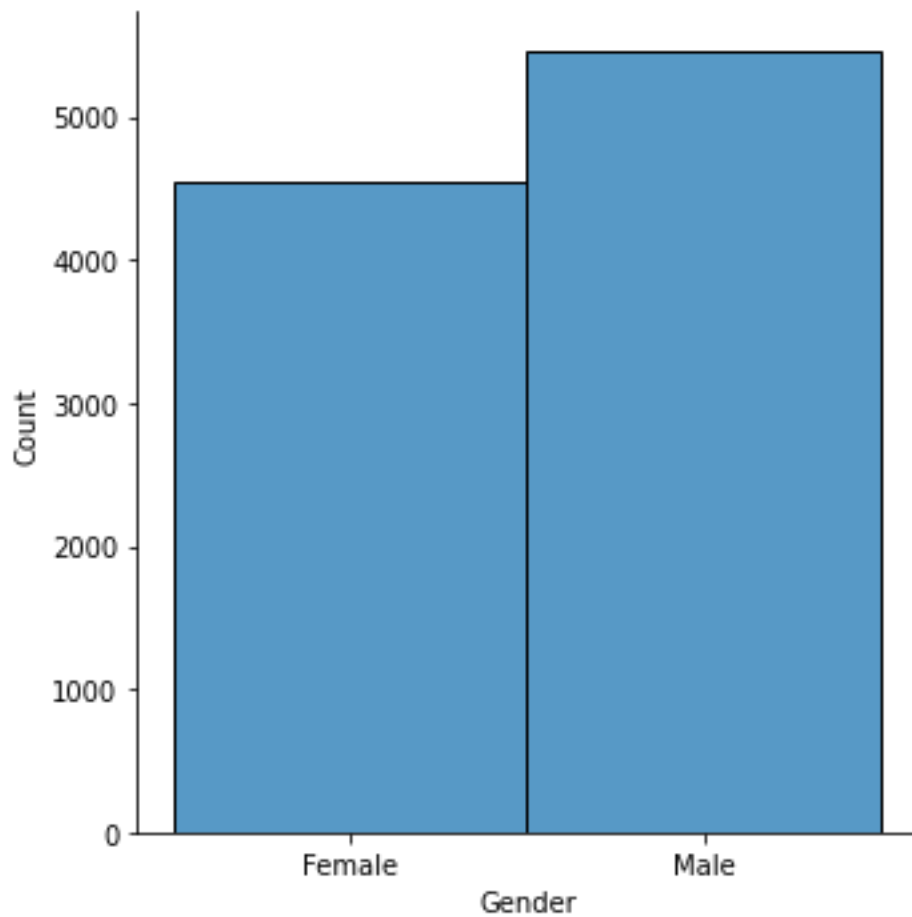
3.Visualizations

a) Univariate Analysis

```
sns.displot(df.Gender)
```

```
<seaborn.axisgrid.FacetGrid at 0x7f2935aae790>
```

Out[]:

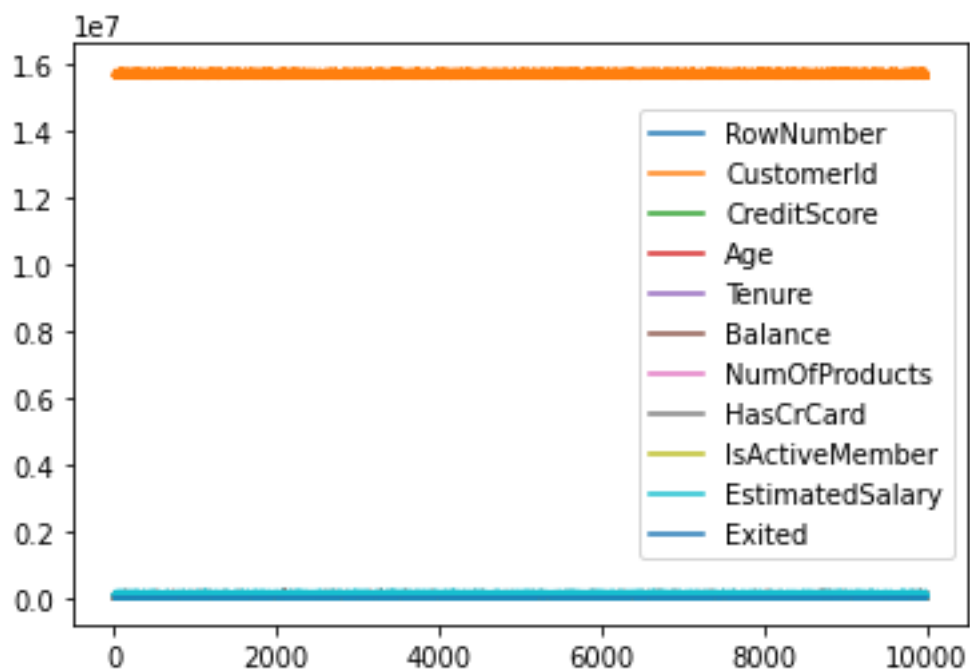


b) Bi-Variate Analysis

```
df.plot.line()
```

Out[]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f2932d2df50>
```

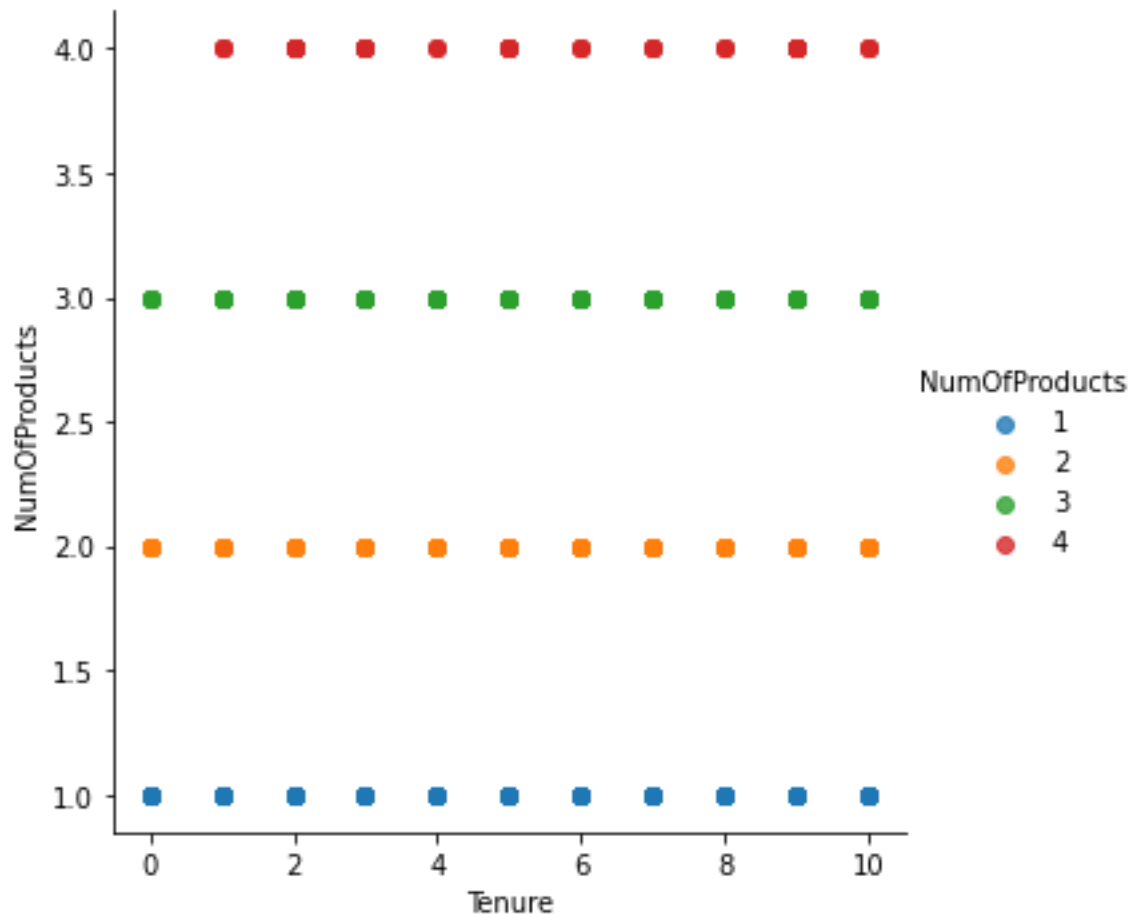


c)Multi - Variate Analysis

```
sns.lmplot("Tenure", "NumOfProducts", df, hue="NumOfProducts", fit_reg=False);
```

/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables as keyword args: x, y, data. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.

FutureWarning



4.Perform descriptive statistics on the dataset.

```
df.describe()
```

Out[]:

	RowN umber	Custo merId	Credit Score	Age	Tenur e	Balanc e	NumOf Product s	HasC rCar d	IsActive Membe r	Estimat edSalar y	Exited
co un t	10000 .0000 0	1.0000 00e+0 4	10000. 00000 0	10000. 00000 0	10000. 00000 0	10000. 000000	10000.0 00000	10000 .0000 0	10000.0 00000	10000.0 00000	10000. 00000 0

	RowN umbe r	Custo merId	Credit Score	Age	Tenur e	Balanc e	NumOf Product s	HasC rCar d	IsActive Membe r	Estimat edSalar y	Exited
m ea n	5000. 50000	1.5690 94e+0 7	650.52 8800	38.921 800	5.0128 00	76485. 889288	1.53020 0	0.705 50	0.51510 0	100090. 239881	0.2037 00
st d	2886. 89568	7.1936 19e+0 4	96.653 299	10.487 806	2.8921 74	62397. 405202	0.58165 4	0.455 84	0.49979 7	57510.4 92818	0.4027 69
mi n	1.000 00	1.5565 70e+0 7	350.00 0000	18.000 000	0.0000 00	0.0000 00	1.00000 0	0.000 00	0.00000 0	11.5800 00	0.0000 00
25 %	2500. 75000	1.5628 53e+0 7	584.00 0000	32.000 000	3.0000 00	0.0000 00	1.00000 0	0.000 00	0.00000 0	51002.1 10000	0.0000 00
50 %	5000. 50000	1.5690 74e+0 7	652.00 0000	37.000 000	5.0000 00	97198. 540000	1.00000 0	1.000 00	1.00000 0	100193. 915000	0.0000 00
75 %	7500. 25000	1.5753 23e+0 7	718.00 0000	44.000 000	7.0000 00	127644 .24000 0	2.00000 0	1.000 00	1.00000 0	149388. 247500	0.0000 00
m ax	10000 .0000 0	1.5815 69e+0 7	850.00 0000	92.000 000	10.000 000	250898 .09000 0	4.00000 0	1.000 00	1.00000 0	199992. 480000	1.0000 00

5.Handle the Missing values.

```
data = pd.read_csv("Churn_Modelling.csv")
pd.isnull(data["Gender"])
```

Out[]:

```
0      False
1      False
2      False
3      False
4      False
...
9995   False
9996   False
9997   False
9998   False
9999   False
Name: Gender, Length: 10000, dtype: bool
```

6. Find the outliers and replace the outliers.

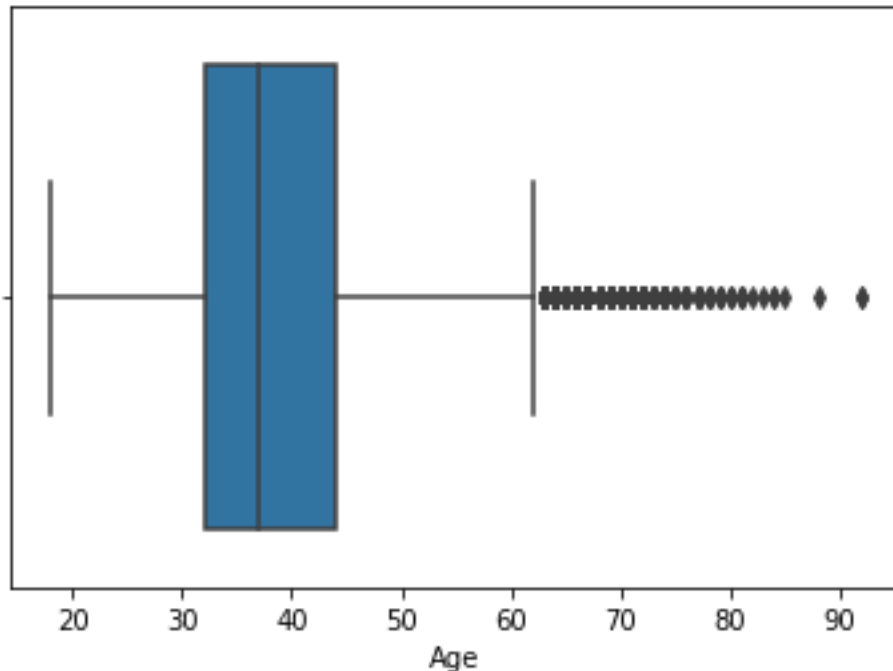
```
sns.boxplot(df['Age'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
```

FutureWarning

Out[]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f2932b8c650>
```



In []:

```
df['Age']=np.where(df['Age']>50,40,df['Age'])
df['Age']
```

Out[]:

```
0      42
1      41
2      42
3      39
4      43
..
9995   39
9996   35
9997   36
9998   42
9999   28
Name: Age, Length: 10000, dtype: int64
```

In []:

```
sns.boxplot(df['Age'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
```

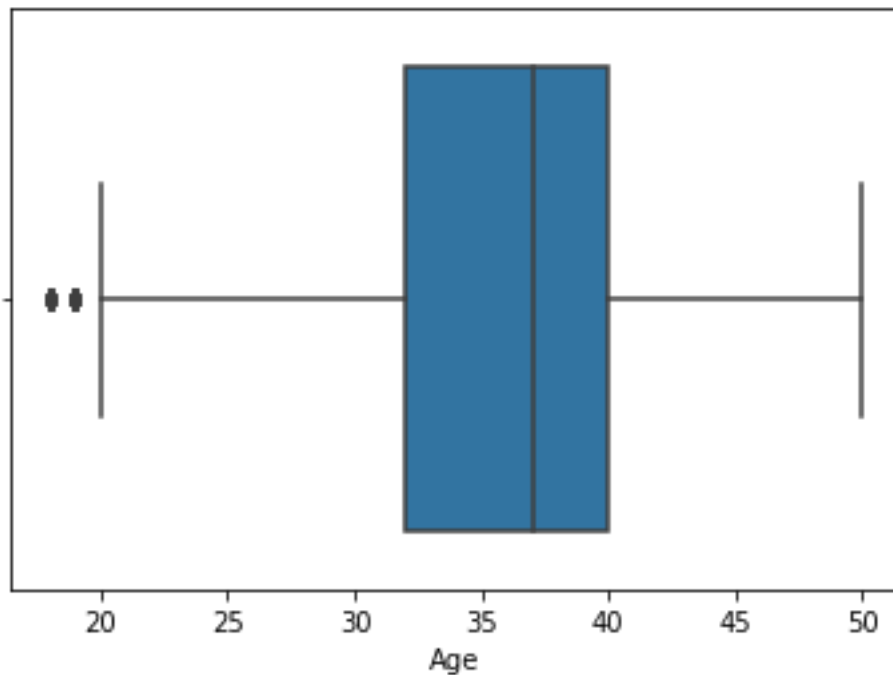
ts without an explicit keyword will result in an error or misinterpretation

.

FutureWarning

Out[]:

<matplotlib.axes._subplots.AxesSubplot at 0x7f2932a9a450>



In []:

```
df['Age']=np.where(df['Age']<20,35,df['Age'])
df['Age']
```

Out[]:

```
0      42
1      41
2      42
3      39
4      43
..
9995   39
9996   35
9997   36
9998   42
9999   28
Name: Age, Length: 10000, dtype: int64
```

7.Check for Categorical columns and perform encoding.

```
pd.get_dummies(df, columns=["Gender", "Age"], prefix=["Age",
"Gender"]).head()
```

Out[]:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Tenure	Balance	NumOfProducts	HasCrCard
0	1	15634602	Hargrave	619	France	2	0.00	1	1

	RowNumber	CustomerId	Surname	CreditScore	Geography	Tenure	Balance	NumOfProducts	HasCrCard
1	2	15647311	Hill	608	Spain	1	83807.86	1	0
2	3	15619304	Onio	502	France	8	159660.80	3	1
3	4	15701354	Boni	699	France	1	0.00	2	0
4	5	15737888	Mitchell	850	Spain	2	125510.82	1	1

5 rows × 45 columns

8.Split the data into dependent and independent variables.

a) Split the data into Independent variables.

```
X = df.iloc[:, :-1].values
print(X)
[[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]]
```

b)Split the data into Dependent variables.

```
Y = df.iloc[:, -1].values
print(Y)
[1 0 1 ... 1 1 0]
```

9.Scale the independent variables

```
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
df[["CustomerId"]] = scaler.fit_transform(df[["CustomerId"]])
```

In []:

```
print(df)
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age
\							
0	1	0.275616	Hargrave	619	France	Female	42
1	2	0.326454	Hill	608	Spain	Female	41
2	3	0.214421	Onio	502	France	Female	42
3	4	0.542636	Boni	699	France	Female	39
4	5	0.688778	Mitchell	850	Spain	Female	43
...
9995	9996	0.162119	Obijiaku	771	France	Male	39
9996	9997	0.016765	Johnstone	516	France	Male	35
9997	9998	0.075327	Liu	709	France	Female	36

9998	9999	0.466637	Sabbatini	772	Germany	Male	42
9999	10000	0.250483	Walker	792	France	Female	28

	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	\
0	2	0.00	1	1	1	
1	1	83807.86	1	0	1	
2	8	159660.80	3	1	0	
3	1	0.00	2	0	0	
4	2	125510.82	1	1	1	
...	
9995	5	0.00	2	1	0	
9996	10	57369.61	1	1	1	
9997	7	0.00	1	0	1	
9998	3	75075.31	2	1	0	
9999	4	130142.79	1	1	0	

	EstimatedSalary	Exited
0	101348.88	1
1	112542.58	0
2	113931.57	1
3	93826.63	0
4	79084.10	0
...
9995	96270.64	0
9996	101699.77	0
9997	42085.58	1
9998	92888.52	1
9999	38190.78	0

[10000 rows x 14 columns]

10.Split the data into training and testing

```
from sklearn.model_selection import train_test_split
train_size=0.8
X = df.drop(columns = ['Tenure']).copy()
y = df['Tenure']
X_train, X_rem, y_train, y_rem = train_test_split(X,y, train_size=0.8)
test_size = 0.5
X_valid, X_test, y_valid, y_test = train_test_split(X_rem,y_rem,
test_size=0.5)
print(X_train.shape), print(y_train.shape)
print(X_valid.shape), print(y_valid.shape)
print(X_test.shape), print(y_test.shape)

(8000, 13)
(8000,)
(1000, 13)
(1000,)
(1000, 13)
(1000,)
```

(None, None)

Out[]:

