Assignment -II

Name	**SANGEETHA M**
Register Number	**820319106018**
Team Size	**4**
Team ID	**PNT2022TMID46416**

1) Importing

In [ ]:

```
import pandas as pd import numpy
as np import seaborn as sns from
matplotlib import pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

2.Load the Dataset

In [ ]:

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

In [43]: data

Out[43]:

	RowNum	CustomerId	Surname	CreditScore	Geography	Gend	Age	Tenure	Balance	NumOfProduc	HasCrCar	IsActiveMemb	EstimatedSalary	Exite
0	1	0.275616	Hargrave	619	France	Female	42	2	0.00	1	1	1	101348.88	1

1	2	0.326454	Hill	608	Spain	Female	41	1	83807.86	1	0	1	112542.58	0
2	3	0.214421	Onio	502	France	Female	42	8	159660.80	3	1	0	113931.57	1
3	4	0.542636	Bondi	699	France	Female	39	1	0.00	2	0	0	93826.63	0
4	5	0.688778	Mitchell	850	Spain	Female	43	2	125510.82	1	1	1	79084.10	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9995	9996	0.162119	Obijiaku	771	France	Male	39	5	0.00	2	1	0	96270.64	0
9996	9997	0.016765	Johnston e	516	France	Male	35	10	57369.61	1	1	1	101699.77	0
9997	9998	0.075327	Liu	709	France	Female	36	7	0.00	1	0	1	42085.58	1

9998	9999	0.466637	Sabbatini	772	Germany	Male	42	3	75075.31	2	1	0	92888.52	1
9999	10000	0.250483	Walker	792	France	Female	28	4	130142.79	1	1	0	38190.78	0

10000 rows × 14 columns

3.Visualizations

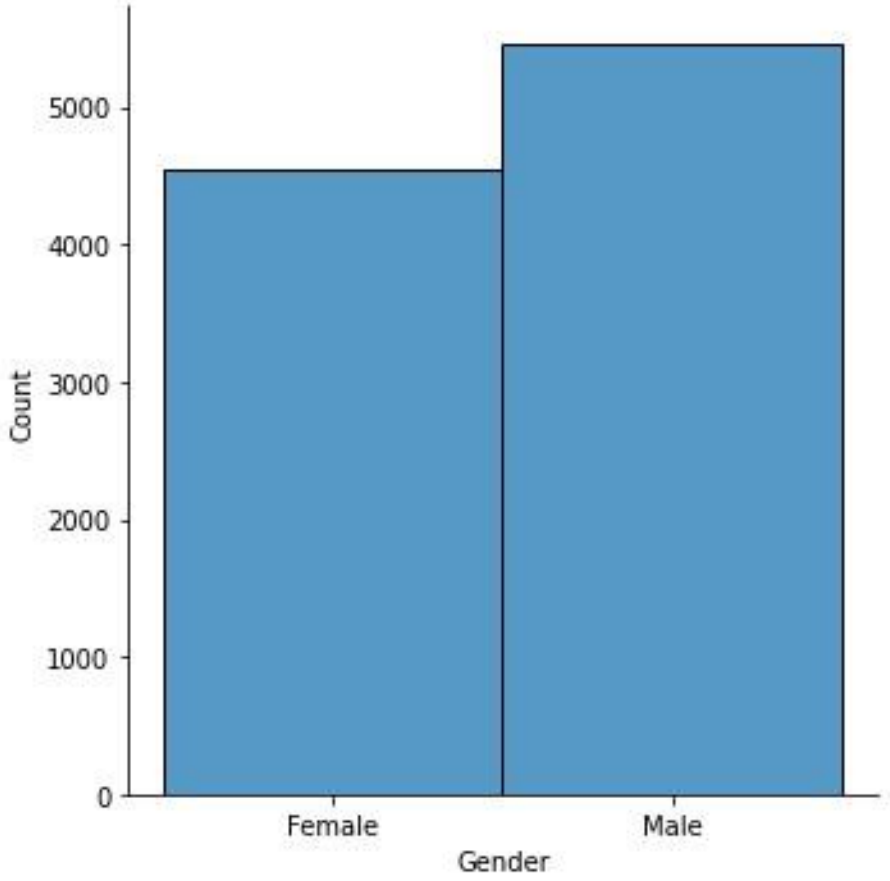
a) Univariate Analysis

In [44]:

```
sns.displot(data.Gender)
```

Out[44]:

<seaborn.axisgrid.FacetGrid at 0x7f80cb07c690>

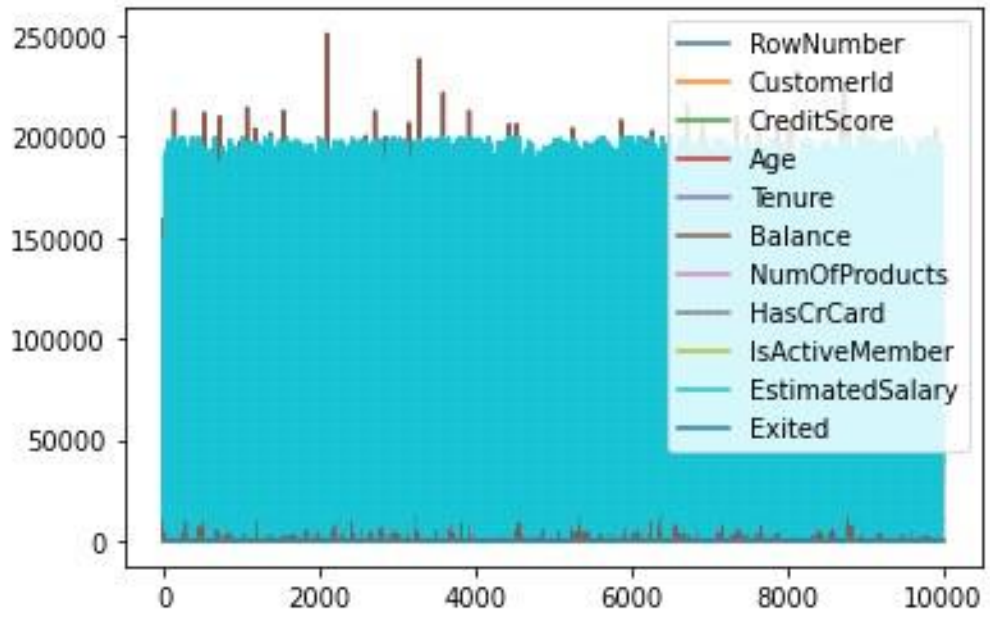


B)Bi-Variate Analysis

In [45]: data.plot.line()

Out[45]:

<matplotlib.axes._subplots.AxesSubplot at 0x7f80cb9a8a50>

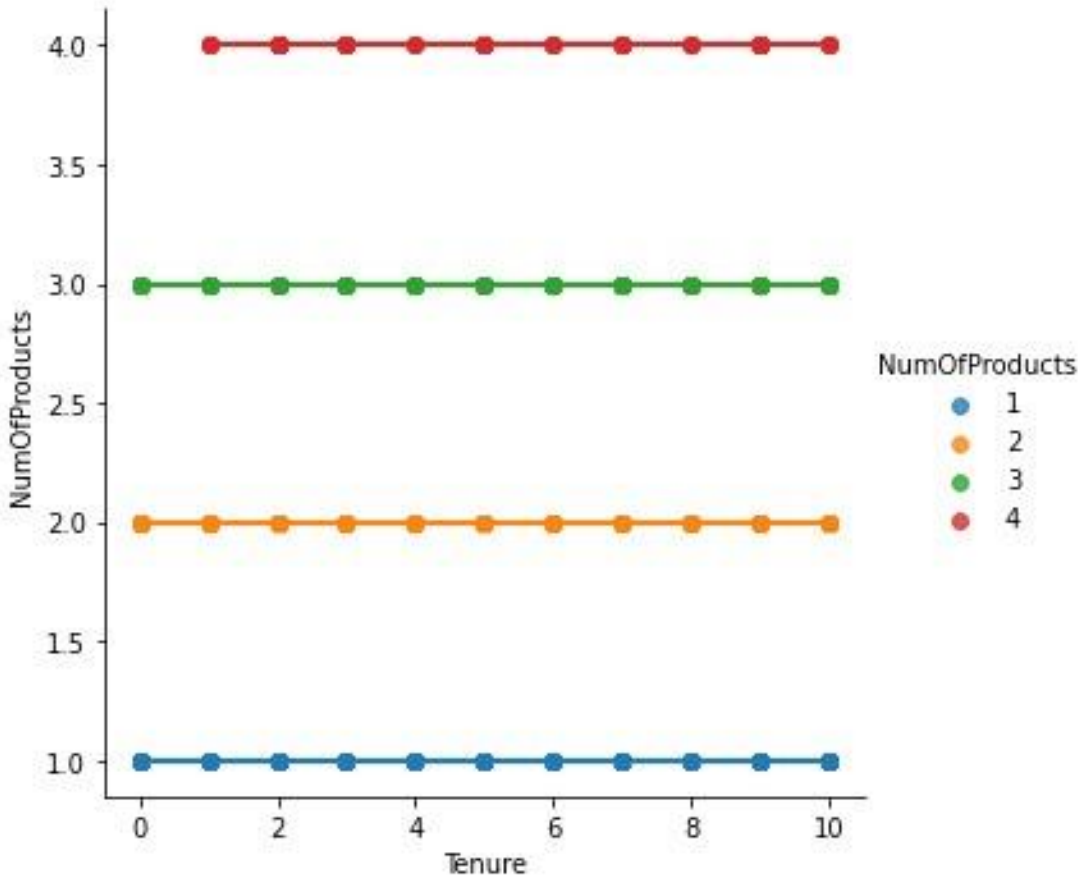


C)Multi - Variate Analysis

In [46]: sns.lmplot("Tenure","NumOfProducts",data,hue="NumOfProducts")

Out[46]:

<seaborn.axisgrid.FacetGrid at 0x7f80cb95fe10>



4)Perform descriptive statistics on the dataset.

In [47]: data.describe()

Out[47]:

	RowNumber	CustomerId	Credit Score	Age	Tenure	Balance	NumOf Products	HasCrCard	IsActive Member	EstimatedSalary	Exited
count	10000.00000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.00000	10000.000000	10000.000000	10000.000000

mean	5000.5	0.5009	650.52	36.533	5.0128	76485.8	1.53020	0.7055	0.515100	100090.2	0.2037
n	0000	80	8800	900	00	89288	0	0		39881	00
std	2886.8 9568	0.2877 57	96.6 53 299	6.4738 43	2.8921 74	62397.4 05202	0.58165 4	0.4558 4	0.499797	57510.49 2818	0.4027 69
min	1.0000 0	0.0000 00	35000 000(	20.000 000	0.0000 00	0.00000 0	1.00000 0	0.0000 0	0.000000	11.58000 0	0.0000 00
25 %	2500.7 5000	0.2513 20	58400 000(	32.000 000	3.0000 00	0.00000 0	1.00000 0	0.0000 0	0.000000	51002.11 0000	0.0000 00
50 %	5000.5 0000	0.5001 70	65200 000(	37.000 000	5.0000 00	97198.5 40000	1.00000 0	1.0000 0	1.000000	100193.9 15000	0.0000 00
75 %	7500.2 5000	0.7501 64	71800 000(	40.000 000	7.0000 00	127644. 240000	2.00000 0	1.0000 0	1.000000	149388.2 47500	0.0000 00
max	10000. 00000	1.0000 00	85000 000(	50.000 000	10.000 000	250898. 090000	4.00000 0	1.0000 0	1.000000	199992.4 80000	1.0000 00

5)Handle the Missing values.

In [ ]:

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

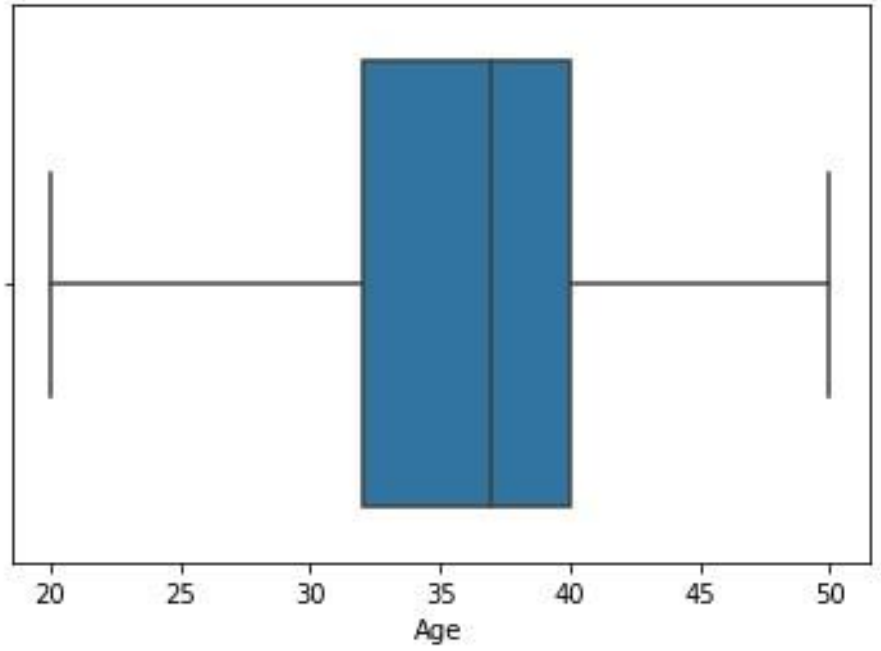
the outliers and replace the outliers

In [48]:

sns.boxplot(data['Age'])

Out[48]:

<matplotlib.axes._subplots.AxesSubplot at 0x7f80caeafc50>



In [28]: data['Age']=np.where(data['Age']>50,40,data['Age'])

data['Age']

Out[28]:

0 42

1 41

2 42

3 39

4 43

..

9995 39

9996 35

9997 36

9998 42

9999 28

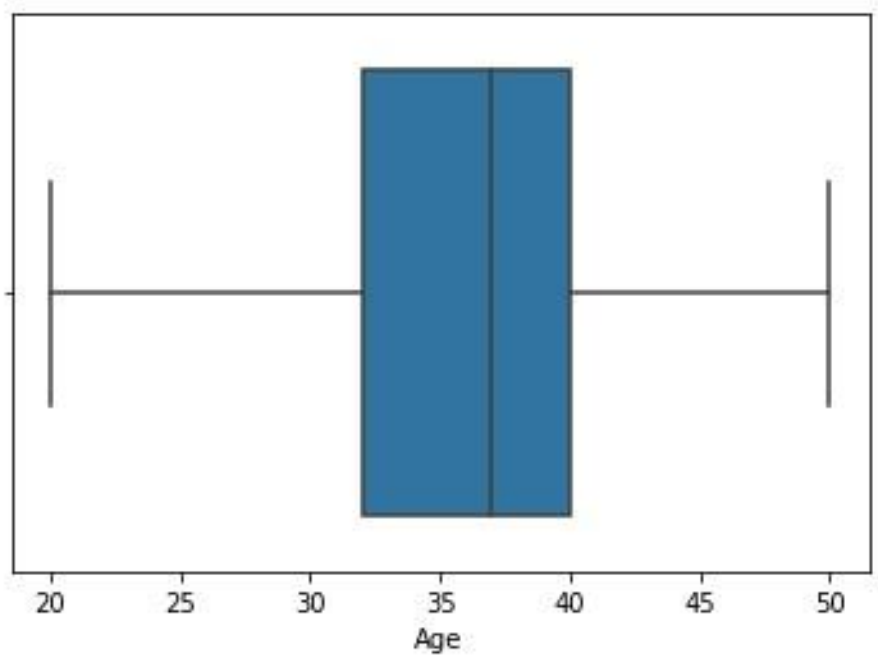
Name: Age, Length: 10000, dtype: int64

In [49]:

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

Out[49]:

<matplotlib.axes._subplots.AxesSubplot at 0x7f80cb95fc10>



In [34]:

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

```
data['Age']
```

Out[34]:

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.

In [50]:

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

Out[50]:

	RowNumber	**CustomerId**	**Surname**	**CreditScore**	**Geography**	**Tenure**	**Balance**	**NumOfProducts**	**HasCrCard**	**IsActiveMember**	**...**	**Gender_41**	**Gender_42**	**Gender_43**	**Gender_44**	**Gender_45**	**Gender_46**	**Gender_47**	**Gender_48**	**Gender_49**	**Gender_50**
0	1	0.27	Har	619	Fran	2	0.0	1	1	1	..	0	1	0	0	0	0	0	0	0	0

		5616	grave		ce		0				.										

1	2	0.3264 54	Hill	608	Spain	1	838 07.86	1	0	1	..	1	0	0	0	0	0	0	0	0	0
2	3	0.2144 21	Onio	502	France	8	159 660 0.80	3	1	0	..	0	1	0	0	0	0	0	0	0	0
3	4	0.5426 36	Boni	699	France	1	0.00	2	0	0	..	0	0	0	0	0	0	0	0	0	0
4	5	0.6887 78	Mitchel l	850	Spain	2	125 51 0.82	1	1	1	..	0	0	1	0	0	0	0	0	0	0

5 rows × 45 columns

8) Split the data into dependent and independent variables.

A) Split the data into Independent variables.

In [37]:

X = data.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.

In [38]:

Y = data.iloc[:, -1].values print(Y)

[1 0 1 ... 1 1 0]

9) Scale the independent variables

In [39]:

import pandas **as** pd **from** sklearn.preprocessing **import** MinMaxScaler scaler = MinMaxScaler()

data[["CustomerId"]] = scaler.fit_transform(data[["CustomerId"]])

In [40]: print(data)

	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

In [42]:

```
from sklearn.model_selection import train_test_split train_size=0.8

X = data.drop(columns = ['Tenure']).copy() y

= data['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,)
```

Out[42]:

```
(None, None)
```