Project Name	AI-powered Nutrition Analyzer for fitness Enthusiasts
Name	YUVARAJAN.R
Roll no	620619104045
Team ID	PNT2022TMID41475

Importing Package

```
from
google.colab
import drive
drive.mount('
/content/drive
')
```

```
impor
t
panda
s as
pd
impor
t
seabo
rn as
snsim
port
nump
y as
np
from matplotlib import pyplot as ply
%matplotlib inline
```

1.Loading dataset

```
CreditScoreGeographyAgedf =pd.read_csv("/content/Churn_Modelling.csv")
```

df							
RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenur

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	

10000 rows × 14 columns

Visualization

a) Univariate analysis

sns.displot (df.Gender)

<seaborn.axisgrid.FacetGrid at 0x7fa2127ec990>

df.plot.line()

<matplotlib.axes._subplots.AxesSubplot at 0x7fa21262e890>

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

CreditScoreAge**/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning FutureWarning**

df.describe()	**RowNumber**	**CustomerId**	**CreditScore**	**Age**	**Tenure**	**Balanc**
count	10000.00000	10000.000000	10000.000000	10000.000000	10000.000000	10000.00000
mean	5000.50000	0.500980	650.528800	36.533900	5.012800	76485.88928
std	2886.89568	0.287757	96.653299	6.473843	2.892174	62397.40520
min	1.00000	0.000000	350.000000	20.000000	0.000000	0.00000
25%	2500.75000	0.251320	584.000000	32.000000	3.000000	0.00000

50%	5000.50000	0.500170	652.000000	37.000000	5.000000	97198.54000
75%	7500.25000	0.750164	718.000000	40.000000	7.000000	127644.24000
max	10000.00000	1.000000	850.000000	50.000000	10.000000	250898.09000

Handle the missing values

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

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**

Find the outliers and replace the outliers

sns.boxplot(df['Age'])

/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning
FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7fa21390b290>

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

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

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7fa213879fd0>
```

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

Check for categorical Columns and perform encoding

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

	RowNumbe	CustomerI	Surnam	CreditScor	Geograph	Tenur	Balance	Nu

r		d	e	e	y	e		m
0	1	0.275616	Hargrave	619	France	2	0.00	
1	2	0.326454	Hill	608	Spain	1	83807.86	
2	3	0.214421	Onio	502	France	8	159660.80	
3	4	0.542636	Boni	699	France	1	0.00	
4	5	0.688778	Mitchell	850	Spain	2	125510.82	

5 rows × 45 columns

Split the data into dependent and independent Variables

- Split the data into independent Variables

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

```
[[1 0.2756161271095934 'Hargrave' ... 1 1 101348.88]
 [2 0.32645436399201344 'Hill' ... 0 1 112542.58]
 [3 0.21442143454311946 'Onio' ... 1 0 113931.57]
 ...
 [9998 0.07532731440183227 'Liu' ... 0 1 42085.58]
 [9999 0.4666365320074064 'Sabbatini' ... 1 0 92888.52]
 [10000 0.25048302125293276 'Walker' ... 1 0 38190.78]]
```

- Split the data into dependent Variables

```
Y =
df.iloc[:, -
1].valuespr
int(Y)
```

```
[1 0 1 ...
 1 1 0]
```

Scale the independent Variables

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

Split the data into training and testing

```
from sklearn.model_selection import
train_test_splittrain_size=0.8X = 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)
```