

ASSIGNMENT -2
Python
Programming

Team ID	PNT2022TMID14463
Project Name	Real Time Communication System Powered By AI For Specially Abled
Roll No	711319EC118

Question-1 :

1 . Importing Required Package

Solution :

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

Question-2 :

2. Loading the Dataset

Solution :

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

df
```

Output:



RowNumber	CustomerID	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
0	1	Hargrave	615	France	Female	42	2	0.00	1	1	1	101348.88	1
1	2	Hair	806	Spain	Female	41	1	83007.86	1	0	1	112542.56	0
2	3	Onio	302	France	Female	42	8	159660.80	3	1	0	113401.87	1
3	4	Boni	596	France	Female	39	1	0.00	2	0	0	93826.63	0
4	5	Mitchell	950	Spain	Female	43	2	125516.82	1	1	1	79004.10	0
...
9995	9996	Ojibaku	771	France	Male	38	5	0.00	2	1	0	96270.64	0
9996	9997	Jonnstone	516	France	Male	35	10	57369.61	1	1	1	101694.77	0
9997	9998	Liu	706	France	Female	36	7	0.00	1	0	1	42093.58	1
9998	9999	Sabbani	772	Germany	Male	42	3	75075.31	2	1	0	82890.52	1
9999	10000	Walker	792	France	Female	28	4	130142.79	1	1	0	38190.78	0

10000 rows x 14 columns

3. Visualizations

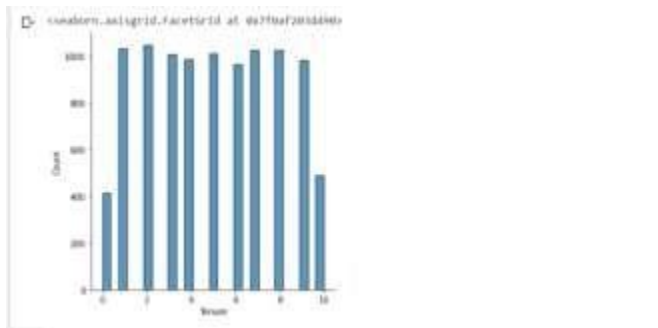
Question-3 :

3.1 Univariate Analysis

Solution:

```
sns.displot(df.Tenure)
```

Output:

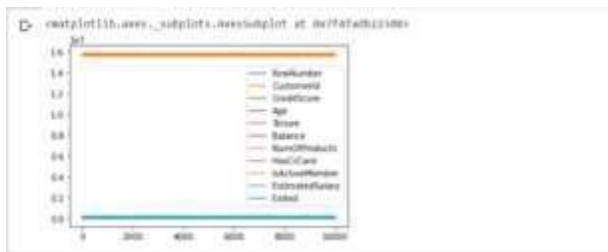


3.2 Bi-Variate Analysis

Solution:

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

Output:



3.3 Multi - Variate Analysis

Solution:

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

Output:



4. Perform descriptive statistics on the dataset.

Question-4 :

Solution:

```
df.describe()
```

Output:

	RouteNumber	CustomerID	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
count	10000.00000	1.000000e+04	10000.00000	10000.00000	10000.00000	10000.00000	10000.00000	10000.00000	10000.00000	10000.00000	10000.00000
mean	3000.50000	1.500000e+07	480.52800	38.52180	0.01200	76483.66098	1.53200	0.70000	0.51510	100000.23488	0.20370
std	2000.89500	7.183618e+04	98.65329	10.40700	2.892174	62397.40292	0.58184	0.45994	0.488797	57510.40258	0.402708
min	1.00000	1.556179e+07	355.00000	18.00000	0.00000	0.00000	1.00000	0.00000	0.00000	11.58000	0.00000
25%	2500.75000	1.562853e+07	384.00000	32.00000	3.00000	0.00000	1.00000	0.00000	0.00000	51002.10000	0.00000
50%	3000.50000	1.568074e+07	452.00000	37.00000	5.00000	87188.54000	1.00000	1.00000	1.00000	100101.81500	0.00000
75%	3500.25000	1.575323e+07	518.00000	44.00000	7.00000	127044.24000	2.00000	1.00000	1.00000	146188.24750	0.00000
max	10000.00000	1.587591e+07	850.00000	62.00000	10.00000	250000.00000	4.00000	1.00000	1.00000	199902.48000	1.00000

5. Handle the Missing values.

Question-5 :

Solution:

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

Output:



The output shows a series of 10000 boolean values, all of which are False, indicating that there are no missing values in the 'Gender' column. The output is truncated with '...' in the middle. The final line shows the name of the series as 'Gender', its length as 10000, and its dtype as bool.

Question-6:

6. Find the outliers and replace the outliers.

Solution:

```
df["Tenure"] = np.where(df["Tenure"] > 10, np.median(df["Tenure"]),
df["Tenure"])
```

Output:



The output shows the 'Tenure' column after replacing outliers. The values are mostly integers, with some values being 10, which is the median used for replacement. The output is truncated with '...' in the middle. The final line shows the name of the series as 'Tenure', its length as 10000, and its dtype as object.

Question-7 :

7. Check for Categorical columns and perform encoding.

Solution:

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

Output:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	...	Gender_78
0	1	15634002	Hargrave	619	France	2	0.00	1	1	1	...	0
1	2	15647311	Hill	608	Spain	1	83607.66	1	0	1	...	0
2	3	15619304	Onio	502	France	8	159680.80	3	1	0	...	0
3	4	15701354	Boni	699	France	1	0.00	2	0	0	...	0
4	5	15737800	Mitchell	650	Spain	2	125510.62	1	1	1	...	0

5 rows × 14 columns

Output:

	HasCrCard	IsActiveMember	...	Gender_78	Gender_79	Gender_80	Gender_81	Gender_82	Gender_83	Gender_84	Gender_85	Gender_86	Gender_92
	1	1	...	0	0	0	0	0	0	0	0	0	0
	0	1	...	0	0	0	0	0	0	0	0	0	0
	1	0	...	0	0	0	0	0	0	0	0	0	0
	0	0	...	0	0	0	0	0	0	0	0	0	0
	1	1	...	0	0	0	0	0	0	0	0	0	0

5 rows × 14 columns

Question-8:

1. Split the data into dependent and independent variables

1.Split the data into Independent variables.

Solution:

```
X = df.iloc[:, :-2].values
print(X)
```

Output:

```
[[1 15634682 'Hargrave' ... 1 1 1]
 [2 15647311 'Hill' ... 1 0 1]
 [3 15619384 'Orio' ... 1 1 0]
 ...
 [9998 15584532 'Liu' ... 1 0 1]
 [9999 15481355 'Sabbatini' ... 2 1 0]
 [10000 15628319 'Walker' ... 1 1 0]]
```

8.2 Split the data into Dependent variables.

Solution:

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

Output:

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

Question-9 :

9. Scale the independent variables

Solution:

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

Output:

	RealEstate	EstimatedValue	Country	Continent	Geography	Gender	Age	%
0	0.0000	1910000	Argentina	SA	France	Female	42	
1	0.0000	1560731	USA	NA	Spain	Female	41	
2	0.0000	1561900	Spain	EU	France	Female	42	
3	0.0000	1190134	Spain	EU	France	Female	39	
4	0.0000	1572700	Spain	EU	Spain	Female	41	
...								
9995	0.0000	1500429	Spain	EU	France	Male	39	
9996	0.0000	1550463	Spain	EU	France	Male	39	
9997	0.0000	1990431	USA	NA	France	Female	39	
9998	0.0000	1502135	Spain	EU	Germany	Male	42	
9999	1.0000	1562016	Spain	EU	France	Female	38	
...								
Tenure	Balance	NumOfProducts	HasACard	EstimatedValue	%			
0	2	0.00	1	1	1			
1	1	0.00	0	0	1			
2	4	0.00	0	1	0			
3	1	0.00	0	0	0			
4	2	0.00	0	1	0			
...								
9995	0	0.00	0	1	0			
9996	20	0.00	0	1	0			
9997	7	0.00	1	0	1			
9998	0	0.00	0	1	0			
9999	0	0.00	1	1	0			
...								
EstimatedValue	Balance							
0	101540.00							
1	112542.50							
2	113913.17							
3	91816.00							
4	70804.16							
...								
9995	90170.00							
9996	104100.77							
9997	43800.50							
9998	92840.52							
9999	90100.70							

[[0.0000, 0.0000] = 15 - 14.00000]

Question-10 :

10. Split the data into training and testing

Solution:

```
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)
```

Output:

```
(8000, 13)
(8000,)
(1000, 13)
(1000,)
(1000, 13)
(1000,)
(None, None)
```


TEAM LEADER: S.THARANI

TEAM MEMBERS:

1. K.V.SUNSHETHA

2. L.P.SINDHUJA

3. S.SRINIVASAN

TEAM ID :PNT2022TMID14463