

ASSIGNMENT – 2

Python Programming

Assignment Date	25-09-2022
Student Name	SHYAM SUNDAR B
Student Roll Number	813819106093
Maximum Marks	2 Mark

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	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	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	Sabbatini	772	Germany	Male	42	3	75075.31	2	1	0	92886.52	1
9999	10000	15628319	Walker	792	France	Female	28	4	130142.79	1	1	0	38190.76	0

10000 rows × 14 columns

3. Visualizations

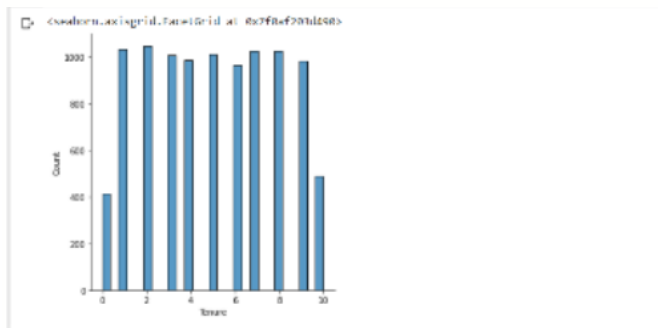
Question-3 :

3.1 Univariate Analysis

Solution:

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

Output:

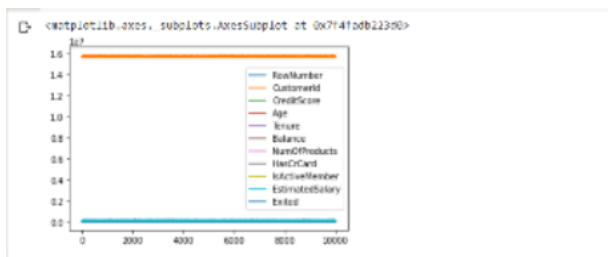


3.2 Bi-Variate Analysis

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:

	Birthdate	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	5000.00000	1.500000e+07	650.00000	38.00000	5.00000	76405.00000	1.50000	0.70000	0.50000	100000.00000	0.50000
std	2880.00000	1.190000e+06	90.00000	10.00000	2.00000	52300.00000	0.50000	0.40000	0.40000	50000.00000	0.50000
min	1.00000	1.000000e+07	300.00000	18.00000	0.00000	0.00000	1.00000	0.00000	0.00000	10000.00000	0.00000
25%	2500.00000	1.000000e+07	500.00000	32.00000	3.00000	0.00000	1.00000	0.00000	0.00000	50000.00000	0.00000
50%	5000.00000	1.500000e+07	650.00000	38.00000	5.00000	76405.00000	1.50000	0.70000	0.50000	100000.00000	0.50000
75%	7500.00000	1.500000e+07	750.00000	44.00000	7.00000	120000.00000	2.00000	1.00000	1.00000	140000.00000	0.00000
max	10000.00000	1.500000e+07	850.00000	50.00000	10.00000	250000.00000	4.00000	1.00000	1.00000	150000.00000	1.00000

5. Handle the Missing values.

Question-5 :

Solution:

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

Output:

```
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
```

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:

```
0      2
1      1
2      8
3      1
4      2
...
9995    5
9996   10
9997    7
9998    3
9999    4
Name: Tenure, Length: 10000, dtype: 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	15634602	Hargrave	619	France	2	0.00	1	1	1	...	0
1	2	15647311	Hill	608	Spain	1	83807.86	1	0	1	...	0
2	3	15619304	Onio	502	France	8	159660.80	3	1	0	...	0
3	4	15701354	Boni	699	France	1	0.00	2	0	0	...	0
4	5	15737888	Mitchell	850	Spain	2	125510.82	1	1	1	...	0

5 rows × 84 columns

Output:

	HasCrCard	IsActiveMember	...	Gender_78	Gender_79	Gender_80	Gender_81	Gender_82	Gender_83	Gender_84	Gender_85	Gender_88	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

Question-8:

8. Split the data into dependent and independent variables

8.1 Split the data into Independent variables.

Solution:

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

Output:

```
[[1 15634602 'Hargrave' ... 1 1 1]
 [2 15647311 'Hill' ... 1 0 1]
 [3 15619304 'Onio' ... 3 1 0]
 ...
 [9998 15584532 'Liu' ... 1 0 1]
 [9999 15682355 '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:

```

RowNumber  CustomerId  Surname  CreditScore  Geography  Gender  Age  \
0          0.0000    15616602  Hargrave        619      France  Female  41
1          0.0001    15647311      Mill        608      Spain  Female  41
2          0.0002    15623084      Davis        582      France  Female  41
3          0.0003    15701354      Ford        690      France  Female  39
4          0.0004    15737888  Mitchell        850      Spain  Female  43
...
9995      0.9996    15606129  Olijakovic        771      France  Male    39
9996      0.9997    15580862  Johnson        515      France  Male    35
9997      0.9998    15580512      Liu        780      France  Female  35
9998      0.9999    15682155  Sahbatkhani        772      Germany  Male    42
9999      1.0000    15628319  Walker        792      France  Female  28

Tenure  Balance  NumOfProducts  HasCrCard  IsActiveMember  \
0        2      0.00            1           1           1
1        1  81807.05            1           0           1
2        0  259650.00            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  57365.61            1           1           1
9997      7      0.00            1           0           1
9998      3  71875.31            2           1           0
9999      4  130142.79            1           1           0

EstimatedSalary  Exited
0      101348.88      1
1      117547.56      0
2      111931.57      1
3      93826.63      0
4      75084.10      0
...
9995      96270.54      0
9996     101699.77      0
9997     42081.58      1
9998     61868.52      1
9999     50190.78      0
[10000 rows x 14 columns]
```

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,)
(1000,)
(None, None)
```