Assignment -2

Python Programming

Assignment Date	27 September 2022
Student Name	Surya P
Student Roll Number	412519104147
Maximum Marks	2 Marks

Question-1:

Download dataset

Solution:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
```

```
df=pd.read_csv('/Churn_Modelling.csv')
```

Question 2:

Dataset details

Solution:

```
df.head()
df.describe()
df.info()
```

```
[ ] df=pd.read_csv('/Churn_Modelling.csv')
```

Dataset Summary

```
[ ] df.head()
```

	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

```
[ ] df.describe()
```

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
count	10000.00000	1.000000e+04	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.00000	10000.000000	10000.000000	10000.000000
mean	5000.50000	1.569094e+07	650.528800	38.921800	5.012800	76485.889288	1.530200	0.70550	0.515100	100090.239881	0.203700
std	2886.89568	7.193619e+04	96.653299	10.487806	2.892174	62397.405202	0.581654	0.45584	0.499797	57510.492818	0.402769
min	1.00000	1.556570e+07	350.000000	18.000000	0.000000	0.000000	1.000000	0.00000	0.000000	11.580000	0.000000

```
[ ] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 14 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   RowNumber        10000 non-null  int64
 1   CustomerId       10000 non-null  int64
 2   Surname          10000 non-null  object
 3   CreditScore      10000 non-null  int64
 4   Geography        10000 non-null  object
 5   Gender           10000 non-null  object
 6   Age              10000 non-null  int64
 7   Tenure           10000 non-null  int64
 8   Balance          10000 non-null  float64
 9   NumOfProducts    10000 non-null  int64
 10  HasCrCard        10000 non-null  int64
 11  IsActiveMember   10000 non-null  int64
 12  EstimatedSalary  10000 non-null  float64
 13  Exited           10000 non-null  int64
dtypes: float64(2), int64(9), object(3)
memory usage: 1.1+ MB
```

Question 3:

Plot of dataset details and etc

Solution:

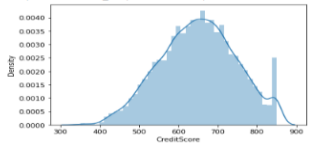
```
sns.distplot(df.Age)

sns.distplot(df.CreditScore)
```

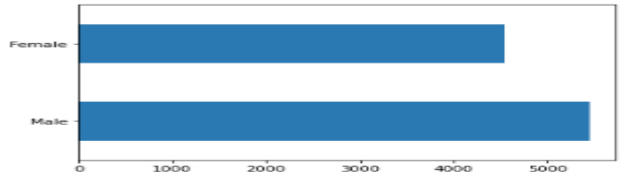
```
[ ] df.head(2)
```

	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

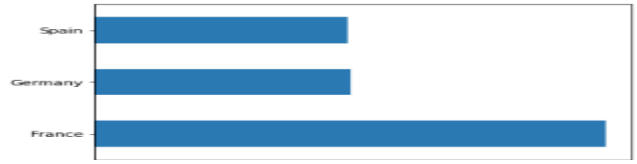
```
[ ] sns.distplot(df.CreditScore)
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f84a5eefd0>
```




```
[ ] df.Gender.value_counts().plot(kind='barh')
<matplotlib.axes._subplots.AxesSubplot at 0x7f84a33fd210>
```




```
[ ] df.Geography.value_counts().plot(kind='barh')
<matplotlib.axes._subplots.AxesSubplot at 0x7f84a338e4d0>
```



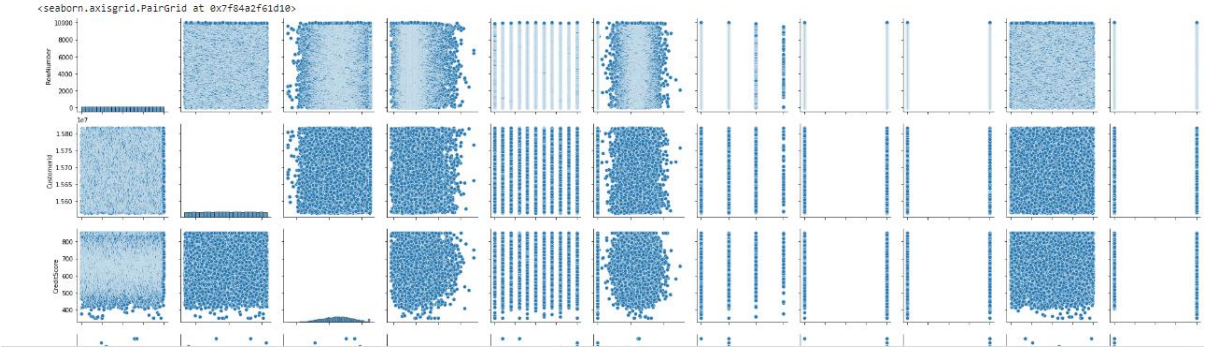
Question 5:

dataset pairplot

Solution:

sns.pairplot(df)

```
[ ] sns.pairplot(df)
<seaborn.axisgrid.PairGrid at 0x7f84a2f61d10>
```



Question 6:

Heat map of dataset features

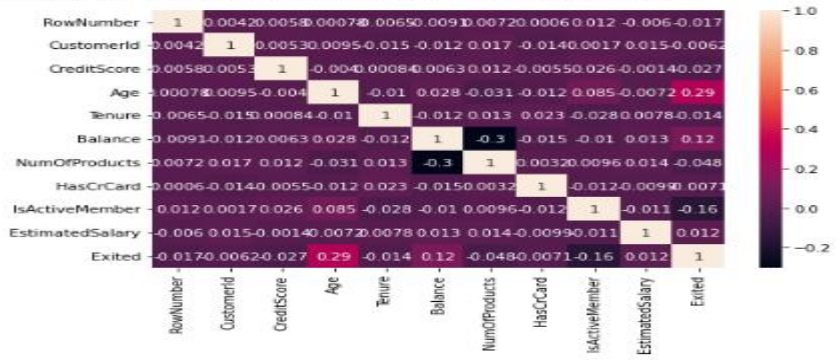
Solution:

plt.figure(figsize=(8,5))

sns.heatmap(df.corr(),annot=True)

```
[ ]  plt.figure(figsize=(8,5))
     sns.heatmap(df.corr(),annot=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f84a20abb90>
```



Question 7:

Explotary data analysis

Solution:

df.Exited.value_counts()

df.isnull().sum()

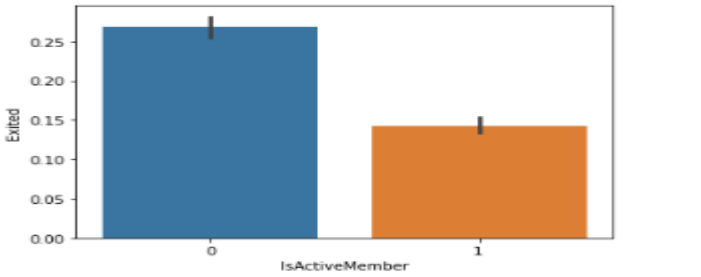
df.head(2)

```
[ ]  df.Exited.value_counts()
```

```
0    7963
1    2037
Name: Exited, dtype: int64
```

```
[ ]  sns.barplot(x='IsActiveMember',y='Exited',data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f84a3100cd0>
```



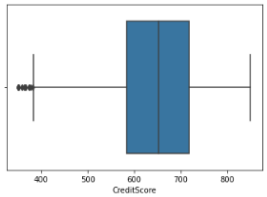
Question 8:

Boxplot

Solution:

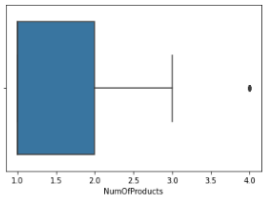
```
sns.boxplot(df.CreditScore)
```

```
/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`,
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f849baff4d0>
```



```
[ ] sns.boxplot(df.NumOfProducts)
```

```
/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`,
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f849ba48790>
```



Question 9:
dataset info

solution:

df.info()

```
[ ] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 14 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   RowNumber        10000 non-null  int64  
 1   CustomerId       10000 non-null  int64  
 2   Surname          10000 non-null  object 
 3   CreditScore      10000 non-null  int64  
 4   Geography        10000 non-null  object 
 5   Gender           10000 non-null  object 
 6   Age              10000 non-null  int64  
 7   Tenure           10000 non-null  int64  
 8   Balance          10000 non-null  float64
 9   NumOfProducts    10000 non-null  float64
 10  HasCrCard        10000 non-null  int64  
 11  IsActiveMember   10000 non-null  int64  
 12  EstimatedSalary  10000 non-null  float64
 13  Exited           10000 non-null  int64  
dtypes: float64(3), int64(8), object(3)
memory usage: 1.1+ MB
```

Question 10:
Preprocessing

Solution:

from sklearn.preprocessing import LabelEncoder

le_geo = LabelEncoder()

le_gen = LabelEncoder()

df['Sex']=le_gen.fit_transform(df.Gender)

df['Country']=le_geo.fit_transform(df.Geography)

df.drop(['Geography','Gender'],axis=1,inplace=True)

```
[ ] from sklearn.preprocessing import LabelEncoder
    le_geo = LabelEncoder()
    le_gen = LabelEncoder()
    df['Sex']=le_gen.fit_transform(df.Gender)
    df['Country']=le_geo.fit_transform(df.Geography)
    df.drop(['Geography','Gender'],axis=1,inplace=True)
```

Question 11:

Preprocessing 1

Solution:

from sklearn.preprocessing import StandardScaler

sc=StandardScaler()

X = sc.fit_transform(X)

```
[ ] from sklearn.preprocessing import StandardScaler
    sc=StandardScaler()
    X = sc.fit_transform(X)
```

```
[ ] from sklearn.model_selection import train_test_split
    x_train,x_test,y_train,y_test=train_test_split(X,y,test_size=0.2,
                                                   random_state=42)
```

```
[ ] x_train.shape, x_test.shape, y_train.shape, y_test.shape

    ((8000, 10), (2000, 10), (8000,), (2000,))
```