

Assignment -2

Assignment Date	29 September 2022
Student Name	Saranya A
Student Roll Number	310819104075
Maximum Marks	2 Marks

PDF LINK: [Assignment_2_saranya.pdf](#)

```
In [ ]:
import numpy as np
import pandas as pd

In [ ]:
# (2) loading dataset
df=pd.read_csv("/content/Churn_Modelling.csv")

In [ ]:
df.head()

Out[ ]:

```

RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard
0	1	Hargrave	619	France	Female	42	2	0.00		1
1	2	Hill	608	Spain	Female	41	1	83807.86		1
2	3	Onio	502	France	Female	42	8	159660.80		3
3	4	Boni	699	France	Female	39	1	0.00		2
4	5	Mitchell	850	Spain	Female	43	2	125510.82		1

```
In [ ]:
# (4) descriptive statistics on the dataset
df.describe()

Out[ ]:

```

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	Is
count	10000.000000	1.000000e+04	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000

```
mean    5000.500000  1.569094e+07  650.528800  38.921800  5.012800  76485.889268  1.530200  0.70550
std     2886.895568  7.193619e+04  96.653299  10.487806  2.892174  62397.405202  0.581654  0.45584
min      1.000000  1.566570e+07  350.000000  18.000000  0.000000  0.000000  1.000000  0.00000
25%    2500.750000  1.562853e+07  584.000000  32.000000  3.000000  0.000000  1.000000  0.00000
50%    5000.500000  1.569074e+07  652.000000  37.000000  5.000000  97198.540000  1.000000  1.00000
75%    7500.250000  1.575323e+07  718.000000  44.000000  7.000000  127644.240000  2.000000  1.00000
max   10000.000000  1.581589e+07  850.000000  92.000000  10.000000  250898.090000  4.000000  1.00000
```

```
In [ ]:
# (5) Handle the Missing values
df.isnull().sum()

Out[ ]:
RowNumber      0
CustomerId      0
Surname         0
CreditScore    0
Geography      0
Gender         0
Age            0
Balance        0
Tenure         0
NumOfProducts  0
HasCrCard      0
IsActiveMember 0
EstimatedSalary 0
Exited         0
dtype: int64

In [ ]:
# (6) finding outliers and replacing
```

```
df['Age'].mean()
Out[ ]:
38.9218

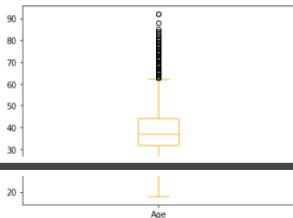
In [ ]:
df['Age'].median()
Out[ ]:
37.0

In [ ]:
df['Age'].std()
Out[ ]:
10.487806451704609

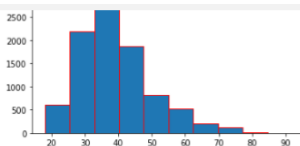
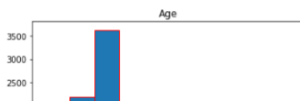
In [ ]:
df['Age'].value_counts()
Out[ ]:
37    478
38    477
35    474
36    456
34    447
...
92      2
82      1
88      1
85      1
83      1
Name: Age, Length: 70, dtype: int64

In [ ]:
```

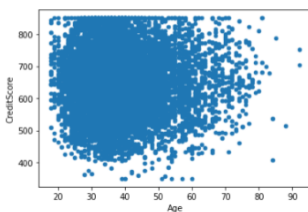
```
import matplotlib.pyplot as plt
df.boxplot(column=['Age'],grid=False, color='orange')
Out[ ]:
<matplotlib.axes._subplots.AxesSubplot at 0x7f0738ad0b10>
```



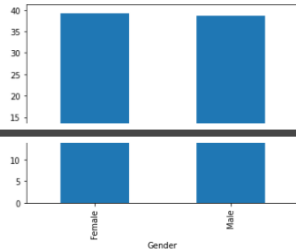
```
In [ ]:
# (7) Check for Categorical columns and perform encoding
df.hist(column='Age',grid=False,edgecolor='red')
Out[ ]:
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7f0738a2c490>]],
      dtype=object)
```



```
In [ ]:
df.plot.scatter('Age','CreditScore')
Out[ ]:
<matplotlib.axes._subplots.AxesSubplot at 0x7f073850ebd0>
```



```
In [ ]:
df.groupby('Gender')['Age'].mean().plot.bar()
Out[ ]:
<matplotlib.axes._subplots.AxesSubplot at 0x7f073848a9d0>
```



In []:

```
from sklearn import preprocessing
import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns
```

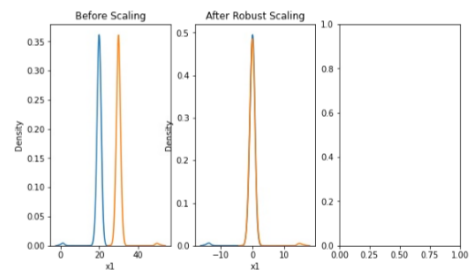
In []:

```
x = pd.DataFrame({
    'x1': np.concatenate([np.random.normal(20, 1, 2000), np.random.normal(1, 1, 20)]),
    'x2': np.concatenate([np.random.normal(30, 1, 2000), np.random.normal(50, 1, 20)]),
})
scaler = preprocessing.RobustScaler()
robust_scaled_df = scaler.fit_transform(x)
robust_scaled_df = pd.DataFrame(robust_scaled_df, columns=['x1', 'x2'])
fig, (ax1, ax2, ax3) = plt.subplots(ncols=3, figsize=(9, 5))
ax1.set_title('Before Scaling')
sns.kdeplot(x['x1'], ax=ax1)
sns.kdeplot(x['x2'], ax=ax1)
ax2.set_title('After Robust Scaling')
sns.kdeplot(robust_scaled_df['x1'], ax=ax2)
```

```
sns.kdeplot(robust_scaled_df['x1'], ax=ax2)
sns.kdeplot(robust_scaled_df['x2'], ax=ax2)
```

Out[]:

<matplotlib.axes._subplots.AxesSubplot at 0x7f072b91e850>



In []:

```
from sklearn.preprocessing import LabelEncoder
```

In []:

```
le = LabelEncoder()
```

In []:

```
from sklearn.model_selection import train_test_split
```

In []:

In []:

```
# (8) splitting of dependent and independent datas
```

```
x=df.iloc[:,0:8].values
y=df.iloc[:,8:15].values
```

In []:

```
# (10) splitting of data into training and testing
```

```
xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size=0.3, random_state=0)
```

In []:

```
# (9) Scale the independent variables
```

```
ytrain.shape, ytest.shape
```

Out[]:

```
((7000, 6), (3000, 6))
```