

Assignment -2

Python Programming

| | |
|-----------------|-------------------|
| Assignment Date | 19 September 2022 |
| Student Name | Pradeepa J |
| Maximum Marks | 2 Marks |

Question-1:

Download the dataset: Dataset

Solution:

https://drive.google.com/file/d/1_HcM0K8wt4b7FMLkc1V1dv0y6l_9ULzy/view

Question-2:

Load the dataset.

Solution:

```
# Read Dataset
data = pd.read_csv('/content/Churn_Modelling.csv')
data = data.iloc[:,3:]
data
```

| | CreditScore | Geography | Gender | Age | Tenure | Balance | NumOfProducts | HasCrCard | IsActiveMember | EstimatedSalary | Exited |
|------|-------------|-----------|--------|-----|--------|-----------|---------------|-----------|----------------|-----------------|--------|
| 0 | 619 | France | Female | 42 | 2 | 0.00 | 1 | 1 | 1 | 101348.88 | 1 |
| 1 | 608 | Spain | Female | 41 | 1 | 83807.86 | 1 | 0 | 1 | 112542.58 | 0 |
| 2 | 502 | France | Female | 42 | 8 | 159660.80 | 3 | 1 | 0 | 113931.57 | 1 |
| 3 | 699 | France | Female | 39 | 1 | 0.00 | 2 | 0 | 0 | 93826.63 | 0 |
| 4 | 850 | Spain | Female | 43 | 2 | 125510.82 | 1 | 1 | 1 | 79084.10 | 0 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 9995 | 771 | France | Male | 39 | 5 | 0.00 | 2 | 1 | 0 | 96270.64 | 0 |
| 9996 | 516 | France | Male | 35 | 10 | 57369.61 | 1 | 1 | 1 | 101699.77 | 0 |
| 9997 | 709 | France | Female | 36 | 7 | 0.00 | 1 | 0 | 1 | 42085.58 | 1 |
| 9998 | 772 | Germany | Male | 42 | 3 | 75075.31 | 2 | 1 | 0 | 92888.52 | 1 |
| 9999 | 792 | France | Female | 28 | 4 | 130142.79 | 1 | 1 | 0 | 38190.78 | 0 |

10000 rows × 11 columns

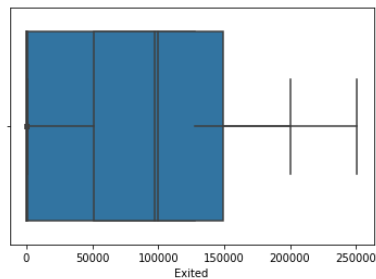
Question-3:

Perform Below Visualizations.

- Univariate Analysis
- Bi - Variate Analysis
- Multi - Variate Analysis

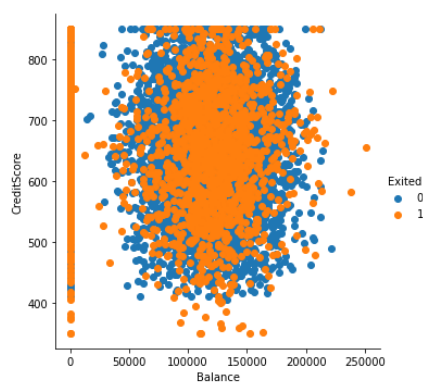
Solution:

```
# 1. Univariate Analysis
for col in data.columns:
    if data.dtypes[col]=='int64' or data.dtypes[col]=='float64':
        sns.boxplot(x=data[col]).set(xlabel=col)
```



```
# 2. Bi-Variate Analysis
sns.FacetGrid(data,hue='Exited',size=5).map(plt.scatter,"Balance","CreditScore").add_legend()
```

/usr/local/lib/python3.7/dist-packages/seaborn/axisgrid.py:337: UserWarning: The `size` parameter has been renamed to `height`; please update your code.
warnings.warn(msg, UserWarning)



Question-4:

Perform descriptive statistics on the dataset.

Solution:

```
data.describe()
```

| | CreditScore | Age | Tenure | Balance | NumOfProducts | HasCrCard | IsActiveMember | EstimatedSalary | Exited |
|-------|--------------|--------------|--------------|---------------|---------------|--------------|----------------|-----------------|--------------|
| count | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 |
| mean | 650.528800 | 38.921800 | 5.012800 | 76485.889288 | 1.530200 | 0.70550 | 0.515100 | 100090.239881 | 0.203700 |
| std | 96.653299 | 10.487806 | 2.892174 | 62397.405202 | 0.581654 | 0.45584 | 0.499797 | 57510.492818 | 0.402769 |
| min | 350.000000 | 18.000000 | 0.000000 | 0.000000 | 1.000000 | 0.000000 | 0.000000 | 11.580000 | 0.000000 |
| 25% | 584.000000 | 32.000000 | 3.000000 | 0.000000 | 1.000000 | 0.000000 | 0.000000 | 51002.110000 | 0.000000 |
| 50% | 652.000000 | 37.000000 | 5.000000 | 97198.540000 | 1.000000 | 1.000000 | 1.000000 | 100193.915000 | 0.000000 |
| 75% | 718.000000 | 44.000000 | 7.000000 | 127644.240000 | 2.000000 | 1.000000 | 1.000000 | 149388.247500 | 0.000000 |
| max | 850.000000 | 92.000000 | 10.000000 | 250898.090000 | 4.000000 | 1.000000 | 1.000000 | 199992.480000 | 1.000000 |

Question-5:

Handle the Missing values.

Solution:

```
data.isnull().sum()
```

```
CreditScore      0
Geography        0
Gender           0
Age              0
Tenure           0
Balance          0
NumOfProducts   0
HasCrCard        0
IsActiveMember   0
EstimatedSalary  0
Exited           0
dtype: int64
```

Question-6:

Find the outliers and replace the outliers

Solution:

```
credit = data.loc[data['CreditScore']<400, 'CreditScore'].median()
prod = data.loc[data['NumOfProducts']>=3.5, 'NumOfProducts'].median()
data.loc[data.CreditScore<400, 'CreditScore']=np.nan
data.fillna(credit,inplace=True)
data.loc[data.NumOfProducts>3, 'NumOfProducts']=np.nan
data.fillna(prod,inplace=True)
```

Question-7:

Check for Categorical columns and perform encoding.

Solution:

```
label = LabelEncoder()
data['Geography'] = label.fit_transform(data['Geography'])
data['Gender'] = label.fit_transform(data['Gender'])
```

Question-8:

Split the data into dependent and independent variables.

Solution:

```
dep = data.iloc[:,1:]
indep = data.iloc[:,0:-1]
```

Question-9:

Scale the independent variables

Solution:

```
indep_var = MinMaxScaler()
show_indep = indep_var.fit_transform(indep)
```

Question-10:

Split the data into training and testing

Solution:

```
xtrain,xtest,ytrain,ytest = train_test_split(show_indep, dep, test_size=0.3)
print(xtrain,xtest,ytrain,ytest)
```

```
[[0.69484536 1.      1.      ... 1.      0.      0.49826033]
 [0.49072165 0.5     1.      ... 1.      1.      0.46418978]
 [0.26597938 0.      1.      ... 1.      1.      0.12240099]
 ...
 [0.70515464 0.      1.      ... 1.      1.      0.38749191]
 [0.43092784 1.      0.      ... 0.      0.      0.92833226]
 [0.76494845 0.      1.      ... 0.      1.      0.4245851 ]] [[0.34845361 0.      0.      ... 1.      1.      0.35087061]
 [0.53195876 0.      1.      ... 1.      1.      0.33056902]
 [0.28865979 0.5     1.      ... 1.      1.      0.29006955]
 ...
 [0.83917526 0.5     1.      ... 1.      1.      0.60536606]
 [0.75670103 0.5     0.      ... 1.      0.      0.60426406]
 [0.66185567 0.5     0.      ... 0.      0.      0.39309194]]      Exited
```

```
1076      0
64         0
2908      0
4415      0
1154      0
...      ...
1079      0
9760      0
5048      0
2684      0
9081      0
```

```
[7000 rows x 1 columns]      Exited
442      0
3865      0
1217      0
3728      0
1827      1
...      ...
1392      0
6629      0
9590      0
9492      1
3211      0
```

```
[3000 rows x 1 columns]
```