

Assignment- 4

Python Programming

Team Id	PND2022TMID13322
Student Name	Puviyasree M
Maximum Marks	2 Marks

Question 1:

Download the dataset: Dataset

https://drive.google.com/file/d/1Z21e5HOZZR81sC_dnfCDPDMEzs-w8ysr/view?usp=sharing

Question 2:

Load the dataset.

```
In [47]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

Downloading and Loading the Dataset

```
In [3]: data = pd.read_csv('/content/Mall_Customers.csv')
data.head()
```

```
Out[3]:
```

	CustomerID	Gender	Age	Annual Income (k\$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

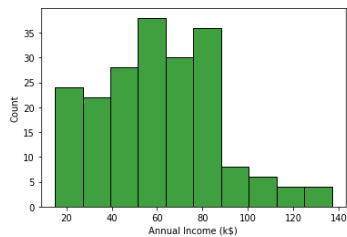
Question 3:

Perform Below Visualizations

- **Univariate Analysis**
- **Bivariate Analysis**
- **Multi-variate Analysis**

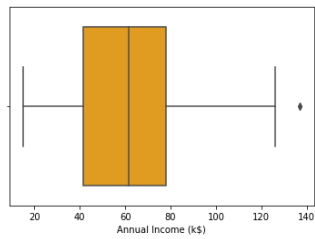
```
In [4]: sns.histplot(data['Annual Income (k$)'], color="green")
```

Out[4]:



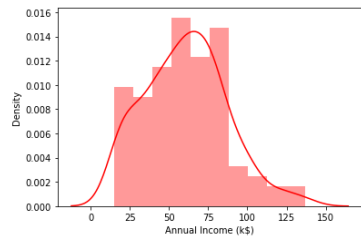
```
In [5]: sns.boxplot(data['Annual Income (k$)'], color="orange")
```

Out[5]:



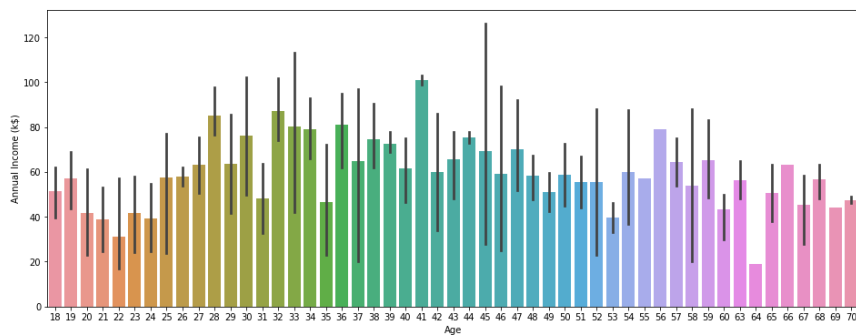
```
In [6]: sns.distplot(data['Annual Income (k$)'], color="red")
```

Out[6]:



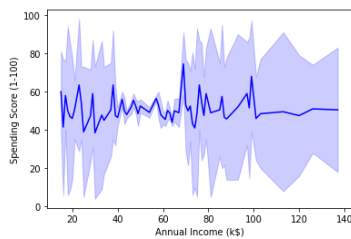
```
In [7]: plt.figure(figsize=(16,6))
sns.barplot(data['Age'],data['Annual Income (k$)'])
```

Out[7]:



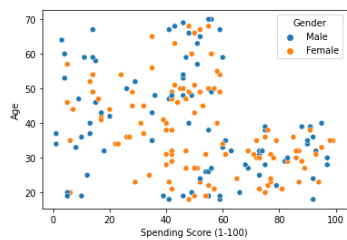
```
In [8]: sns.lineplot(data['Annual Income (k$)'], data['Spending Score (1-100)'], color="blue")
```

Out[8]:

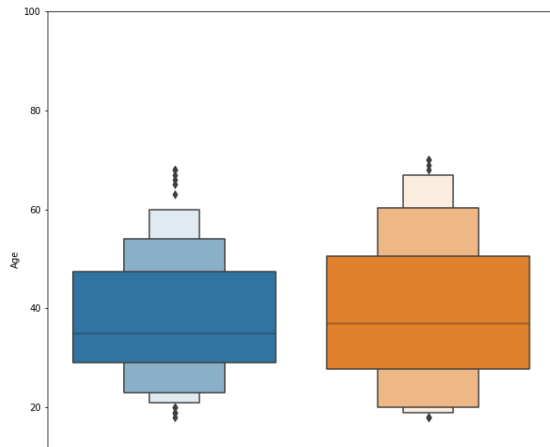


```
In [13]: sns.scatterplot(data['Spending Score (1-100)'], data['Age'], hue = data['Gender'])
```

Out[13]:

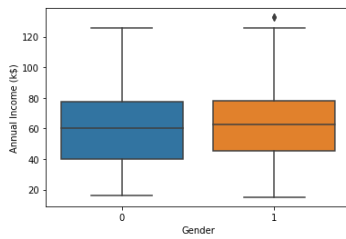


```
In [48]: temp = pd.concat([data['Age'], data['Gender']], axis=1)
f, ax = plt.subplots(figsize=(10,10))
fig = sns.boxplot(x='Gender', y='Age', data=temp)
fig.axis(ymin=0, ymax=100);
```



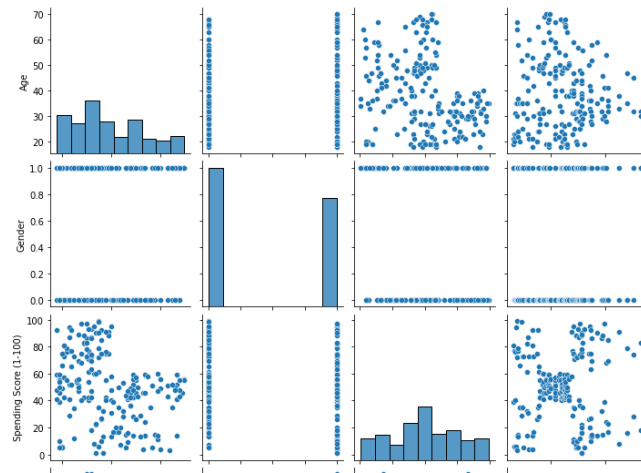
```
In [49]: sns.boxplot(x=data['Gender'], y=data['Annual Income (k$)'])
```

Out[49]:



```
In [50]: sns.pairplot(data=data[['Age', 'Gender', 'Spending Score (1-100)', 'Annual Income (k$)']])
```

Out[50]:



```
In [18]: sns.heatmap(data.corr(), annot=True)
```

Out[18]:



Question 4:

Perform descriptive statistics on the dataset.

```
In [19]: data.describe()
```

	CustomerID	Age	Annual Income (k\$)	Spending Score (1-100)
count	200.000000	200.000000	200.000000	200.000000
mean	100.500000	38.850000	60.560000	50.200000
std	57.879185	13.969007	26.264721	25.823522
min	1.000000	18.000000	15.000000	1.000000
25%	50.750000	28.750000	41.500000	34.750000
50%	100.500000	36.000000	61.500000	50.000000
75%	150.250000	49.000000	78.000000	73.000000
max	200.000000	70.000000	137.000000	99.000000

```
In [20]: data.info
```

```
Out[20]:
```

```
In [21]: data.shape
```

```
Out[21]: (200, 5)
```

Question 5:

Handle the Missing values.

```
In [22]: data.isnull().any() #Inference: The dataset has no null values
```

```
Out[22]: CustomerID      False
          Gender         False
          Age           False
          Annual Income (k$) False
          Spending Score (1-100) False
          dtype: bool
```

```
In [23]: data.drop('CustomerID',axis=1,inplace=True)
          data.head()
```

	Gender	Age	Annual Income (k\$)	Spending Score (1-100)
0	Male	19	15	39
1	Male	21	15	81
2	Female	20	16	6
3	Female	23	16	77
4	Female	31	17	40

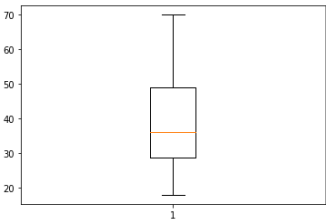
Question 6:

Find the outliers and replace the outliers

```
In [24]: for i in data:
          if data[i].dtype=='int64':
              q1=data[i].quantile(0.25)
              q3=data[i].quantile(0.75)
              iqr=q3-q1
              upper=q3+1.5*iqr
              lower=q1-1.5*iqr
              data[i]=np.where(data[i] >upper, upper, data[i])
              data[i]=np.where(data[i] <lower, lower, data[i])
```

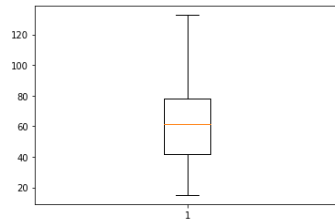
```
In [25]: plt.boxplot(data['Age'])
```

```
Out[25]: {'whiskers': [ ,
 ],
          'caps': [ ,
 ],
          'boxes': [ ],
          'medians': [ ],
          'fliers': [ ],
          'means': [ ]}
```



```
In [26]: plt.boxplot(data['Annual Income (k$)'])
```

```
Out[26]: {'whiskers': [,
],
'caps': [,
],
'boxes': [],
'medians': [],
'fliers': [],
'means': []}
```



```
In [27]: plt.boxplot(data['Spending Score (1-100)'])
```

```
Out[27]: {'whiskers': [,
],
'caps': [,
],
'boxes': [],
'medians': [],
'fliers': [],
'means': []}
```

Question 7:

Check for Categorical columns and perform encoding.

```
In [28]: from sklearn.preprocessing import LabelEncoder
l_en = LabelEncoder()
```

```
In [30]: data['Gender'] = l_en.fit_transform(data['Gender'])
data.head()
```

```
Out[30]:
```

	Gender	Age	Annual Income (k\$)	Spending Score (1-100)
0	1	19.0	15.0	39.0
1	1	21.0	15.0	81.0
2	0	20.0	16.0	6.0
3	0	23.0	16.0	77.0
4	0	31.0	17.0	40.0

Question 8:

Scaling the data

```
In [24]: from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
data_scaled = scaler.fit_transform(data)
data_scaled[0:5]
```

```
Out[24]: array([[1.          , 0.01923077, 0.          , 0.3877551 ],
 [1.          , 0.05769231, 0.          , 0.81632653],
 [0.          , 0.03846154, 0.00849257, 0.05102041],
 [0.          , 0.09615385, 0.00849257, 0.7755102 ],
 [0.          , 0.25          , 0.01698514, 0.39795918]])
```

Question 9:

Perform any clustering algorithms

```
In [25]: from sklearn.cluster import KMeans
km = KMeans()
res = km.fit_predict(data_scaled)
res
```

```
Out[25]: array([[0, 0, 1, 1, 1, 1, 5, 1, 2, 1, 2, 1, 5, 1, 3, 0, 1, 0, 2, 1, 0, 0,
5, 0, 5, 0, 5, 0, 5, 1, 2, 1, 2, 0, 5, 1, 5, 1, 5, 1, 5, 0, 2, 1,
5, 1, 5, 1, 1, 1, 5, 0, 1, 2, 5, 2, 5, 2, 1, 2, 2, 0, 5, 5, 2, 0,
5, 5, 0, 1, 2, 5, 5, 5, 2, 0, 5, 0, 1, 5, 2, 0, 2, 5, 1, 2, 5, 1,
1, 5, 5, 0, 2, 5, 1, 0, 5, 1, 2, 0, 1, 5, 2, 0, 2, 1, 5, 2, 2, 2,
2, 1, 4, 0, 1, 1, 5, 5, 5, 5, 0, 4, 6, 7, 4, 6, 3, 7, 2, 7, 3, 7,
4, 6, 3, 6, 4, 7, 3, 6, 4, 7, 4, 6, 3, 7, 3, 6, 4, 7, 3, 7, 4, 6,
4, 6, 3, 6, 3, 6, 5, 6, 3, 6, 3, 6, 3, 6, 4, 7, 3, 7, 3, 7, 4, 6,
3, 7, 3, 7, 4, 6, 3, 6, 4, 7, 4, 7, 4, 6, 4, 6, 3, 6, 4, 6, 4, 7,
3, 7], dtype=int32)
```

```
In [26]: datal = pd.DataFrame(data_scaled, columns = data.columns)
datal.head()
```

```
Out[26]:
```

	Gender	Age	Annual Income (k\$)	Spending Score (1-100)
0	1.0	0.019231	0.000000	0.387755
1	1.0	0.057692	0.000000	0.816327
2	0.0	0.038462	0.008493	0.051020
3	0.0	0.096154	0.008493	0.775510
4	0.0	0.250000	0.016985	0.397959

```
In [27]: datal['kclus'] = pd.Series(res)
datal.head()
```

```
Out[27]:
```

	Gender	Age	Annual Income (k\$)	Spending Score (1-100)	kclus
0	1.0	0.019231	0.000000	0.387755	0
1	1.0	0.057692	0.000000	0.816327	0
2	0.0	0.038462	0.008493	0.051020	1
3	0.0	0.096154	0.008493	0.775510	1
4	0.0	0.250000	0.016985	0.397959	1

```
In [28]: datal['kclus'].unique()
```

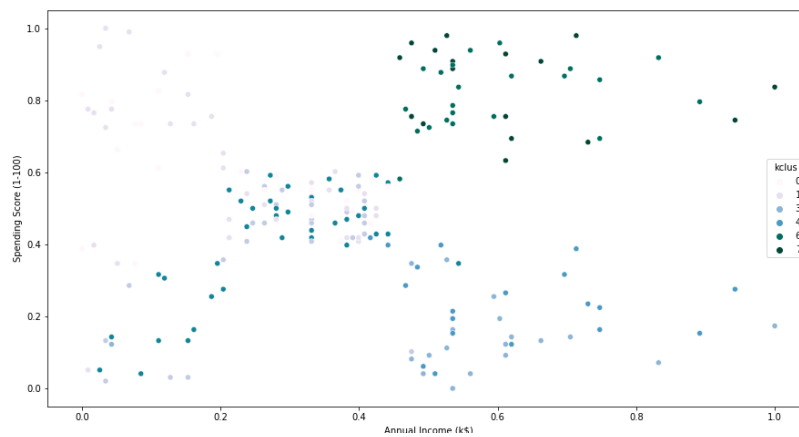
```
Out[28]: array([0, 1, 5, 2, 3, 4, 6, 7], dtype=int32)
```

```
In [29]: datal['kclus'].value_counts()
```

```
Out[29]:
5    37
1    34
2    26
0    24
6    22
3    20
4    19
7    18
Name: kclus, dtype: int64
```

```
In [30]: import matplotlib.pyplot as plt

fig,ax = plt.subplots(figsize=(15,8))
sns.scatterplot(x=datal['Annual Income (k$)'],
y=datal['Spending Score (1-100)'],
hue=datal['kclus'],
palette='PuBuGn')
plt.show()
```



```
In [31]: ind = data1.iloc[:,0:4]
ind.head()
```

```
Out[31]:
```

	Gender	Age	Annual Income (k\$)	Spending Score (1-100)
0	1.0	0.019231	0.000000	0.387755
1	1.0	0.057692	0.000000	0.816327
2	0.0	0.038462	0.008493	0.051020
3	0.0	0.096154	0.008493	0.775510
4	0.0	0.250000	0.016985	0.397959

```
In [32]: dep = data1.iloc[:,4:]
dep.head()
```

```
Out[32]:
```

	kclus
0	0
1	0
2	1
3	1
4	1

Question 10:

Split the data into training and testing

```
In [33]: from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(ind,dep,test_size=0.3,random_state=1)
x_train.head()
```

```
Out[33]:
```

	Gender	Age	Annual Income (k\$)	Spending Score (1-100)
116	0.0	0.865385	0.424628	0.428571
67	0.0	0.961538	0.280255	0.479592
78	0.0	0.096154	0.331210	0.520408
42	1.0	0.576923	0.203822	0.357143
17	1.0	0.038462	0.050955	0.663265

```
In [34]: x_test.head()
```

```
Out[34]:
```

	Gender	Age	Annual Income (k\$)	Spending Score (1-100)
58	0.0	0.173077	0.263270	0.510204
40	0.0	0.903846	0.195329	0.346939
34	0.0	0.596154	0.152866	0.132653
102	1.0	0.942308	0.399151	0.591837
184	0.0	0.442308	0.713376	0.387755

```
In [35]: y_train.head()
```

```
Out[35]:
```

	kclus
116	5
67	5
78	1
42	2
17	0

```
In [36]: y_test.head()
```

```
Out[36]:
```

	kclus
58	1
40	5
34	5
102	2
184	4

Question 11:

Build the model

```
In [37]: from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(x_train,y_train)
```

```
Out[37]: LinearRegression()
```

Question 12:

Train and test the model

```
In [38]: pred_test = lr.predict(x_test)
         pred_test[0:5]
```

```
Out[38]: array([[2.6187044 ],
                [4.18578692],
                [2.36471457],
                [4.66483923],
                [5.65372864]])
```

Question 13:

Measure the performance using Evaluation Metrics

```
In [39]: from sklearn.metrics import mean_squared_error, mean_absolute_error
         from sklearn.metrics import accuracy_score
         mse = mean_squared_error(pred_test, y_test)
         print("The Mean squared error is: ", mse)
         rmse = np.sqrt(mse)
         print("The Root mean squared error is: ", rmse)
         mae = mean_absolute_error(pred_test, y_test)
         print("The Mean absolute error is: ", mae)
         acc = lr.score(x_test, y_test)
         print("The accuracy is: ", acc)
```

```
The Mean squared error is: 3.0172335232092844
The Root mean squared error is: 1.737018573075511
The Mean absolute error is: 1.5010421648471637
The accuracy is: 0.3647557936982617
```
