

Assignment - 4

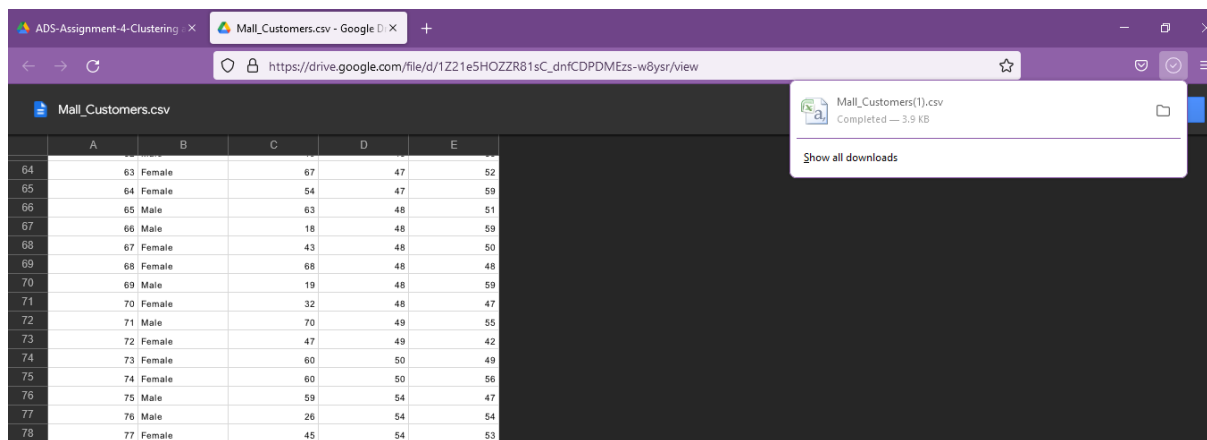
Clustering And Classification

Assignment Date	15 October 2022
Student Name	Manjunathan V
Student Roll Number	727719EUCS080
Maximum Marks	2 Marks

Question-1:

Download the dataset: Dataset

Solution:



The screenshot shows a web browser window with a Google Drive link: https://drive.google.com/file/d/1Z21e5HOZZR81sC_dnfCDPMEzs-w8ysr/view. The file 'Mall_Customers.csv' is displayed in a table format with columns A, B, C, D, and E. A download notification for 'Mall_Customers(1).csv' is visible in the top right corner.

	A	B	C	D	E
64	63	Female	67	47	52
65	64	Female	54	47	59
66	65	Male	63	48	51
67	66	Male	18	48	59
68	67	Female	43	48	50
69	68	Female	68	48	48
70	69	Male	19	48	59
71	70	Female	32	48	47
72	71	Male	70	49	55
73	72	Female	47	49	42
74	73	Female	60	50	49
75	74	Female	60	50	56
76	75	Male	59	54	47
77	76	Male	26	54	54
78	77	Female	45	54	53

Question-2:

Load the dataset into the tool

Solution:

```
import pandas as pd
df=pd.read_csv("D://Mall_Customers.csv")
df.head()
```

	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

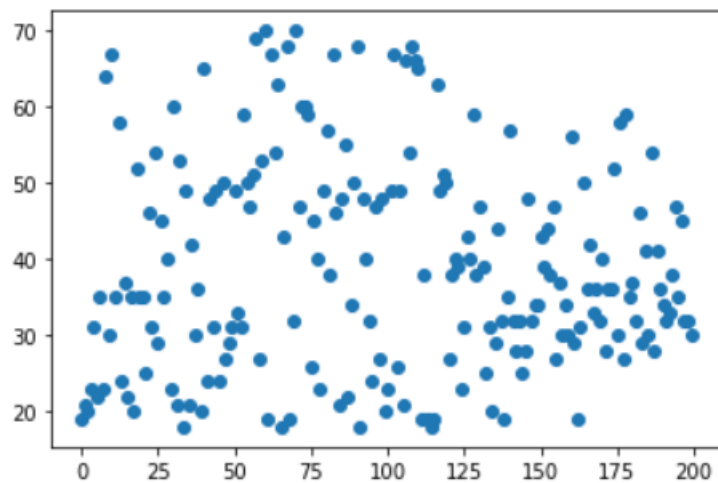
Question-3:

Perform Below Visualizations.

- Univariate analysis

Solution:

```
import matplotlib.pyplot as plt
plt.scatter(df.index,df['Age'])
plt.show()
```

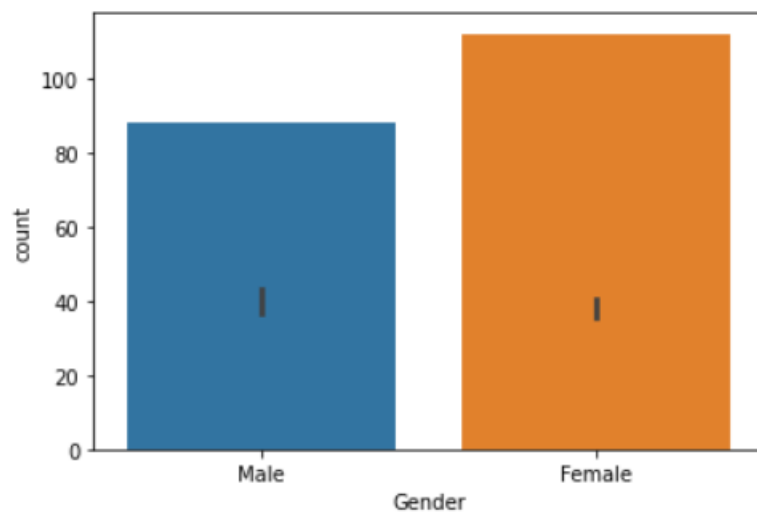


- Bi-variate analysis

Solution:

```
import seaborn as sns
sns.barplot(x='Gender',y='Age',data=df)
sns.countplot(x='Gender',data=df)
```

<AxesSubplot:xlabel='Gender', ylabel='count'>

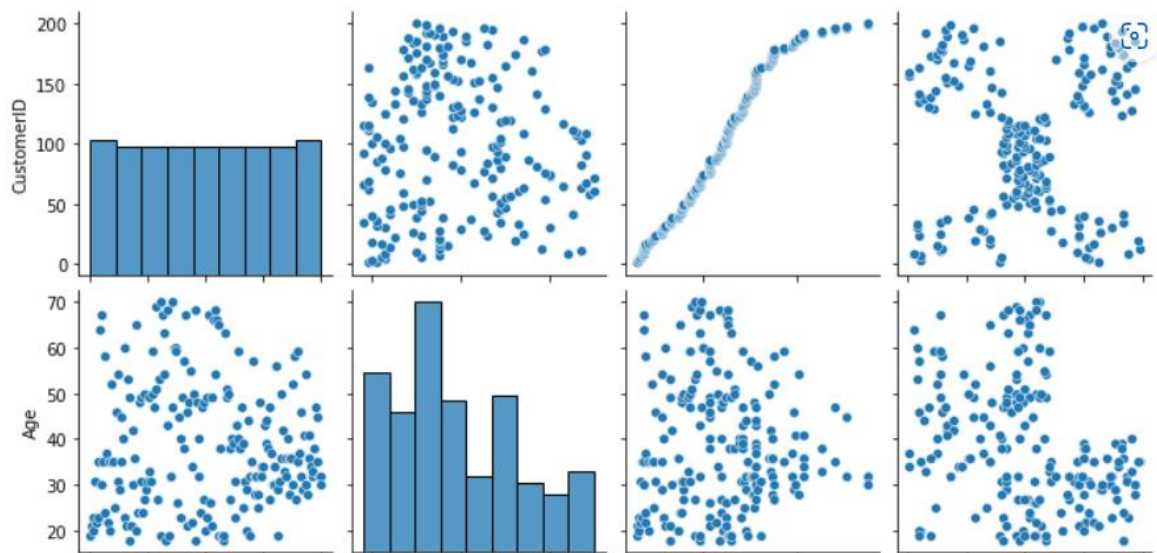


- Multi-variate analysis

Solution:

```
import seaborn as sns
sns.pairplot(df)
```

<seaborn.axisgrid.PairGrid at 0x21049a56b20>



Question-4:

Perform descriptive statistics on the dataset.

Solution:

```
df.mean()
```

CustomerID	100.50
Gender	0.56
Age	38.85
Annual Income (k\$)	60.56
Spending Score (1-100)	50.20
dtype:	float64

```
df.median()
```

CustomerID	100.5
Gender	1.0
Age	36.0
Annual Income (k\$)	61.5
Spending Score (1-100)	50.0
dtype:	float64

```
df.mode()
```

	CustomerID	Gender	Age	Annual Income (k\$)	Spending Score (1-100)
0	1	Female	32.0	54.0	42.0
1	2	NaN	NaN	78.0	NaN
2	3	NaN	NaN	NaN	NaN
3	4	NaN	NaN	NaN	NaN
4	5	NaN	NaN	NaN	NaN
...
195	196	NaN	NaN	NaN	NaN
196	197	NaN	NaN	NaN	NaN
197	198	NaN	NaN	NaN	NaN
198	199	NaN	NaN	NaN	NaN
199	200	NaN	NaN	NaN	NaN

200 rows × 5 columns

```
df.skew()
```

```
CustomerID      0.000000
Gender          -0.243578
Age             0.485569
Annual Income (k$)  0.321843
Spending Score (1-100) -0.047220
dtype: float64
```

```
df.kurt()
```

```
CustomerID      -1.200000
Gender          -1.960375
Age            -0.671573
Annual Income (k$) -0.098487
Spending Score (1-100) -0.826629
dtype: float64
```

```
qu=df['Age'].quantile(q=[0.75,0.25])
qu
```

```
0.75    49.00
0.25    28.75
Name: Age, dtype: float64
```

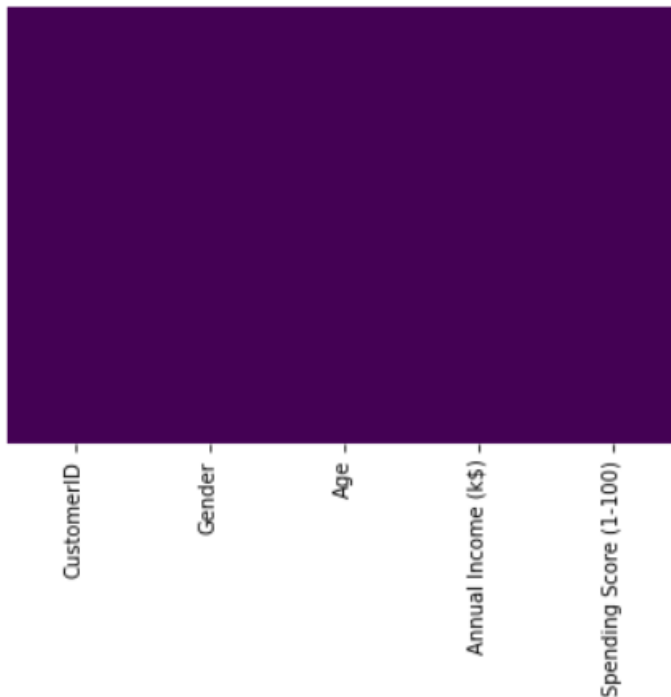
Question-5:

Check for Missing values and deal with them.

Solution:

```
sns.heatmap(df.isnull(),yticklabels=False,cbar=False,cmap="viridis")  
#there is no missing values as per heatmap
```

<AxesSubplot:>

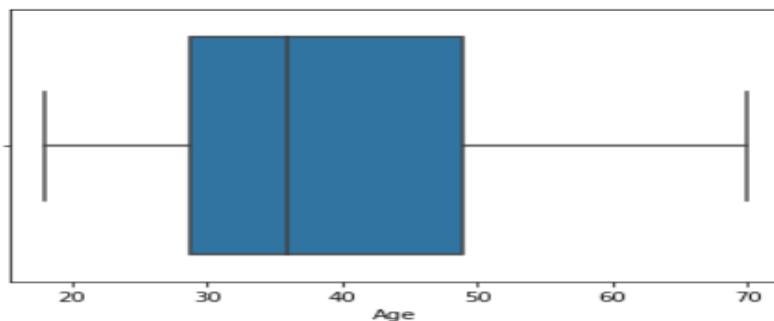


Question-6:

Find the outliers and replace the outliers

Solution:

```
sns.boxplot(df['Age'])  
<AxesSubplot:xlabel='Age'>
```



```
Q1=df.Age.quantile(0.25)  
Q2=df.Age.quantile(0.75)  
IQR=Q2-Q1  
print(IQR)
```

20.25

```
df=df[~((df.Age<(Q1-1.5*IQR))|(df.Age>(Q2+1.5*IQR)))]
df
```

	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
...
195	196	Female	35	120	79
196	197	Female	45	126	28
197	198	Male	32	126	74
198	199	Male	32	137	18
199	200	Male	30	137	83

200 rows × 5 columns

Question-7:

Check for Categorical columns and perform encoding.

Solution:

```
df['Gender'].replace({'Female':1,'Male':0},inplace=True)
df.head()
```

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

Question-8:

Scaling the data

Solution:

```
from sklearn.preprocessing import StandardScaler
std=StandardScaler()
X=std.fit_transform(X)
X
```

```
[ -1.13750203, -1.62449091,  1.70038436 ],
[  1.80493225, -1.58632148, -1.83237767 ],
[ -0.6351352 , -1.58632148,  0.84631002 ],
[  2.02023231, -1.58632148, -1.4053405  ],
[ -0.27630176, -1.58632148,  1.89449216 ],
[  1.37433211, -1.54815205, -1.36651894 ],
[ -1.06573534, -1.54815205,  1.04041783 ],
[ -0.13276838, -1.54815205, -1.44416206 ],
[ -1.20926872, -1.54815205,  1.11806095 ],
[ -0.27630176, -1.50998262, -0.59008772 ],
[ -1.3528021 , -1.50998262,  0.61338066 ],
[  0.94373197, -1.43364376, -0.82301709 ],
[ -0.27630176, -1.43364376,  1.8556706  ],
[ -0.27630176, -1.39547433, -0.59008772 ],
[ -0.99396865, -1.39547433,  0.88513158 ],
[  0.51313183, -1.3573049 , -1.75473454 ],
[ -0.56336851, -1.3573049 ,  0.88513158 ],
[  1.08726535, -1.24279661, -1.4053405  ],
[ -0.70690189, -1.24279661,  1.23452563 ],
[  0.44136514, -1.24279661, -0.7065524  ],
```

Question-9:

Perform any of the clustering algorithms

Solution:

```
from sklearn.cluster import MeanShift
clus = MeanShift(bandwidth=2).fit(X)
```

```
from sklearn.cluster import KMeans
kmeans=KMeans(n_clusters=3,random_state=42)
labels=kmeans.fit_predict(X)
```

Question-10:

Add the cluster data with the primary dataset

Solution:

```
clus.labels_
```

[illegible]

Question-11:

Split the data into dependent and independent variables.

Solution:

```
X=df.iloc[:,2:5].values
```

```
y=df.iloc[:,4].values
```

Question-12:

Split the data into training and testing

Solution:

```
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=0)
```

Question-13:

Build the Model

Solution:

```
from sklearn.linear_model import LinearRegression
r=LinearRegression()
r.fit(x_train,y_train)
```

LinearRegression()

Question-14:

Train the Model

Solution:

x_train

```
array([[ -1.3528021,  0.4748277, -1.75473454],
       [ 0.29783176, -0.47940803, -0.00776431],
       [ 0.44136514, -1.24279661, -0.7065524 ],
       [-1.42456879,  0.13130284, -0.16305055],
       [-0.20453507,  1.00919971, -0.90066021],
       [ 1.08726535, -0.51757746,  0.34162973],
       [ 1.80493225, -1.58632148, -1.83237767],
       [-0.92220196, -0.25039146,  0.14752193],
       [ 0.87196528,  0.24581112, -0.27951524],
       [-0.49160182,  0.58933599,  1.42863343],
       [ 0.58489852, -0.4412386 , -0.3183368 ],
       [-1.13750203,  0.36031941, -0.82301709],
       [ 0.15429838,  1.46723286, -0.43480148],
       [-0.85043527, -0.02137488, -0.00776431],
       [-0.34806844,  0.66567484,  1.54509812],
       [ 1.08726535, -1.24279661, -1.4053405 ],
       [ 1.51786549, -1.16645776, -1.7935561 ],
       [ 1.23079873,  0.70384427, -0.59008772],
       [ 1.87669894, -0.86110232, -0.59008772],
       [ 0.07106530,  0.63300575,  0.00776431]]
```

y_train

```
array([ 5, 50, 32, 46, 27, 59,  3, 54, 43, 87, 42, 29, 39, 50, 90, 14,  4,
       35, 35, 50, 75, 47, 86, 98, 76, 49, 45, 93, 60, 58, 72, 55, 73, 48,
       12, 46, 48, 13, 61, 93, 68, 55, 10, 79, 43, 52,  6, 46,  7, 74, 61,
       14, 16, 56, 28, 14, 86, 35, 40, 42,  1, 69, 52, 39, 42, 52, 52, 51,
       95, 92, 47, 42, 42, 46, 77, 83, 73, 24, 88, 35, 79, 52, 56, 54, 41,
       53, 77, 43, 26, 18,  6, 59, 57, 40, 89, 75,  1, 41, 83, 99, 57, 59,
       81, 46, 59, 81, 56,  5, 36, 42, 34, 92, 66, 26, 71, 60, 78, 11, 14,
       31, 48, 88, 73, 20, 95, 15,  4, 40, 63, 74, 87, 49, 41, 42, 50, 22,
       91, 49, 48, 82, 75, 55, 17, 13, 23, 75, 51,  5, 60, 55, 55, 17, 73,
       72, 55, 48,  8, 59, 47, 10], dtype=int64)
```

Question-15:

Test the Model

Solution:

```
x_test
```

```
array([[ 0.94373197, -1.43364376, -0.82301709],
       [ 0.08253169,  1.00919971, -1.44416206],
       [ 1.08726535,  0.09313341, -0.16305055],
       [ 0.65666521,  0.01679455, -0.3183368 ],
       [-0.85043527,  1.04736914,  0.72984534],
       [ 0.51313183,  1.42906343, -1.36651894],
       [-1.20926872, -1.66266033,  1.00159627],
       [ 0.65666521,  0.62750542, -0.55126616],
       [ 1.37433211, -1.54815205, -1.36651894],
       [ 0.36959845,  0.66567484, -1.17241113],
       [-1.42456879, -0.55574689,  0.18634349],
       [-0.56336851,  0.36031941,  1.04041783],
       [-0.13276838,  1.390894  , -0.7065524 ],
       [ 0.58489852,  0.66567484, -1.32769738],
       [ 1.30256542, -0.25039146,  0.03105725],
       [-1.13750203, -1.62449091,  1.70038436],
       [-1.49633548, -1.05194947,  1.62274124],
       [ 0.58489852,  0.39848884, -1.5994483 ],
       [-0.6351352 , -1.01378004,  0.88513158],
       [ 1.4460988 , -0.25039146, -0.12422899],
       [-0.70690189,  1.42906343,  1.46745499],
       [-0.77866858,  0.62750542,  1.81684904],
       [-1.06573534, -0.82293289,  0.5745591 ],
       [-0.6351352 ,  0.66567484,  0.88513158],
       [ 2.23553238, -0.55574689,  0.22516505],
       [ 0.010765  ,  0.32214998,  1.58391968],
       [-0.27630176,  1.23821628,  1.54509812],
```

```
y_test
```

```
array([29, 13, 46, 42, 69, 15, 76, 36, 15, 20, 55, 77, 32, 16, 51, 94, 92,
        9, 73, 47, 88, 97, 65, 73, 56, 91, 90, 97, 58, 28, 35, 41, 17, 54,
        5, 85, 75, 40, 44, 50], dtype=int64)
```

Question-16:

Measure the performance using Evaluation Metrics.

Solution:

```
from sklearn.metrics import silhouette_score
acc=silhouette_score(X,labels)
print(acc)
```

```
0.357793388710272
```