ASSIGNMENT 5

HARSH KUMAR

21BDS0391

harsh.kumar2021d@vitstudent.ac.in


```
# HARSH KUMAR
# 21BDS0391
# harsh.kumar2021d@vitstudent.ac.in
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import silhouette_score
```

```
# Load the dataset (assuming it's a CSV file)
data = pd.read_csv('Mall_Customers.csv')
```

1) Understanding the data

```
print(data.head(10))
```

```
   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
5           6  Female   22                  17                      76
6           7  Female   35                  18                       6
7           8  Female   23                  18                      94
8           9    Male   64                  19                       3
9          10  Female   30                  19                      72
```

```
print(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
```

```
print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   CustomerID              200 non-null    int64
 1   Gender                  200 non-null    object
 2   Age                     200 non-null    int64
 3   Annual Income (k$)      200 non-null    int64
 4   Spending Score (1-100)  200 non-null    int64
dtypes: int64(4), object(1)
memory usage: 7.9+ KB
None
```

2) Data Preprocessing

```
X = data[['Annual Income (k$)', 'Spending Score (1-100)']]
```

```
# Standardize features (scaling)
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
X_scaled
```

```
       [ 0.58933599, -0.39597992],
       [ 0.58933599,  1.42863343],
       [ 0.62750542, -1.48298362],
       [ 0.62750542,  1.81684904],
       [ 0.62750542, -0.55126616],
       [ 0.62750542,  0.92395314],
       [ 0.66567484, -1.09476801],
       [ 0.66567484,  1.54509812],
       [ 0.66567484, -1.28887582],
       [ 0.66567484,  1.46745499],
       [ 0.66567484, -1.17241113],
       [ 0.66567484,  1.00159627],
       [ 0.66567484, -1.32769738],
       [ 0.66567484,  1.50627656],
       [ 0.66567484, -1.91002079],
       [ 0.66567484,  1.07923939],
       [ 0.66567484, -1.91002079],
       [ 0.66567484,  0.88513158],
       [ 0.70384427, -0.59008772],
       [ 0.70384427,  1.27334719],
       [ 0.78018313, -1.75473454],
       [ 0.78018313,  1.6615628 ],
       [ 0.93286085, -0.93948177],
       [ 0.93286085,  0.96277471],
       [ 0.97103028, -1.17241113],
       [ 0.97103028,  1.73920592],
       [ 1.00919971, -0.90066021],
       [ 1.00919971,  0.49691598],
       [ 1.00919971, -1.44416206],
       [ 1.00919971,  0.96277471],
       [ 1.00919971, -1.56062674],
       [ 1.00919971,  1.62274124],
       [ 1.04736914, -1.44416206],
       [ 1.04736914,  1.38981187],
       [ 1.04736914, -1.36651894],
       [ 1.04736914,  0.72984534],
       [ 1.23821628, -1.4053405 ],
       [ 1.23821628,  1.54509812],
       [ 1.390894  , -0.7065524 ],
       [ 1.390894  ,  1.38981187],
       [ 1.42906343, -1.36651894],
       [ 1.42906343,  1.46745499],
       [ 1.46723286, -0.43480148],
       [ 1.46723286,  1.81684904],
       [ 1.54357172, -1.01712489],
       [ 1.54357172,  0.69102378],
       [ 1.61991057, -1.28887582],
       [ 1.61991057,  1.35099031],
       [ 1.61991057, -1.05594645],
       [ 1.61991057,  0.72984534],
       [ 2.00160487, -1.63826986],
       [ 2.00160487,  1.58391968],
       [ 2.26879087, -1.32769738],
       [ 2.26879087,  1.11806095],
       [ 2.49780745, -0.86183865],
       [ 2.49780745,  0.92395314],
       [ 2.91767117, -1.25005425],
       [ 2.91767117,  1.27334719]])
```

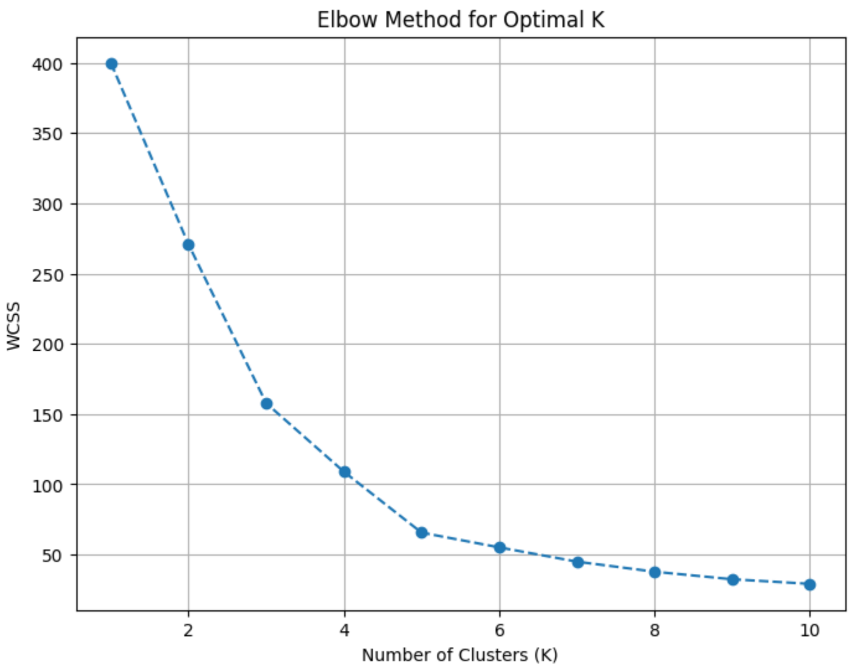
3) Unsupervised Machine Learning Approach (K-Means Clustering)

```
# Determine the optimal number of clusters using the Elbow method
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', random_state=42)
    kmeans.fit(X_scaled)
    wcss.append(kmeans.inertia_)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change fr
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change fr
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change fr
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change fr
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change fr
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change fr
  warnings.warn(
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change fr
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change fr
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change fr
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change fr
  warnings.warn(
```

```python
# Plot the Elbow method graph
plt.figure(figsize=(8, 6))
plt.plot(range(1, 11), wcss, marker='o', linestyle='--')
plt.title('Elbow Method for Optimal K')
plt.xlabel('Number of Clusters (K)')
plt.ylabel('WCSS')
plt.grid(True)
plt.show()
```



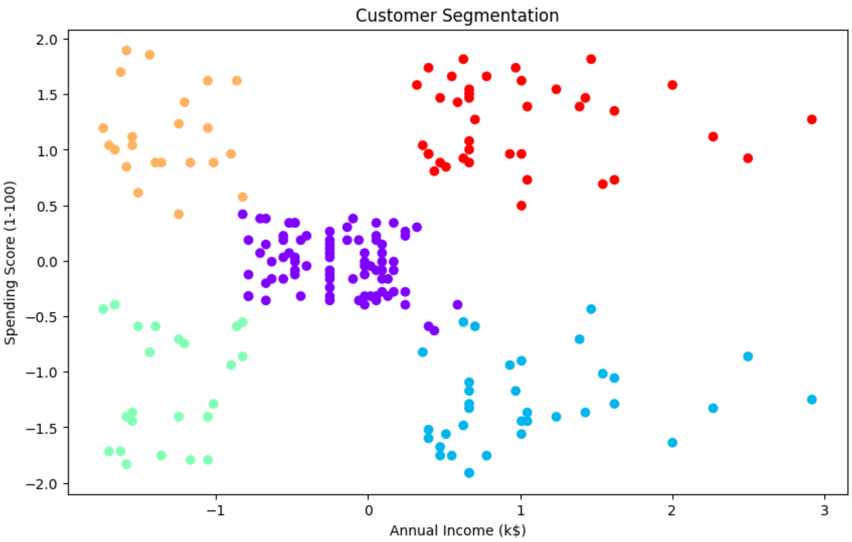

```python
# Based on the Elbow method, K=5
kmeans = KMeans(n_clusters=5, init='k-means++', random_state=42)
cluster_labels = kmeans.fit_predict(X_scaled)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change fr
  warnings.warn(
```

```python
data['Cluster'] = cluster_labels
data.head()
```

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

```python
# Visualizing the clusters
plt.figure(figsize=(10, 6))
plt.scatter(X_scaled[:, 0], X_scaled[:, 1], c=cluster_labels, cmap='rainbow')
plt.title('Customer Segmentation')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.show()
```




```
# Evaluating quality using silhouette score
silhouette_avg = silhouette_score(X_scaled, cluster_labels)
print(f'Silhouette Score: {silhouette_avg}')
```

```
Silhouette Score: 0.5546571631111091
```