22CB903

MINI-PROJECT 3

OBJECTIVE:

The goal is to group customers into distinct segments based on their characteristics.

ALGORITHM:

Data Preprocessing:

1. Load the customer dataset using pandas
2. Inspect the dataset to understand its structure using .head() and .info() methods.
3. Check for missing values using isnull().sum().

Feature Selection:

1. Extract the features of interest: Annual Income (k$) and Spending Score (1-100) using the iloc method. These will be used as input for clustering (X = customer_data.iloc[:, [3,4]].values).

Finding Optimal Number of Clusters (Elbow Method):

1. Initialize an empty list wcss = [] to store the Within-Cluster Sum of Squares (WCSS) for different numbers of clusters.

2. For each cluster number i (ranging from 1 to 10), perform the following:
    - Initialize a K-Means model with i clusters using kmeans = KMeans(n_clusters=i, init='k-means++', random_state=42).
    - Fit the model to the selected features using kmeans.fit(X).
    - Append the WCSS (inertia) to the list wcss.append(kmeans.inertia_).
3. Plot the WCSS against the number of clusters to visualize the "elbow" point. This elbow point indicates the optimal number of clusters.

Applying K-Means Clustering:

1. Based on the elbow method, choose the optimal number of clusters.
2. Initialize the K-Means model with the chosen number of clusters (e.g., kmeans = KMeans(n_clusters=5, init='k-means++', random_state=0)).
3. Fit the model to the data and assign each customer to a cluster based on the closest centroid.

Visualization:

1. Plot the results of the clustering, where each cluster is represented by a different color to show how customers are grouped.

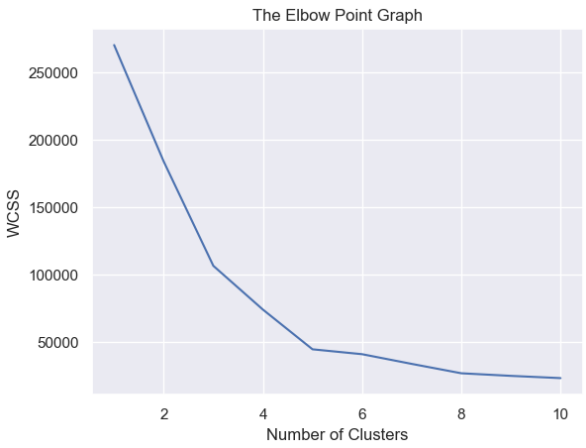
CODE:

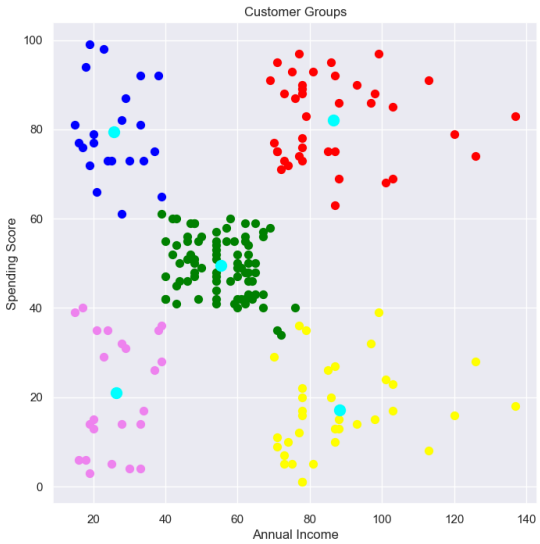
```
# %%
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans

# %%
# loading the data from csv file to a Pandas DataFrame
customer_data = pd.read_csv('Mall_Customers.csv')

# %%
# first 5 rows in the dataframe
customer_data.head()

# %%
# finding the number of rows and columns
customer_data.shape

# %%
# getting some informations about the dataset
```

```
customer_data.info()

# %%
# checking for missing values
customer_data.isnull().sum()

# %%
X = customer_data.iloc[:,[3,4]].values
print(X)

# %%
# finding wcss value for different number of clusters

wcss = []

for i in range(1,11):
  kmeans = KMeans(n_clusters=i, init='k-means++', random_state=42)
  kmeans.fit(X)

  wcss.append(kmeans.inertia_)

# %%
# plot an elbow graph

sns.set()
plt.plot(range(1,11), wcss)
plt.title('The Elbow Point Graph')
plt.xlabel('Number of Clusters')
plt.ylabel('WCSS')
plt.show()

# %%
kmeans = KMeans(n_clusters=5, init='k-means++', random_state=0)
```

```
# return a label for each data point based on their cluster
Y = kmeans.fit_predict(X)


print(Y)


# %%
# plotting all the clusters and their Centroids


plt.figure(figsize=(8,8))
plt.scatter(X[Y==0,0], X[Y==0,1], s=50, c='green', label='Cluster 1')
plt.scatter(X[Y==1,0], X[Y==1,1], s=50, c='red', label='Cluster 2')
plt.scatter(X[Y==2,0], X[Y==2,1], s=50, c='yellow', label='Cluster 3')
plt.scatter(X[Y==3,0], X[Y==3,1], s=50, c='violet', label='Cluster 4')
plt.scatter(X[Y==4,0], X[Y==4,1], s=50, c='blue', label='Cluster 5')


# plot the centroids
plt.scatter(kmeans.cluster_centers_[:,0], kmeans.cluster_centers_[:,1],
s=100, c='cyan', label='Centroids')


plt.title('Customer Groups')
plt.xlabel('Annual Income')
plt.ylabel('Spending Score')
plt.show()
```

OUTPUT:





By

S Sharvesh Guru

CSBS | 111722202043