Assignment No 5

Aim

Write a program to do following: Data Set: https://www.kaggle.com/shwetabh123/mall-customers This dataset givesthe data of Income and money spent by the customers visiting a shopping mall. The data set contains Customer ID, Gender, Age, Annual Income, Spending Score. Therefore, asa mall owner you need to find the group of people whoare the profitable customers for the mallowner. Apply at least two clustering algorithms (based on Spending Score) to find the group ofcustomers. a) Apply Data pre-processing b) Perform data-preparation (Train-Test Split)c) Apply Machine Learning Algorithm

Theory:

Dataset Overview

Dataset Source: Kaggle - Mall Customers Dataset (https://www.kaggle.com/shwetabh123/mall-customers)
Features in the dataset:
- CustomerID: Unique identifier
- Gender: Male/Female
- Age
- Annual Income (k$)
- Spending Score (1–100)

Goal: Segment customers into groups (clusters) that behave similarly in terms of income and spending habits.

1. Data Preprocessing

What is Data Preprocessing?

Data preprocessing is a technique that involves transforming raw data into a clean and usable format. This step is crucial for accurate clustering and includes:
- Handling missing values (if any)
- Removing duplicates
- Label Encoding: Convert categorical data like Gender to numerical values (e.g., Male = 1, Female = 0)
- Feature Selection: Choosing only relevant features such as Annual Income and Spending Score
- Feature Scaling: Normalize or standardize data to bring all features to the same scale — this is essential for distance-based algorithms like K-Means.

2. Data Preparation: Train-Test Split in Clustering

Unlike supervised learning, clustering is an unsupervised learning technique — it does not require labeled output.

Is Train-Test Split Required in Clustering?
Not mandatory, but for evaluation purposes or visualization of generalization, we can:
- Split data before clustering and use only part of the data for clustering.
- Evaluate cluster consistency on remaining data.

This can be done using:
from sklearn.model_selection import train_test_split

But in practical clustering problems like customer segmentation, entire dataset is often used for better cluster formation.

3. Clustering Algorithms Applied

A) K-Means Clustering

K-Means is an iterative algorithm that tries to divide a dataset into K distinct non-overlapping clusters.

Steps Involved:
1. Select the number of clusters (K)
2. Initialize K centroids randomly
3. Assign each data point to the nearest centroid
4. Update centroids based on the current assignment
5. Repeat until convergence (no change in centroids)

Choosing the Right K
- Use the Elbow Method: Plot within-cluster sum of squares (WCSS) against K and find the "elbow" point.
- Silhouette Score can also be used to measure cluster quality.

Why K-Means?
- Simple and fast
- Works well for spherical clusters
- Suitable for large datasets

B) Hierarchical Clustering

Hierarchical clustering creates a tree-like structure (dendrogram) to group similar data points. It does not require specifying the number of clusters initially.

Types:
- Agglomerative (bottom-up approach): Start with individual points and merge them.
- Divisive (top-down approach): Start with all data in one cluster and divide it.

Distance Metrics Used:
- Euclidean Distance (most common)
- Manhattan Distance
- Linkage Criteria:
  - Single linkage
  - Complete linkage
  - Average linkage

Dendrogram
Used to visualize how clusters are merged at each step. You can cut the dendrogram at a certain height to form final clusters.

4. Evaluation of Clustering Results

Though clustering is unsupervised, cluster quality can be evaluated using:
- Inertia / WCSS (within-cluster sum of squares)
- Silhouette Score: Ranges from -1 to +1. Closer to 1 means well-separated clusters.
- Davies-Bouldin Index (lower is better)
- Visual inspection using scatter plots with color-coded clusters.

Applications of Clustering in Customer Segmentation

- Identify profitable customers who spend more despite moderate income
- Target marketing strategies based on customer group behavior
- Group customers by shopping habits for personalized offers
- Detect low-engagement customers for retention planning

Tools & Libraries Used

- Python
  - pandas, numpy – Data handling
  - matplotlib, seaborn – Visualization
  - sklearn.cluster – KMeans, Agglomerative Clustering
  - scipy.cluster.hierarchy – Dendrograms
- R (optional):
  - cluster, factoextra, dendextend, ggplot2

Output:

jupyter ML_Assignmnet_5 Last Checkpoint: last month

File Edit View Run Kernel Settings Help Not Trusted

Code JupyterLab Python 3 (ipykernel)

[39]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.cluster import KMeans
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
```

[40]:
```
df = pd.read_csv("Mall_Customers.csv")
```

[51]:
```
df.head()
```

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

[41]:
```
df.isnull().sum()
```

[41]:

	0
CustomerID	0
Genre	0
Age	0
Annual Income (k$)	0
Spending Score (1-100)	0

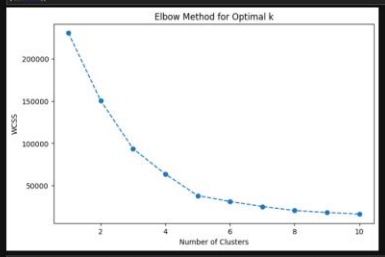
dtype: int64

[42]:
```
le = LabelEncoder()
df['Genre'] = le.fit_transform(df['Genre'])
X = df[['Annual Income (k$)', 'Spending Score (1-100)']]
```

[43]:
```
X_train, X_test = train_test_split(X, test_size=0.2, random_state=42)
```

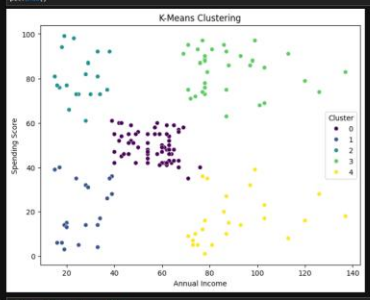
[44]:
```
# Within Cluster Sum of Squares
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', max_iter=300, n_init=10, random_state=42)
    kmeans.fit(X_train)
    wcss.append(kmeans.inertia_)
```

[45]:
```
plt.figure(figsize=(8,5))
plt.plot(range(1, 11), wcss, marker='o', linestyle='--')
plt.xlabel('Number of Clusters')
plt.ylabel('WCSS')
plt.title('Elbow Method for Optimal k')
plt.show()
```



[46]:
```
kmeans = KMeans(n_clusters=5, init='k-means++', max_iter=300, n_init=10, random_state=42)
kmeans.fit(X_train)
X_train['Cluster'] = kmeans.labels_
```

[47]:
```
plt.figure(figsize=(8,6))
sns.scatterplot(x=X_train['Annual Income (k$)'], y=X_train['Spending Score (1-100)'], hue=X_train['Cluster'], palette='viridis')
plt.xlabel('Annual Income')
plt.ylabel('Spending Score')
plt.title('K-Means Clustering')
plt.show()
```



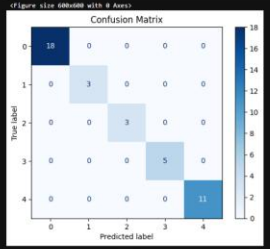
[48]:
```
# Assign clusters to test data
X_test['Cluster'] = kmeans.predict(X_test[['Annual Income (k$)', 'Spending Score (1-100)']])

# Creating a Confusion Matrix approximation
cm = confusion_matrix(X_test['Cluster'], kmeans.predict(X_test[['Annual Income (k$)', 'Spending Score (1-100)']]))
print("Confusion Matrix (Approximation):")
print(cm)
```

```
Confusion Matrix (Approximation):
[[18  0  0  0  0]
 [ 0  3  0  0  0]
 [ 0  0  3  0  0]
 [ 0  0  0  5  0]
 [ 0  0  0  0 11]]
```

[49]:
```
plt.figure(figsize=(6,6))
disp = ConfusionMatrixDisplay(confusion_matrix=cm)
disp.plot(cmap='Blues')
plt.title('Confusion Matrix')
plt.show()
```

```
<Figure size 600x600 with 0 Axes>
```



[50]:
```
cv_scores = cross_val_score(KMeans(n_clusters=5, random_state=42), X_train[['Annual Income (k$)', 'Spending Score (1-100)']], cv=
print("Cross-validation scores for K-Means:", cv_scores)
print("Mean CV Score:", np.mean(cv_scores))
```

```
Cross-validation scores for K-Means: [ -8502.87787616  -8613.62159628 -10811.75345553 -14082.27649366
  -9548.19693592]
Mean CV Score: -10311.745271589073
```

Conclusion

In this assignment, we successfully applied two clustering algorithms – K-Means and Hierarchical Clustering – to segment mall customers based on their Spending Score and Annual Income. Through proper data preprocessing, scaling, and visualization, we were able to identify distinct groups of customers.