Roll No. 21EC	**Experiment No.** 04	**Marks :**
BATCH - C		**Sign :**

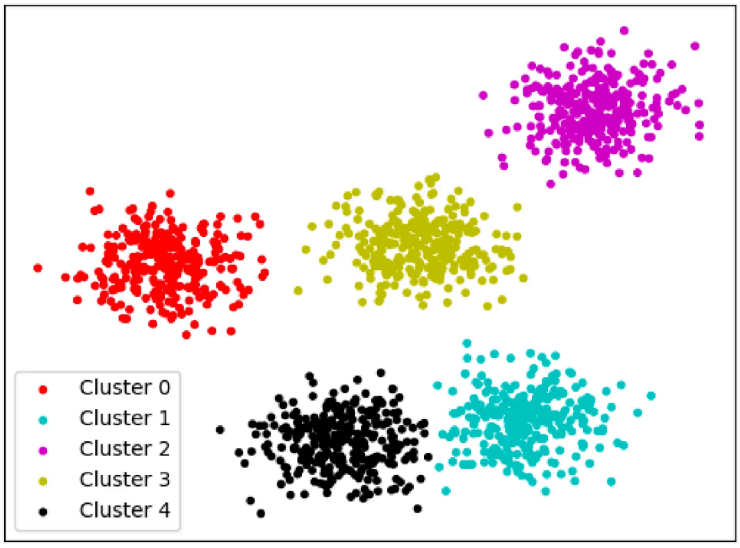
Aim: Implementation of Clustering algorithm (K-means/Agglomerative)

Apparatus: Google Colab

Theory:

What is Clustering in Data Mining?

Clustering in data mining is a technique that groups similar data points together based on their features and characteristics. It can also be referred to as a process of grouping a set of objects so that objects in the same group (called a cluster) are more similar to each other than those in other groups (clusters). It is an unsupervised learning technique that aims to identify similarities and patterns in a dataset. Clustering algorithms typically require defining the number of clusters, similarity measures, and clustering methods. These algorithms aim to group data points together in a way that maximizes similarity within the groups and minimizes similarity between different groups, as shown in the picture below.



Clustering techniques in data mining can be used in various applications, such as image segmentation, document clustering, and customer segmentation. The goal is to obtain meaningful insights from the data and improve decision-making processes.

What is a Cluster?

In data mining, a cluster refers to a group of data points with similar characteristics or features. These characteristics or features can be defined by the analyst or identified by the clustering algorithm while grouping similar data points together. The data points within a cluster are typically more similar to each other than those outside the cluster. For example, in the above figure, there are 5 clusters present.

A cluster can have the following properties -

- The data points within a cluster are similar to each other based on some pre-defined criteria or similarity measures.
- The clusters are distinct from each other, and the data points in one cluster are different from those in another cluster.
- The data points within a cluster are closely packed together.
- A cluster is often represented by a centroid or a center point that summarizes the properties of the data points within the cluster.
- A cluster can have any number of data points, but a good cluster should not be too small or too large.

Applications of Clustering in Data Mining:

Clustering is a widely used technique in data mining and has numerous applications in various fields. Some of the common applications of clustering in data mining include -

- **Customer Segmentation**
  Clustering techniques in data mining can be used to group customers with similar behavior, preferences, and purchasing patterns to create more targeted marketing campaigns.
- **Image Segmentation**
  Clustering techniques in data mining can be used to segment images into different regions based on their pixel values, which can be useful for tasks such as object recognition and image compression.
- **Anomaly Detection**
  Clustering techniques in data mining can be used to identify outliers or anomalies in datasets that deviate significantly from normal behavior.
- **Text Mining**
  Clustering techniques in data mining can be used to group documents or texts with similar content, which can be useful for tasks such as document summarization and topic modeling.
- **Biological Data Analysis**
  Clustering techniques in data mining can be used to group genes or proteins with similar characteristics or expression patterns, which can be useful for tasks such as drug discovery and disease diagnosis.

Clustering Methods in Data Mining:

There are several clustering techniques in data mining, each with its own strengths and weaknesses. Some of the most commonly used clustering techniques in data mining include -

1. **K-means Clustering**
   K-means clustering is a partitioning method that divides the data points into k clusters, where k is a pre-defined number. It works by iteratively moving the centroid of each cluster to the mean of the data points assigned to it until convergence. K-means aims to minimize the sum of squared distances between each data point and its assigned cluster centroid.
2. **Agglomerative Clustering**
   Agglomerative Clustering is a type of hierarchical clustering algorithm. It is an unsupervised machine learning technique that divides the population into several clusters such that data points in the same cluster are more similar and data points in different clusters are dissimilar.
3. **Hierarchical Clustering**
   Hierarchical clustering in data mining is a method that builds a tree-like hierarchy of clusters, either by merging smaller clusters into larger ones (agglomerative or bottom-up) or by splitting larger clusters into smaller ones (divisive or top-down). It does not require a pre-defined number of clusters.
4. **Density-Based Clustering**
   Density-based clustering is a method that identifies clusters based on regions of high density in the data space. Points that are not in any high-density region are considered noise or outliers. The most commonly used density-based clustering algorithm is DBSCAN.
5. **Model-Based Clustering**
   Model-based clustering is a method that assumes that a probabilistic model, such as a mixture of Gaussian distributions generates the data points. It seeks to identify the model parameters that best fit the data and assigns data points to clusters based on their likelihood under the model.
6. **Fuzzy Clustering**
   Fuzzy clustering is a method that assigns data points to clusters based on their degree of membership in each cluster. This allows a data point to belong to multiple clusters with different degrees of membership.

Why is Clustering Required in Data Mining?

Clustering is a critical technique in the data mining process, and it has various advantages, as mentioned below -

- ☐ **Scalability**
  Clustering algorithms in data mining can handle large datasets efficiently, making it possible to extract useful insights and knowledge from massive amounts of data.
- ☐ **High Dimensionality**
  Clustering algorithms in data mining can efficiently handle high-dimensional datasets, making it possible to find patterns and relationships that may not be apparent in lower dimensions.
- ☐ **Discovery of Clusters with Arbitrary Shape**
  Clustering algorithms in data mining can discover clusters that have different shapes and sizes, making it possible to identify groups of data points that share common properties or features.

- ☐ **Interpretability**
  Clustering results can be easily interpreted by humans, making it possible to extract useful insights and knowledge from the data.
- ☐ **Ability to Deal with Different Kinds of Data**
  Clustering algorithms in data mining can handle different types of data, such as categorical, numerical, and binary, making it possible to cluster a wide range of data types.

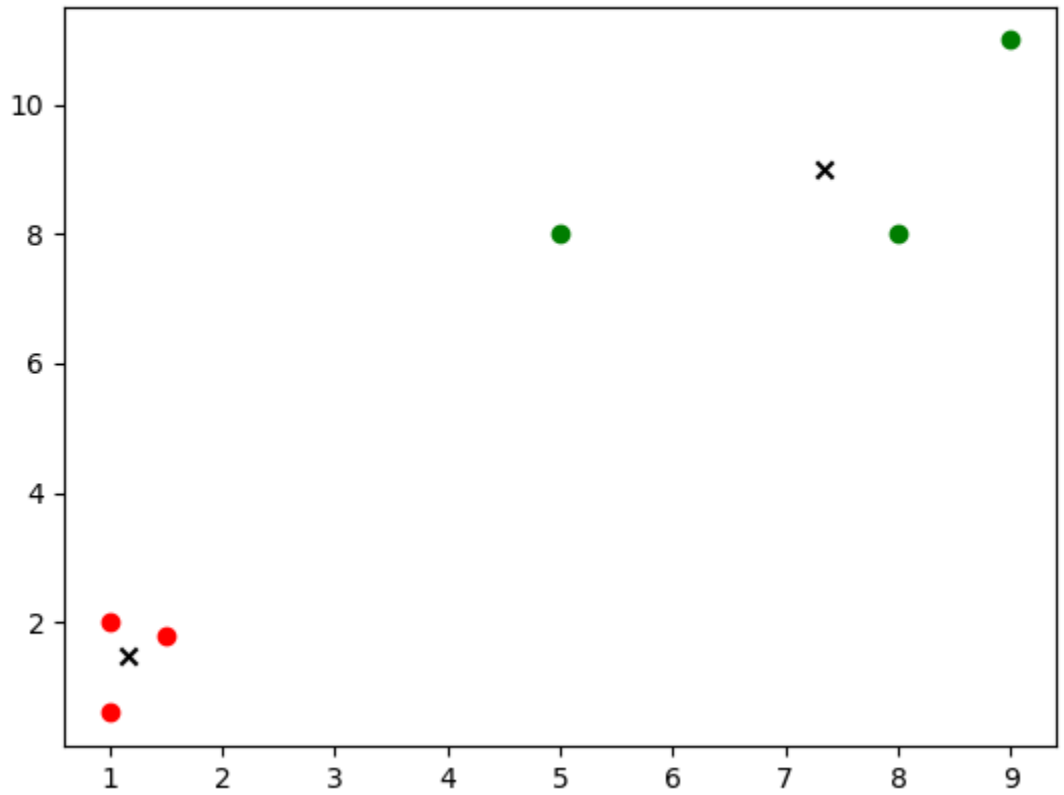
Implementation:

1. **K-Means Clustering :**

```
import numpy as np
import matplotlib.pyplot as plt

class KMeans:
    def __init__(self, n_clusters=3, max_iters=100):
        self.n_clusters = n_clusters
        self.max_iters = max_iters

    def fit(self, X):
        self.centroids = X[np.random.choice(range(len(X)), self.n_clusters, replace=False)]

        for _ in range(self.max_iters):
            clusters = [[] for _ in range(self.n_clusters)]
            for point in X:
                distances = [np.linalg.norm(point - centroid) for centroid in self.centroids]
                cluster_idx = np.argmin(distances)
                clusters[cluster_idx].append(point)
            new_centroids = [np.mean(cluster, axis=0) for cluster in clusters]
            if np.allclose(self.centroids, new_centroids):
                break
            self.centroids = new_centroids
        self.labels_ = np.zeros(len(X))
        for i, cluster in enumerate(clusters):
            self.labels_[[X.tolist().index(p.tolist()) for p in cluster]] = i
        return self.labels_
```

```
# Example usage:
X = np.array([[1, 2], [1.5, 1.8], [5, 8], [8, 8], [1, 0.6], [9, 11]])

kmeans = KMeans(n_clusters=2)
labels = kmeans.fit(X)

colors = ['r', 'g', 'b', 'y', 'c', 'm']
for i in range(len(X)):
    plt.scatter(X[i][0], X[i][1], color=colors[int(labels[i])])
plt.scatter(np.array(kmeans.centroids)[:, 0], np.array(kmeans.centroids)[:, 1], color='k', marker='x')
plt.show()
```

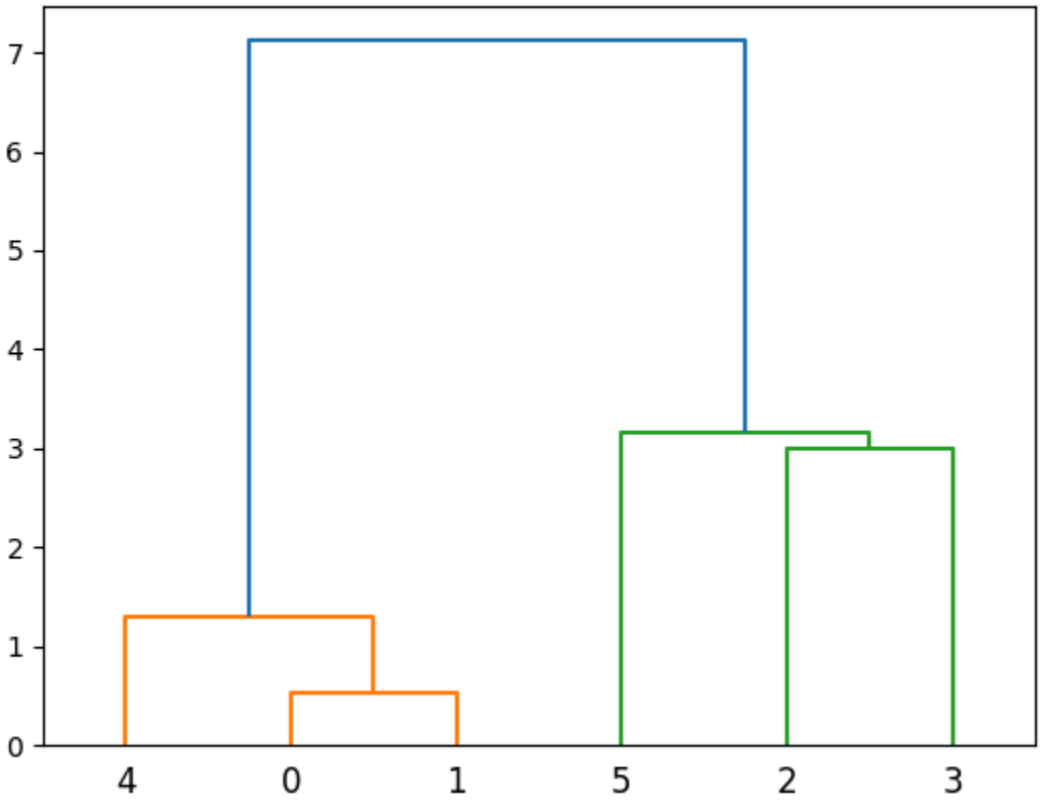
OUTPUT:



2. Agglomerative Clustering :

```
from scipy.cluster.hierarchy import dendrogram, linkage
import matplotlib.pyplot as plt

class AgglomerativeClustering:
    def __init__(self, n_clusters=2, linkage_method='single'):
        self.n_clusters = n_clusters
        self.linkage_method = linkage_method

    def fit_predict(self, X):
        Z = linkage(X, method=self.linkage_method)
        dendrogram(Z)
        plt.show()
        return Z

# Example usage:
X = [[1, 2], [1.5, 1.8], [5, 8], [8, 8], [1, 0.6], [9, 11]]

agg = AgglomerativeClustering(n_clusters=2, linkage_method='single')
agg.fit_predict(X)
```

OUTPUT:



Conclusion:

In Conclusion, both K-means and Agglomerative clustering algorithms were implemented in this Collab notebook. K-means partitions data into K clusters, while Agglomerative clustering merges close points iteratively. Each method has strengths and limitations depending on the dataset and goals.