CHAPTER 7

Clustering

Learning Objectives

At the end of this chapter, you will be able to:

- Define clustering
- Describe the commonly used clustering algorithms

7.1 Introduction to Clustering

Clustering refers to the process of arranging or organizing objects according to specific criteria. It plays a crucial role in uncovering concealed knowledge in large data sets. Clustering involves dividing or grouping the data into smaller data sets based on similarities/dissimilarities. Depending on the requirements, this grouping can lead to various outcomes, such as partitioning of data, data re-organization, compression of data and data summarization.

7.1.1 Partitioning of Data

Clustering of data is a crucial aspect of efficient data access in database-based applications.

Example 1: To illustrate, let us take the example of an EMPLOYEE data table that contains 30,000 records of fixed length. We assume that there are 1000 distinct values of DEPT_CODE and that the employee records are evenly distributed among these values.

If this data table is clustered by DEPT_CODE, accessing all the employees of the department with DEPT_CODE = "15" requires $\log_2(1000) + 30$ accesses. The first term involves accessing the index table that is constructed using DEPT_CODE, which is achieved through binary search. The second term involves fetching 30 (that is, 30000/1000) records from the clustered (that is, grouped based on DEPT_CODE) employee table, which is indicated by the fetched entry in the index table.

Without clustering, accessing 30 employee records from a department would require, on average, $30 \times 30000/2$ accesses.

7.1.3 Data Compression

Clustering can also assist in data compression by reducing the time complexity of accessing the data features and minimizing the space required to store them.

EXAMPLE 3: To demonstrate, consider the data set shown in Table 7.1. Through clustering, we can compress the data set, resulting in the table shown in Table 7.4. Note that the frequent pattern (FP) tree and pattern count (PC) tree structures are some examples for achieving such compression of data.

TABLE 7.4 Compressed data

Pattern	f1	f2	f3	f4	f5	f6	Count
1, 3	1	0	1	0	1	0	2
2, 4	0	1	0	1	0	1	2

7.1.4 Summarization

The objective of data summarization is to extract a representative subset of samples from a large data set. The purpose is to simplify and expedite analysis on a smaller data set.

EXAMPLE 4: For instance, when dealing with a data set of the marks of 100 students in a subject, statistical measures like mean (the numerical average of the marks), mode (the most frequently repeated marks) and median (the value in the middle of all the marks when the marks are ranked in order) can provide summarized information. Such summarized information can be derived through clustering.

7.1.5 Matrix Factorization

It is possible to view clustering as matrix factorization.

Let there be n data points in an l-dimensional space. We can represent it as a matrix $X_{n\times l}$. It is possible to approximate X as a product of two matrices $B_{n\times K}$ and $C_{K\times l}$. So, $X \approx BC$, where B is the cluster assignment matrix and C is the representatives matrix.

EXAMPLE 5: Consider the data matrix

$$X = \begin{bmatrix} 6 & 6 & 6 \\ 6 & 6 & 8 \\ 2 & 4 & 2 \\ 2 & 2 & 2 \end{bmatrix}$$

There are four patterns in a three-dimensional space. If we use the leader algorithm with a threshold of 3 units, we get two clusters: (6,6,6) and (6,6,8) that belong to cluster 1 with (6,6,6) as its leader and (2,4,2) and (2,2,2) that belong to cluster 2 with (2,4,2) as its leader.

So, we have $B = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$. It indicates that the first pattern, (6,6,6), is assigned to cluster 1. Correspondingly in B we have a 1 in the first row and first column. Matrix B has four rows

corresponding to four patterns in X; B has two columns corresponding to two clusters. Matrix C will have K rows if there are K clusters and the i^{th} row is the representative of the i^{th} cluster.

In the current example, $C = \begin{bmatrix} 6 & 6 & 6 \\ 2 & 4 & 2 \end{bmatrix}$. So the resulting matrix factorization is given by

cluster-1 ← cluster-2 ←

$$\begin{bmatrix} 6 & 6 & 6 \\ 6 & 6 & 8 \\ 2 & 4 & 2 \\ 2 & 2 & 2 \end{bmatrix} \approx \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} 6 & 6 & 6 \\ 2 & 4 & 2 \end{bmatrix}$$

Leader is a hard clustering algorithm where each pattern is assigned fully, that is, with a value 1, to a single cluster. So, each row of the B matrix will have one 1 and the rest as 0s.

It is possible to have soft clustering. In this case, we can have multiple non-zero entries in the range [0,1] in each row, indicating that a pattern is assigned to more than one cluster. The sum of the entries in a row is 1.

Instead of the leader, if we use the centroid of points in the cluster as its representative, then $C = \begin{bmatrix} 6 & 6 & 7 \\ 2 & 3 & 2 \end{bmatrix}$. There is no change in B. Note that the centroid of {(6,6,6), (6,6,8)} is (6,6,7) and the centroid of the second cluster {(2,4,2), (2,2,2)} is (2,3,2).

7.2 Clustering of Patterns

Clustering is a technique that involves grouping a set of patterns, resulting in the creation of cohesive clusters or groups from a given collection of patterns. The process of clustering aims to group similar patterns together while keeping dissimilar patterns separate. Figure 7.1 illustrates the clustering of a two-dimensional data set (represented by f1 and f2), where three clusters are visually identifiable.

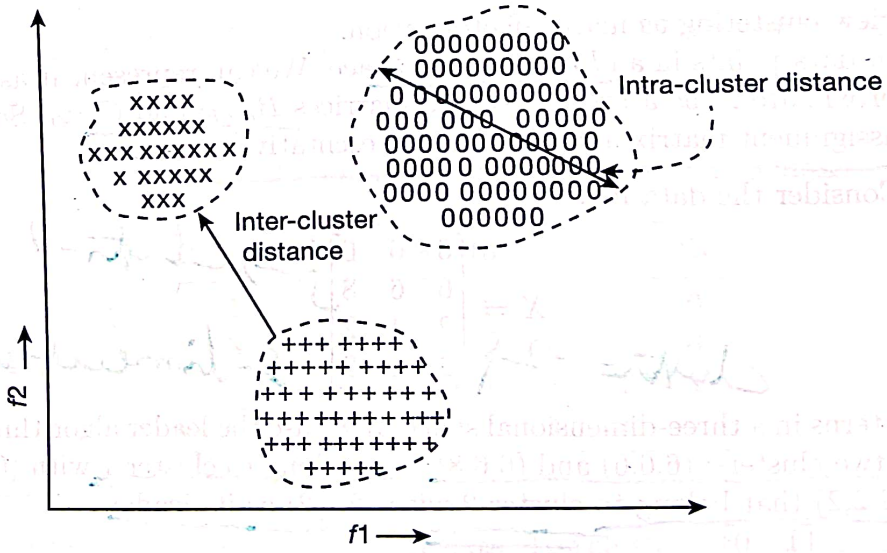

Fig. 7.1 Inter- and intra-cluster distances

The clustering process may ensure that the distance between any two points within the same cluster (intra-cluster distance), as measured by a dissimilarity measure such as Euclidean distance, is smaller than the distance between any two points belonging to different clusters (inter-cluster distance).

This indicates that the similarity between points within a cluster is higher than the similarity between clusters. We illustrate this idea with the following example.

EXAMPLE 6: Let us consider a two-dimensional data set with 11 data points having two features f1 and f2 as listed in Table 7.5.

TABLE 7.5 Data set with two features

Data point	f1	f2
x_1	1	1
x_2	1	3
x_3	2	2
x_4	3	1
x_5	3	3
x_6	4	7
x_7	5	7
x_8	6	9
x_9	7	1
x_{10}	7	3
x_{11}	9	1

Any two points are placed in the same cluster if the distance between them is lower than a certain threshold. In this example, the squared Euclidean distance is used to measure the distance between the points, and a threshold of 10 units is set to cluster them. The squared Euclidean distance (d) between two points, x_i and x_j, is calculated as follows:

$$d(x_i, x_j) = \sum_{k=1}^{l} (x_i(k) - x_j(k))^2,$$

where l represents the dimensionality of the points.

In this example, since we are dealing with two-dimensional data points, l equals 2. Using this formula, the squared Euclidean distance between all pairs of points can be found and is presented in Table 7.6.

In Table 7.6, the clusters are clearly visible within the matrix itself. The table contains three sub-matrices of sizes 5×5, 3×3 and 3×3 that meet the condition of having a maximum value of 10 in any entry. For instance, the sub-matrix of size 5×5, corresponding to the first five patterns (say, Cluster_A $= \{x_1, x_2, x_3, x_4, x_5\}$) has values ranging from 0 to 8, with every other entry in the first five rows exceeding 10. Similarly, it can be deduced that patterns x_6, x_7 and x_8 belong to the second cluster (Cluster_B $= \{x_6, x_7, x_8\}$) while patterns x_9, x_{10} and x_{11} belong to the third cluster (Cluster_C $= \{x_9, x_{10}, x_{11}\}$).

One can observe from Table 7.6 that none of the data points is present in more than one cluster. Such a clustering is called *hard clustering*, otherwise it is known as *soft or overlapping clustering*. We discuss soft clustering in Section 7.6.

TABLE 7.6 Squared Euclidean distance between pairs of data points listed in Table 7.4

Data point	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}
x_1	0	4	2	4	8	45	52	89	36	40	64
x_2	4	0	2	8	4	25	32	61	40	36	68
x_3	2	2	0	2	2	29	34	65	26	26	50
x_4	4	8	2	0	4	37	40	73	16	20	36
x_5	8	4	2	4	0	17	20	45	20	16	40
x_6	45	25	29	37	17	0	1	8	45	25	61
x_7	52	32	34	40	20	1	0	5	40	20	52
x_8	89	61	65	73	45	8	5	0	65	37	73
x_9	36	40	26	16	20	45	40	65	0	4	4
x_{10}	40	36	26	20	16	25	20	37	4	0	8
x_{11}	64	68	50	36	40	61	52	73	4	8	0

Let $\mathfrak{X} = \{x_1, x_2, \ldots, x_n\}$ be n data points which are mapped to C clusters, $\mathfrak{C} = \{X_1, X_2, \ldots, X_c\}$ such that

- $\cup_{i=1}^{C} X_i = \mathfrak{X}$
- $X_i \neq \phi, \forall i \in \{1, 2, \ldots, c\}$

If $\forall i, j, i \neq j, x_i \cap x_j = \phi$, then the clustering is **hard**, else it is **soft**.

7.2.1 Data Abstraction

Clustering is a useful method for data abstraction, and it can be applied to generate clusters of data points that can be represented by their centroid, medoid, leader or some other suitable entity.

The centroid is computed as the sample mean of the data points in a cluster, and it is given by $\frac{1}{n_C} \times \sum(x_i \in C)$, where n_C is the number of patterns in the cluster C. Note that it may not coincide with any specific data point, as shown in Fig. 7.2.

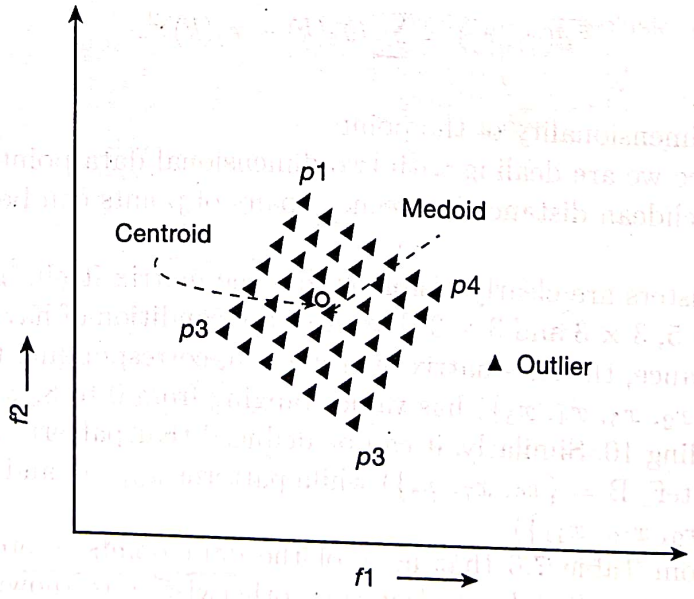

FIG. 7.2 Visualization of centroid and medoid

Clustering algorithms can be classified into hard and soft clustering, depending on whether clusters share data points or not. Hard clustering algorithms generate a partition of the given data set, while hierarchical algorithms generate a nested sequence of partitions. On the other hand, soft clustering algorithms utilize fuzzy sets, rough sets or evolutionary algorithms.

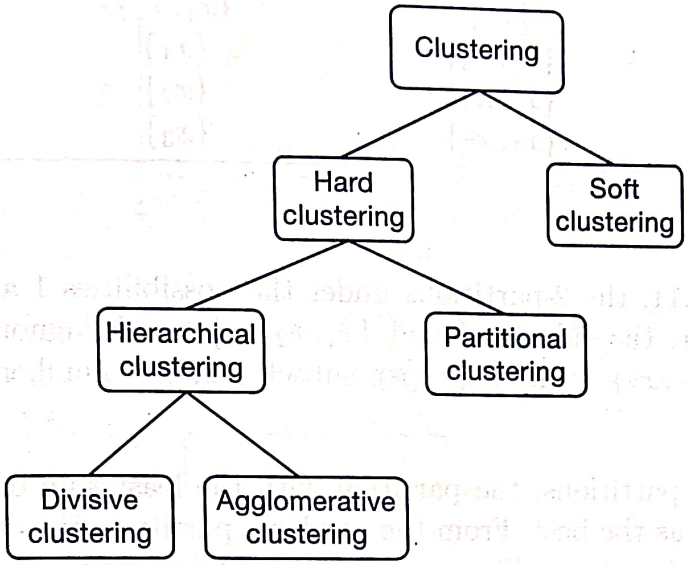

FIG. 7.3 Classification of clustering algorithms

Hierachical Algorithms

The process of generating a sequence of data partitions using hierarchical algorithms can be represented using a tree structure called a **dendrogram**. There are two types of hierarchical algorithms – divisive and agglomerative.

Divisive algorithms follow a top-down approach, where a single cluster with all the patterns is split into smaller clusters at each step until there is only one pattern in each cluster or a collection of singleton clusters. On the other hand, agglomerative algorithms follow a bottom-up approach, where n singleton clusters are created for n input patterns. At each level, the two most similar clusters are merged to reduce the size of the partition by 1.

Once two patterns are placed in the same cluster in agglomerative algorithms, they remain in the same cluster in subsequent levels. Similarly, once two patterns are placed in different clusters in divisive algorithms, they remain in different clusters in subsequent levels.

In the next sections, we discuss divisive clustering, agglomerative clustering and partitional clustering in detail.

7.3 DIVISIVE CLUSTERING

Divisive algorithms are either polythetic, where the division is based on more than one feature, or monothetic, when only one feature is considered at a time. The **polythetic scheme** is based on finding all possible 2-partitions of the data and choosing the best among them. If there are n patterns, the number of distinct 2-partions is given by $\frac{2^n-2}{2} = 2^{n-1} - 1$.

EXAMPLE 10: For example, if the data set contains three patterns, x_1, x_2 and x_3, the possible 2-partitions among these patterns are given in Table 7.11.

TABLE 7.11 All possible 2-partition sets for the patterns x_1, x_2 and x_3

Possibilities	Non-empty subset	Its complement	2-partition
1	$\{x_1\}$	$\{x_2, x_3\}$	$\{x_1, \{x_2, x_3\}\}$
2	$\{x_2\}$	$\{x_1, x_3\}$	$\{x_2, \{x_1, x_3\}\}$
3	$\{x_3\}$	$\{x_1, x_2\}$	$\{x_3, \{x_1, x_2\}\}$
4	$\{x_2, x_3\}$	$\{x_1\}$	$\{\{x_2, x_3\}, x_1\}$
5	$\{x_1, x_3\}$	$\{x_2\}$	$\{\{x_1, x_3\}, x_2\}$
6	$\{x_1, x_2\}$	$\{x_3\}$	$\{\{x_1, x_2\}, x_3\}$

Note that in Table 7.11, the 2-partitions under the possibilities 1 and 4, 2 and 5 and 3 and 6 are repetitions. Further, the subsets ϕ and $\{x_1, x_2, x_3\}$ can be ignored. This is because ϕ is an empty set and $\{x_1, x_2, x_3\}$ is an improper subset. So, the number of possible 2-partitions is $\frac{2^3-2}{2} = 3$.

Among all possible 2-partitions, the partition with the least sum of the sample variances of the two clusters is chosen as the best. From the resulting partition, the cluster with the maximum sample variance is selected and is split into an optimal 2-partition. This process is repeated till we get singleton clusters. If a collection of patterns (data points) is split into two clusters with p patterns $x_1, \ldots, x_p$ in one cluster and q patterns $y_1, \ldots, y_q$ in the other cluster, with the centroids of the two clusters being C1 and C2, respectively, then the sum of the sample variances will be

$$\sum_{i=1}^{p}(x_i - C1)^2 + \sum_{j=1}^{q}(y_j - C2)^2$$

EXAMPLE 11: Consider the data set containing eight points (patterns) with two features as shown in Table 7.12. Figure 7.4 shows the visual representation of these eight data patterns.

TABLE 7.12 Example of data patterns with two features

Patterns	f1	f2
x_1	5.5	5.5
x_2	7	6.5
x_3	7	5.5
x_4	10	6
x_5	10.75	6
x_6	10	8
x_7	10.5	8
x_8	7	8

Figure 7.5 shows the dendrogram corresponding to the divisive clustering using the procedure discussed above. The top of the dendrogram shows a single cluster consisting of all eight data points. By evaluating all possible 2-partitions (out of $2^7 - 1 = 127$), the best 2-partition $\{\{x_1, x_2, x_3, x_8\}, \{x_4, x_5, x_6, x_7\}\}$ is obtained and shown in the dendrogram.

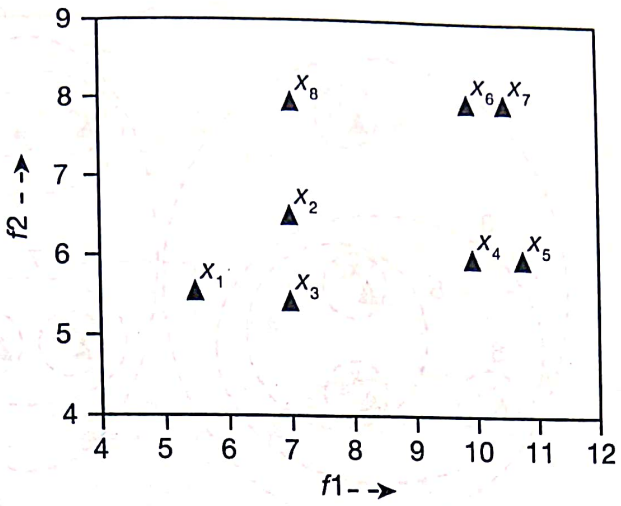

FIG. 7.4 Visual representation of data patterns listed in Table 7.12

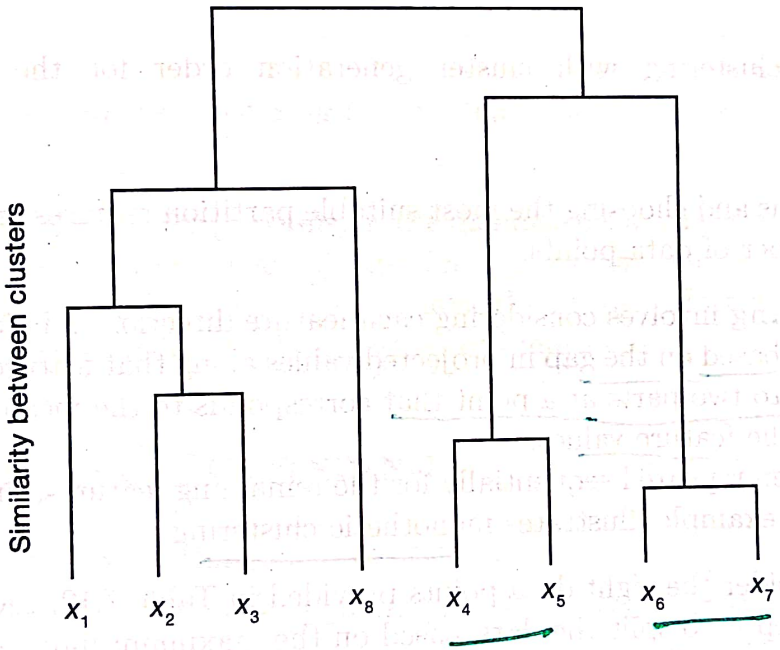

FIG. 7.5 Divisive clustering for polythetic clustering for the data points listed in Table 7.12

The cluster $\{x_4, x_5, x_6, x_7\}$ is then selected to split into two clusters, $\{x_4, x_5\}$ and $\{x_6, x_7\}$. After the split, the three clusters at that level of the dendrogram are $\{x_1, x_2, x_3, x_8\}$, $\{x_4, x_5\}$ and $\{x_6, x_7\}$.

At the subsequent levels, the cluster $\{x_1, x_2, x_3, x_8\}$ is split into $\{x_1, x_2, x_3\}$ and $\{x_8\}$, and $\{x_1, x_2, x_3\}$ is further divided into $\{x_1\}$ and $\{x_2, x_3\}$. This results in five clusters: $\{x_1\}$, $\{x_2, x_3\}$, $\{x_8\}$, $\{x_4, x_5\}$ and $\{x_6, x_7\}$.

The dendrogram in Fig. 7.5 illustrates these clusters, as well as the partitions with 6, 7 and 8 clusters at successive levels. The same is represented in an onion layer diagram (Fig. 7.6), which shows the cluster generation order numerically. Observe that at the final level, each cluster has only one point; such clusters are called singleton clusters.

It is important to highlight that in order to find the optimal 2-partition for a cluster of size m, it is necessary to consider $2^m - 1$ possible 2-partitions and select the best one. Therefore, generating

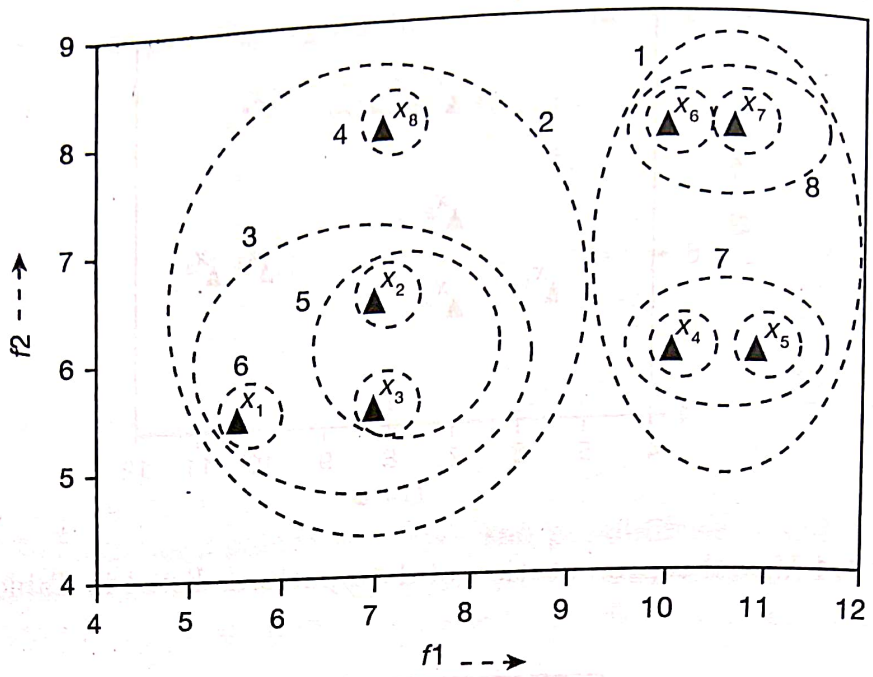

FIG. 7.6 Divisive clustering with cluster generation order for the data points listed in Table 7.12

all possible 2-partitions and choosing the most suitable partition requires an effort of $\mathcal{O}(2^n)$, where n represents the number of data points.

Monothetic clustering involves considering each feature direction individually and dividing the data into two clusters based on the gap in projected values along that feature direction. Specifically, the data set is split into two parts at a point that corresponds to the mean value of the maximum gap observed among the feature values.

This process is then repeated sequentially for the remaining features, further partitioning each cluster. The following example illustrates monothetic clustering.

EXAMPLE 12: Consider the eight data points provided in Table 7.12, each having two features, f1 and f2. The first step is to split the data based on the maximum inter-pattern gap observed in the f1 direction. Sorting the f1 values in ascending order yields $x_1 : 5.5$, $x_2 : 7$, $x_3 : 7$, $x_8 : 7$, $x_4 : 10$, $x_6 : 10$, $x_7 : 10.5$ and $x_5 : 10.75$. The largest gap of 3 units is between x_8 and x_4. We select the mid-point between 7 and 10, which is 8.5 (that is, 7 + 1.5), and use it to split the data into two clusters: C1 = $\{x_1, x_2, x_3, x_8\}$ and C2 = $\{x_4, x_5, x_6, x_7\}$.

Next, each of these clusters is further divided based on the f2 values. Sorting the patterns in C1 by their f2 values gives $x_1 : 5.5$, $x_3 : 5.5$, $x_2 : 6.5$ and $x_8 : 8$. The largest gap of 1.5 units (8 − 6.5 = 1.5 units) occurs between x_2 and x_8. We split C1 at the midpoint 7.25 (that is, 6.5 + 0.75) along the f2 direction, resulting in two clusters: C11 = $\{x_8\}$ and C12 = $\{x_1, x_2, x_3\}$. Similarly, by splitting C2 using the value 7 for f2, we obtain C21 = $\{x_6, x_7\}$ and C22 = $\{x_4, x_5\}$; by splitting C12 using the value 6.25 for f1, we obtain C121 = $\{x_1\}$ and C122 = $\{x_2, x_3\}$. Figure 7.7 shows this monothetic clustering.

However, a challenge with this approach is that in the worst case, the initial data set may be divided into 2^l clusters when considering all l features. Having such a large number of clusters might not be practical, necessitating an additional merging phase. In this phase, pairs of clusters are selected based on their proximity and merged into a single cluster. This process is repeated until the desired number of clusters is achieved.

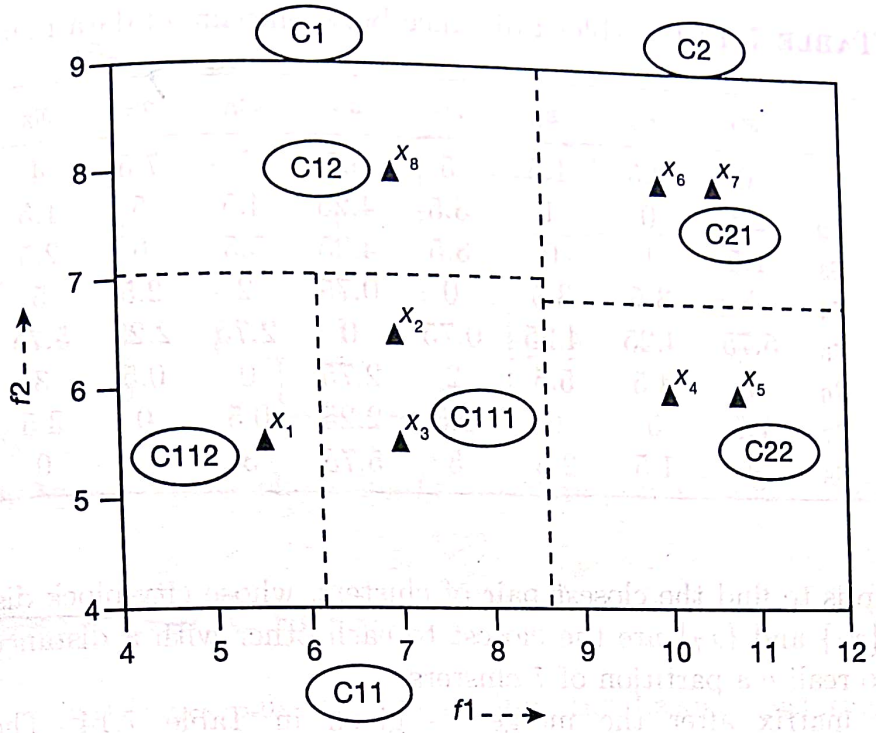

FIG. 7.7 Monothetic clustering for the data points listed in Table 7.12

One commonly used criterion for proximity is the distance between the centroids of the clusters. The time complexity of sorting the n elements and finding the maximum inter-pattern gap in each feature direction is $\mathcal{O}(n \log n)$. Considering l features, the overall effort is $\mathcal{O}(ln \log n)$. However, this approach may become infeasible for large values of l and n.

7.4 AGGLOMERATIVE CLUSTERING

An agglomerative clustering algorithm generally uses the following steps:

1. Compute the proximity matrix for all pairs of patterns in the data set.
2. Find the closest pair of clusters based on the computed proximity measure and merge them into a single cluster. Update the proximity matrix to reflect the merge, adjusting the distances between the newly formed cluster and the remaining clusters.
3. If all the patterns belong to a single cluster, terminate the algorithm. Otherwise, go back to Step 2 and repeat the process until all the patterns are in one cluster.

By iteratively merging the closest clusters, the algorithm gradually builds a hierarchy of clusters, with each iteration reducing the number of clusters until a stopping criterion is met. The following example illustrates agglomerative clustering.

EXAMPLE 13: Consider the eight data points specified in Table 7.12. To start with, each data point is a singleton cluster. As per the method, the first step is to compute the proximity between the data points. Table 7.13 shows the proximity between the data points using city-block distance. Note that the city-block (Manhattan) distance between any two data points, x_1 and x_2, is given by $\mathrm{M}(x_1, x_2) = \sum_{i=1}^{l}(\mid x_1(i) - x_2(i) \mid)$, where l is the number of features in the data point.

TABLE 7.13 City-block distance between pairs of data points

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
x_1	0	2.5	1.5	5	5.5	7	7.5	4
x_2	2.5	0	1	3.5	4.25	4.5	5	1.5
x_3	1.5	1	0	3.5	4.25	5.5	6	2.5
x_4	5	3.5	3.5	0	0.75	2	2.5	5
x_5	5.75	4.25	4.25	0.75	0	2.75	2.25	5.75
x_6	7	4.5	5.5	2	2.75	0	0.5	3
x_7	7.5	5	6	2.5	2.25	0.5	0	3.5
x_8	4	1.5	2.5	5	5.75	3	3.5	0

The second step is to find the closest pair of clusters, whose city-block distance is minimum. Since the clusters $\{x_6\}$ and $\{x_7\}$ are the closest to each other with a distance of 0.5 units, they are merged as C1 to realise a partition of 7 clusters.

The proximity matrix after the merger is given in Table 7.14. The merging uses a *single-link* strategy. That is, the distance between any pair of clusters Cp and Cq is $min_{x_i \in Cp \text{ and } x_j \in Cq}(d(x_i, x_j))$.

TABLE 7.14 City-block distance among the data points after merging x_6 and x_7 as one cluster

	x_1	x_2	x_3	x_4	x_5	C1=$\{x_6, x_7\}$	x_8
x_1	0	2.5	1.5	5	5.5	7	4
x_2	2.5	0	1	3.5	4.25	4.5	1.5
x_3	1.5	1	0	3.5	4.25	5.5	2.5
x_4	5	3.5	3.5	0	0.75	2	5
x_5	5.75	4.25	4.25	0.75	0	2.75	5.75
$C1 = \{x_6, x_7\}$	7	4.5	5.5	2	2.25	0	3
x_8	4	1.5	2.5	5	5.75	3	0

In the third step, we need to repeat Step 2 till we reach a single cluster of the required number of clusters. In Table 7.14, one can observe that the pair of clusters $\{x_4\}$ and $\{x_5\}$ has a minimum distance (that is, 0.75 units) and hence the clusters are merged next to form C2 = $\{x_4, x_5\}$. Similarly, the clusters $\{x_2\}$ and $\{x_3\}$ are merged to create C3 = $\{x_2, x_3\}$. Then, C3 is merged with cluster $\{x_1\}$, resulting in C4 = $\{x_1, x_2, x_3\}$. Subsequently, C4 is merged with cluster $\{x_8\}$, leading to C5 = $\{x_1, x_2, x_3, x_8\}$. Clusters C2 and C1 are then merged to form C6 = $\{x_4, x_5, x_6, x_7\}$.

At this point, there are two clusters remaining, namely, C5 and C6. The process can be terminated once the desired number of clusters is achieved. The dendrogram in Fig. 7.8 illustrates the merging of clusters at different levels.

The time complexity of agglomerative clustering is $\mathcal{O}(n^2)$, where n is the number of data points. This is to compute the proximity between all pairs of data points in the data set. Since the computed proximity matrix needs to be stored, the space complexity is also $\mathcal{O}(n^2)$.

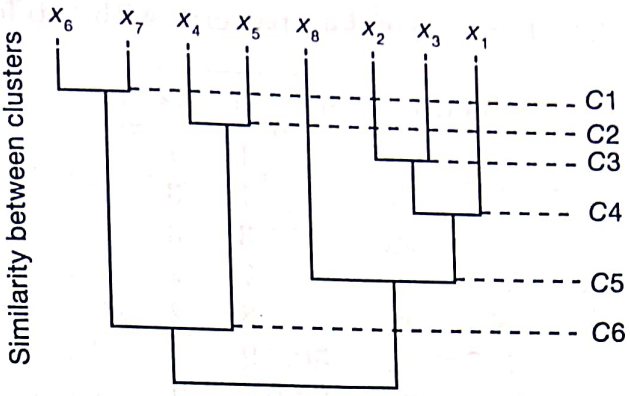

FIG. 7.8 Agglomerative clustering for the data points listed in Table 7.12

7.5 PARTITIONAL CLUSTERING

We have discussed a partitional clustering algorithm, the Leader clustering algorithm, in an earlier chapter. One of the most popular algorithms under this category is the k-means clustering algorithm.

7.5.1 *K*-Means Clustering

The k-means algorithm can be described in the following steps:

1. Select k initial cluster centres from the given n data points. The remaining $(n - k)$ data points are then assigned to one of the k clusters based on their proximity to the closest cluster centre.
2. Compute the new cluster centres based on the current assignment of data points. This is done by calculating the mean or centroid of all the data points belonging to each cluster.
3. Assign each of the n data points to the cluster centre that is closest in proximity to it.
4. Check if there have been any changes in the assignment of data points to clusters. If no changes have occurred, terminate the algorithm; otherwise, go back to Step 2 and repeat the process.

By iteratively updating the cluster centres and reassigning data points until convergence, the k-means algorithm aims to find a partition that minimizes the overall within-cluster variance or distance.

We illustrate the algorithm using the following example.

EXAMPLE 14: Consider the data set containing eight points with two features as shown in Table 7.15.

Let us assume that the number of clusters, $k = 3$. If we select the initial clusters as x_1, x_4 and x_7, then Cluster 1 has (1, 1) as its cluster centre, Cluster 2 has (7, 2) as its cluster centre and Cluster 3 has (7, 9) as its cluster centre.

Table 7.16 shows the Euclidean distance (a proximity measure) between the cluster centre and the other data points. The patterns x_2, x_3, x_5, x_6 and x_8 are assigned to their respective clusters as shown in the last column of Table 7.16.

TABLE 7.15 Example of data patterns with two features

Data points	f1	f2
x_1	1	1
x_2	1	3
x_3	3	3
x_4	7	2
x_5	8	2
x_6	9	3
x_7	7	9
x_8	8	9

TABLE 7.16 Euclidean distances with cluster assignments

Data points	Euclidean distance from C1	Euclidean distance from C2	Euclidean distance from C3	Assigned cluster
x_1	**0**	$\sqrt{37}$	10	C1
x_2	**2**	$\sqrt{37}$	$\sqrt{72}$	C1
x_3	$\sqrt{8}$	$\sqrt{17}$	$\sqrt{52}$	C1
x_4	$\sqrt{37}$	**0**	7	C2
x_5	$\sqrt{50}$	**1**	$\sqrt{50}$	C2
x_6	$\sqrt{68}$	$\sqrt{5}$	$\sqrt{40}$	C2
x_7	10	7	**0**	C3
x_8	$\sqrt{113}$	$\sqrt{50}$	**1**	C3

The next step involves computing the new cluster centres. The new cluster centre of C1 will be the mean of the patterns in Cluster 1(mean of x_1, x_2 and x_3), which will be (1.67, 2.33). The cluster centre of C2 will be (8,2.33) and the cluster centre of $C3$ will be (7.5, 9).

The patterns are again assigned to the closest cluster depending on the distance from the cluster centres. Now, x_1, x_2 and x_3 are assigned to Cluster 1, x_4, x_5 and x_6 are assigned to Cluster 2 and x_7 and x_8 are assigned to Cluster 3. Table 7.17 shows the Euclidean distance from the new cluster centre to the data points with their new assigned clusters.

Since there is no change in the clusters formed, this is the final set of clusters. The visual representation of the clusters is given in Fig. 7.9.

One of the important points to be noted in k-means clustering is that the algorithm is *sensitive to the selection of initial centroids*. That is, cluster formation changes with the initially chosen cluster centres. For example, if we select the initial cluster centre as x_1, x_2 and x_3, the clustering generated is as shown in Fig. 7.10. (Cluster formation using k-means clustering with $k = 3$ is left as an exercise to the reader.)

The goodness of the k-means clustering algorithm is based on the sum of the squared deviations of data points in a cluster from its centre. More formally, if C_i is the i^{th} cluster and μ_i is its centre,

TABLE 7.17 Euclidean distances with cluster assignments

Data points	Euclidean Euclidean from C1	Euclidean distance from C2	Euclidean distance from C3	Assigned cluster
x_1	**1.49**	7.13	10.3	C1
x_2	**0.95**	7	8.85	C1
x_3	**1.49**	5.24	7.5	C1
x_4	5.34	**1.05**	7.02	C2
x_5	6.34	**0.33**	7.02	C2
x_6	7.36	**1.2**	6.18	C2
x_7	8.54	6.74	**0.5**	C3
x_8	9.2	6.67	**0.5**	C3

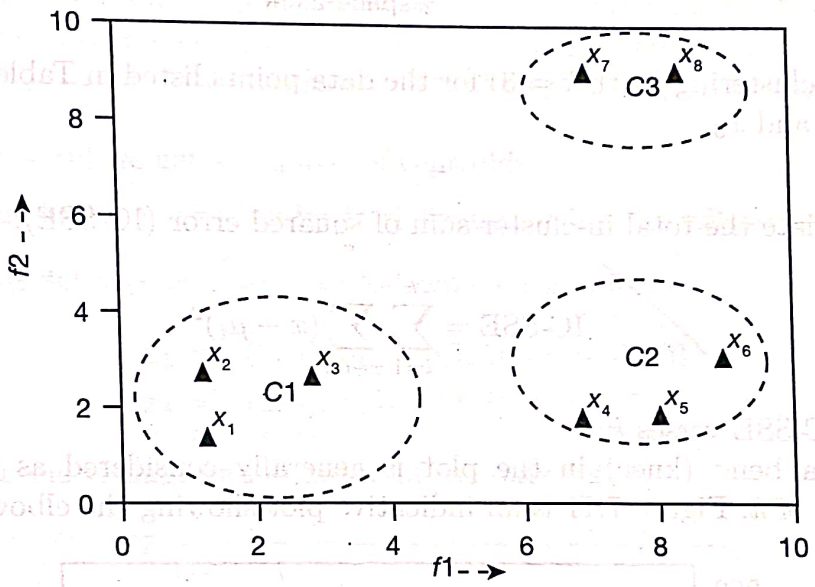

FIG. 7.9 K-means clustering (with $k = 3$) for the data points listed in Table 7.15

then the criterion function minimised by the algorithm is

$$\sum_{i=1}^{k} \sum_{x \in C_i} (x - \mu_i)^t (x - \mu_i)$$

This is called the **sum-of-squared-error criterion**. This should be minimal for optimal clustering.

Choosing a suitable value for k is a practical challenge in k-means clustering. A commonly employed strategy is to initialize the cluster centres for k in a way that maximizes their distance from each other, which has shown to be effective in real-world scenarios. Alternatively, the **elbow method** is a popular approach to determine the value of k. The steps used in the elbow method are as follows:

1. Perform the k-means clustering algorithm (as discussed earlier) for different values of k.

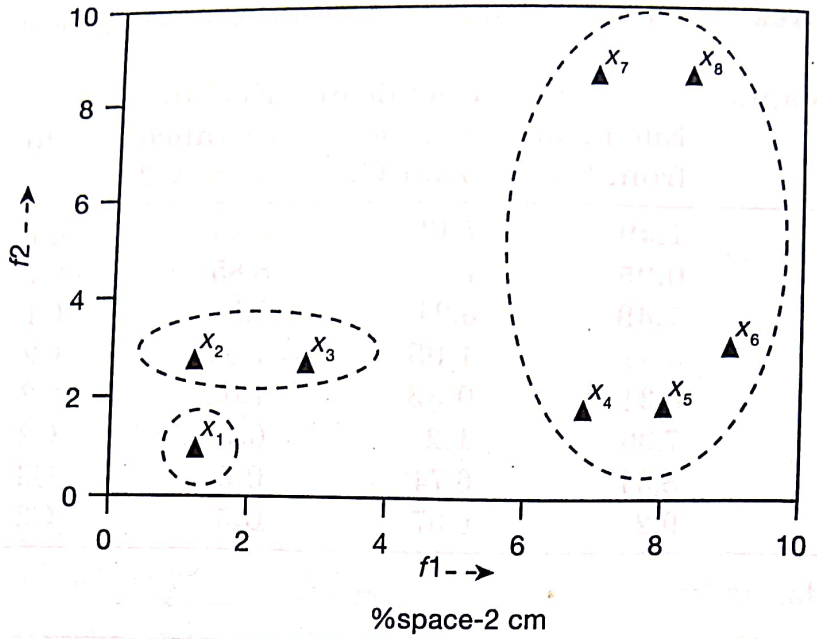

FIG. 7.10 K-means clustering (with $k = 3$) for the data points listed in Table 7.15 but with initial cluster centre, x_1, x_2 and x_3

2. For each k, calculate the total in-cluster sum of squared error (IC-SSE):

$$\text{IC-SSE} = \sum_{i=1}^{k} \sum_{x \in C_i} (x - \mu_i)^2$$

3. Plot the curve, IC-SSE verses k.
4. The location of a bend (knee) in the plot is generally considered as an indicator of the appropriate value of k. Figure 7.11 is an indicative plot showing the elbow method.

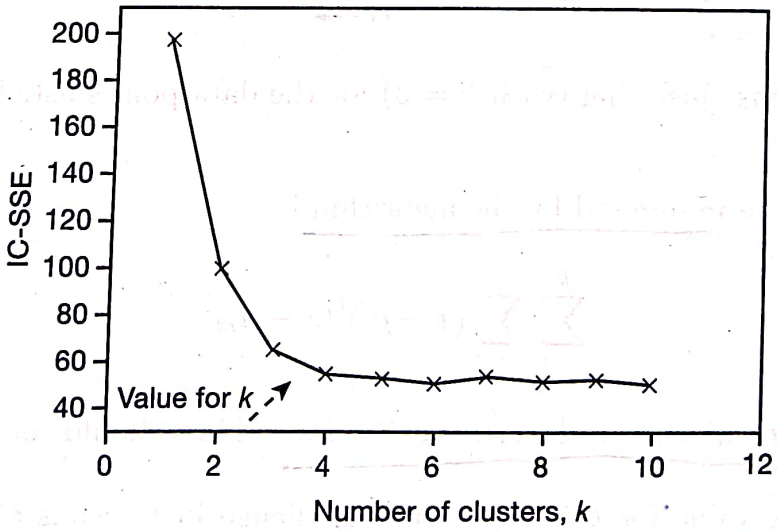

FIG. 7.11 Elbow method for approximating the k value

The k-means clustering algorithm has a time complexity of $\mathcal{O}(nlkp)$, where n represents the number of data points, l denotes the number of features in each data point, k specifies the number of clusters and p indicates the number of iterations. As for space requirements, it is $\mathcal{O}(kn)$. These

TABLE 7.18 Euclidean distances and probabilities between x_1 and other data points

Data point	$d(x_1, x_i)$	$\text{Min}(d(x_1, x_i))$	$Prob_i = \frac{d(x_1,x_i)}{\sum d(x_1,x_i)}$
x_1	-	-	-
x_2	1.41	1.41	0.066
x_3	5.39	5.39	0.254
x_4	4.12	4.12	0.194
x_5	6.08	6.08	**0.286**
x_6	4.24	4.24	0.2

Table 7.19 gives the Euclidean distances and probabilities between x_1, x_5 and other data points.

TABLE 7.19 Euclidean distances and probabilities between x_1, x_5 and other data points

Data point	$d(x_1, x_i), d(x_5, x_i)$	$\text{Min}(d(x_1, x_i), d(x_5, x_i))$	$Prob_i = \frac{Min(d(x_1,x_i),d(x_5,x_i))}{\sum Min(d(x_1,x_i),d(x_5,x_i))}$
x_1	-	-	-
x_2	1.41, 7.0	1.41	0.09
x_3	5.39, 7.2	5.39	**0.36**
x_4	4.12, 2	4.12	0.27
x_5	-	-	-
x_6	4.24, 9.22	4.24	0.28

Choose x_3 as the third centroid, as its probability in the list is maximum. We now run the k-means clustering algorithm with initial centroids, x_1, x_2 and x_3.

Note that in k-means++, we have some initialization (set-up) cost when compared to the conventional k-means algorithm. But convergence tends to be faster and better with lower heterogeneity. One can visualize this in Fig. 7.13.

Figure 7.13 represents cluster formation using random initial centroids and selection of initial centroids by k-means++. Figure 7.13 (a) shows the randomly selected initial three centroids (circled), Fig. 7.13 (b) shows three possible clusters after executing the k-means algorithm. Figure 7.13 (c) shows the initial 3-centroids (circled) by k-means++ and Fig. 7.13 (d) shows cluster formation after executing the k-means clustering algorithm.

Visually it is clear that (i) the amount of centroid movement is less if we use the k-means++ initialization step (the grey arrows in both the cases shows this) and (ii) the cluster formed has lower heterogeneity with k-means++ initialization.

7.5.3 Soft Partitioning

The conventional k-means algorithm assigns data points exclusively to the cluster which is nearest to it, and this leads to a hard partition. Due to inadequate selection of the initial cluster centre, the algorithm may result in a locally optimal solution. Various solutions were proposed in the literature to address this issue.

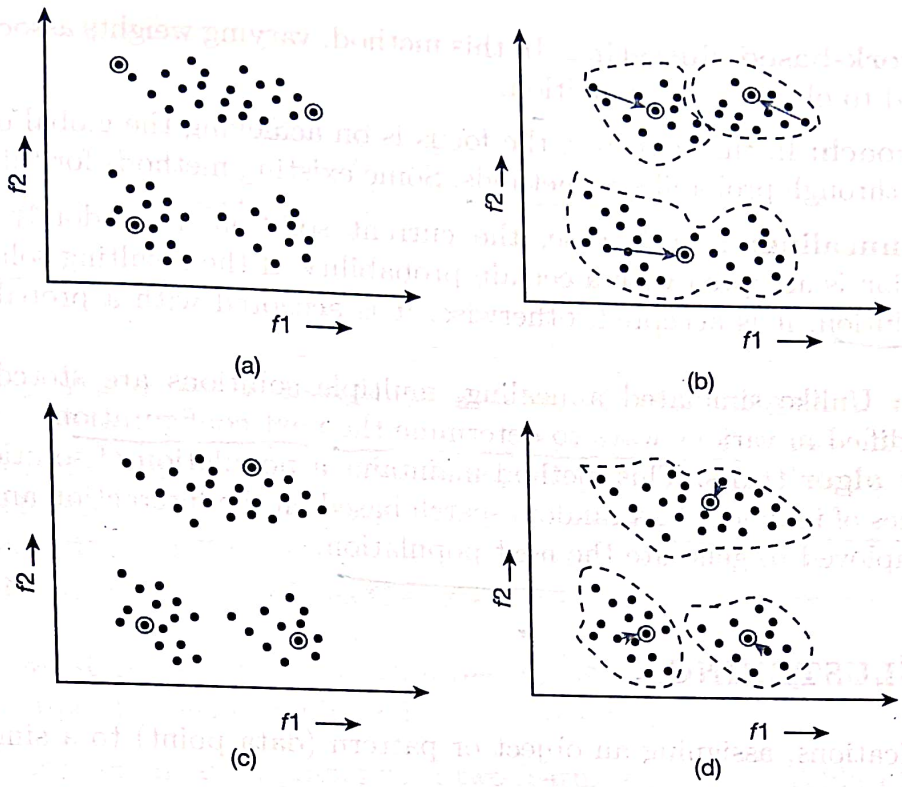

FIG. 7.13 Visualization of goodness of using k-means clustering for initialization (for colour figure, please see Colour Plate 2)

Split and merge strategy: Here, clusters generated by the k-means algorithm are evaluated for potential splitting and merging.

If the sample variance of a cluster exceeds a specified threshold value (τ_v), the cluster is divided into two clusters by selecting the most dissimilar pair of patterns as initial seeds. Likewise, if the distance between the centres of two clusters is below a distance threshold (τ_d), they are merged into a single cluster. By appropriately setting τ_v and τ_d, it becomes possible to obtain an optimal partition. For example, the cluster $\{x_4, x_5, x_6, x_7, x_8\}$ in Fig. 7.10 is selected for splitting based on the variance surpassing the user-defined threshold, τ_v. By choosing either x_4, x_5 or x_6 as initial centroids for one cluster and either x_7 or x_8 for the other, we obtain the clusters $\{x_4, x_5, x_6\}$ and $\{x_7, x_8\}$.

Additionally, by merging the clusters $\{x_1\}$ and $\{x_2, x_3\}$ because the distance between x_1 and the centroid of $\{x_2, x_3\}$ falls below the user-defined threshold τ_d, the partition shown in Fig. 7.9 can be achieved. However, implementing this strategy is challenging due to the difficulty of estimating τ_v and τ_d.

Multiple membership approach: This approach involves the influence of not only the closest cluster centre to a given data point but also neighboring cluster centres. These cluster centroids are affected to varying degrees by the data point, with the closest centre being more influenced than others. Several existing methods for this approach include:

- **Fuzzy clustering:** Each data point is assigned to multiple clusters, typically more than one, based on a membership value. The value is computed using the data point and the corresponding cluster centroid.
- **Rough clustering:** Each cluster is assumed to have both a non-overlapping part and an overlapping part. Data points in the non-overlapping portion belong exclusively to that cluster, while data points in the overlapping part may belong to multiple clusters.

- **Neural network-based clustering:** In this method, varying weights associated with the data points are used to obtain a soft partition.

Stochastic approach: In this category, the focus is on achieving the global optimal solution for cluster formation through probabilistic methods. Some existing methods for this approach include:

- **Simulated annealing:** In this case, the current solution is randomly updated, and the resulting solution is accepted with a certain probability. If the resulting solution is better than the current solution, it is accepted; otherwise, it is accepted with a probability ranging from 0 to 1.
- **Tabu search:** Unlike simulated annealing, multiple solutions are stored, and the current solution is modified in various ways to determine the next configuration.
- **Evolutionary algorithms:** This method maintains a population of solutions. In addition to the fitness values of individuals, a random search based on the interaction among solutions with mutation is employed to generate the next population.

7.6 Soft Clustering

In numerous applications, assigning an object or pattern (data point) to a single cluster is often not feasible.

For instance, when assigning a document in text mining, it is possible for the same document to fall under both the "politics" and "sports" categories. Similarly, a person in a social network can belong to multiple subject areas within social media. This means that an object or pattern can be assigned to more than one category. This type of categorization, where an object or pattern can be associated with multiple groups, is known as soft partitioning or soft clustering.

The general form of soft clustering involves considering n data points, denoted as $x_1, x_2, \ldots, x_n$, and k clusters, denoted as $C_1, C_2, \ldots, C_k$. In soft clustering, a data point can belong to one or more clusters. This relationship can be represented by an $n \times k$ matrix, where a non-zero entry in the cell (i, j) indicates that the data point x_i belongs to the cluster C_j, while $(i, j) = 0$ indicates otherwise.

In each row, the number of non-zero entries can range from 1 to k. Excluding the all-zero vector, which implies that x_i must be assigned to at least one of the k clusters, there are $(2^k - 1)$ possible choices for each row in terms of the non-zero entries. Therefore, the total number of possibilities for all n rows is $(2^k - 1)^n$. If we store one of p possible values in each cell to indicate the extent to which x_i belongs to C_j, then the number of possibilities is bounded by $(p^k - 1)^n$, considering the number of distinct values each cell can assume as p.

7.6.1 Fuzzy C-Means Clustering

In this type, each data point x_i can be assigned to a cluster C_j with a membership value μ_{ij}, which represents the degree to which x_i belongs to C_j. The membership value μ_{ij} is assumed to range between 0 and 1, and for each data point x_i, the sum of all μ_{ij} values across clusters is equal to 1.

The fuzzy c-means clustering algorithm follows an iterative scheme similar to the k-means algorithm. It begins by identifying c initial cluster centres from the data set or membership values between each data point and the cluster. The following steps are then performed iteratively:

1. Compute membership values:

$$\mu_{ij} = \frac{\mathrm{d}(x_j, C_i)^{-1/(M-1)}}{\sum_{l=1}^{c} \mathrm{d}(x_j, C_l)^{-1/(M-1)}}$$

2. Compute the fuzzy cluster centroid:

$$C_i = \frac{\sum_{j=1}^{m} (\mu_{ij})^M \times x_j}{\sum_{j=1}^{m} (\mu_{ij})^M}$$

In the above expressions, μ_{ij} represents the membership value of data point x_j in cluster C_i, where $j = 1, 2, \ldots, n$ and $i = 1, 2, \ldots, c$; $d(x_j, C_i)$ denotes the Euclidean distance between x_j and C_i; C_i represents the centroid of the ith cluster and M is the fuzzyifying constant which determines the behaviour of the algorithm. When $M = 1$, the fuzzy algorithm functions similar to the hard k-means algorithm, where $\mu_{ij} = 1$ when X_j is assigned to C_i. As M increases or tends to infinity, μ_{ij} approaches $\frac{1}{c}$, as the exponents in both the numerator and denominator tend towards 0.

3. Repeat the above steps, and the algorithm terminates when there is no significant change in the computed values for membership and cluster centroid.

EXAMPLE 16: Consider the data points with two features shown in Table 7.21.

TABLE 7.20 Example of a data set with two features

Data points	f1	f2
x_1	1	1
x_2	2	2
x_3	4	3
x_4	5	3

Let the number of clusters, $c = 2$ and $M = 2$. To start with, assume the following membership values between the data points and the clusters($\mu^{(0)}$):

	x_1	x_2	x_3	x_4
C1	1	1	0	0
C2	0	0	1	1

The cluster centres are calculated as follows:

$$C1_{f1} = \frac{\mu_{11}^2 * x_1(f1) + \mu_{12}^2 * x_2(f1) + \mu_{13}^2 * x_3(f1) + \mu_{14}^2 * x_4(f1)}{\mu_{11}^2 + \mu_{12}^2 + \mu_{13}^2 + \mu_{14}^2}$$

$$C1(f1) = \frac{1^2 * 1 + 1^2 * 2 + 0^2 * 4 + 0^2 * 5}{1^2 + 1^2 + 0^2 + 0^2}$$

$$C1(f1) = \frac{3}{2} = 1.5$$

$$C1(f2) = \frac{\mu_{11}^2 * X1_{f2} + \mu_{12}^2 * X2_{f2} + \mu_{13}^2 * X3_{f2} + \mu_{14}^2 * X4_{f2}}{\mu_{11}^2 + \mu_{12}^2 + \mu_{13}^2 + \mu_{14}^2}$$

$$C1(f2) = \frac{1^2 * 1 + 1^2 * 2 + 0^2 * 3 + 0^2 * 3}{1^2 + 1^2 + 0^2 + 0^2}$$

$$C1(f2) = \frac{3}{2} = 1.5$$

So, the updated cluster centre for $C1 = (1.5, 1.5)$.

Similarly,

$$C2(f1) = \frac{\mu_{21}^2 * x_1(f1) + \mu_{22}^2 * x_2(f1) + \mu_{23}^2 * x_3(f1) + \mu_{24}^2 * x_4(f1)}{\mu_{21}^2 + \mu_{22}^2 + \mu_{23}^2 + \mu_{24}^2}$$

$$C2(f1) = \frac{0^2 * 1 + 0^2 * 2 + 1^2 * 4 + 1^2 * 5}{0^2 + 0^2 + 1^2 + 1^2}$$

$$C2(f1) = \frac{9}{2} = 4.5$$

$$C2(f2) = \frac{\mu_{21}^2 * X1_{f2} + \mu_{22}^2 * X2_{f2} + \mu_{23}^2 * X3_{f2} + \mu_{24}^2 * X4_{f2}}{\mu_{21}^2 + \mu_{22}^2 + \mu_{23}^2 + \mu_{24}^2}$$

$$C2(f2) = \frac{0^2 * 1 + 0^2 * 2 + 1^2 * 3 + 1^2 * 3}{0^2 + 0^2 + 1^2 + 1^2}$$

$$C2(f2) = \frac{6}{2} = 3$$

So, the updated cluster centre for $C2 = (4.5, 3)$.

The next step is recalculation of membership values. As a part of this step, let us first calculate the Euclidean distance between the cluster centres and the data points. This is done as follows:

$$\mathrm{d}(C1, x_1) = \mathrm{d}((1.5, 1.5), (1, 1)) = \sqrt{1.5 - 1)^2 + (1.5 - 1)^2} = 0.71$$

$$\mathrm{d}(C1, x_2) = \mathrm{d}((1.5, 1.5), (2, 2)) = \sqrt{(1.5 - 2)^2 + (1.5 - 2)^2} = 0.71$$

$$\mathrm{d}(C1, x_3) = \mathrm{d}((1.5, 1.5), (4, 3)) = \sqrt{(1.5 - 4)^2 + (1.5 - 3)^2} = 2.92$$

$$\mathrm{d}(C1, x_4) = \mathrm{d}((1.5, 1.5), (5, 3)) = \sqrt{(1.5 - 5)^2 + (1.5 - 3)^2} = 3.81$$

$$\mathrm{d}(C2, x_1) = \mathrm{d}((4.5, 3), (1, 1)) = \sqrt{(4.5 - 1)^2 + (3 - 1)^2} = 4.03$$

$$\mathrm{d}(C2, x_2) = \mathrm{d}((4.5, 3), (2, 2)) = \sqrt{(4.5 - 2)^2 + (3 - 2)^2} = 2.69$$

$$\mathrm{d}(C2, x_3) = \mathrm{d}((4.5, 3), (4, 3)) = \sqrt{(4.5 - 4)^2 + (3 - 3)^2} = 0.25$$

$$\mathrm{d}(C2, x_4) = \mathrm{d}((4.5, 3), (5, 3)) = \sqrt{(4.5 - 5)^2 + (3 - 3)^2} = 0.25$$

The membership values are:

$$\mu_{11} = \frac{\frac{1}{\mathrm{d}(C1, x_1)}}{\frac{1}{\mathrm{d}(C1, x_1)} + \frac{1}{\mathrm{d}(C2, x_1)}}$$

$$\mu_{11} = \frac{\frac{1}{0.71}}{\frac{1}{0.71} + \frac{1}{4.03}} = 0.85$$

$$\mu_{12} = \frac{\frac{1}{d(C1,x_2)}}{\frac{1}{d(C1,x_2)} + \frac{1}{d(C2,x_2)}}$$

$$\mu_{12} = \frac{\frac{1}{0.71}}{\frac{1}{0.71} + \frac{1}{2.69}} = 0.79$$

$$\mu_{13} = \frac{\frac{1}{d(C1,x_3)}}{\frac{1}{d(C1,x_3)} + \frac{1}{d(C2,x_3)}}$$

$$\mu_{13} = \frac{\frac{1}{2.92}}{\frac{1}{2.92} + \frac{1}{0.25}} = 0.08$$

$$\mu_{14} = \frac{\frac{1}{d(C1,x_4)}}{\frac{1}{d(C1,x_4)} + \frac{1}{d(C2,x_4)}}$$

$$\mu_{14} = \frac{\frac{1}{3.81}}{\frac{1}{3.81} + \frac{1}{0.25}} = 0.06$$

Similarly, μ_{21}, μ_{22}, μ_{23}, μ_{24} can be computed and their values are 0.15, 0.21, 0.92 and 0.94, respectively. Note that for a given data point x_j, $\sum_{i=1}^{2}(\mu_{ij}) = 1$.

Now the updated membership values $\mu^{(1)}$ are given below:

	x_1	x_2	x_3	x_4
$C1$	0.85	0.79	0.08	0.06
$C2$	0.15	0.21	0.92	0.94

Since we do not end up with the same cluster centres or the same membership values, we iterate for the next step. [Note: To converge in a finite number of iterations, some threshold values, either for the difference between the current and previous membership values or for the difference between the current and the previous cluster centre values, can be specified.]

The updated cluster centres for the updated membership values ($\mu^{(1)}$) are as follows:

$$C1(f1) = \frac{0.85^2 * 1 + 0.79^2 * 2 + 0.08^2 * 4 + 0.06^2 * 5}{0.85^2 + 0.79^2 + 0.08^2 + 0.06^2}$$

$$C1(f1) = \frac{2.04}{1.36} = 1.5$$

$$C1_{f2} = \frac{0.85^2 * 1 + 0.79^2 * 2 + 0.08^2 * 3 + 0.06^2 * 3}{0.85^2 + 0.79^2 + 0.08^2 + 0.06^2}$$

$$C1(f2) = \frac{2.01}{1.36} = 1.48$$

So, the updated cluster centre for $C1 = (1.5, 1.48)$.

$$C2(f1) = \frac{0.15^2 * 1 + 0.21^2 * 2 + 0.92^2 * 4 + 0.94^2 * 5}{0.15^2 + 0.21^2 + 0.92^2 + 0.94^2}$$

$$C2(f1) = \frac{7.9143}{1.797} = 4.4$$

$$C2(f2) = \frac{0.15^2 * 1 + 0.21^2 * 2 + 0.92^2 * 3 + 0.94^2 * 3}{0.15^2 + 0.21^2 + 0.92^2 + 0.94^2}$$

$$C2(f2) = \frac{5.307}{1.7966} = 2.95$$

So, the updated cluster centre for $C2 = (4.4, 2.95)$.

The next step is recalculation of membership values. As a part of this step, let us first calculate the Euclidean distance between the cluster centres and the data points. This is done as follows:

$$\mathrm{d}(C1, x_1) = \mathrm{d}((1.5, 1.48), (1, 1)) = \sqrt{(1.5-1)^2 + (1.48-1)^2} = 0.693$$

$$\mathrm{d}(C1, x_2) = \mathrm{d}((1.5, 1.48), (2, 2)) = \sqrt{(1.5-2)^2 + (1.48-2)^2} = 0.721$$

$$\mathrm{d}(C1, x_3) = \mathrm{d}((1.5, 1.48), (4, 3)) = \sqrt{(1.5-4)^2 + (1.48-3)^2} = 2.925$$

$$\mathrm{d}(C1, x_4) = \mathrm{d}((1.5, 1.48), (5, 3)) = \sqrt{(1.5-5)^2 + (1.48-3)^2} = 3.815$$

$$\mathrm{d}(C2, x_1) = \mathrm{d}((4.4, 2.95), (1, 1)) = \sqrt{(4.4-1)^2 + (2.95-1)^2} = 3.919$$

$$\mathrm{d}(C2, x_2) = d((4.4, 2.95), (2, 2)) = \sqrt{(4.4-2)^2 + (2.95-2)^2} = 2.58$$

$$\mathrm{d}(C2, x_3) = \mathrm{d}((4.4, 2.95), (4, 3)) = \sqrt{(4.4-4)^2 + (2.95-3)^2} = 0.40$$

$$\mathrm{d}(C2, x_4) = \mathrm{d}((4.4, 2.95), (5, 3)) = \sqrt{(4.4-5)^2 + (2.95-3)^2} = 0.60$$

The updated membership values are:

$$\mu_{11} = \frac{\frac{1}{0.693}}{\frac{1}{0.693} + \frac{1}{3.919}} = 0.85$$

$$\mu_{12} = \frac{\frac{1}{0.721}}{\frac{1}{0.721} + \frac{1}{2.58}} = 0.79$$

$$\mu_{13} = \frac{\frac{1}{2.925}}{\frac{1}{2.925} + \frac{1}{0.40}} = 0.12$$

$$\mu_{14} = \frac{\frac{1}{3.815}}{\frac{1}{3.815} + \frac{1}{0.60}} = 0.14$$

Similarly, μ_{21}, μ_{22}, μ_{23}, μ_{24} can be computed and their values are 0.15, 0.21, 0.88 and 0.86, respectively.

Now the updated membership values $\mu^{(2)}$ are given below:

	x_1	x_2	x_3	x_4
$C1$	0.85	0.79	0.12	0.14
$C2$	0.15	0.21	0.88	0.86

Since we do not end up with the same cluster point or the same membership values, we iterate for the next step.

The updated cluster centres for the updated membership values ($\mu^{(2)}$) are as follows:

$$C1(f1) = \frac{0.85^2 * 1 + 0.79^2 * 2 + 0.12^2 * 4 + 0.14^2 * 5}{0.85^2 + 0.79^2 + 0.12^2 + 0.14^2}$$

$$C1(f1) = \frac{2.04}{1.355} = 1.5$$

$$C1(f2) = \frac{0.85^2 * 1 + 0.79^2 * 2 + 0.12^2 * 3 + 0.14^2 * 3}{0.85^2 + 0.79^2 + 0.12^2 + 0.14^2}$$

$$C1(f2) = \frac{2.01}{1.355} = 1.48$$

The updated cluster centre for $C1 = (1.5, 1.48)$.

$$C2(f1) = \frac{0.15^2 * 1 + 0.21^2 * 2 + 0.92^2 * 4 + 0.94^2 * 5}{0.15^2 + 0.21^2 + 0.92^2 + 0.94^2}$$

$$C2(f1) = \frac{7.9143}{1.797} = 4.4$$

$$C2(f2) = \frac{0.15^2 * 1 + 0.21^2 * 2 + 0.92^2 * 3 + 0.94^2 * 3}{0.15^2 + 0.21^2 + 0.92^2 + 0.94^2}$$

$$C2(f2) = \frac{5.307}{1.797} = 2.95$$

The updated cluster centre for $C2 = (4.4, 2.95)$.

Since there is no change in the cluster centre, the algorithm stops. The final membership value and cluster allocation for the data points is given in Table 7.21.

TABLE 7.21 Cluster formation using fuzzy *c*-means algorithm

Data point	μ^{C1}	μ^{C2}	Allocated cluster
x_1	0.85	0.15	$C1$
x_2	0.79	0.21	$C1$
x_3	0.12	0.88	$C2$
x_4	0.14	0.86	$C2$

7.7 ROUGH CLUSTERING

It is possible that some of the data points clearly belong to only one cluster while the other data points may belong to two or more clusters. This is abstracted by a rough set which is represented using two sets; these sets are called **lower approximation** and **upper approximation.** A data point can belong to at most one lower approximation. If a data point does not belong to any lower approximation, then it belongs to two or more upper approximations.

We can define a rough set using the lower approximation ($\underline{R}(S)$) and the upper approximation ($\overline{R}(S)$), where $S \subseteq \mathfrak{X}$, a set of data points, is as follows:

$$\underline{R}(S) = \bigcup_i G_i, \text{ where } \quad G_i \subseteq S$$

$$\overline{R}(S) = \bigcup_i G_i, \text{ where } \quad G_i \cap S \neq \emptyset$$

In the lower approximation, each equivalence class must be entirely contained within S. On the other hand, in the upper approximation, equivalence classes that partially overlap with S are included. Consequently, the lower approximation comprises patterns that definitely belong to S, whereas the upper approximation contains patterns that may potentially belong to S.

Further note that if $\overline{R}(S) - \underline{R}(S) = \emptyset$, then there is no roughness in S with respect to R. We discuss one of the popular clustering algorithms, rough k-means clustering, in the next subsection which is based on these concepts.

7.7.1 Rough *K*-Means Clustering Algorithm

Let n be the number of data points, $\mathfrak{X} = \{x_1, x_2, \ldots, x_n\}$, having l features, k be the number of clusters, w_l and w_u be the weights associated with the lower and upper approximations and ϵ be the threshold value.

1. Randomly assign each data object to exactly one lower approximation $\underline{R}(k)$. Note that by definition, the data object also belongs to the upper approximation $\overline{R}(k)$ of the same cluster.
2. Compute the cluster centroid C_j as follows:

   If $\underline{R}(k) \neq \emptyset$ and $\overline{R}(k) - \underline{R}(k) = \emptyset$

   $$C_j = \sum_{x_i \in \underline{R}(k)} \frac{x_i}{|\underline{R}(k)|}$$

   Else if $\underline{R}(k) = \emptyset$ and $\overline{R}(k) - \underline{R}(k) \neq \emptyset$

   $$C_j = \sum_{x_i \in \overline{R}(k) - \underline{R}(k)} \frac{x_i}{|\overline{R}(k) - \underline{R}(k)|}$$

   Else

   $$C_j = w_l \times \sum_{x_i \in \underline{R}(k)} \frac{x_i}{|\underline{R}(k)|} + w_u \times \sum_{x_i \in \overline{R}(k) - \underline{R}(k)} \frac{x_i}{|\overline{R}(k) - \underline{R}(k)|}$$

3. For each data point in $\mathfrak{X}$, compute the Euclidean distance with each cluster, C_j, $1 \leq j \leq k$ [that is, $d(x_i, C_j)$, $\forall i \in m$, $\forall j \in k$].
   Let $d(X, C_p) = min_{1 \leq j \leq k}(x, Cj)$.

4. Use the ratio, $\frac{d(x,C_j)}{d(x,C_p)}$, where $1 \leq i,p \leq k$, to determine the membership of X as follows:
   - If the ratio $\not\leq \epsilon$, then the corresponding data point x will not be part of the lower approximation.
   - Else, the corresponding data point x will be part of the lower approximation
5. Repeat from Step 2 until convergence (that is, Old centroid = New centroid)

EXAMPLE 17: Consider the data set shown in Table 7.22 with $k = 2$, $w_l = 0.7$, $w_u = 0.3$ and $\epsilon = 2$.

TABLE 7.22 Example of data set with two features

Data points	f1	f2
x_1	1	1
x_2	2	2
x_3	4	1
x_4	5	2
x_5	7	0
x_6	8	0

1. Randomly assign each data point to exactly one lower approximation (two clusters).

$$\underline{R}(k_1) = \{(1,1),(2,2),(4,1)\}$$
$$\underline{R}(k_2) = \{(5,2),(7,0),(8,0)\}$$

2. Since (i) $\underline{R}(k_1) \neq \emptyset$ and $\overline{R}(k_1) - \underline{R}(k_1) = \emptyset$; (ii) $\underline{R}(k_2) \neq \emptyset$ and $\overline{R}(k_2) - \underline{R}(k_2) = \emptyset$, the centroid is calculated using $C_j = \sum_{x_i \in \underline{R}(k)} \frac{x_i}{|\underline{R}(k)|}$.

$$C_1 = \left(\frac{1+2+4}{3}, \frac{1+2+1}{3}\right) = (2.33, 1.33)$$
$$C_2 = \left(\frac{5+7+8}{3}, \frac{2=0+0}{3}\right) = (6.67, 0.67)$$

3. Find the Euclidean distances between each data point and the cluster centre.
   - With reference to C_1

$$d((1,1),(2.33,1.33)) = \sqrt{(1-2.33)^2 + (1-1.33)^2} = 1.37$$
$$d((2,2),(2.33,1.33)) = \sqrt{(2-2.33)^2 + (2-1.33)^2} = 0.75$$
$$d((4,1),(2.33,1.33)) = \sqrt{(4-2.33)^2 + (1-1.33)^2} = 1.70$$
$$d((5,2),(2.33,1.33)) = \sqrt{(5-2.33)^2 + (2-1.33)^2} = 2.75$$
$$d((7,0),(2.33,1.33)) = \sqrt{(7-2.33)^2 + (0-1.33)^2} = 4.86$$
$$d((8,0),(2.33,1.33)) = \sqrt{(8-2.33)^2 + (0-1.33)^2} = 5.82$$

- With reference to C_2

$$d((1,1),(6.67,0.67)) = \sqrt{(1-6.67)^2+(1-0.67)^2} = 5.68$$
$$d((2,2),(6.67,0.67)) = \sqrt{(2-6.67)^2+(2-0.67)^2} = 4.86$$
$$d((4,1),(6.67,0.67)) = \sqrt{(4-6.67)^2+(1-0.67)^2} = 2.69$$
$$d((5,2),(6.67,0.67)) = \sqrt{(5-6.67)^2+(2-0.67)^2} = 2.14$$
$$d((7,0),(6.67,0.67)) = \sqrt{(7-6.67)^2+(0-0.67)^2} = 0.75$$
$$d((8,0),(6.67,0.67)) = \sqrt{(8-6.67)^2+(0-0.67)^2} = 1.49$$

4. Use the ratio, $\frac{d(x,C_j)}{d(x,C_p)}$, where $1 \leq i,p \leq k$, to determine the membership.

$(1,1) \Longrightarrow \dfrac{d((1,1),(6.67,0.67))}{d((1,1),(2.33,1.33))} = \dfrac{5.68}{1.37} = 4.14 \not\leq 2$. So, x_1 will be part of $\underline{R}(k_1)$

$(2,2) \Longrightarrow \dfrac{d((2,2),(6.67,0.67))}{d((2,2),(2.33,1.33))} = \dfrac{4.86}{0.75} = 6.48 \not\leq 2$. So, x_2 will be part of $\underline{R}(k_1)$

$(4,1) \Longrightarrow \dfrac{d((4,1),(6.67,0.67))}{d((4,1),(2.33,1.33))} = \dfrac{2.69}{1.7} = 1.58 < 2$. So, x_3 will not be part of $\underline{R}(k_1)$ and $\underline{R}(k_2)$

$(5,2) \Longrightarrow \dfrac{d((5,2),(2.33,1.33))}{d((5,2),(6.67,0.67))} = \dfrac{2.75}{2.14} = 1.28 < 2$. So, x_4 will not be part of $\underline{R}(k_1)$ and $\underline{R}(k_2)$

$(7,0) \Longrightarrow \dfrac{d((7,0),(2.33,1.33))}{d((7,0),(6.67,0.67))} = \dfrac{4.86}{0.75} = 6.48 \not\leq 2$. So, x_5 will be part of $\underline{R}(k_2)$

$(8,0) \Longrightarrow \dfrac{d((8,0),(2.33,1.33))}{d((8,0),(6.67,0.67))} = \dfrac{5.82}{1.49} = 3.91 \not\leq 2$. So, x_4 will be part of $\underline{R}(k_2)$

Now we have clusters

$$\underline{R}(k_1) = \{(1,1),(2,2)\} \qquad \overline{R}(k_1) = \{(1,1),(2,2),(4,1),(5,2)\}$$
$$\underline{R}(k_2) = \{(7,0),(8,0)\} \qquad \overline{R}(k_2) = \{(4,1),(5,2),(7,0),(8,0)\}$$

Here, (i) $\underline{R}(k_1) \neq \emptyset$ and $\overline{R}(k_1) - \underline{R}(k_1) \neq \emptyset$; (ii) $\underline{R}(k_2) \neq \emptyset$ and $\overline{R}(k_2) - \underline{R}(k_2) \neq \emptyset$. So, the centroid is calculated using

$$C_j = w_l \times \sum_{x \in \underline{R}(k)} \frac{x_i}{|\underline{R}(k)|} + w_u \times \sum_{x_i \in \overline{R}(k) - \underline{R}(k)} \frac{x_i}{|\overline{R}(k) - \underline{R}(k)|}$$

$$C_1 = 0.7 \times \left(\frac{1+2}{2}, \frac{1+2}{2}\right) + 0.3 \times \left(\frac{4+5}{2}, \frac{1+2}{2}\right) = (2.4, 1.5)$$

$$C_2 = 0.7 \times \left(\frac{7+8}{2}, \frac{0+0}{2}\right) + 0.3 \times \left(\frac{4+5}{2}, \frac{1+2}{2}\right) = (6.6, 0.45)$$

5. Repeat from Step 3 until convergence.

7.8 EXPECTATION MAXIMIZATION-BASED CLUSTERING

One of the approaches to mathematically represent clusters is through parametric probability distributions. In this method, the data set is regarded as a combination of these distributions, and clustering involves using a finite mixture density model consisting of k probability distributions, where each distribution represents a cluster. Hence, the clustering problem entails estimating the parameters of the probability distributions to best fit the data.

The EM (expectation maximization) algorithm is a suitable method for parameter estimation. In EM, a data point is assigned to a cluster based on its membership probability. The basic concept of the EM algorithm is as follows: it begins with an initial estimation of the mixture model parameters. Subsequently, it iteratively recalculates the probabilities for the data points based on the mixture density generated using these parameters. The parameters are then updated using these recalculated probabilities.

Let $\{x_1, x_2, \ldots, x_n\}$ be the data set to be clustered. Let Z_{ij} be the probability that $x_i \in \mathfrak{c}_j$ (a cluster).

1. Computing the probability that a data point belongs to a cluster:

$$Prob(x_i \in \mathfrak{c}_j) = Z_{ij} = \frac{exp[-\frac{1}{2\sigma^2}(x_i - \mu_j)^2]}{\sum_{l=1}^{k} exp[-\frac{1}{2\sigma^2}(x_i - \mu_l)^2]} \tag{7.1}$$

2. The μ values are updated using Z_{ij} as follows:

$$\mu_j = \frac{\sum_{i=1}^{n} Z_{ij} x_i}{\sum_{i=1}^{n} Z_{ij}} \tag{7.2}$$

   Notice the similarity between the EM algorithm and the k-means algorithm of choosing the centroids (μ_j) and assigning data points to the nearest cluster means to get Z_{ij}.
3. Repeat the above two steps until there is no change in μ values.

The following example illustrate the above steps.

EXAMPLE 18: Let the data points be $x_1 = 1$, $x_2 = 6$, $x_3 = 6$, $x_4 = 7$. Let us consider two clusters, $\mathfrak{c}_1$ and $\mathfrak{c}_2$, and $\sigma = 1$. Let the two extreme points $x_1 = 1$ and $x_4 = 7$ be the initial cluster centres of $\mathfrak{c}_1$ and $\mathfrak{c}_2$, respectively. The values of Z_{ij} are calculated using Equation 7.1 and is shown in Table 7.23. For example,

$$Z_{11} = \frac{e^{-\frac{1}{2}(1-1)^2}}{e^{-\frac{1}{2}(1-1)^2} + e^{-\frac{1}{2}(1-7)^2}} \approx 0.9999$$

The updated μ values using Equation 7.2 for $\mathfrak{c}_1$ and $\mathfrak{c}_2$ are $\mu_1 = 1.5$ and $\mu_2 = 6.5$, respectively.

TABLE 7.23 Z_{ij} values

Data point i	Z_{i1}	Z_{i2}
$x_1 = 1$	0.9999	0.0001
$x_2 = 2$	0.9999	0.0001
$x_3 = 6$	0.0001	0.9999
$x_4 = 7$	0.0001	0.999

Let us add the data item $x_5 = 4$. Starting with the same initial conditions as earlier, we get $Z_{51} = 0.5$ and $Z_{52} = 0.5$. The updated $\mu_1 = 2$ and $\mu_2 = 6$. The Z_{ij} values are given in Table 7.24. The updated $\mu_1 = 2$ and $\mu_2 = 6$ and are the same as the previous values, so the algorithm terminates.

TABLE 7.24 Z_{ij} values

Data points	Z_{i1}	Z_{i2}
$x_1 = 1$	0.9999	0.0001
$x_2 = 2$	0.9996	0.0004
$x_3 = 6$	0.0004	0.9996
$x_4 = 7$	0.0001	0.999
$x_5 = 4$	0.5	0.5

7.9 SPECTRAL CLUSTERING

Clustering algorithms like k-means are effective for data with isotropic or spherical clusters; they are not suitable for non-isotropic clusters. For instance, when clusters are elongated in a specific direction or when they are concentric with similar centroids (similar to two circles or spheres with varying radii and points distributed along each circle or sphere).

Spectral clustering algorithms are well-suited for data sets that contain non-isotropic clusters, such as chain-like or concentric clusters. The spectral clustering algorithm assumes that the data set is in the form of a graph which can be represented by triplets, $< V, E, S >$. Here,

- $V = \{x_1, x_2, \ldots, x_n\}$, where each x_i in V corresponds to a data point in the collection.
- $E\{< x_i, x_j >: x_i \in V$ and $x_j \in V\}$ for $i, j = 1, 2, \cdots, n$. So, each element of E characterizes an edge between a pair of vertices.
- $S = \{A_{ij} : x_i, x_j \in V\}$. Each element of A characterizes similarity between a pair of nodes. $A_{ij} = 0$ if x_i and x_j are not similar (or not connected); and $A_{ij} = 1$ if x_i and x_j are similar (or connected). We assume that the graph is undirected; so, $A_{ij} = A_{ji}$. Further, we are assuming that the similarity values are binary, either 0 or 1.

Consider the graph shown in Fig. 7.14.

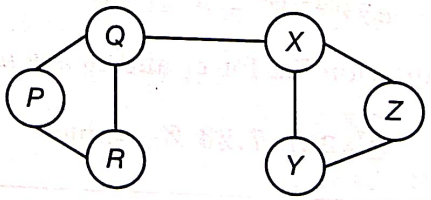

FIG. 7.14 Graphical representation of a data set with six vertices

The corresponding S matrix is given by

$$S = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

where we are assuming that a node is similar to itself and so the diagonal entries are all 1.

Weight (Degree) matrix (D) is a diagonal matrix and $D_{ii} = \sum S_{ij}$ and $D_{ij} = 0$ if $i \neq j; i, j \in V$. That is, the ith diagonal element in D is the sum of the elements in the ith row of A. Matrix D for the graph (Fig. 7.14) is

$$D = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 \end{bmatrix}$$

Let C_1 (a cluster) be a subset of V and C_2 (another cluster), the complement of C_1, be $V - C_1$. $Cut(C_1, C_2) = \sum_{X_i \in C_1, X_j \in C_2} A_{ij}$. C_1 and C_2 are said to be best partitions, provided $Cut(C_1, C_2)$ is minimum. For example, let $C_1 = \{P, Q, R\}$ and $C_2 = \{X, Y, Z\}$. Then $Cut(C_1, C_2) = 1$, which is minimum among any set of vertices chosen between C_1 and C_2.

It is possible to formalize the minimum cut expression by considering the following:

- Let $I_i = 0$, if $X_i \in C_1$ and $I_i = 1$ if $X_i \in C_2$. Note that $(I_i - I_j)^2 = 1$ if X_i and X_j are in different clusters and it is 0 if they are in the same cluster. So, $Cut(C_1, C_2) = \sum_{x_i \in C_1, x_j \in C_2} A_{ij}(I_i - I_j)^2$.
- It is possible to show that $Cut(C_1, C_2) = I^t L I$, where I is the index vector of size n (where there are n vertices in the graph), Laplacian $L = D - S$, where D is the degree matrix and S is the adjacency matrix. So, minimizing the Cut amounts to finding the index vector I such that $I^t D I$ is minimized; once I is known, it is possible to obtain the clusters based on the polarity of the entries in I.
- L exhibits symmetry, similar to S and D. The eigenvalue with the lowest magnitude in L is 0, with its corresponding eigenvector being denoted as $\mathbf{1} = (1, 1, \ldots, 1)^t$ since $L\mathbf{1} = 0 = 0\mathbf{1}$.
- By utilizing the eigenvector $\mathbf{1}$ as the value of I, it can be demonstrated that $I^t L I$ equals 0 because $LI = L\mathbf{1} = 0$. However, this particular choice of I does not yield a two-partition outcome as it results in a single (positive) cluster.
- Instead of selecting the smallest eigenvalue, opt for the subsequent smallest eigenvalue to maintain a small value for $I^t L I$. In this case, I represents the eigenvector corresponding to the second smallest eigenvalue. Moreover, because L is symmetric, its eigenvectors can be chosen to be orthonormal, and the eigenvalues are real. Consequently, by choosing I as the eigenvector linked to the second smallest eigenvalue, a new I is obtained that is orthogonal to $\mathbf{1}$.
- This implies that I comprises both negative and positive entries capturing two clusters. Hence, I is determined to be the eigenvector associated with the second smallest eigenvalue.
- For Fig. 7.14, the Laplacian matrix L is given by

$$L = \begin{bmatrix} 2 & -1 & -1 & 0 & 0 & 0 \\ -1 & 3 & -1 & -1 & 0 & 0 \\ -1 & -1 & 2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 3 & -1 & -1 \\ 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & -1 & 2 \end{bmatrix}$$

- The eigenvalues of L are 0, $\frac{5-\sqrt{17}}{2}$, 3, 3, 3, $\frac{5+\sqrt{17}}{2}$. The first two eigenvectors are 1 and $\left(1, \frac{-3+\sqrt{17}}{2}, 1, \frac{3-\sqrt{17}}{2}, \frac{-7+\sqrt{17}}{4}, \frac{3-\sqrt{17}}{2}\right)^t$. Note that the first three entries are positive and the remaining three are negative in the second eigenvector. So, the clusters are $C_2 = \{P, Q, R\}$ and $C_1 = \{X, Y, Z\}$, where C_1 is the negative cluster and C_2 is the positive cluster.
- Figure 7.14 is an example of a two-partition scenario. However, the number of clusters (k) can be more than two in general. In such cases, we have to consider k eigenvectors corresponding to the k smallest eigenvalues. A point to be noted is that each eigenvector has n dimensions (that is, number of data points), so the k eigenvectors can provide a k-dimensional representation of the n patterns by arranging them as k columns in a matrix of size $n \times k$. Moreover, these k eigenvectors are orthogonal to each other, allowing us to cluster the n rows (data points) into k clusters.

7.10 Clustering Large Data Sets

The development of technology has led to advancements in data acquisition, collection and processing capabilities. With these advancements, the size of the data sets has also increased, to the point where they cannot fit entirely into the main memory of the system being used.

Instead, they reside in secondary storage devices and need to be transferred to the main memory as required. However, accessing data from secondary storage is significantly slower compared to accessing data from the main memory. When comparing the time complexity of clustering algorithms, it is possible for them to have the same time complexity. However, the feasibility of the algorithms can differ based on the specific conditions. For example, the k-means algorithm has a time complexity of $\mathcal{O}(n)$ for fixed values of k (number of clusters) and the number of iterations, p.

If the value of p is, say 100, the algorithm needs to scan the entire data set 100 times. In the case of large data sets, where the data needs to be fetched from secondary storage, the k-means algorithm may not generate the desired partition within an acceptable time frame due to the slower access of secondary storage. Therefore, the number of data set scans also affects the performance of the clustering algorithm.

Another important consideration is handling incremental changes in the data set. In some cases, the data in the data set may change incrementally, and the clustering algorithm should be capable of accommodating these changes without referring to the previously seen data.

These requirements are addressed by the following two types of algorithms:

Incremental clustering algorithms: These algorithms cluster new data points based on existing cluster representatives and specific threshold parameters. They do not reconsider data points that were seen earlier.

Leader clustering algorithm is an example of an incremental clustering algorithm. It can be used in both offline scenarios, where the entire data set is available from the beginning, and online scenarios, where data is made available incrementally. An important characteristic of this algorithm is that the clusters are formulated using a *single scan of the data set.*

The core concept of this algorithm revolves around grouping data points that are close to each other into the same cluster, utilizing a predefined distance threshold. In essence, if a data point is located within the threshold distance of a cluster's representative (leader), it is assigned to that cluster. However, if there are no existing clusters within the proximity (threshold distance) of the data point, a new cluster is created with the point designated as the leader of this newly formed cluster.