We have described the construction of distances and similarities. It is always possible to construct similarities from distances. For example, we might set

$$S_{ik} = \frac{1}{(1 + d_{ik})} \tag{11.6}$$

where $1 \geq S_{ik} \geq 0$ is the similarity between items i and k and d_{ik} is the corresponding distance. With the non negative condition and with the maximum similarity scaled as $S_{ii} = 1$, the measure

$$d_{ik} = \sqrt{2(1 - S_{ik})} \tag{11.7}$$

has the properties of a distance.

11.1.2 Basic Types of Clustering

Broadly speaking there are two types of clustering: (refer Fig. 11.10)

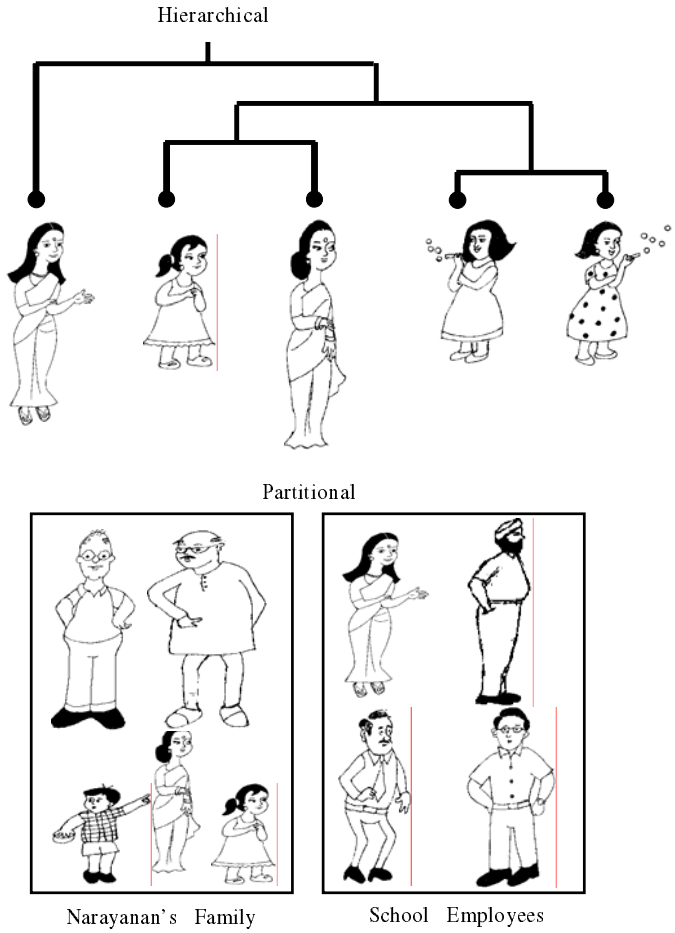


Fig. 11.10 Basic types of clustering.

1. *Partitional algorithms:* In partitional algorithms, we construct various partitions and then evaluate them by some criterion.
2. *Hierarchical algorithms:* In hierarchical algorithms, we create a hierarchical decomposition of the set of objects using some criterion.

Consider Fig. 11.10 that shows the differences in the output of the two approaches.

Desirable features of a Clustering Algorithm. Following are the desirable features of a clustering algorithm.

- Scalability (in terms of both time and space)
- Ability to deal with different data types
- Discovery of clusters with arbitrary shape
- Minimal requirements for domain knowledge to determine input parameters
- Able to deal with noise and outliers
- Insensitive to order of input records
- Incorporation of user-specified constraints
- Interpretability and usability

Hierarchical clustering: We can build hierarchical trees in two ways:

Bottom-up (agglomerative): Initially we assume that all items belong to a separate cluster. Then, we find the best pair to merge into a new cluster. In Fig. 11.11 clustering process start from the bottom and the final result is shown at the top.

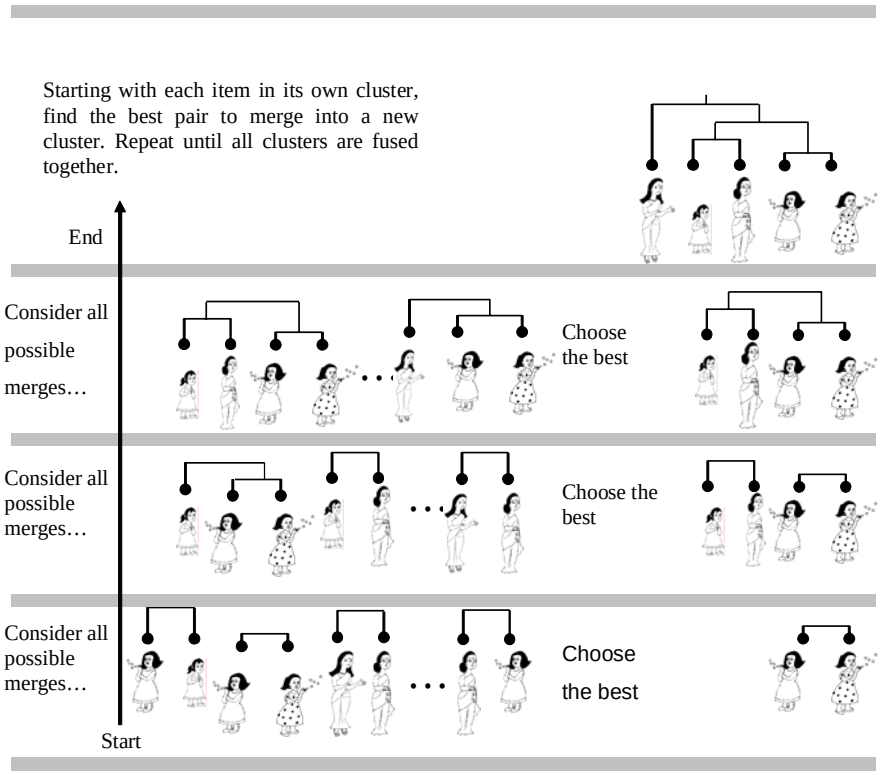


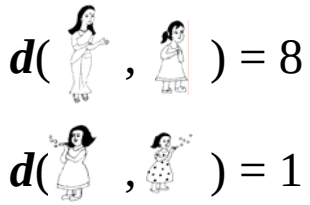
Fig. 11.11 Bottom-up agglomerative clustering.

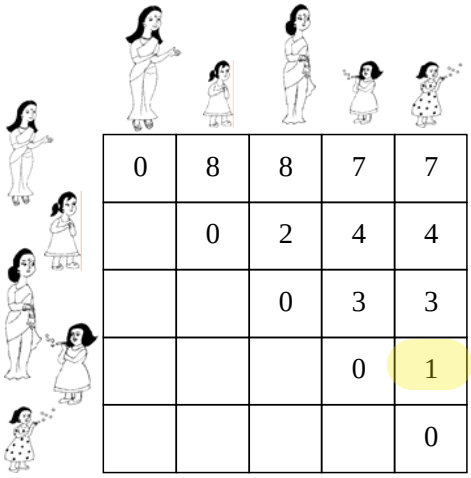
Top-down (divisive): Starting with all the data in a single cluster, consider every possible way to divide the cluster into two. Choose the best division and recursively operate on both sides.

Most commonly used one is Agglomerative method and hence we describe here in detail.

Agglomerative hierarchical clustering. The process of clustering starts by finding the clusters with the closest distance and putting those two clusters into one cluster by means of the common distance measure. This process starts by assigning each object as a cluster, each of which is defined by chosen distance measure. However, once several objects have been linked together, how do we determine the distances between those new clusters? Figures 11.12 and 11.13 illustrate the problem.

We begin with a distance matrix that contains the distances between every pair of objects in our database.



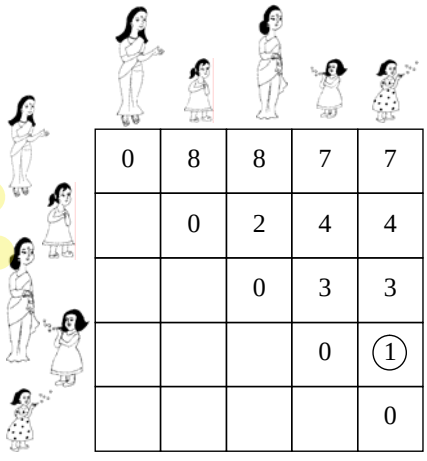


0	8	8	7	7
	0	2	4	4
		0	3	3
			0	1
				0

Fig. 11.12 Hierarchy clustering step 1: making pair wise distance matrix.

With respect to Fig. 11.12, the distance between objects 4 and 5 is 1 and hence we merge them. Now we have only 4 objects—three individual objects and a composite object consisting of two individuals. We proceed to make another 4×4 distance matrix. Since the first three objects are same as before, matrix entries do not change (refer Fig. 11.13). Since the fourth object is a composite object, finding distance of it from other three is a problem. How do we define distance between two composite objects?

In the first iteration of agglomerative clustering we merge  So we need to remove them from the matrix.



0	8	8	7	7
	0	2	4	4
		0	3	3
			0	(1)
				0

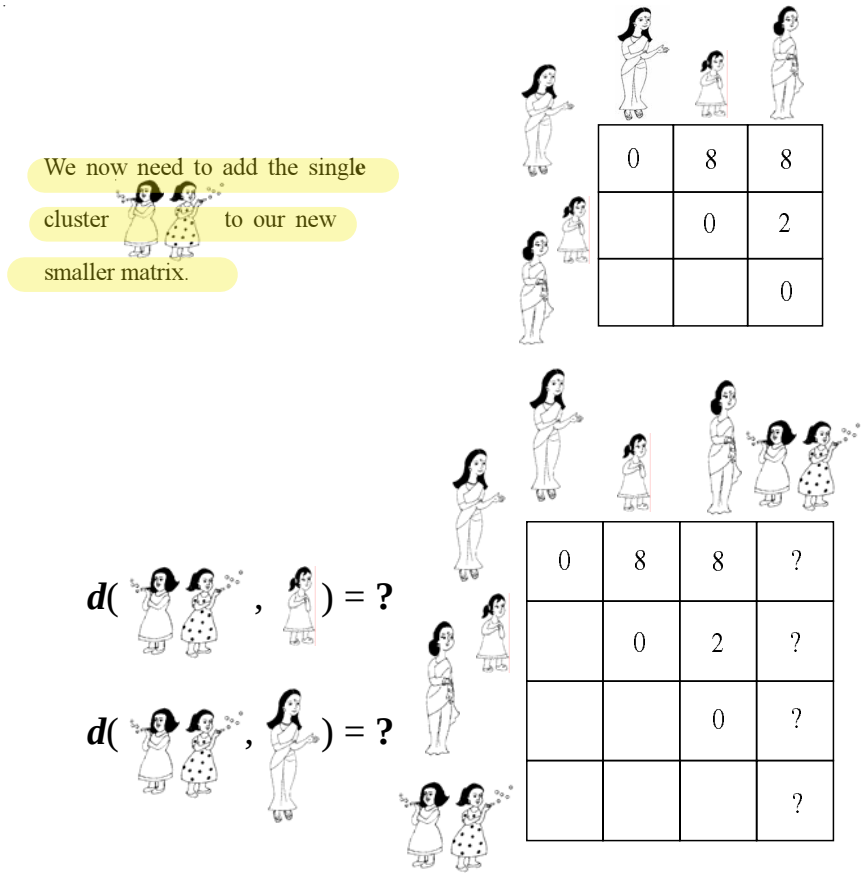


Fig. 11.13 Need for linkage (Amalgamation) in hierarchical clustering.

In other words, we need a linkage or amalgamation rule to determine when two clusters are sufficiently similar to be linked together. There are various possibilities, for example, we could link two clusters together when *any* two objects in the two clusters are closer together than the respective linkage distance. Put another way, we use the "nearest neighbours" across clusters to determine the distances between clusters; this method is called **single linkage**. This rule produces "stringy" types of clusters, that is, clusters "chained together" by only single objects that happen to be close together. Alternatively, we may use the neighbours across clusters that are furthest away from each other; this method is called **complete linkage**. There are numerous other linkage rules such as these that have been proposed.

Single linkage (nearest neighbour). As described earlier, in this method the distance between two clusters is determined by the distance of the two closest objects (nearest neighbours) in the different clusters. This rule will, in a sense, *string* objects together to form clusters, and the resulting clusters tend to represent long "chains." In short, the closest distance between two samples belonging to two different clusters (Fig. 11.14 and Fig. 11.15).

$$d(c_1, c_2) = \min_{o \in C_1,\, O \in C_2} \{d(o, O)\} \tag{11.8}$$

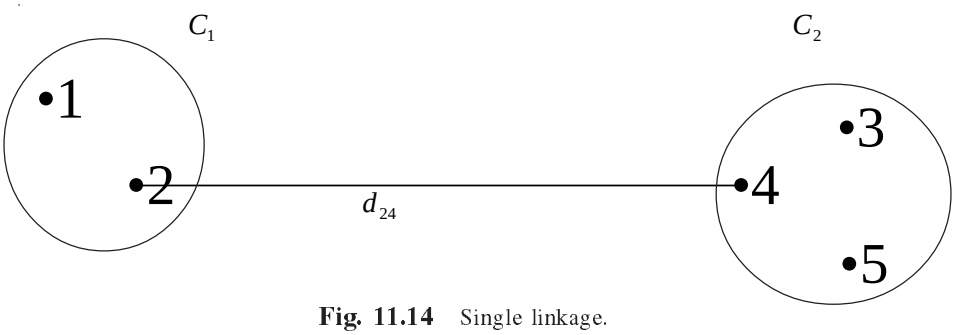
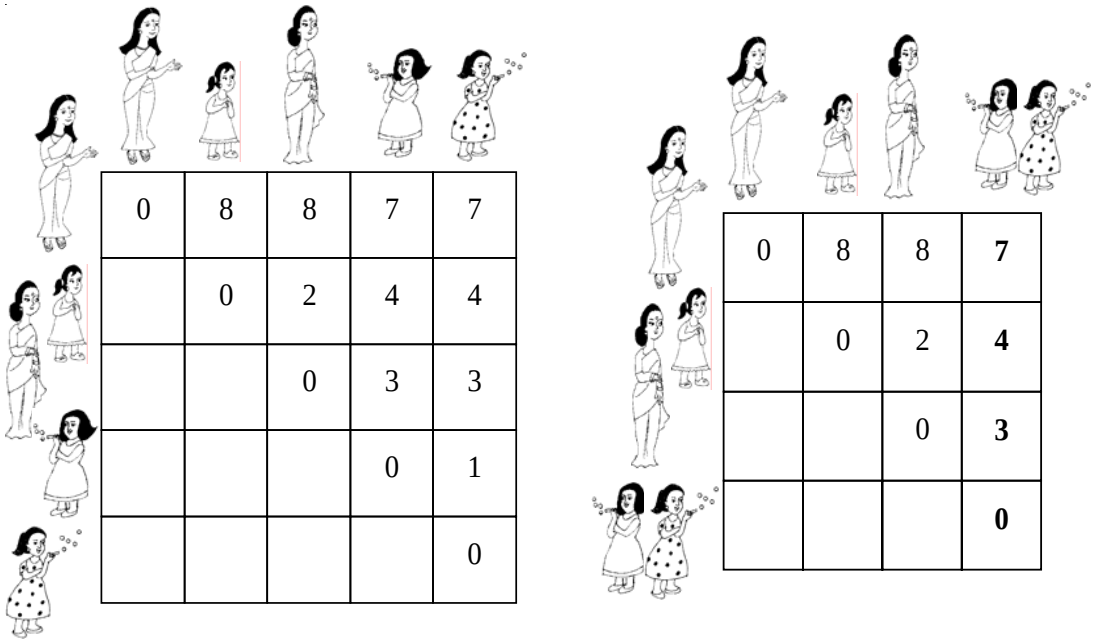


Fig. 11.14 Single linkage.

Fig. 11.15 illustrates the method pictorially.



Using Single linkage (nearest neighbour)...

$d($, $) = \min[\, d($, $),\ d($, $)\,] = 4$

$d($, $) = \min[\, d($, $),\ d($, $)\,] = 7$

Fig. 11.15 Example of applying single linkage rule to do hierarchical clustering.

Complete linkage (furthest neighbour). In this method, the distances between clusters are determined by the greatest distance between any two objects in the different clusters (i.e., by the "furthest neighbours"). This method usually performs quite well in cases when the objects actually form naturally distinct "clumps." If the clusters tend to be somehow elongated or of a "chain" type nature then this method is inappropriate. Figure 11.16 illustrates computation of complete linkage clustering.

$$d(c_1, c_2) = \max_{o \in C_1,\, O \in C_2} \{d(o, O)\} \tag{11.9}$$

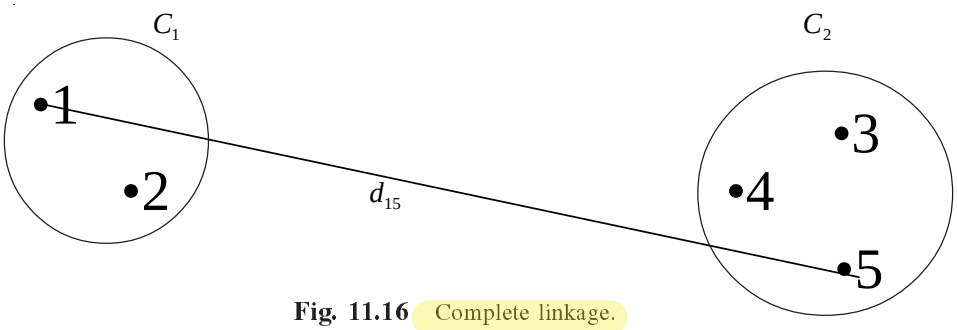
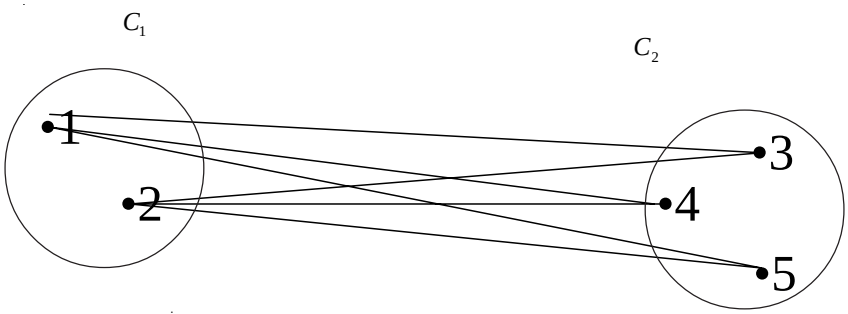


Fig. 11.16 Complete linkage.

Unweighted pair-group average. In this method, the distance between two clusters is calculated as the average distance between all pairs of objects in the two different clusters (refer Fig. 11.17). This method is also very efficient when the objects form natural distinct "clumps," however, it performs equally well with elongated, "chain" type clusters. Note that in their book, Sneath and Sokal (1973) [1] introduced the abbreviation UPGMA to refer to this method as *unweighted pair-group method using arithmetic averages. The average distance between all of the samples belonging to two different clusters.*

$$d(c_1, c_2) = \frac{1}{n_1 n_2} \sum_{o \in C_1,\, O \in C_2} \{d(o, O)\} \tag{11.10}$$

where n_1 and n_2 is the number of samples in clusters.



$$\text{Distance} = \frac{(d_{13} + d_{14} + d_{15} + d_{23} + d_{24} + d_{25})}{6}$$

Fig. 11.17 Unweighted pair-group average.

Weighted pair-group average. This method is identical to the *unweighted pair-group average* method, except that in the computations, the size of the respective clusters (i.e., the number of objects contained in them) is used as a weight. Thus, this method (rather than the previous method) should be used when the cluster sizes are suspected to be greatly uneven. *WPGMA* is referred to as *weighted pair-group method using arithmetic averages.*

Unweighted pair-group centroid. The *centroid* of a cluster is the average point in the multidimensional space defined by the dimensions. In a sense, it is the *centre of gravity* for the respective cluster. In this method, the distance between two clusters is determined as the difference between centroids. Sneath and Sokal (1973) use the abbreviation *UPGMC* to refer to this method as *unweighted pair-group method using the centroid average.*

Weighted pair-group centroid (median). This method is identical to the previous one, except that the weighting is introduced into the computations to take into consideration differences in cluster sizes (i.e., the number of objects contained in them). Thus, when there are (or one suspects there to be) considerable differences in cluster sizes, this method is preferable to the previous one. Sneath and Sokal (1973) use the abbreviation *WPGMC* to refer to this method as *weighted pair-group method using the centroid average.*

Ward's method. This method is distinct from all other methods because it uses an analysis of variance approach to evaluate the distances between clusters. In short, this method attempts to minimize the Sum of Squares (SS) of any two (hypothetical) clusters that can be formed at each step. Refer to Ward (1963)[2] for details concerning this method. In general, this method is regarded as very efficient, however, it tends to create clusters of small size.

Examples of clustering using different linkages

Consider the array of distances as given in Table 11.7.

Table 11.7 Mutual Distance of 4 Objects

$D = \{d_{jk}\} =$

	O_1	O_2	O_3	O_4
O_1	0			
O_2	1	0		
O_3	11	2	0	
O_4	5	3	4	0

The values represent distances between the objects O_1, O_2, O_3, O_4. To illustrate linkage and its various types let us consider the data given in Table 11.7 observe its various types with regard to above problem.

Example using single linkage. In the matrix of distances, select the distance value that is minimum. Here $d(O_1, O_2)$ is the least and is equal to 1. To combine into clusters the single linkage measure is used as distance measure. Now, a new object $O_{1,2}$ is formed.

Table 11.8 Distance Matrices on Applying Single Linkage rule

	O_1	O_2	O_3	O_4
O_1	0			
O_2	(1)	0		
O_3	11	2	0	
O_4	5	3	4	0

	$O_{1,2}$	O_3	O_4
$O_{1,2}$	0		
O_3	(2)	0	
O_4	3	4	0

	$O_{1,2,3}$	O_4
$O_{1,2,3}$	0	
O_4	(3)	0

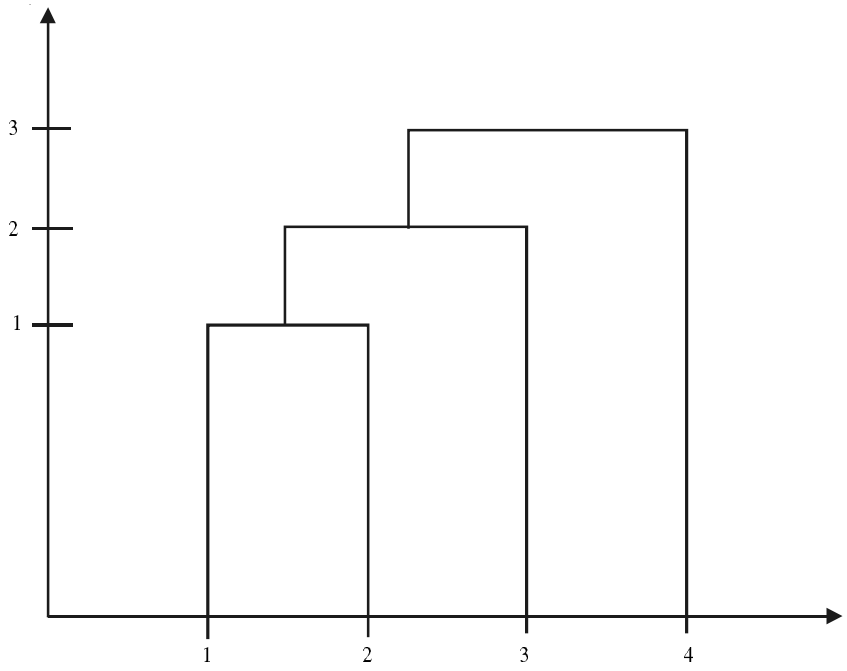


Fig. 11.18 Single linkage illustration.

The various steps involved are shown in Table 11.8. We have now three objects out of which object $O_{1,2}$ is a sub-cluster (a composite object). The calculation of distance between these three objects is to be carried out for further clustering.

According to single linkage,

$$\text{Distance}(O_{1,2}, O_3) = \min\{\text{Distance}(O_1, O_3), \text{Distance}(O_2, O_3)\}$$

$$= \min\{11, 2\} = 2$$

The above distances are taken from Table 11.7

Similarly,

$$\text{Distance}(O_{1,2}, O_4) = \min\{\text{Distance}(O_1, O_4), \text{Distance}(O_2, O_4)\}$$

$$= \min\{5, 3\} = 3$$

Distance (O_3, O_4) is 4 as per Table 11.7.

With the above three distances, we form the second matrix shown in Table 11.9.

It is same as the second matrix in Table 11.8.

Table 11.9 Inter Cluster Distances using Single Linkage

	$O_{1,2}$	O_3	O_4
$O_{1,2}$	0		
O_3	2	0	
O_4	3	4	0

The singe linkage rule of finding inter cluster distances can be written as:

$$\text{Distance}(P+Q, R) = \min\{\text{Distance}(P, R), \text{Distance}(Q, R)\} \quad (11.11)$$

where $P + Q$ represents union of two sub-clusters.

We now combine objects $O_{1,2}$, and O_3 to form a new cluster $O_{1,2,3}$ since objects. $O_{1,2}$ and O_3 are now closest objects according to single linkage criteria.

The elements of the last matrix in Table 11.8 is arrived at as per the following calculation:

$$\text{Distance}(O_{1,2,3}, O_4) = \min\{\text{Distance}(O_{1,2}, O_4), \text{Distance}(O_3, O_4)\}$$

$$= \min\{3, 4\} = 3$$

Note that to compute this distances we need to refer only the matrix derived in the previous stage, i.e., matrix given in Table 11.9.

Finally a combined cluster $O_{1,2,3,4}$ is obtained. The hierarchy of clusters formed can be represented by a dendrogram. See the dendrogram in Fig. 11.18.

According to the dendrogram shown in Fig. 11.18, at a distance of units 1, objects O_1 and O_2 form a cluster, at distance of 2, objects O_1, O_2 and O_3 form a cluster and at a distance of 3, objects O_1, O_2, O_3, O_4 form a combined cluster.

Example using Complete Linkage. For illustrating complete linkage, consider the same example as earlier. Here the process is similar as in simple linkage, however, the rule is that it selects the maximum distance between the object of R with those of cluster P, Q so as to make an assumption that the all members are within some maximum distance, i.e.,

$$\text{Distance}(P+Q, R) = \max\{\text{Distance}(P, R), \text{Distance}(Q, R)\}$$

Let us return to the matrix example. In this case,

The objects O_1 and O_2 are closest and they form the first sub-cluster $O_{1,2}$. We now form inter-cluster distance matrix using complete linkage rules.

$$\text{Distance}(O_{1,2}, O_3) = \max\{\text{Distance}(O_1, O_3), \text{Distance}(O_2, O_3)\}$$

$$= \max\{11, 2\} = 11$$

Similarly

$$\text{Distance}(O_{1,2}, O_4) = \max\{\text{Distance}(O_1, O_4), \text{Distance}(O_2, O_4)\}$$

$$= \max\{5, 3\} = 5$$

$$\text{Distance}(O_{1,2}, O_4) = \max\{\text{Distance}(O_1, O_4), \text{Distance}(O_2, O_4)\}$$

$$= \max\{5, 3\} = 5$$

Distance $(O_3, O_4) = 4$ as per Table 11.9.

With the above three distances, we form the matrix shown in Table 11.10.

It is the third matrix in Table 11.12, which shows the whole process.

Table 11.10 Inter Cluster Distances using Complete Linkage

	$O_{1,2}$	O_3	O_4
$O_{1,2}$	0		
O_3	11	0	
O_4	5	(4)	0

The minimum distance in matrix in Table 11.10 is 4. So we combine objects O_3 and O_4 to form the new cluster of objects $O_{3,4}$.

We proceed to find the next inter-cluster distance matrix. Now, we have only two clusters —$O_{1,2}$ and $O_{3,4}$— to consider.

Table 11.11 Distance Matrix between $O_{1,2}$ and $O_{3,4}$

$O_{1,2}$	$O_{3,4}$	
$O_{1,2}$	0	
$O_{3,4}$	11	0

To fill the matrix elements, we refer 11.10 and apply complete linkage rule.

$$\text{Distance}(O_{1,2}, O_{3,4}) = \max\{\text{Distance}(O_{1,2}, O_3), \text{Distance}(O_{1,2}, O_4)\}$$
$$= \max\{11, 5\} = 11$$

Table 11.12 Complete Linkage Illustration

$D = \{d_{jk}\}$

	O_1	O_2	O_3	O_4
O_1	0			
O_2	1	0		
O_3	11	2	0	
O_4	5	3	4	0

	O_1	O_2	O_3	O_4
O_1	0			
O_2	(1)	0		
O_3	11	2	0	
O_4	5	3	4	

	$O_{1,2}$	O_3	O_4	0
$O_{1,2}$	0			
O_3	~~2~~ 11	0		
O_4	~~2~~ 5	(4)	0	

	$O_{1,2}$	$O_{3,4}$
$O_{1,2}$	0	
$O_{3,4}$	(11)	0

The dendrogram is shown in Fig. 11.19.

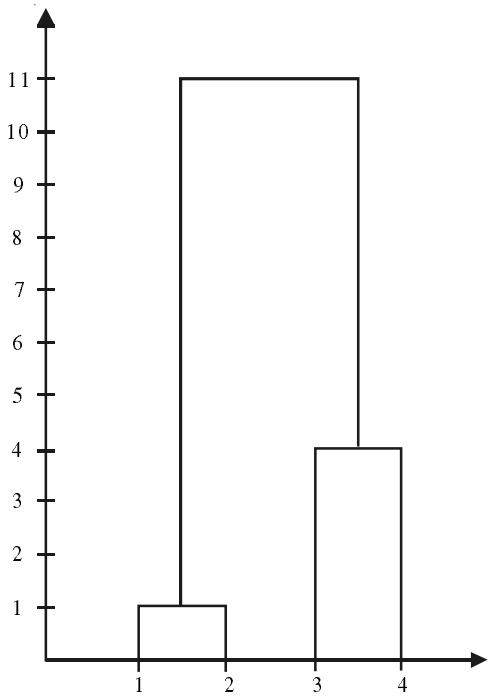


Fig. 11.19 Dendrogram for the data in Table 11.7 using complete linkage.

Example using average linkage. The process is similar as above except that the linkage takes average distance between the cluster P, Q and object of R.

To start with we combine again objects O_1 and O_2 to form $O_{1,2}$. The inter-cluster distance matrix elements according to average linkage are computed as follows:

$$\text{Distance}(O_{1,2}, O_3) = \left(\frac{1}{2}\right)\{\text{Distance}(O_1,\ O_3) + \text{Distance}(O_2,\ O_3)\}$$

$$= \left(\frac{1}{2}\right)(11+2) = 6.5$$

Similarly,

$$\text{Distance}(O_{1,2}, O_4) = \left(\frac{1}{2}\right)\{\text{Distance}(O_1,\ O_4) + \text{Distance}(O_2, O_4)\}$$

$$= \left(\frac{1}{2}\right)\{5+3\} = 4$$

Distance (O_3, O_4) = 4 as per Table 11.7

Table 11.13 shows the resulting matrix.

Table 11.13 Resulting Matrix

	$O_{1,2}$	O_3	O_4
$O_{1,2}$	0		
O_3	6.5	0	
O_4	4	4	0

Referring Table 11.13, we observe that two entries in matrix have the minimum value. We can choose any one of them, however, it leads two slightly different forms of dendrograms. Let us choose to merge objects O_3 and O_4 to form $O_{3,4}$.

We now form inter cluster distance between the two clusters formed, i.e. $O_{1,2}$ and $O_{3,4}$.

$$\text{Distance}\,(O_{1,2}, O_{3,4}) = \left(\frac{1}{2}\right)\{\text{Distance}\,(O_{1,2}, O_3) + \text{Distance}\,(O_{1,2}, O_4)\}$$

$$= \left(\frac{1}{2}\right)(6.5 + 4) = 5.25$$

This distance can also be computed as:

$$\left(\frac{1}{4}\right)\{\text{Distance}\,(O_1, O_3) + \text{Distance}\,(O_1, O_4) + \text{Distance}\,(O_2, O_3) + \text{Distance}\,(O_2, O_4)\}$$

$$= \left(\frac{1}{4}\right)(11 + 5 + 2 + 3) = 5.25$$

Table 11.14 shows the whole process in one picture. Corresponding dendrogram is exhibited in Fig. 11.20

Table 11.14 Average Linkage Illustration.

$D = \{d_{jk}\}$

	O_1	O_2	O_3	O_4
O_1	0			
O_2	1	0		
O_3	11	2	0	
O_4	5	3	4	0

	O_1	O_2	O_3	O_4
O_1	0			
O_2	(1)	0		
O_3	11	2	0	
O_4	5	3	4	0

	$O_{1,2}$	O_3	O_4
$O_{1,2}$	0		
O_3	6.5	0	
O_4	4	4	0

	$O_{1,2}$	$O_{3,4}$
$O_{1,2}$	0	
$O_{3,4}$	5.25	0

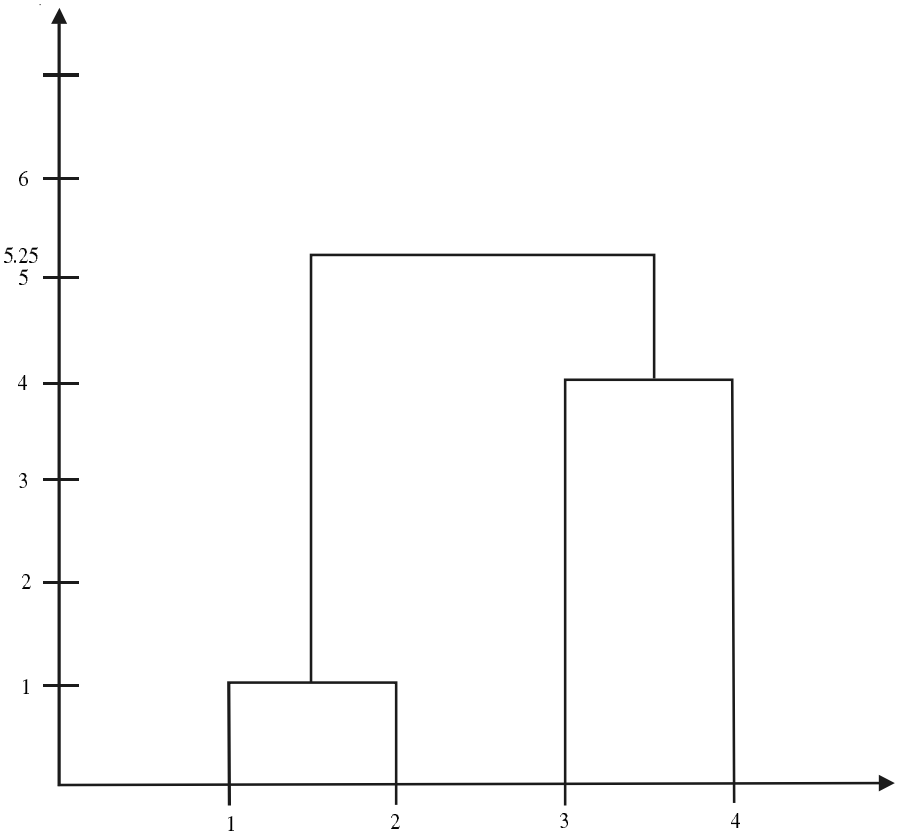


Fig. 11.20 Dendrogram for the data in Table 11.7 using Average linkage.

Advantages of hierarchical Clustering

1. We can look at the dendrogram to determine the "correct" number of clusters. See Figs. 11.21 below. In this case, the two highly separated subtrees are highly suggestive of two clusters. (Things are rarely this clear cut, unfortunately).

These two highly separated subtrees are highly suggestive of two clusters

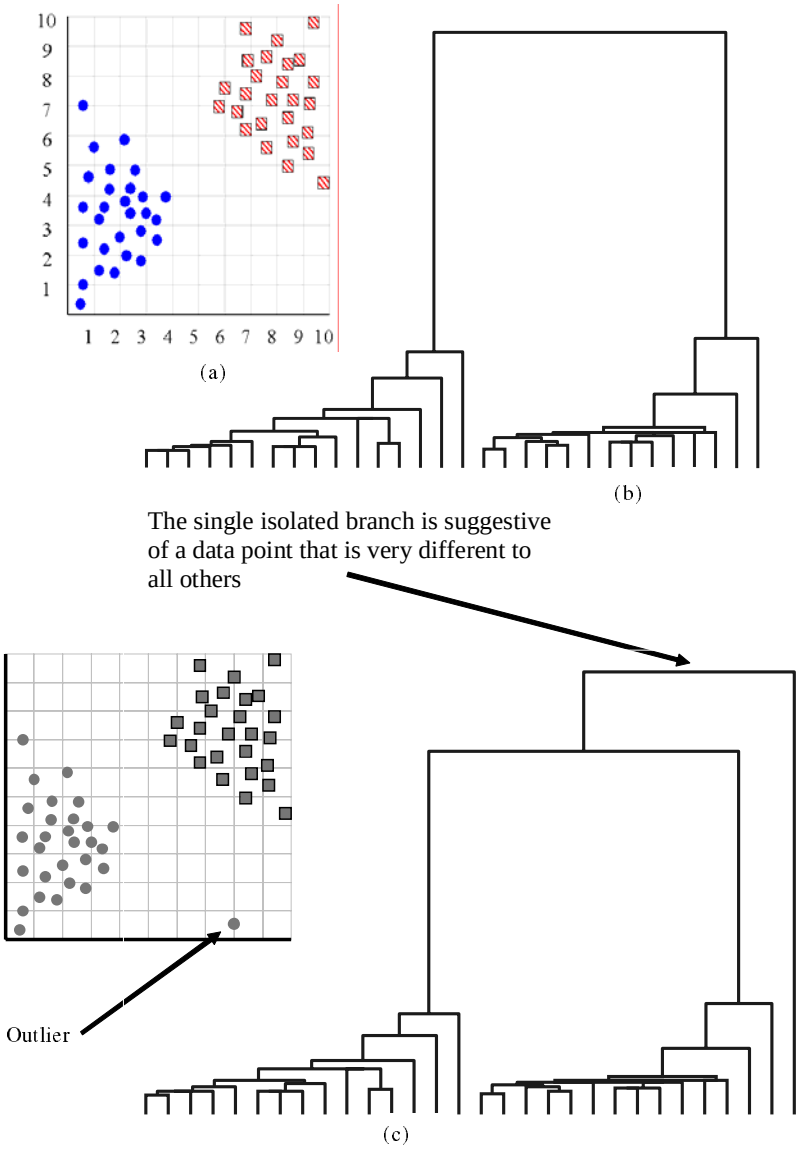


Fig. 11.21 Finding outliers using dendrogram.

2. Hierarchal nature maps nicely onto human intuition for some domains.
3. One potential use of a dendrogram is to detect outliers.

Pitfalls. Hierarchal clustering can sometimes show patterns that are meaningless or spurious. For example, in this clustering shown in Fig. 11.22, the tight grouping of Australia, Anguilla, St. Helena, etc. is meaningful, since all these countries are former UK colonies. However the tight grouping of Niger and India is completely spurious, there is no connection between the two.

The flag of Niger (see Fig. 11.22) is orange over white over green, with an orange disc on the central white stripe, symbolizing the sun. The orange stands the Sahara desert, which borders Niger to the north. Green stands for the grassy plains of the south and west and for the River Niger, which sustains them. It also stands for fraternity and hope. White generally symbolizes purity and hope.

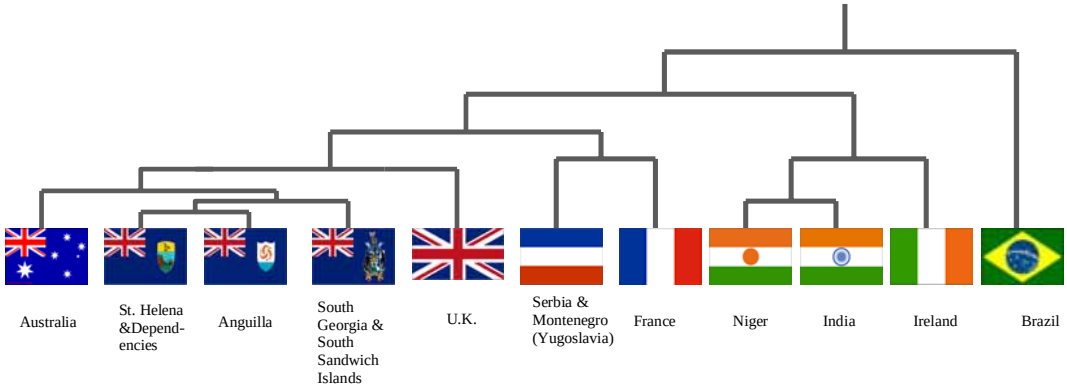


Fig. 11.22 Patterns shown by clusters may sometimes be spurious.

The Indian flag is a horizontal tricolour in equal proportion of deep saffron on the top, white in the middle and dark green at the bottom. In the centre of the white band, there is a wheel in navy blue to indicate the Dharma Chakra, the wheel of law in the Sarnath Lion Capital. This centre symbol or the 'CHAKRA' is a symbol dating back to 2nd century BC. The saffron stands for courage and sacrifice; the white, for purity and truth; the green for growth and auspiciousness.

Playing with applets: The CD accompanying this book contains an applet in the directory Chapter 11. You can play with this applet. The instructions are as follows.

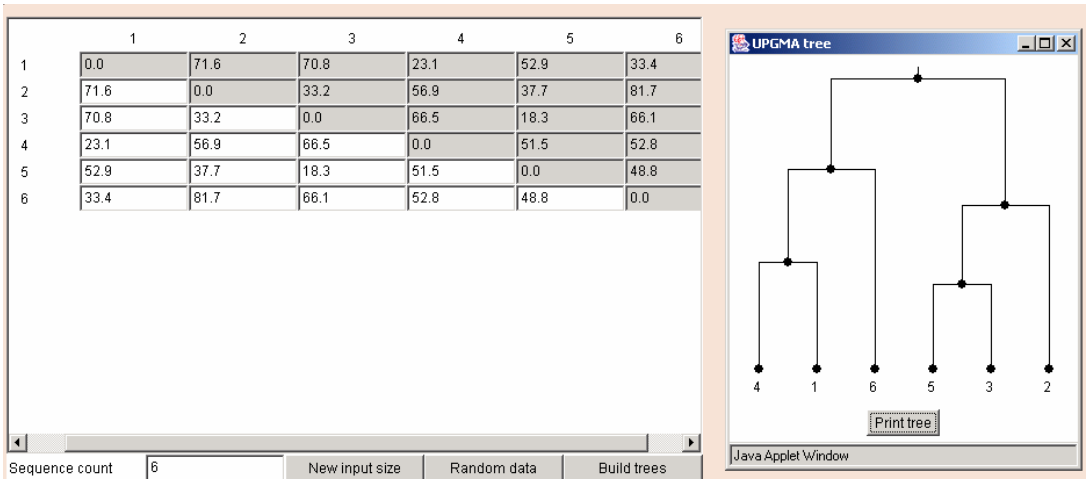


Fig. 11.23 Dendrogram applet.

- Press 'Build trees' to (re)build the UPGMA.
- Edit the fields of the distance matrix (only the lower ones are editable; the matrix is automatically made symmetric when you rebuild the trees).
- Edit the 'Sequence count' field and press 'New input size' if you want to change the size of the distance matrix.
- Press 'Random data' to fill the matrix with random distance data.
- For security reasons, the 'Print tree' button does not work when the applet is run inside a browser.

Summary of Hierarchical Clustering Methods

- No need to specify the number of clusters in advance.
- Hierarchical nature maps nicely onto human intuition for some domains.
- They do not scale well: time complexity of at least $O(n^2)$, where n is the number of total objects.
- Like any heuristic search algorithms, local optima are a problem.
- Interpretation of results is subjective.

11.2 PARTITIONAL CLUSTERINGS

k-Means Clustering

k-means (MacQueen, 1967) is one of the simplest unsupervised learning algorithms that solve the well-known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed a priori. The main idea is to define k centroids, one for each cluster. These centroids should be placed in a cunning way because of different location causes different result. So, the better choice is to place them as much as possible far away from each other. The next step is to take each point belonging to a given dataset and associate it to the nearest centroid. When no point is pending, the first step is completed and an early groupage is done. At this point we need to re-calculate k new centroids as barycentres of the clusters resulting from the previous step. After we have these k new centroids, a new binding has to be done between the same dataset points and the nearest new centroid. A loop has been generated. As a result of this loop we may notice that the k centroids change their location step by step until no more changes are done. In other words centroids do not move any more.

It is well suited to generating globular clusters. The *k*-means method is numerical, unsupervised, non-deterministic and iterative.

Finally, this algorithm aims at minimizing an *objective function*, in this case a squared error function. The objective function

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^{(j)} - C_j \right\|^2$$

where $\left\| x_i^{(j)} - C_j \right\|^2$ is a chosen distance measure between a data point $x_i^{(j)}$ and the cluster centre C_j, is an indicator of the distance of the n data points from their respective cluster centres.

The algorithm is composed of the following steps:

1. Decide on a value for k.
2. Initialize the k cluster centres (randomly, if necessary).
3. Decide the class memberships of the N objects by assigning them to the nearest cluster centres.
4. Re-estimate the k cluster centres, by assuming the memberships found above are correct.
5. If none of the N objects changed membership in the last iteration, exit. Otherwise goto 3.1.

Although it can be proved that the procedure will always terminate, the k-means algorithm does not necessarily find the most optimal configuration, corresponding to the global objective function minimum. The algorithm is also significantly sensitive to the initial randomly selected cluster centres. The k-means algorithm can be run multiple times to reduce this effect.

An example. Suppose that we have n sample feature vectors $\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_n}$ all from the same class, and we know that they fall into k compact clusters, $k < n$. Let $\mathbf{m_i}$ be the mean of the vectors in cluster i. If the clusters are well separated, we can use a minimum-distance classifier to separate them. That is, we can say that $\mathbf{x}$ is in cluster i if $\| \mathbf{x} - \mathbf{m_i} \|$ is the minimum of all the k distances. This suggests the following procedure for finding the k means:

- Make initial guesses for the means $\mathbf{m_1}, \mathbf{m_2}, ..., \mathbf{m_k}$.
- Until there are no changes in any mean

Use the estimated means to classify the samples into clusters
  For i from 1 to k
  Replace $\mathbf{m_i}$ with the mean of all of the samples for cluster i
  end for
end until

Here is an example Figs. 11.24 to 11.28 showing how the means m_1, m_2 and m_3 move into the centres of three clustres.

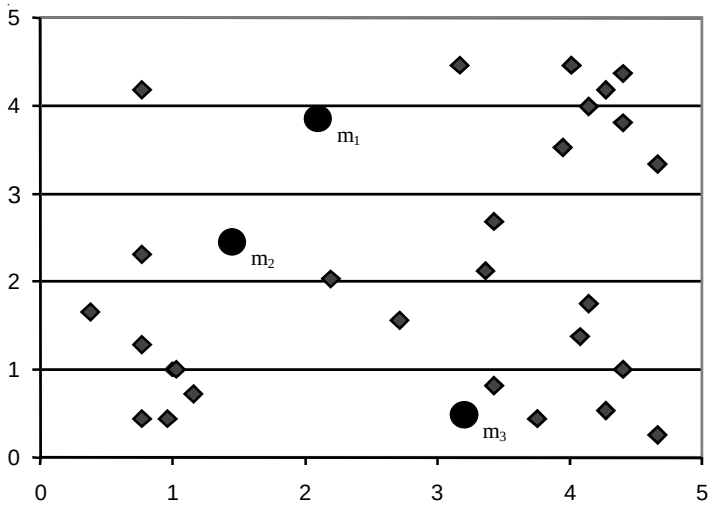


Fig. 11.24 k-means clustering: Step 1.

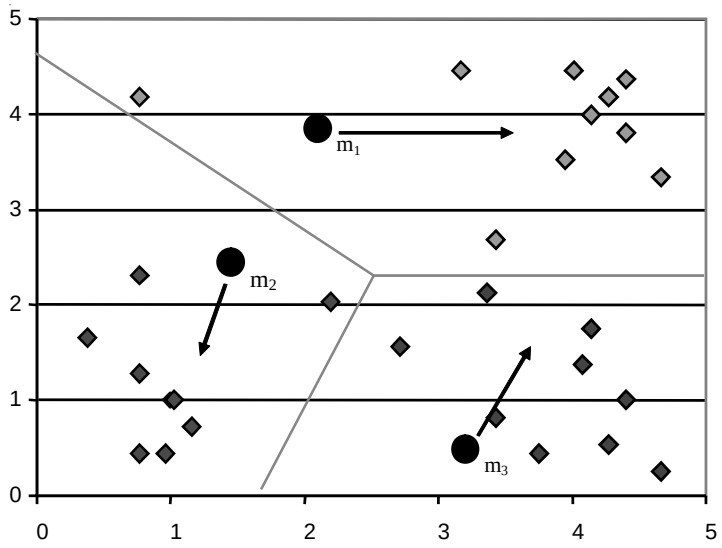


Fig. 11.25 *k*-means clustering: Step 2

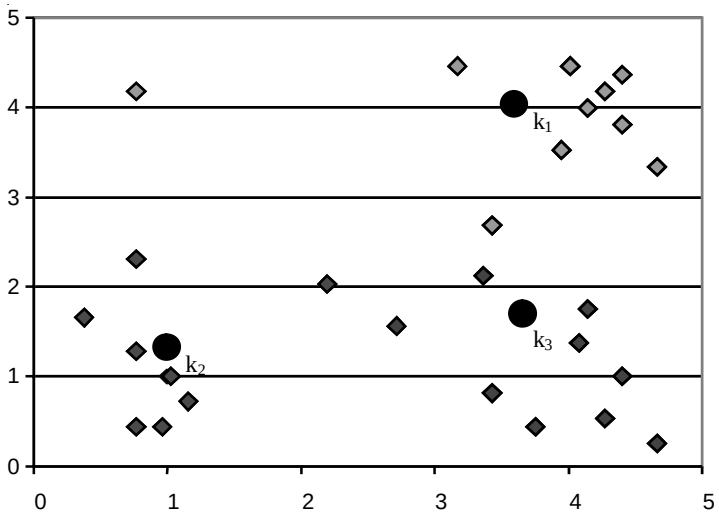


Fig. 11.26 *k*-means clustering: Step 3

Algorithm: k-means, Distance metric: Euclidean distance.

Remarks. This is a simple version of the *k*-means procedure. It can be viewed as a greedy algorithm for partitioning the *n* samples into *k* clusters so as to minimize the sum of the squared distances to the cluster centres. It does have some weaknesses:

- The way to initialize the means was not specified. One popular way to start is to randomly choose *k* of the samples.

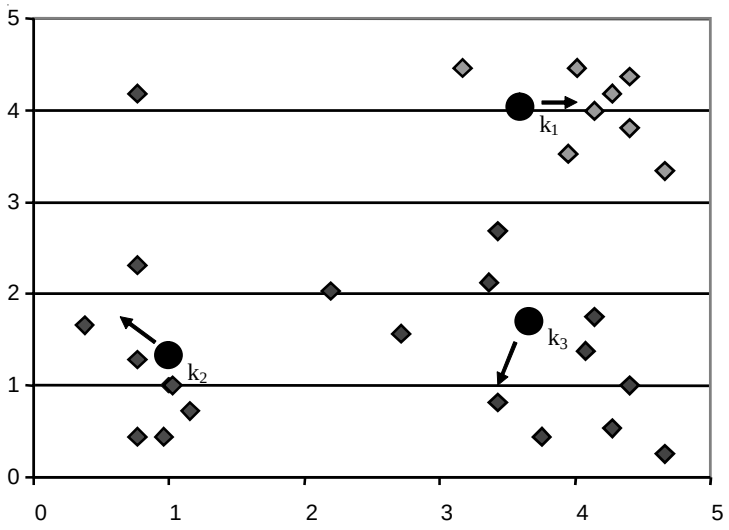


Fig. 11.27 *k*-means clustering: Step 4

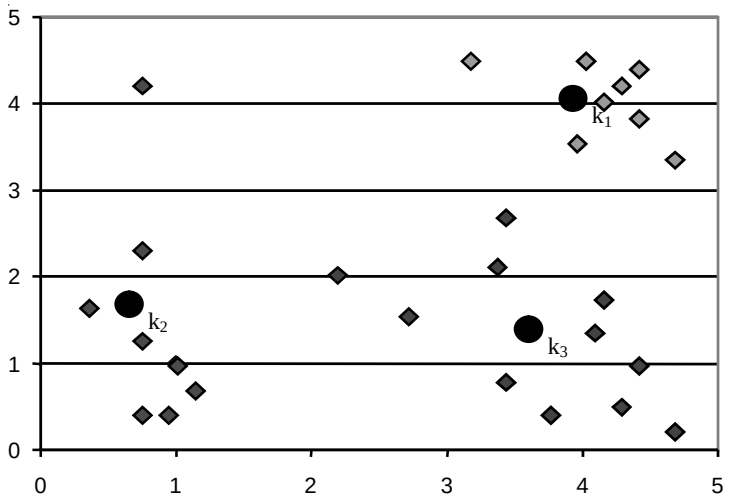


Fig. 11.28 *k*-means clustering: Step 5.

- The results produced depend on the initial values for the means, and it frequently happens that sub-optimal partitions are found. The standard solution is to try a number of different starting points.
- It can happen that the set of samples closest to $\mathbf{m}_i$ is empty, so that $\mathbf{m}_i$ cannot be updated. This is an annoyance that must be handled in an implementation, but that we shall ignore.
- The results depend on the metric used to measure $\| \mathbf{x} - \mathbf{m}_i \|$. A popular solution is to normalize each variable by its standard deviation, though this is not always desirable.
- The results depend on the value of k.

This last problem is particularly troublesome, since we often have no way of knowing how many clusters exist. Unfortunately there is no general theoretical solution to find the optimal

number of clusters for any given dataset. A simple approach is to compare the results of multiple runs with different k classes and choose the best one according to a given criterion, but we need to be careful because increasing k results in smaller error function values by definition, but also an increasing risk of overfitting.

Advantages to Using this Technique

- With a large number of variables, k-means may be computationally faster than hierarchical clustering (if k is small).
- k-means may produce tighter clusters than hierarchical clustering, especially if the clusters are globular.

Disadvantages to Using this Technique

- Difficulty in comparing quality of the clusters produced (e.g. for different initial partitions or values of k affect outcome).
- Fixed number of clusters can make it difficult to predict what k should be.
- Does not work well with non-globular clusters.

Different initial partitions can result in different final clusters. It is helpful to rerun the program using the same as well as different k values, to compare the results achieved.

11.3 *k*-MEDOIDS

Both k-means and k-medoids have similar procedures. In the k-medoids algorithm, only data points in the space can become medoids; however, in the k-means algorithm any point in the space near the data points or data points themselves can be mean points. Based on the cost calculated between a point and an assumed medoid the points are swapped or retained as medoids until there is no net change for all points for the medoid assumed.

k-medoid is a typical partitioning algorithm. The objective of using this algorithm is, for a given k; find k representatives in the dataset so that, when assigning each object to the closest representative, the sum of the distances between representatives and objects, which are assigned to them, is minimal.

Algorithm

1. Arbitrarily choose k objects as the initial medoids (representatives).
2. Repeat
   Assign each remaining object to the cluster with nearest medoids
   Randomly select a non-medoids object, O random;
   Compute the total cost S of swapping O_j with O random;
   If $S < 0$ then swap O_j with O random to form the new set of k medoids;
   Until no change;
3. Done

Working of *k*-medoid Algorithm

The basic strategy of k-medoids clustering algorithm is to find k clusters in n objects by first arbitrarily finding a representative object for each cluster in the data. These representative objects

are called **medoids** and the remaining objects are called **non-medoids**. The distance from all non-medoid to each medoid is calculated and each of them is assigned to the nearest cluster, i.e., to the medoid to which the distance is minimum. The process is repeated by iteratively replaces one of the medoids by one of the non-medoids as long as the quality of the clusters is improved. This quality is estimated using a cost function that measures the average dissimilarity between an object and the medoid of its cluster.

1. Select B representative objects arbitrarily.
2. Compute TC_{ih} for all pairs of objects O_i, O_h where O_i is currently selected, and O_h is not.
3. Select the pair O_i, O_h that corresponds to min (O_i, O_h, TC_{ih}). If the minimum TC_{ih} is negative, replace O_i with O_h, and go back to Step (2).
4. Otherwise, for each non-selected object, find the most similar representative object.

11.4 MODERN CLUSTERING METHODS

Classical methods falls into two groups which we have covered in the previous section. The modern clustering methods can be classified into five groups which also include the classical Hierarchical methods and Partitioning methods.

1. Hierarchical methods
2. Partitioning methods
3. Density-based methods
4. Grid-based methods
5. Model-based methods

We now briefly describe main features of each group and the main algorithms in that group, which have appeared in the literature recently.

1. Hierarchical Methods

The methods proceed successively by either merging the smaller clusters into lager ones, or by splitting the lager clusters. The result of the algorithm is a tree of clusters, called **dendrogram** (creates a hierarchical decomposition of the given data objects). The method can be classified as being either agglomerative (bottom-up, starts with each object forming the separate group) or divisive (top-down, starts with all the objects in the same cluster), based on how the hierarchical decomposition is formed. If one step (merge or split) is done, it can never be undone. To compensate for rigidity of merge or split, the quality of hierarchical agglomeration could be improved by analyzing object linkages at each hierarchical partitioning or integrating other clustering techniques, such as iterative relocation.

Examples:

- ***BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies)***—It uses a hierarchical data structure called **CF-tree** for partitioning the incoming data points in an incremental and dynamic way. CF-tree is a height-balanced tree, which stores the clustering features and it is based on two parameters: branching factor B and threshold T, which referred to diameter of the cluster of each cluster must be less than T.

- ***CURE (Clustering Using REpresentatives).*** It represents each cluster by a certain number of points that are generated by selecting well-scattered points and then shrinking them toward the cluster centroid by specified fraction. It uses a combination of random sampling and partition clustering to handle large databases.
- ***ROCK.*** Robust Clustering Algorithms for Boolean and Categorical data. It produces the point's neighbour and link concepts and bases on them in order to measure the similarity/ proximity between a pair of data points.

Partitioning Methods

The methods attempt to directly decompose the data into disjoint clusters (first creates an initial k partitions, where parameter k is the number of partitions to construct; then it uses an iterative relocation technique that attempts to improve the partitioning by moving objects from one group to another). The method works well for finding spherical-shaped clusters in small or medium-sized databases.

Examples

- ***k-means.*** each cluster is represented by the mean value of the objects in the cluster. Then the algorithms assigns each object of the dataset to the cluster whose centre is the nearest, and re-computes the centres. The process continues until the centres of the clusters stop changing.
- ***PAM (Partitioning Around Medoids).*** Each cluster is represented by one of the objects located near the centre of the cluster. The algorithm begins by selecting an object as medoid for each of n clusters, then each of non-selected objects is grouped with the medoid to which it is the most similar. The algorithm swaps the medoids with other non-selected objects until all objects qualify as medoid.
- ***CLARA (Clustering LARge Application).*** It is the implementation of the PAM algorithm in a subset of the dataset. It draws multiple samples of the dataset, applies PAM on the samples, and then outputs the best clustering out of these samples.
- ***CLARANS (Clustering Large Application based on RANdomised Search).*** Searching a graph where every node is a potential solution, that is, a set of k medoids. It selects the node and compares it to a user-defined number of their neighbours searching for a local minimum and moves to the neighbour's node. If the local optimum is found, it starts a new randomly selected node in search for a new local optimum.
- ***k-mode.*** based on k-means algorithm, but it aims at clustering categorical data.

Density-Based Methods

The methods group neighbouring objects into clusters based on density conditions. It either grows cluster according to the density of neighbourhood or according to some density function.

Examples

- ***DENCLUE (DENsity-based CLUstEring).*** Models the overall point density analytically as the sum of influence functions of the data points; the clusters can be identified by determining density attractors.

- ***DBSCAN (Density-Based Spatial Clustering of Application with Noise).*** For each points in the cluster, the neighbourhood of a given radius has to contain at least a minimum number of points. The algorithm can handle noise and discover the clusters of arbitrary shape.
- ***OPTICS (Ordering Points To Identify the Clustering Structure).*** Computes an augmented clustering ordering for automatic and interactive cluster analysis.

Grid-based Methods

The methods quantize the space into a finite number of the cells that form a grid structure, and then perform clustering on the grid structure.

Examples

- ***STING (STatistical INformation Grid-based method).*** Divides the spatial area into rectangular cells using a hierarchical structure; it goes through the dataset and computes statistical parameters of each numerical feature of the objects within cells, then it generates a hierarchical structure of the grid cells to represent the clustering information at different levels.
- ***Wave cluster.*** Based on signal processing techniques to convert the spatial data into frequency domain; it first summarizes the data by imposing a multidimensional grid structure onto the data space, each grid cell summarizes the information of a group of points that man into the cell, then it uses a wavelet transformation to transform the original feature space. It identifies the clusters by finding the dense regions in the transformed domain.

Model-based Methods

The methods hypothesize a model for each of the clusters and find the best fit of the data to that model. Typical model-based methods involve statistical approaches (**COBWEB, CLASSIT, AutoClass**) or neural network approaches (**Competitive learning** and **Self-organizing maps**).

Some popular clustering algorithms are discussed in the forthcoming sections. The multitudes of clustering algorithms have come up mainly by incorporating difficulties faced in its predecessor or having different kind of data.

11.5 BIRCH

Zhang et al. [13] presented an algorithm—BIRCH (Balanced Iterative Reducing and Clustering)—for clustering of large sets of points. The method they presented is the incremental one with possibility of adjustment of memory requirements to the size of memory that is available. The authors used concepts called **Clustering Feature** and **CF tree**.

A **clustering feature CF** is the triple summarizing information about subclusters of points. Given N d-dimensional points in the subcluster: $\{\vec{X}_i, i = 1, 2, \ldots, N\}$, CF is defined as:

$$CF = (N, \overrightarrow{LS}, \overrightarrow{SS})$$