19 Clustering

19.1 Introduction

In this chapter we continue with the theme of extracting information from unlabelled data and turn to the important topic of *clustering*. Clustering is concerned with grouping together objects that are similar to each other and dissimilar to the objects belonging to other clusters.

In many fields there are obvious benefits to be had from grouping together similar objects. For example

– In an economics application we might be interested in finding countries whose economies are similar.

– In a financial application we might wish to find clusters of companies that have similar financial performance.

– In a marketing application we might wish to find clusters of customers with similar buying behaviour.

– In a medical application we might wish to find clusters of patients with similar symptoms.

– In a document retrieval application we might wish to find clusters of documents with related content.

– In a crime analysis application we might look for clusters of high volume crimes such as burglaries or try to cluster together much rarer (but possibly related) crimes such as murders.

M. Bramer, *Principles of Data Mining*, Undergraduate Topics in Computer Science, DOI 10.1007/978-1-4471-7307-6_19

There are many algorithms for clustering. We will describe two methods for which the similarity between objects is based on a measure of the distance between them.

In the restricted case where each object is described by the values of just two attributes, we can represent them as points in a two-dimensional space (a plane) such as Figure 19.1.

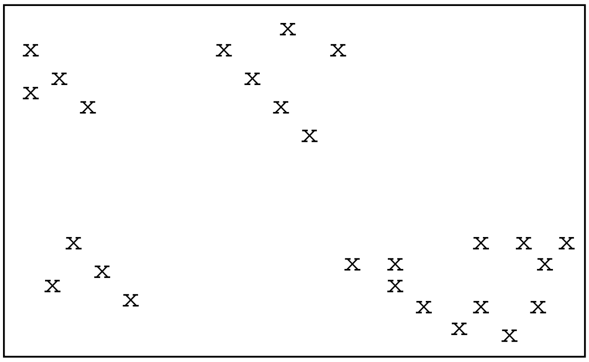

Figure 19.1 Objects for Clustering

It is usually easy to visualise clusters in two dimensions. The points in Figure 19.1 seem to fall naturally into four groups as shown by the curves drawn surrounding sets of points in Figure 19.2.

However there is frequently more than one possibility. For example are the points in the lower-right corner of Figure 19.1 one cluster (as shown in Figure 19.2) or two (as shown in Figure 19.3)?

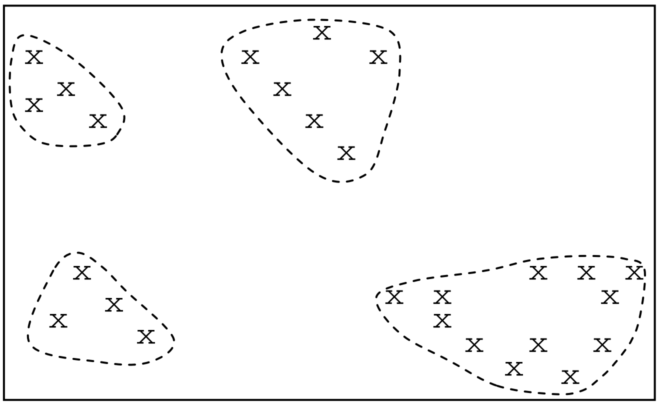

Figure 19.2 Clustering of Objects in Figure 19.1(first version)

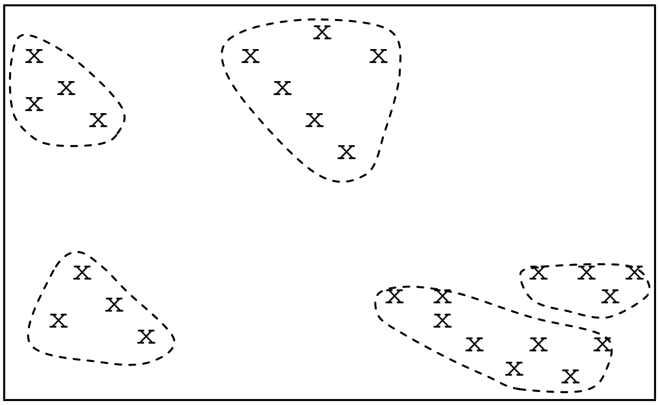

Figure 19.3 Clustering of Objects in Figure 19.1(second version)

In the case of three attributes we can think of the objects as being points in a three-dimensional space (such as a room) and visualising clusters is generally straightforward too. For larger dimensions (i.e. larger numbers of attributes) it soon becomes impossible to visualise the points, far less the clusters.

The diagrams in this chapter will use only two dimensions, although in practice the number of attributes will usually be more than two and can often be large.

Before using a *distance-based clustering algorithm* to cluster objects, it is first necessary to decide on a way of measuring the distance between two points. As for nearest neighbour classification, discussed in Chapter 3, a measure commonly used when clustering is the Euclidean distance. To avoid complications we will assume that all attribute values are continuous. (Attributes that are categorical can be dealt with as described in Chapter 3.)

We next need to introduce the notion of the 'centre' of a cluster, generally called its *centroid.*

Assuming that we are using Euclidean distance or something similar as a measure we can define the centroid of a cluster to be the point for which each attribute value is the average of the values of the corresponding attribute for all the points in the cluster.

So the centroid of the four points (with 6 attributes)

8.0	7.2	0.3	23.1	11.1	−6.1
2.0	−3.4	0.8	24.2	18.3	−5.2
−3.5	8.1	0.9	20.6	10.2	−7.3
−6.0	6.7	0.5	12.5	9.2	−8.4

would be

0.125	4.65	0.625	20.1	12.2	−6.75

The centroid of a cluster will sometimes be one of the points in the cluster, but frequently, as in the above example, it will be an ‘imaginary’ point, not part of the cluster itself, which we can take as marking its centre. The value of the idea of the centroid of a cluster will be illustrated in what follows.

There are many methods of clustering. In this book we will look at two of the most commonly used: *k-means clustering* and *hierarchical clustering*.

19.2 *k*-Means Clustering

k-means clustering is an *exclusive clustering algorithm*. Each object is assigned to precisely one of a set of clusters. (There are other methods that allow objects to be in more than one cluster.)

For this method of clustering we start by deciding how many clusters we would like to form from our data. We call this value k. The value of k is generally a small integer, such as 2, 3, 4 or 5, but may be larger. We will come back later to the question of how we decide what the value of k should be.

There are many ways in which k clusters might potentially be formed. We can measure the quality of a set of clusters using the value of an *objective function* which we will take to be the sum of the squares of the distances of each point from the centroid of the cluster to which it is assigned. We would like the value of this function to be as small as possible.

We next select k points (generally corresponding to the location of k of the objects). These are treated as the centroids of k clusters, or to be more precise as the centroids of k potential clusters, which at present have no members. We can select these points in any way we wish, but the method may work better if we pick k initial points that are fairly far apart.

We now assign each of the points one by one to the cluster which has the nearest centroid.

When all the objects have been assigned we will have k clusters based on the original k centroids but the ‘centroids’ will no longer be the true centroids of the clusters. Next we recalculate the centroids of the clusters, and then repeat the previous steps, assigning each object to the cluster with the nearest centroid etc. The entire algorithm is summarised in Figure 19.4.

1. Choose a value of k.
2. Select k objects in an arbitrary fashion. Use these as the initial set of k centroids.
3. Assign each of the objects to the cluster for which it is nearest to the centroid.
4. Recalculate the centroids of the k clusters.
5. Repeat steps 3 and 4 until the centroids no longer move.

Figure 19.4 The k-Means Clustering Algorithm

19.2.1 Example

We will illustrate the k-means algorithm by using it to cluster the 16 objects with two attributes x and y that are listed in Figure 19.5.

x	y
6.8	12.6
0.8	9.8
1.2	11.6
2.8	9.6
3.8	9.9
4.4	6.5
4.8	1.1
6.0	19.9
6.2	18.5
7.6	17.4
7.8	12.2
6.6	7.7
8.2	4.5
8.4	6.9
9.0	3.4
9.6	11.1

Figure 19.5 Objects for Clustering (Attribute Values)

The 16 points corresponding to these objects are shown diagrammatically in Figure 19.6. The horizontal and vertical axes correspond to attributes x and y, respectively.

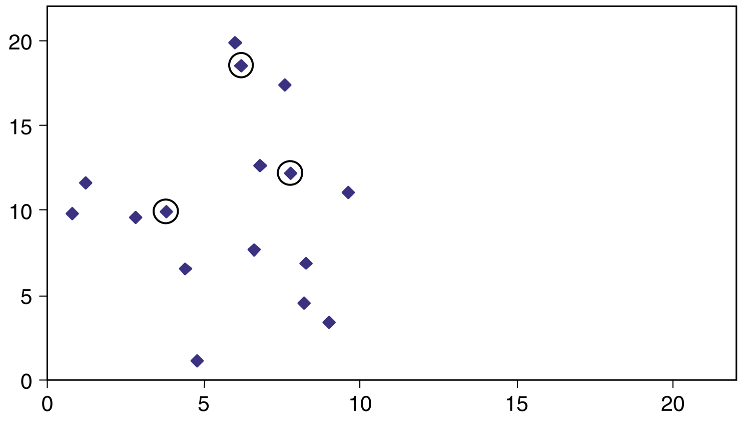

Figure 19.6 Objects for Clustering

Three of the points shown in Figure 19.6 have been surrounded by small circles. We will assume that we have chosen $k = 3$ and that these three points have been selected to be the locations of the initial three centroids. This initial (fairly arbitrary) choice is shown in Figure 19.7.

	Initial	
	x	y
Centroid 1	3.8	9.9
Centroid 2	7.8	12.2
Centroid 3	6.2	18.5

Figure 19.7 Initial Choice of Centroids

The columns headed $d1$, $d2$ and $d3$ in Figure 19.8 show the Euclidean distance of each of the 16 points from the three centroids. For the purposes of this example, we will not normalise or weight either of the attributes, so the distance of the first point $(6.8, 12.6)$ from the first centroid $(3.8, 9.9)$ is simply

$$\sqrt{(6.8 - 3.8)^2 + (12.6 - 9.9)^2} = 4.0 \text{ (to one decimal place)}$$

The column headed 'cluster' indicates the centroid closest to each point and thus the cluster to which it should be assigned.

The resulting clusters are shown in Figure 19.9 below.

The centroids are indicated by small circles. For this first iteration they are also actual points within the clusters. The centroids are those that were used to construct the three clusters but are not the true centroids of the clusters once they have been created.

x	y	$d1$	$d2$	$d3$	cluster
6.8	12.6	4.0	1.1	5.9	2
0.8	9.8	3.0	7.4	10.2	1
1.2	11.6	3.1	6.6	8.5	1
2.8	9.6	1.0	5.6	9.5	1
3.8	9.9	0.0	4.6	8.9	1
4.4	6.5	3.5	6.6	12.1	1
4.8	1.1	8.9	11.5	17.5	1
6.0	19.9	10.2	7.9	1.4	3
6.2	18.5	8.9	6.5	0.0	3
7.6	17.4	8.4	5.2	1.8	3
7.8	12.2	4.6	0.0	6.5	2
6.6	7.7	3.6	4.7	10.8	1
8.2	4.5	7.0	7.7	14.1	1
8.4	6.9	5.5	5.3	11.8	2
9.0	3.4	8.3	8.9	15.4	1
9.6	11.1	5.9	2.1	8.1	2

Figure 19.8 Objects for Clustering (Augmented)

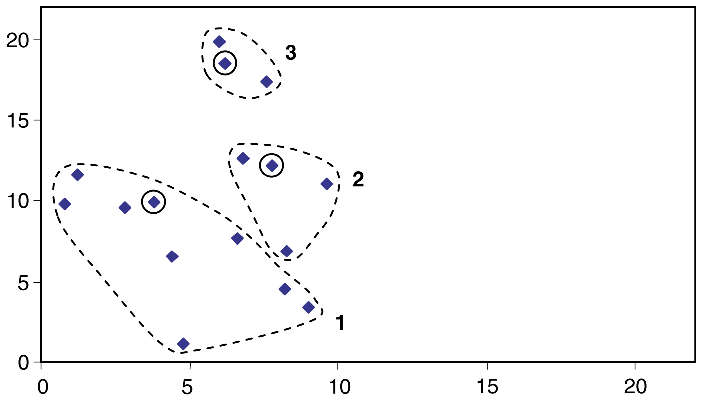

Figure 19.9 Initial Clusters

We next calculate the centroids of the three clusters using the x and y values of the objects currently assigned to each one. The results are shown in Figure 19.10.

The three centroids have all been moved by the assignment process, but the movement of the third one is appreciably less than for the other two.

	Initial		After first iteration	
	x	y	x	y
Centroid 1	3.8	9.9	4.6	7.1
Centroid 2	7.8	12.2	8.2	10.7
Centroid 3	6.2	18.5	6.6	18.6

Figure 19.10 Centroids After First Iteration

We next reassign the 16 objects to the three clusters by determining which centroid is closest to each one. This gives the revised set of clusters shown in Figure 19.11.

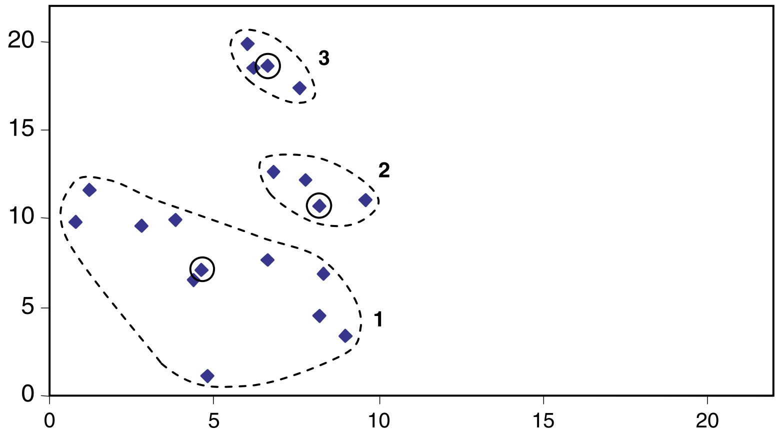

Figure 19.11 Revised Clusters

The centroids are again indicated by small circles. However from now on the centroids are 'imaginary points' corresponding to the 'centre' of each cluster, not actual points within the clusters.

These clusters are very similar to the previous three, shown in Figure 19.9. In fact only one point has moved. The object at $(8.3, 6.9)$ has moved from cluster 2 to cluster 1.

We next recalculate the positions of the three centroids, giving Figure 19.12.

The first two centroids have moved a little, but the third has not moved at all.

We assign the 16 objects to clusters once again, giving Figure 19.13.

These are the same clusters as before. Their centroids will be the same as those from which the clusters were generated. Hence the termination condition of the k-means algorithm 'repeat ... until the centroids no longer move' has

	Initial		After first iteration		After second iteration	
	x	y	x	y	x	y
Centroid 1	3.8	9.9	4.6	7.1	5.0	7.1
Centroid 2	7.8	12.2	8.2	10.7	8.1	12.0
Centroid 3	6.2	18.5	6.6	18.6	6.6	18.6

Figure 19.12 Centroids After First Two Iterations

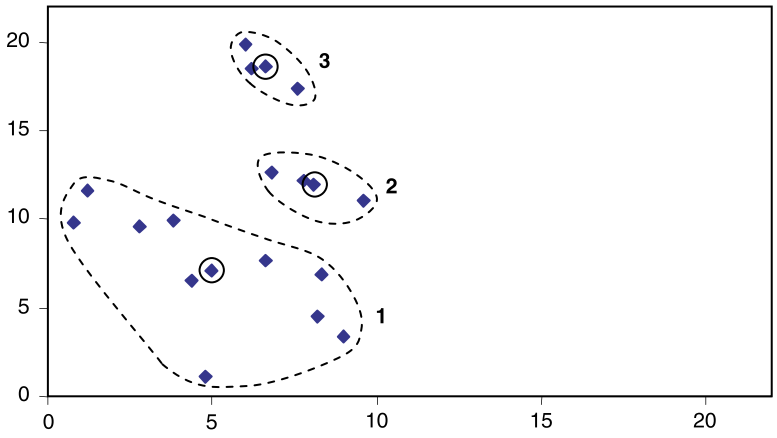

Figure 19.13 Third Set of Clusters

been met and these are the final clusters produced by the algorithm for the initial choice of centroids made.

19.2.2 Finding the Best Set of Clusters

It can be proved that the k-means algorithm will always terminate, but it does not necessarily find the best set of clusters, corresponding to minimising the value of the objective function. The initial selection of centroids can significantly affect the result. To overcome this, the algorithm can be run several times for a given value of k, each time with a different choice of the initial k centroids, the set of clusters with the smallest value of the objective function then being taken.

The most obvious drawback of this method of clustering is that there is no principled way to know what the value of k ought to be. Looking at the final set of clusters in the above example (Figure 19.13), it is not clear that $k = 3$ is the most appropriate choice. Cluster 1 might well be broken into several separate clusters. We can choose a value of k pragmatically as follows.

If we imagine choosing $k = 1$, i.e. all the objects are in a single cluster, with the initial centroid selected in a random way (a very poor idea), the value of the objective function is likely to be large. We can then try $k = 2$, $k = 3$ and $k = 4$, each time experimenting with a different choice of the initial centroids and choosing the set of clusters with the smallest value. Figure 19.14 shows the (imaginary) results of such a series of experiments.

Value of k	Value of objective function
1	62.8
2	12.3
3	9.4
4	9.3
5	9.2
6	9.1
7	9.05

Figure 19.14 Value of Objective Function for Different Values of k

These results suggest that the best value of k is probably 3. The value of the function for $k = 3$ is much less than for $k = 2$, but only a little better than for $k = 4$. It is possible that the value of the objective function drops sharply after $k = 7$, but even if it does $k = 3$ is probably still the best choice. We normally prefer to find a fairly small number of clusters as far as possible.

Note that we are *not* trying to find the value of k with the smallest value of the objective function. That will occur when the value of k is the same as the number of objects, i.e. each object forms its own cluster of one. The objective function will then be zero, but the clusters will be worthless. This is another example of the *overfitting* of data discussed in Chapter 9. We usually want a fairly small number of clusters and accept that the objects in a cluster will be spread around the centroid (but ideally not too far away).

19.3 Agglomerative Hierarchical Clustering

Another very popular clustering technique is called *Agglomerative Hierarchical Clustering*.

As for k-means clustering we need to choose a way of measuring the distance between two objects. Also as for that method a commonly used distance measure is Euclidean distance (defined in Chapter 3). In two dimensions Euclidean distance is just the 'straight line' distance between two points.

The idea behind Agglomerative Hierarchical Clustering is a simple one. We start with each object in a cluster of its own and then repeatedly merge the closest pair of clusters until we end up with just one cluster containing everything. The basic algorithm is given in Figure 19.15.

1. Assign each object to its own single-object cluster. Calculate the distance between each pair of clusters.
2. Choose the closest pair of clusters and merge them into a single cluster (so reducing the total number of clusters by one).
3. Calculate the distance between the new cluster and each of the old clusters.
4. Repeat steps 2 and 3 until all the objects are in a single cluster.

Figure 19.15 Agglomerative Hierarchical Clustering: Basic Algorithm

If there are N objects there will be $N - 1$ mergers of two objects needed at Step 2 to produce a single cluster. However the method does not only produce a single large cluster, it gives a *hierarchy* of clusters as we shall see.

Suppose we start with eleven objects A, B, C, . . . , K located as shown in Figure 19.16 and we merge clusters on the basis of Euclidean distance.

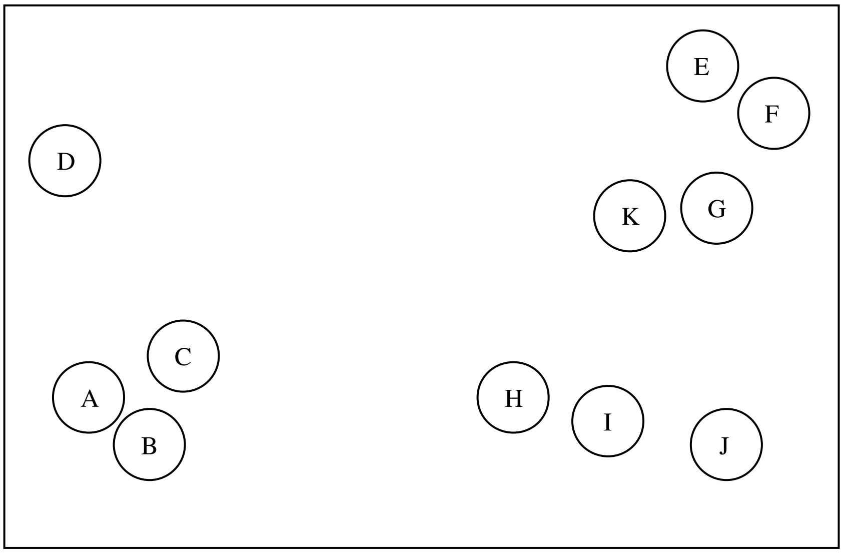

Figure 19.16 Original Data (11 Objects)

It will take 10 'passes' through the algorithm, i.e. repetitions of Steps 2 and 3, to merge the initial 11 single object clusters into a single cluster. Let us assume the process starts by choosing objects A and B as the pair that are closest and merging them into a new cluster which we will call AB. The next step may be to choose clusters AB and C as the closest pair and to merge them. After two passes the clusters then look as shown in Figure 19.17.

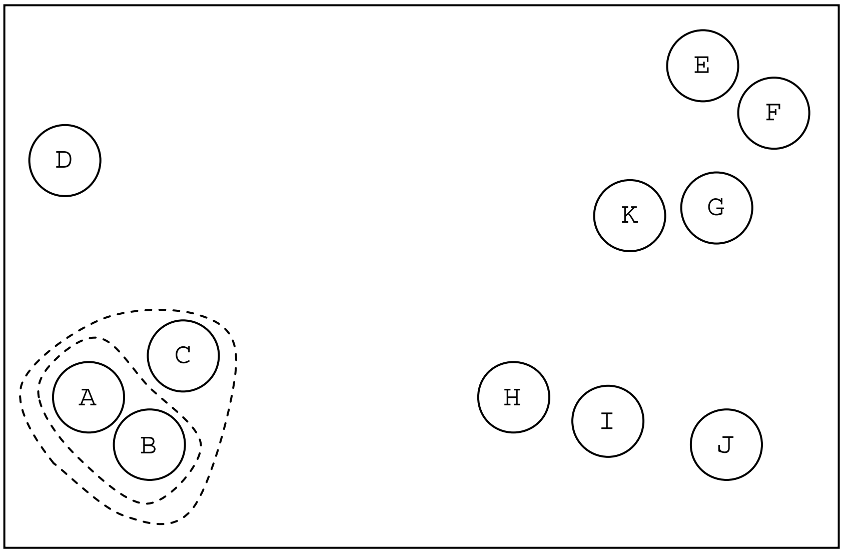

Figure 19.17 Clusters After Two Passes

We will use notation such as A and B → AB to mean 'clusters A and B are merged to form a new cluster, which we will call AB'.

Without knowing the precise distances between each pair of objects, a plausible sequence of events is as follows.

1. A and B → AB
2. AB and C → ABC
3. G and K → GK
4. E and F → EF
5. H and I → HI
6. EF and GK → EFGK
7. HI and J → HIJ
8. ABC and D → ABCD
9. EFGK and HIJ → EFGKHIJ
10. ABCD and EFGKHIJ → ABCDEFGKHIJ

The final result of this hierarchical clustering process is shown in Figure 19.18, which is called a *dendrogram*. A dendrogram is a binary tree (two branches at each node). However, the positioning of the clusters does not correspond to their physical location in the original diagram. All the original objects are placed at the same level (the bottom of the diagram), as leaf nodes. The root of the tree is shown at the top of the diagram. It is a cluster containing all the objects. The other nodes show smaller clusters that were generated as the process proceeded.

If we call the bottom row of the diagram level 1 (with clusters A, B, C, ..., K), we can say that the level 2 clusters are AB, HI, EF and GK, the level 3 clusters are ABC, HIJ and EFGK, and so on. The root node is at level 5.

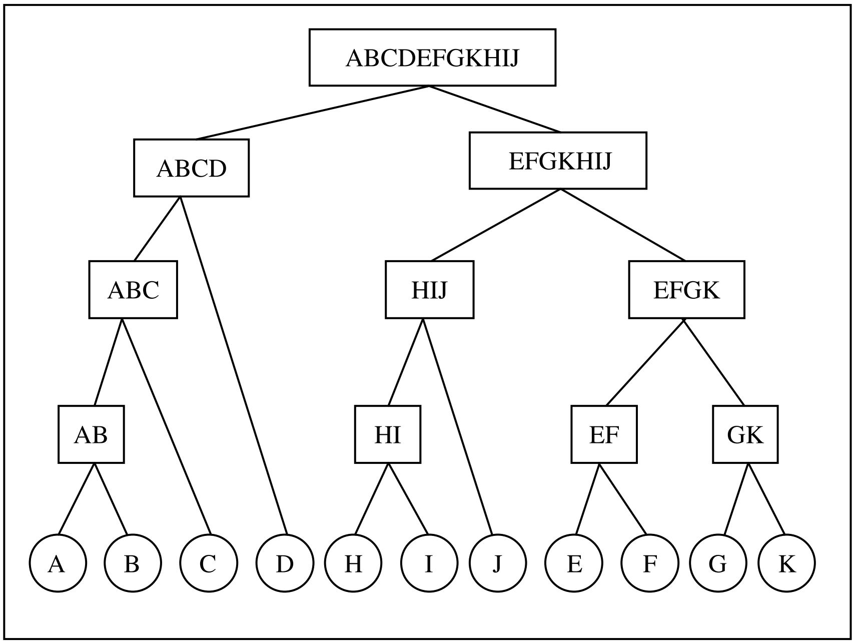

Figure 19.18 A Possible Dendrogram Corresponding to Figure 19.16

19.3.1 Recording the Distance Between Clusters

It would be very inefficient to calculate the distance between each pair of clusters for each pass through the algorithm, especially as the distance between those clusters not involved in the most recent merger cannot have changed.

The usual approach is to generate and maintain a *distance matrix* giving the distance between each pair of clusters.

If we have six objects a, b, c, d, e and f, the initial distance matrix might look like Figure 19.19.

	a	b	c	d	e	f
a	0	12	6	3	25	4
b	12	0	19	8	14	15
c	6	19	0	12	5	18
d	3	8	12	0	11	9
e	25	14	5	11	0	7
f	4	15	18	9	7	0

Figure 19.19 Example of a Distance Matrix

Note that the table is symmetric, so not all values have to be calculated (the distance from c to f is the same as the distance from f to c etc.). The values on the diagonal from the top-left corner to the bottom-right corner must always be zero (the distance from a to a is zero etc.).

From the distance matrix of Figure 19.19 we can see that the closest pair of clusters (single objects) are a and d, with a distance value of 3. We combine these into a single cluster of two objects which we will call ad. We can now rewrite the distance matrix with rows a and d replaced by a single row ad and similarly for the columns (Figure 19.20).

The entries in the matrix for the various distances between b, c, e and f obviously remain the same, but how should we calculate the entries in row and column ad?

	ad	b	c	e	f
ad	0	?	?	?	?
b	?	0	19	14	15
c	?	19	0	5	18
e	?	14	5	0	7
f	?	15	18	7	0

Figure 19.20 Distance Matrix After First Merger (Incomplete)

We could calculate the position of the centroid of cluster ad and use that to measure the distance of cluster ad from clusters b, c, e and f. However for hierarchical clustering a different approach, which involves less calculation, is generally used.

In *single-link clustering* the distance between two clusters is taken to be the shortest distance from any member of one cluster to any member of the other cluster. On this basis the distance from *ad* to *b* is 8, the shorter of the distance from *a* to *b* (12) and the distance from *d* to *b* (8) in the original distance matrix.

Two alternatives to *single-link clustering* are *complete-link clustering* and *average-link clustering*, where the distance between two clusters is taken to be the longest distance from any member of one cluster to any member of the other cluster, or the average such distance respectively.

Returning to the example and assuming that we are using single-link clustering, the position after the first merger is given in Figure 19.21.

	ad	*b*	*c*	*e*	*f*
ad	0	8	6	11	4
b	8	0	19	14	15
c	6	19	0	5	18
e	11	14	5	0	7
f	4	15	18	7	0

Figure 19.21 Distance Matrix After First Merger

The smallest (non-zero) value in the table is now 4, which is the distance between cluster *ad* and cluster *f*, so we next merge these clusters to form a three-object cluster *adf*. The distance matrix, using the single-link method of calculation, now becomes Figure 19.22.

	adf	*b*	*c*	*e*
adf	0	8	6	7
b	8	0	19	14
c	6	19	0	5
e	7	14	5	0

Figure 19.22 Distance Matrix After Two Mergers

The smallest non-zero is now 5, the distance from cluster *c* to cluster *e*. These clusters are now merged into a single new cluster *ce* and the distance matrix is changed to Figure 19.23.

Clusters *adf* and *ce* are now the closest, with distance 6 so we merge them into a single cluster adfce. The distance matrix becomes Figure 19.24.

At the final stage clusters *adfce* and *b* are merged into a single cluster *adfceb* which contains all the original six objects. The dendrogram corresponding to this clustering process is shown in Figure 19.25.

	adf	*b*	*ce*
adf	0	8	6
b	8	0	14
ce	6	14	0

Figure 19.23 Distance Matrix After Three Mergers

	adfce	*b*
adfce	0	8
b	8	0

Figure 19.24 Distance Matrix After Four Mergers

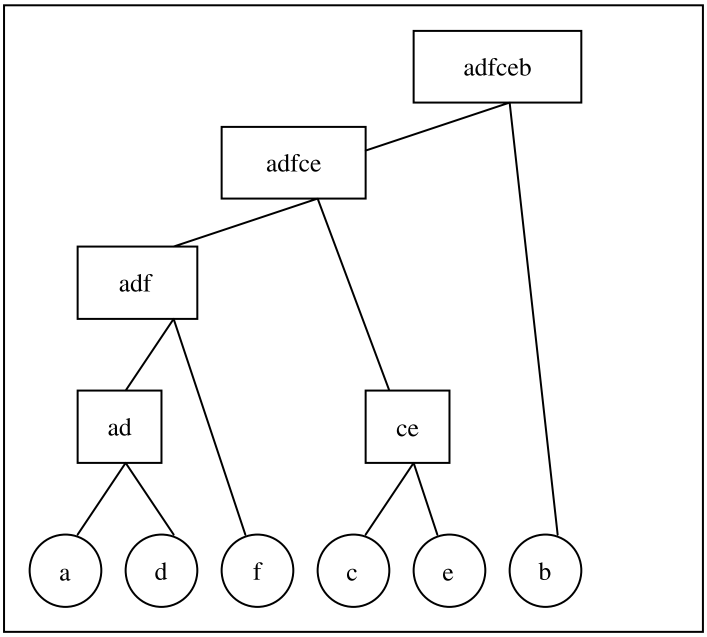

Figure 19.25 Dendrogram Corresponding to Hierarchical Clustering Process

19.3.2 Terminating the Clustering Process

Often we are content to allow the clustering algorithm to produce a complete cluster hierarchy. However we may prefer to end the merger process when we have converted the original N objects to a ‘small enough’ set of clusters.

We can do this in several ways. For example we can merge clusters until only some pre-defined number remain. Alternatively we can stop merging when a newly created cluster fails to meet some criterion for its compactness, e.g. the average distance between the objects in the cluster is too high.

19.4 Chapter Summary

This chapter continues with the theme of extracting information from unlabelled data. Clustering is concerned with grouping together objects that are similar to each other and dissimilar to objects belonging to other clusters.

There are many methods of clustering. Two of the most widely used, *k-means clustering* and *hierarchical clustering* are described in detail.

19.5 Self-assessment Exercises for Chapter 19

1. Using the method shown in Section 19.2, cluster the following data into three clusters, using the k-means method.

x	y
10.9	12.6
2.3	8.4
8.4	12.6
12.1	16.2
7.3	8.9
23.4	11.3
19.7	18.5
17.1	17.2
3.2	3.4
1.3	22.8
2.4	6.9
2.4	7.1
3.1	8.3
2.9	6.9
11.2	4.4
8.3	8.7

2. For the example given in Section 19.3.1, what would be the distance matrix after each of the first three mergers if complete-link clustering were used instead of single-link clustering?