12

Clustering

The second class of models for unsupervised learning is represented by *clustering* methods, which will be covered in this chapter. By defining appropriate metrics and the induced notions of distance and similarity between pairs of observations, the purpose of clustering methods is the identification of homogeneous groups of records called *clusters*. With respect to the specific distance selected, the observations belonging to each cluster must be close to one another and far from those included in other clusters.

We begin this chapter by reviewing the main features of clustering models. We will then illustrate the most popular measures of distance between pairs of observations, in relation to the nature of the attributes contained in the dataset. Partition methods will then be described, focusing in particular on K-means and K-medoids algorithms. Finally, we will illustrate both agglomerative and divisive hierarchical methods, in connection with the main metrics that express the inhomogeneity among distinct clusters. We will also introduce some indicators of the quality of clustering models.

12.1 Clustering methods

The aim of clustering models is to subdivide the records of a dataset into homogeneous groups of observations, called *clusters*, so that observations belonging to one group are similar to one another and dissimilar from observations included in other groups.

Grouping objects by affinity is a typical reasoning pattern applied by the human brain. Also for this reason clustering models have long been used in various disciplines, such as social sciences, biology, astronomy, statistics, image recognition, processing of digital information, marketing and data mining.

Clustering models are useful for various purposes. In some applications, the clusters generated may provide a meaningful interpretation of the phenomenon

Business Intelligence: Data Mining and Optimization for Decision Making Carlo Vercellis
 ISBN: 978-0-470-51138-1

of interest. For example, grouping customers based on their purchase behaviors may reveal the existence of a cluster corresponding to a market niche to which it might be appropriate to address specific marketing actions for promotional purposes. Furthermore, subdivision into clusters may be the preliminary phase of a data mining project that will be followed by the application of other methodologies within each cluster. In a retention analysis, a preliminary subdivision into clusters may be followed by the development of distinct classification models, with the aim of identifying with greater accuracy the customers characterized by a high probability of churning. Finally, grouping into clusters may prove useful in the course of exploratory data analysis to highlight outliers and to identify an observation that might represent on its own an entire cluster, in order to reduce the size of the dataset.

Clustering methods must fulfill a few general requirements, as indicated below.

Flexibility. Some clustering methods can be applied to numerical attributes only, for which it is possible to use the Euclidean metrics to calculate the distances between observations. However, a flexible clustering algorithm should also be able to analyze datasets containing categorical attributes. Algorithms based on the Euclidean metrics tend to generate spherical clusters and have difficulty in identifying more complex geometrical forms.

Robustness. The robustness of an algorithm manifests itself through the stability of the clusters generated with respect to small changes in the values of the attributes of each observation. This property ensures that the given clustering method is basically unaffected by the noise possibly existing in the data. Moreover, the clusters generated must be stable with respect to the order of appearance of the observations in the dataset.

Efficiency. In some applications the number of observations is quite large and therefore clustering algorithms must generate clusters efficiently in order to guarantee reasonable computing times for large problems. In the case of massive datasets, one may also resort to the extraction of samples of reduced size in order to generate clusters more efficiently. However, this approach inevitably implies a lower robustness for the clusters so generated. Clustering algorithms must also prove efficient with respect to the number of attributes existing in the dataset.

12.1.1 Taxonomy of clustering methods

Clustering methods can be classified into a few main types based on the logic used for deriving the clusters: *partition* methods, *hierarchical* methods, *density based* methods and *grid* methods.

Partition methods. Partition methods, described in Section 12.2, develop a subdivision of the given dataset into a predetermined number K of non-empty subsets. They are suited to obtaining groupings of a spherical or at most convex shape, and can be applied to datasets of small or medium size.

Hierarchical methods. Hierarchical methods, described in Section 12.3, carry out multiple subdivisions into subsets, based on a tree structure and characterized by different homogeneity thresholds within each cluster and inhomogeneity thresholds between distinct clusters. Unlike partition methods, hierarchical algorithms do not require the number of clusters to be predetermined.

Density-based methods. Whereas the two previous classes of algorithms are founded on the notion of distance between observations and between clusters, density-based methods derive clusters from the number of observations locally falling in a neighborhood of each observation. More precisely, for each record belonging to a specific cluster, a neighborhood with a specified diameter must contain a number of observations which should not be lower than a minimum threshold value. Density-based methods can identify clusters of non-convex shape and effectively isolate any possible outliers.

Grid methods. Grid methods first derive a discretization of the space of the observations, obtaining a grid structure consisting of cells. Subsequent clustering operations are developed with respect to the grid structure and generally achieve reduced computing times, despite a lower accuracy in the clusters generated.

A second distinction is concerned with the methods for assigning the observations to each single cluster. It is possible to include each observation *exclusively* in a single cluster or to place it by *superposition* into multiple clusters. Furthermore, *fuzzy* methods have been developed which assign the observations to the clusters with a weight between 0 (the observation is totally extraneous to the cluster) and 1 (the observation exclusively belongs to the cluster), with the additional condition that the sum of the weights over all clusters be equal to 1. Finally, a distinction should be made between *complete* clustering methods, which assign each observation to at least one cluster, and *partial* methods, which may leave some observations outside the clusters. The latter methods are useful for identifying outliers.

Most clustering methods are heuristic in nature, in the sense that they generate a subdivision into clusters of good quality although not necessarily optimal. Actually, an exhaustive method based on a complete enumeration of the possible subdivisions of m observations into K clusters would require

$$\frac{1}{K!}\sum_{h=1}^{K}\binom{K}{h}h^m. \tag{12.1}$$

combinations to be examined. Therefore, it would be inapplicable even to small datasets, due to the exponential growth in computing time. In terms of computational complexity, clustering problems actually belong to the class of difficult (*NP*-hard) problems whenever $K \geq 3$.

12.1.2 Affinity measures

Clustering models are usually based on a measure of similarity between observations. In many instances this can be obtained by defining an appropriate notion of distance between each pair of observations. In this section we will review the most popular metrics, in connection with the type of attributes analyzed.

Given a dataset $\mathcal{D}$, we can represent the m observations by means of n-dimensional vectors of attributes, so as to formally represent the data using a matrix $\mathbf{X}$ with m rows and n columns, as described in Section 5.2. Let

$$\mathbf{D} = [d_{ik}] = \begin{bmatrix} 0 & d_{12} & \cdots & d_{1,m-1} & d_{1m} \\ & 0 & \cdots & d_{2,m-1} & d_{2m} \\ & & \cdots & \vdots & \vdots \\ & & & 0 & d_{m-1,m} \\ & & & & 0 \end{bmatrix} \tag{12.2}$$

be the symmetric $m \times m$ matrix of distances between pairs of observations, obtained by setting

$$d_{ik} = \text{dist}(\mathbf{x}_i, \mathbf{x}_k) = \text{dist}(\mathbf{x}_k, \mathbf{x}_i), \quad i, k \in \mathcal{M}, \tag{12.3}$$

where $\text{dist}(\mathbf{x}_i, \mathbf{x}_k)$ denotes the distance between observations $\mathbf{x}_i$ and $\mathbf{x}_k$.

Notice that it is possible to transform the distance d_{ik} between two observations into a similarity measure s_{ik}, by alternatively using

$$s_{ik} = \frac{1}{1 + d_{ik}}, \quad \text{or} \quad s_{ik} = \frac{d_{\max} - d_{ik}}{d_{\max}}, \tag{12.4}$$

in which $d_{\max} = \max_{i,k} d_{ik}$ denotes the maximum distance between the observations in the dataset $\mathcal{D}$.

As shown in the rest of this section, the definition of an appropriate notion of distance depends on the nature of the attributes that make up the dataset $\mathcal{D}$, which may be numerical, binary, nominal categorical, ordinal categorical or of mixed composition.

Numerical attributes

If all n attributes in a dataset are numerical, we may turn to the *Euclidean distance* between the vectors associated with the pair of observations

$\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{in})$ and $\mathbf{x}_k = (x_{k1}, x_{k2}, \ldots, x_{kn})$ in n-dimensional space, which is defined as

$$\begin{aligned} \text{dist}(\mathbf{x}_i, \mathbf{x}_k) &= \sqrt{\sum_{j=1}^{n} (x_{ij} - x_{kj})^2} \\ &= \sqrt{(x_{i1} - x_{k1})^2 + (x_{i2} - x_{k2})^2 + \cdots + (x_{in} - x_{kn})^2}. \end{aligned} \tag{12.5}$$

As an alternative we may consider the *Manhattan distance*

$$\begin{aligned} \text{dist}(\mathbf{x}_i, \mathbf{x}_k) &= \sum_{j=1}^{n} |x_{ij} - x_{kj}| \\ &= |x_{i1} - x_{k1}| + |x_{i2} - x_{k2}| + \cdots + |x_{in} - x_{kn}|, \end{aligned} \tag{12.6}$$

which is so called because in order to reach one point from another we have to travel along two sides of a rectangle having the two points as its opposite vertices. Figure 12.1 shows the difference between Euclidean and Manhattan metrics, using a two-dimensional example.

A third option, which generalizes both the Euclidean and Manhattan metrics, is the *Minkowski distance*, defined as

$$\begin{aligned} \text{dist}(\mathbf{x}_i, \mathbf{x}_k) &= \sqrt[q]{\sum_{j=1}^{n} |x_{ij} - x_{kj}|^q} \\ &= \sqrt[q]{|x_{i1} - x_{k1}|^q + |x_{i2} - x_{k2}|^q + \ldots, |x_{in} - x_{kn}|^q}, \end{aligned} \tag{12.7}$$

for some positive integer q. The Minkowski distance reduces to the Manhattan distance when $q = 1$, and to the Euclidean distance when $q = 2$.

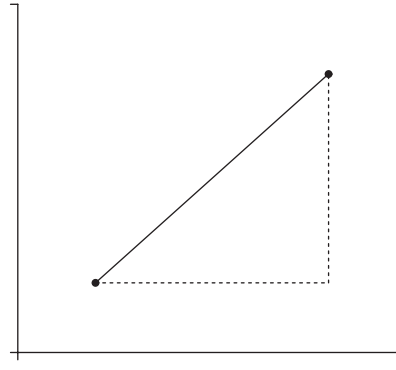

Figure 12.1 Euclidean distance (full line) and Manhattan distance (dotted line) between two points in the plane

A further generalization of the Euclidean distance can be obtained through the *Mahalanobis distance*, defined as

$$\text{dist}(\mathbf{x}_i, \mathbf{x}_k) = \sqrt{(\mathbf{x}_i - \mathbf{x}_k)\mathbf{V}_{ik}^{-1}(\mathbf{x}_i - \mathbf{x}_k)'}, \tag{12.8}$$

where $\mathbf{V}_{ik}^{-1}$ is the inverse of the covariance matrix of the pair of observations $\mathbf{x}_i$ and $\mathbf{x}_k$. If the observations $\mathbf{x}_i$ and $\mathbf{x}_k$ are independent, so that the covariance matrix reduces to the identity matrix, the Mahalanobis distance coincides with the Euclidean distance. The Mahalanobis distance is used when the observations are highly correlated, with different variances and a different range.

The expressions described above for computing the distance between pairs of observations are affected by the possible existence of attributes taking on absolute values significantly greater than the others. In extreme situations, a single prevailing attribute could affect the groupings generated by clustering algorithms.

A few tactics can be adopted in order to avoid such imbalance in the evaluation of distances between observations. On the one hand, one may standardize the values of the numerical attributes, using the techniques described in Chapter 6, so as to obtain new values on a uniform scale, for example in the interval [0, 1] or [−1, 1].

Alternatively, one may assign to each attribute $\mathbf{a}_j$ a weight w_j that is inversely proportional to the greatest value assumed by that same attribute, modifying the previous definitions to obtain weighted distances and thus balancing the importance of each attribute. For example, one can set

$$\text{dist}(\mathbf{x}_i, \mathbf{x}_k) = \sqrt[q]{\sum_{j=1}^{n} w_j |x_{ij} - x_{kj}|^q}. \tag{12.9}$$

A further possibility is to use the *arccosine distance*,[1] which depends on the angle formed by the vectors associated with the observations. Besides being independent of the absolute value of the components of the observations, this measure can be applied not only to numerical attributes but also to categorical and binary attributes. For this metric, the similarity coefficient based on the cosine of the angle between the two vectors $\mathbf{x}_i$ and $\mathbf{x}_k$ should be defined using the expression

$$B_{\cos}(\mathbf{x}_i, \mathbf{x}_k) = \cos(\mathbf{x}_i, \mathbf{x}_k) = \frac{\sum_{j=1}^{n} x_{ij} x_{kj}}{\sqrt{\sum_{j=1}^{n} x_{ij}^2 \sum_{j=1}^{n} x_{kj}^2}}. \tag{12.10}$$

The coefficient $B_{\cos}(\mathbf{x}_i, \mathbf{x}_k)$ lies in the interval [0, 1], and it is closer to 1 when the two vectors $\mathbf{x}_i$ and $\mathbf{x}_k$ are similar to each other (parallel), and closer to 0

[1] The arccosine $\arccos(\alpha)$ is the inverse function of the cosine and gives the angle corresponding to the value α of the cosine.

when they are dissimilar (orthogonal). Starting from the coefficient $B_{\cos}(\mathbf{x}_i, \mathbf{x}_k)$, a notion of distance between the two observations can be derived in different ways, for example using the arccosine or the sine of the angle, as in

$$\begin{aligned} \text{dist}(\mathbf{x}_i, \mathbf{x}_k) &= \arccos(\mathbf{x}_i, \mathbf{x}_k), \\ \text{dist}(\mathbf{x}_i, \mathbf{x}_k) &= \sin(\mathbf{x}_i, \mathbf{x}_k) = \sqrt{1 - \cos^2(\mathbf{x}_i, \mathbf{x}_k)}. \end{aligned} \tag{12.11}$$

Binary attributes

Suppose that a given attribute $\mathbf{a}_j = (x_{1j}, x_{2j}, \ldots, x_{mj})$ is binary, so that it assumes only one of the two values 0 or 1. Even if it is possible to formally calculate the difference $x_{ij} - x_{kj}$ for every two observations of the dataset, it is clear that this quantity does not represent a distance that can be meaningfully associated with the metrics defined for numerical attributes, since the values 0 and 1 are purely conventional, and their meanings could be interchanged. The need thus arises to define an appropriate notion of proximity that can be applied to binary attributes.

Assume for the moment that all n attributes in the dataset $\mathcal{D}$ are binary. In order to define a metric, reference should be made to the contingency table associated with the n attributes at two distinct observations $\mathbf{x}_i$ and $\mathbf{x}_k$, as indicated in Table 12.1. The value p is the number of binary attributes for which both observations $\mathbf{x}_i$ and $\mathbf{x}_k$ assume the value 0. Similarly, q is the number of attributes for which observation $\mathbf{x}_i$ takes the value 0 and observation $\mathbf{x}_k$ takes the value 1. The value u is the number of attributes for which $\mathbf{x}_i$ assumes the value 1 and $\mathbf{x}_k$ assumes the value 0. Finally, there are v attributes for which the value 1 is assumed by both observations.

As remarked in Chapter 11, binary attributes can be *symmetric* or *asymmetric*. In the first case the presence of the value 0 is as interesting as the presence of the value 1. For instance, customer authorization of promotional mailings assumes the values {no, yes}. In the second case we are mostly interested in the presence of the value 1, which can usually be found in a small proportion of the observations. In order to discriminate between those individuals suffering from a medical condition and those who are not affected, it is clearly more

Table 12.1 Contingency table for a binary attribute

		observation $\mathbf{x}_k$		
		0	1	total
observation $\mathbf{x}_i$	0	p	q	$p+q$
	1	u	v	$u+v$
	total	$p+u$	$q+v$	n

significant to analyze the presence of the value 1, assuming that this codes the presence of the condition.

If all the n binary attributes are symmetric, we can define the degree of similarity between pairs of observations through the *coefficient of similarity*, given by

$$\text{dist}(\mathbf{x}_i, \mathbf{x}_k) = \frac{q+u}{p+q+u+v} = \frac{q+u}{n}. \tag{12.12}$$

Assume instead that all n attributes are binary and asymmetric. As stated before, for a pair of asymmetric attributes it is much more interesting to match *positives*, i.e. records possessing the property relative to each attribute and therefore being coded by the value 1, with respect to matching *negatives*, representing observations coded by the value 0. For binary variables, the *Jaccard coefficient* is therefore used, defined as

$$\text{dist}(\mathbf{x}_i, \mathbf{x}_k) = \frac{q+u}{q+u+v}. \tag{12.13}$$

Finally, if the dataset includes both symmetric and asymmetric binary attributes, as well as for simultaneous numerical and binary attributes, the methodologies described in the section on mixed composition attributes should be applied.

Nominal categorical attributes

A nominal categorical attribute, defined in Chapter 5, can be interpreted as a generalization of a symmetric binary attribute, for which the number of distinct values is greater than two. As a consequence, for nominal attributes we can also use an extension of the similarity coefficient, previously defined for symmetric binary variables, using the new expression

$$\text{dist}(\mathbf{x}_i, \mathbf{x}_k) = \frac{n-f}{n}, \tag{12.14}$$

in which f is the number of attributes for which the observations $\mathbf{x}_i$ and $\mathbf{x}_k$ take the same nominal value.

Alternatively, as shown in Section 8.3.3, it is possible to represent each nominal categorical variable by means of a number of asymmetric binary attributes equal to the number of distinct values assumed by the nominal variable. Once the nominal variables in the dataset have been replaced by the categorical attributes, we can therefore use the Jaccard coefficient (12.13) to express the degree of affinity.

Ordinal categorical attributes

Ordinal categorical attributes can be placed on a natural ordering scale, whose numerical values are, however, arbitrary. Therefore, they require a preliminary standardization so that they can be adapted to the metrics defined for numerical attributes.

To clarify this concept, assume that the values of each ordinal categorical attribute are represented through the corresponding position in the natural order. If, for example, the ordinal variable corresponds to the level of education, and can be expressed by the values {elementary school, high school graduate, bachelor's degree, postgraduate}, the elementary level is matched with numerical value 1, the high school graduate level with 2, and so on. Let $\mathcal{H}_j = \{1, 2, \ldots H_j\}$ be the ordered values associated with the ordinal attribute $\mathbf{a}_j$.

To standardize the values assumed by the attribute $\mathbf{a}_j$ to the interval [0,1], the following transformation is used

$$x'_{ij} = \frac{x_{ij} - 1}{H_j - 1}. \tag{12.15}$$

After we have carried out the transformation indicated for all the ordinal variables, we can use the measures of distance previously introduced for numerical attributes.

Mixed composition attributes

Suppose that the n attributes in the dataset $\mathcal{D}$ have a mixed composition, in the sense that some of them are numerical, while others are binary symmetric or binary asymmetric or nominal categorical or ordinal categorical. We wish to define an overall affinity measure that can be used to evaluate the degree of similarity between two observations $\mathbf{x}_i$ and $\mathbf{x}_k$ even if mixed composition attributes are present.

Let us define a binary indicator δ_{ikj} that takes the value 0 if and only if one of the following cases occurs:

- at least one of the two values x_{ij} or x_{kj} is missing in the corresponding records of the dataset;
- the attribute $\mathbf{a}_j$ is binary asymmetric and, moreover, $x_{ij} = x_{kj} = 0$.

For all the remaining cases the indicator δ_{ikj} takes the value 1. Also define the contribution Δ_{ikj} of the variable $\mathbf{a}_j$ to the similarity between $\mathbf{x}_i$ and $\mathbf{x}_k$, based on the nature of the attribute $\mathbf{a}_j$, as follows:

- if the attribute $\mathbf{a}_j$ is binary or nominal, we set $\Delta_{ikj} = 0$ if $x_{ij} = x_{kj}$ and $\Delta_{ikj} = 1$ otherwise;
- if the attribute $\mathbf{a}_j$ is numerical, we set

$$\Delta_{ikj} = \frac{|x_{ij} - x_{kj}|}{\max_l x_{lj}}; \tag{12.16}$$

- if the attribute $\mathbf{a}_j$ is ordinal, its standardized value is computed as previously described and we set Δ_{ikj} as for numerical attributes.

Therefore, we can define the similarity coefficient between the observations $\mathbf{x}_i$ and $\mathbf{x}_k$ as

$$\text{dist}(\mathbf{x}_i, \mathbf{x}_k) = \frac{\sum_{j=1}^{n} \delta_{ikj} \Delta_{ikj}}{\sum_{j=1}^{n} \delta_{ikj}}. \tag{12.17}$$

12.2 Partition methods

Given a dataset $\mathcal{D}$ of m observations, each represented by a vector in n-dimensional space, partition methods construct a subdivision of $\mathcal{D}$ into a collection of non-empty subsets $\mathcal{C} = \{C_1, C_2, \ldots, C_K\}$, where $K \leq m$. In general, the number K of clusters is predetermined and assigned as an input to partition algorithms. The clusters generated by partition methods are usually exhaustive and mutually exclusive, in the sense that each observation belongs to one and only one cluster. There are, however, *fuzzy* partition methods that assign each observation to different clusters according to a specific proportion.

Partition methods start with an initial assignment of the m available observations to the K clusters. Then, they iteratively apply a reallocation technique whose purpose is to place some observations in a different cluster, so that the overall quality of the subdivision is improved. Although alternative measures of the clustering quality can be used, all the various criteria tend to express the degree of homogeneity of the observations belonging to the same cluster and their heterogeneity with respect to the records included in other clusters. Partition algorithms usually stop when during the same iteration no reallocation occurs, and therefore the subdivision appears stable with respect to the evaluation criterion chosen.

Partition methods are therefore of a heuristic nature, in the sense that they are based on a myopic logic typical of the class of so-called greedy methods, and at each step they make the choice that locally appears the most advantageous. Operating in this way, there is no guarantee that a globally optimal clustering will be reached, but only that a good subdivision will be obtained, at least for the majority of the datasets.

The *K-means* method and the *K-medoids* method, which will be described next, are two of the best-known partition algorithms. They are rather efficient clustering methods that are effective in determining clusters of spherical shape.

12.2.1 *K*-means algorithm

The *K-means* algorithm receives as input a dataset $\mathcal{D}$, a number K of clusters to be generated and a function $\text{dist}(\mathbf{x}_i, \mathbf{x}_k)$ that expresses the inhomogeneity

between each pair of observations, or equivalently the matrix **D** of distances between observations.

Given a cluster C_h, $h = 1, 2, \ldots K$, the *centroid* of the cluster is defined as the point $\mathbf{z}_h$ having coordinates equal to the mean value of each attribute for the observations belonging to that cluster, that is,

$$z_{hj} = \frac{\sum_{\mathbf{x}_i \in C_h} x_{ij}}{\text{card}\{C_h\}}. \tag{12.18}$$

Procedure 12.1 – K-means algorithm

1. During the initialization phase, K observations are arbitrarily chosen in $\mathcal{D}$ as the centroids of the clusters.
2. Each observation is iteratively assigned to the cluster whose centroid is the most similar to the observation, in the sense that it minimizes the distance from the record.
3. If no observation is assigned to a different cluster with respect to the previous iteration, the algorithm stops.
4. For each cluster, the new centroid is computed as the mean of the values of the observations belonging to the cluster, and then the algorithm returns to step 2.

The algorithm, described in Procedure 12.1, starts by arbitrarily selecting K observations that constitute the initial centroids. For example, the K points can be randomly selected among the m observations in $\mathcal{D}$. At each subsequent iteration, each record is assigned to the cluster whose centroid is the closest, that is, which minimizes the distance from the observation among all centroids. If no observation is reallocated to a cluster different from the one to which it belongs, determined during the previous iteration, the procedure stops, since any subsequent iteration cannot alter the current subdivision in clusters. Otherwise the new centroid for each cluster is computed and a new assignment made.

Figure 12.2 illustrates the application of the K-means algorithm to a set of observations in the Euclidean plane. Suppose that we have decided on $K = 3$ as the desired number of clusters. Figure 12.2(a) shows the points corresponding to the two-dimensional observations that are to be subdivided into three clusters. During the initialization phase, three observations are arbitrarily selected, which are indicated in the figure as large dots and constitute the three initial centroids. At the first iteration, each observation is assigned to the closest centroid, generating the clusters highlighted in Figure 12.2(b), where the

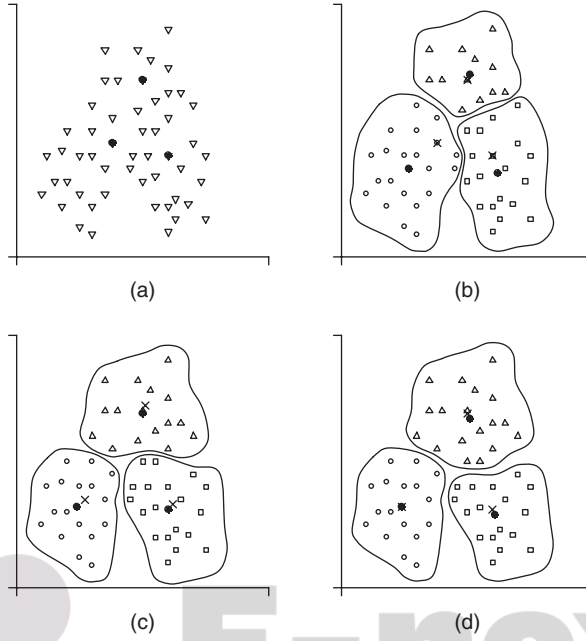

Figure 12.2 An example of application of the K-means algorithm

observations belonging to distinct clusters are indicated by three different graphical symbols. Figure 12.2(b) also shows the new centroids, computed as the mean of the attributes for the observations belonging to each cluster and represented as large dots. The three centroids of the previous iteration are indicated in diagram (b) by the symbol $\times$. Figures 12.2(c) and (d) respectively show the second and the third iterations of the algorithm. Since at the end of the third iteration no observation is assigned to a different cluster, the algorithm stops.

As stated earlier, if the attributes are numerical or ordinal categorical it is possible to transform them into a standardized representation in n-dimensional Euclidean space. In this case, one may also express the function of the overall heterogeneity between each observation and the point $\mathbf{w}_h$ representing the cluster C_h to which it is assigned, by means of the minimization of the squared error

$$\begin{aligned} \min_{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_K} \mathrm{EQ} &= \sum_{h=1}^{K} \sum_{\mathbf{x}_i \in C_h} (\mathrm{dist}(\mathbf{x}_i, \mathbf{w}_h))^2 \\ &= \sum_{h=1}^{K} \sum_{\mathbf{x}_i \in C_h} \sum_{j=1}^{n} (x_{ij} - w_{hj})^2, \end{aligned} \tag{12.19}$$

where the function dist$(\cdot, \cdot)$ represents the Euclidean distance between two points.

The function EQ is minimized by setting $\mathbf{w}_h = \mathbf{z}_h$, $h = 1, 2, \ldots, K$. This means that the centroids, calculated as the mean of the observations belonging to each cluster, minimize the sum of squared differences of the observations from the K points that represent the clusters, as these points vary.

If the dataset also includes binary or nominal categorical attributes, it is possible to use other notions of distance, as seen in Section 12.1.2. However, the problem remains of how to replace the mean of the observations, which is not defined for nominal variables, with a corresponding indicator that allows the centroid of a cluster to be identified. The algorithm that uses the mode of each attribute, calculated for the observations belonging to each cluster, in place of its mean, is called the *K-modes* method.

Several variants and extensions of the K-means algorithm have been proposed. Since the initial assignment tends to strongly influence the final subdivision into clusters, it is useful to generate different initial random assignments and then derive for each of them a different clustering, finally choosing the best one out of those that have been generated. Outliers can also affect the final result, since they take large numerical values in the quadratic objective function EQ. Therefore the K-means method is applied preferably only after the outliers have been identified and removed. Finally, it is possible to apply a posteriori a few tactics to improve the quality of the subdivision generated by the algorithm, for example by separating one or more clusters and increasing in this way the total number K of clusters identified. On the other hand, it may be appropriate to merge two or more clusters into a single one, reducing the number of clusters.

12.2.2 K-medoids algorithm

The *K-medoids* algorithm, also known as *partitioning around medoids*, is a variant of the K-means method. It is based on the use of *medoids* instead of the means of the observations belonging to each cluster, with the purpose of mitigating the sensitivity of the partitions generated with respect to the extreme values in the dataset.

Given a cluster C_h, a *medoid* $\mathbf{u}_h$ is the most *central* observation, in a sense that will be formally defined below, among those that are assigned to C_h. Once the medoid representing each cluster has been identified, the K-medoids algorithm proceeds like the K-means method, using medoids instead of centroids.

The initialization phase involves the arbitrary selection of K observations representing the medoids at the first iteration. Each of the remaining observations is assigned to the medoid to which it is most similar, so as to minimize the distance measure dist$(\mathbf{x}_i, \mathbf{u}_h)$. During each subsequent iteration, the algorithm

tries to substitute each medoid with an observation that does not represent a medoid, provided that the overall quality of the subdivision in clusters is improved. The quality of the subdivision is evaluated through a measure of average inhomogeneity between each observation and the medoid associated with the cluster to which it belongs.

Let U be the set of K medoids at a given iteration. To assess the advantage afforded by an exchange of medoids, the algorithm considers all the $(m - K)K$ pairs $(\mathbf{x}_i, \mathbf{u}_h)$ composed of an observation $\mathbf{x}_i \notin U$ and a medoid $\mathbf{u}_h \in U$. For each pair, every observation $\mathbf{x}_k \notin U$ different from $\mathbf{x}_i$ is considered, and the contribution R_{ihk} made by $\mathbf{x}_k$ to the exchange between $\mathbf{x}_i$ and $\mathbf{u}_h$ is evaluated. The following four cases are then considered:

- $\mathbf{x}_k$ is currently assigned to the cluster corresponding to medoid $\mathbf{u}_h$, and also $\text{dist}(\mathbf{x}_i, \mathbf{x}_k) \leq \min_{\mathbf{u}_e \in U, e \neq h} \text{dist}(\mathbf{u}_e, \mathbf{x}_k)$. In this first case the observation $\mathbf{x}_k$ is assigned to the new medoid $\mathbf{x}_i$ intended to represent cluster C_h, and the contribution to the exchange, which can be positive, zero or negative, is given by

$$R_{ihk} = \text{dist}(\mathbf{x}_i, \mathbf{x}_k) - \text{dist}(\mathbf{u}_h, \mathbf{x}_k). \tag{12.20}$$

- $\mathbf{x}_k$ is currently assigned to the cluster corresponding to medoid $\mathbf{u}_h$, and also $\text{dist}(\mathbf{x}_i, \mathbf{x}_k) \leq \min_{\mathbf{u}_e \in U, e \neq h} \text{dist}(\mathbf{u}_e, \mathbf{x}_k)$. In this second case the observation $\mathbf{x}_k$ is assigned to a different cluster, and the contribution to the exchange is given by

$$R_{ihk} = \min_{\mathbf{u}_e \in U, e \neq h} \text{dist}(\mathbf{u}_e, \mathbf{x}_k) - \text{dist}(\mathbf{u}_h, \mathbf{x}_k). \tag{12.21}$$

- $\mathbf{x}_k$ is not currently assigned to the cluster corresponding to medoid $\mathbf{u}_h$, and also $\text{dist}(\mathbf{x}_i, \mathbf{x}_k) \geq \min_{\mathbf{u}_e \in U, e \neq h} \text{dist}(\mathbf{u}_e, \mathbf{x}_k)$. In this third case the observation $\mathbf{x}_k$ remains assigned to a different cluster, and the contribution to the exchange is given by

$$R_{ihk} = 0.$$

- $\mathbf{x}_k$ is not currently assigned to the cluster corresponding to medoid $\mathbf{u}_h$, and also $\text{dist}(\mathbf{x}_i, \mathbf{x}_k) < \min_{\mathbf{u}_e \in U, e \neq h} \text{dist}(\mathbf{u}_e, \mathbf{x}_k)$. In this last case the observation $\mathbf{x}_k$ is assigned to cluster C_h, and the contribution to the exchange is given by

$$R_{ihk} = \text{dist}(\mathbf{x}_i, \mathbf{x}_k) - \min_{\mathbf{u}_e \in U, e \neq h} \text{dist}(\mathbf{u}_e, \mathbf{x}_k). \tag{12.22}$$

Once the contribution R_{ihk} made by each observation $\mathbf{x}_k \notin U$ to the exchange between $\mathbf{x}_i$ and $\mathbf{u}_h$ has been calculated, the overall contribution can be evaluated as

$$T_{ih} = \sum_{\mathbf{x}_k \notin U} R_{ihk}. \tag{12.23}$$

If the minimum value among the overall contributions T_{ih} is negative, the pair $(\mathbf{x}_i, \mathbf{u}_h)$ that corresponds to such minimum value is actually exchanged and the algorithm proceeds with a new iteration, stopping when no exchange occurs, that is, when the minimum value of the overall contribution is non-negative.

The K-medoids algorithm requires a large number of iterations and is not suited to deriving clusters for large datasets. A few variants have therefore been proposed, based on the extraction of samples of observations to which the K-medoids method can be applied.

12.3 Hierarchical methods

Hierarchical clustering methods are based on a tree structure. Unlike partition methods, they do not require the number of clusters to be determined in advance. Hence, they receive as input a dataset $\mathcal{D}$ containing m observations and a matrix of distances $\text{dist}(\mathbf{x}_i, \mathbf{x}_k)$ between all pairs of observations.

In order to evaluate the distance between two clusters, most hierarchical algorithms resort to one of five alternative measures: minimum distance, maximum distance, mean distance, distance between centroids, and Ward distance.

Suppose that we wish to calculate the distance between two clusters C_h and C_f and let $\mathbf{z}_h$ and $\mathbf{z}_f$ be the corresponding centroids.

Minimum distance. According to the criterion of *minimum distance*, also called the *single linkage* criterion, the dissimilarity between two clusters is given by the minimum distance among all pairs of observations such that one belongs to the first cluster and the other to the second cluster, that is,

$$\text{dist}(C_h, C_f) = \min_{\substack{\mathbf{x}_i \in C_h \\ \mathbf{x}_k \in C_f}} \text{dist}(\mathbf{x}_i, \mathbf{x}_k). \tag{12.24}$$

Maximum distance. According to the criterion of *maximum distance*, also called the *complete linkage* criterion, the dissimilarity between two clusters is given by the maximum distance among all pairs of observations such that one belongs to the first cluster and the other to the second cluster, that is,

$$\text{dist}(C_h, C_f) = \max_{\substack{\mathbf{x}_i \in C_h \\ \mathbf{x}_k \in C_f}} \text{dist}(\mathbf{x}_i, \mathbf{x}_k). \tag{12.25}$$

Mean distance. The *mean distance* criterion expresses the dissimilarity between two clusters via the mean of the distances between all pairs of observations belonging to the two clusters, that is,

$$\text{dist}(C_h, C_f) = \frac{\sum_{\mathbf{x}_i \in C_h} \sum_{\mathbf{x}_k \in C_f} \text{dist}(\mathbf{x}_i, \mathbf{x}_k)}{\text{card}\{C_h\}\,\text{card}\{C_f\}}. \tag{12.26}$$

Distance between centroids. The criterion based on the *distance between centroids* determines the dissimilarity between two clusters through the distance between the centroids representing the two clusters, that is,

$$\text{dist}(C_h, C_f) = \text{dist}(\mathbf{z}_h, \mathbf{z}_f). \tag{12.27}$$

Ward distance. The criterion of *Ward distance*, based on the analysis of the variance of the Euclidean distances between the observations, is slightly more complex than the criteria described above. Indeed, it requires the algorithm to first calculate the sum of squared distances between all pairs of observations belonging to a cluster. Afterwards, all pairs of clusters that could be merged at the current iteration are considered, and for each pair the total variance is computed as the sum of the two variances between the distances in each cluster, evaluated in the first step. Finally, the pair of clusters associated with the minimum total variance are merged. Methods based on the Ward distance tend to generate a large number of clusters, each containing a few observations.

Hierarchical methods can be subdivided into two main groups: *agglomerative* and *divisive* methods.

12.3.1 Agglomerative hierarchical methods

Agglomerative methods are *bottom-up* techniques in which each single observation initially represents a distinct cluster. These clusters are then aggregated during subsequent iterations, deriving clusters of increasingly larger cardinalities. The algorithm is stopped when a single cluster including all the observations has been reached. It is then necessary for the user to decide on a cut point and thus determine the number of clusters.

Procedure 12.2 describes the general structure of an agglomerative method. At the end of the algorithm it is possible to graphically represent the process of subsequent mergers using a *dendrogram*, indicating on one axis the value of the minimum distance corresponding to each merger and the observations on the other axis.[2]

[2]There is no standard convention relative to the placement of the axes in dendrograms, since sometimes the observations are positioned on the horizontal axis and sometimes on the vertical axis.

Procedure 12.2 – Agglomerative algorithm

1. In the initialization phase, each observation constitutes a cluster. The distance between clusters therefore corresponds to the matrix **D** of the distances between all pairs of observations.
2. The minimum distance between the clusters is then computed, and the two clusters C_h and C_f with the minimum distance are merged, thus deriving a new cluster C_e. The corresponding minimum distance dist(C_h, C_f) originating the merger is recorded.
3. The distance between the new cluster C_e, resulting from the merger between C_h and C_f, and the preexisting clusters is computed.
4. If all the observations are included into a single cluster, the procedure stops. Otherwise it is repeated from step 2.

Figure 12.3 shows the dendrograms for the clustering obtained by applying an agglomerative hierarchical algorithm to the *mtcars* dataset, described in Appendix B, using (a) the mean Euclidean distance and (b) the mean Manhattan distance. As we can see, the observations are placed on the horizontal axis and the distance value on the vertical axis. Actually, each dendrogram provides

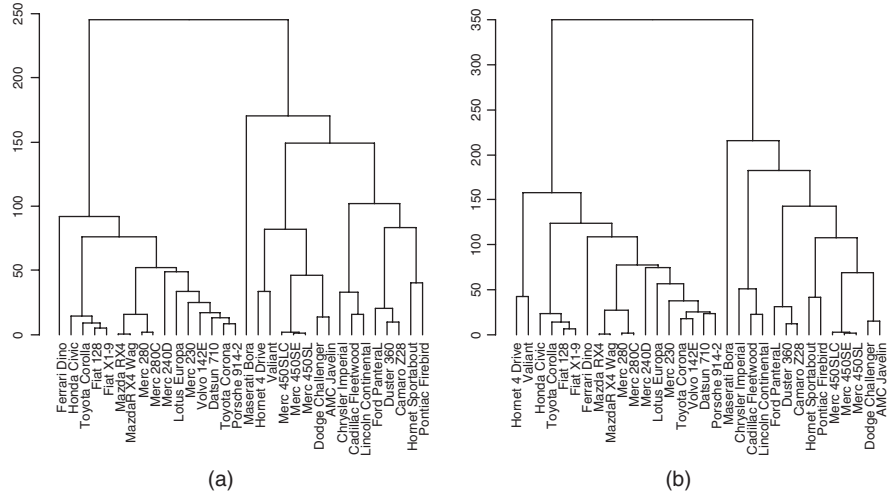

Figure 12.3 Dendrograms for an agglomerative hierarchical algorithm applied to the mtcars dataset with (a) the mean Euclidean distance and (b) the mean Manhattan distance

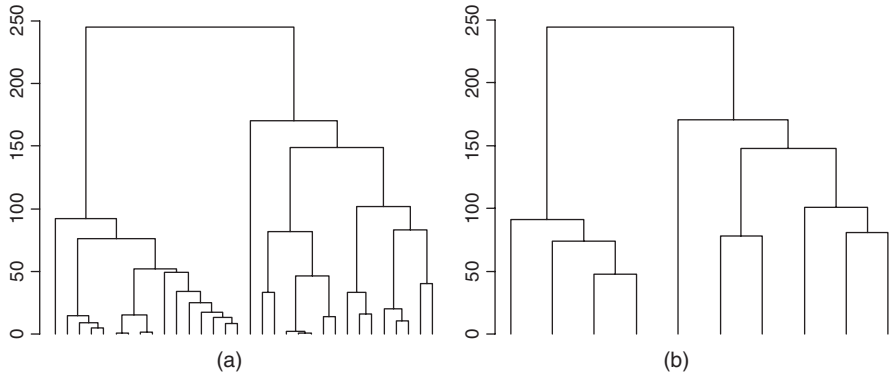

Figure 12.4 Dendrograms for an agglomerative hierarchical algorithm applied to the mtcars dataset with the mean Euclidean distance and two different levels of clusters

a whole hierarchy of clusters, corresponding to different threshold values for the minimum distance between clusters, indicated on the vertical axis. In the dendrogram shown in Figure 12.3(a), four clusters can be obtained if the tree is truncated at the distance 125, while nine clusters are generated by cutting the tree at 70. It can be observed that the dendrograms and the corresponding hierarchies of clusters obtained using the Euclidean and the Manhattan distances differ significantly.

Figure 12.4, in which the labels of each single observation have been removed, illustrates the effect of tree pruning. Figure 12.4(a) shows the same tree as in Figure 12.3(a), while Figure 12.4(b) has been obtained by means of a cut that results in ten clusters. In this way, a more aggregated view of the phenomenon can be achieved that leads to an easier interpretation.

Figures 12.5 and 12.6 show the effect of changes in the metrics used by the agglomerative hierarchical algorithm. Figure 12.5(a) shows the dendrogram obtained with the minimum distance and Figure 12.5(b) the dendrogram generated with the maximum distance, respectively. Figure 12.6(a) shows the dendrogram obtained with the Ward distance and Figure 12.6(b) the dendrogram generated with the distance between the centroids, respectively. Confirming previous remarks, the Ward method produces a tree with several ramifications toward the leaves and therefore a structure made up of many small clusters. More generally, these figures suggest that the type of metric selected has a relevant influence on the clustering generated.

12.3.2 Divisive hierarchical methods

Divisive algorithms are the opposite of agglomerative methods, in that they are based on a *top-down* technique, which initially places all the observations in

a single cluster. This is then subdivided into clusters of smaller size, so that the distances between the generated subgroups are minimized. The procedure is repeated until clusters containing a single observation are obtained, or until an analogous stopping condition is met.

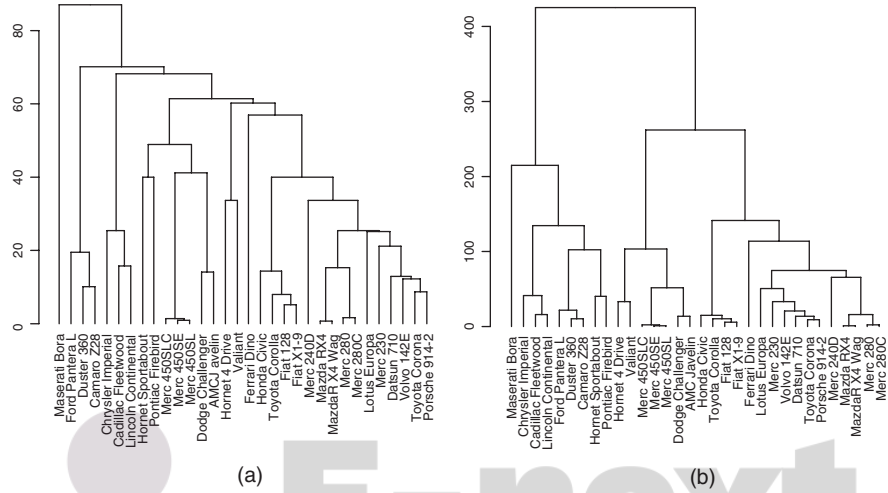

Figure 12.5 Dendrograms for an agglomerative hierarchical algorithm applied to the mtcars dataset with (a) the mean Euclidean distance and (b) the maximum Euclidean distance

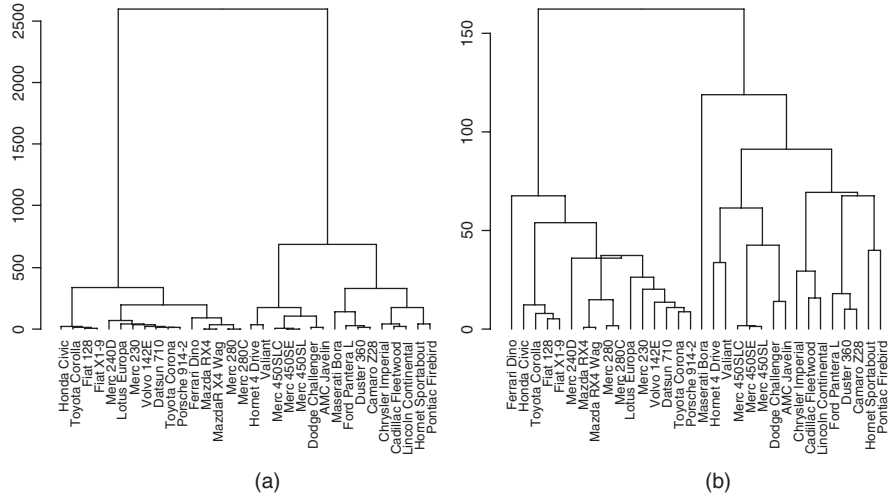

Figure 12.6 Dendrograms for an agglomerative hierarchical algorithm applied to the mtcars dataset with (a) the Ward distance and (b) the distance of centroids

Unlike agglomerative methods, in order to keep computing times within reasonable limits, divisive algorithms require a strict limitation on the number of combinations that can be analyzed. In order to understand the motivations for this choice, consider the first step of the algorithm. The only cluster containing all the observations must be subdivided into two subsets, so that the distance between the two resulting clusters is maximized. Since there are 2^{m-2} possible partitions of the whole set into two non-empty disjoint subsets, this results in an exponential number of operations already at the first iteration.

To circumvent this difficulty, at any given iteration divisive hierarchical algorithms usually determine for each cluster the two observations that are furthest from each other, and subdivide the cluster by assigning the remaining records to the one or the other, based on their proximity.

Figure 12.7(a) shows the dendrogram obtained by applying a divisive hierarchical algorithm to the *mtcars* dataset, using the mean Euclidean distance. This can be compared with the corresponding dendrogram shown in Figure 12.3(a), obtained using the same metrics but with an agglomerative hierarchical algorithm. As we can see, for this dataset the type of algorithm seems to exert less influence than the metric adopted.

12.4 Evaluation of clustering models

We have seen in previous chapters that for supervised learning methods, such as classification, regression and time series analysis, the evaluation of the

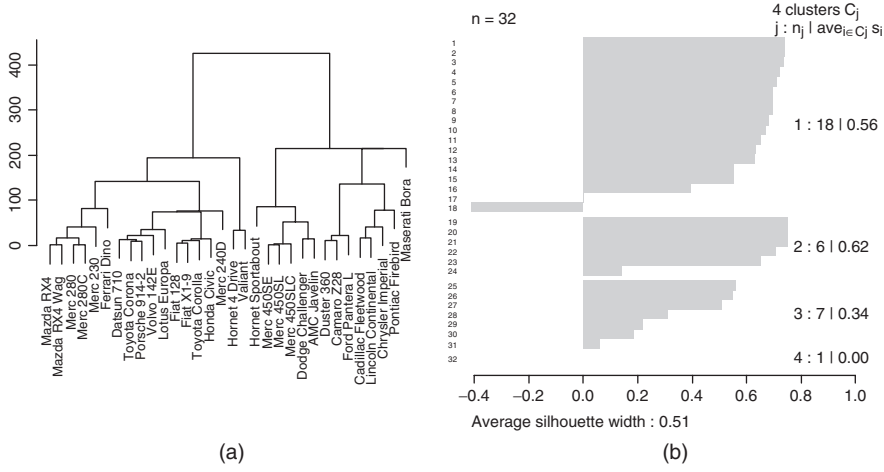

Figure 12.7 Dendrograms for a divisive hierarchical algorithm applied to the mtcars dataset (a) with the mean Euclidean distance and (b) the corresponding silhouette with four clusters

predictive accuracy is part of the development process of a model and is based on specific numerical indicators. The same does not apply to clustering methods and, more generally, to unsupervised learning models. Even though the absence of a target attribute makes the evaluation of an unsupervised model less direct and intuitive, it is possible to define reasonable measures of quality and significance for clustering methods.

To evaluate a clustering method it is first necessary to verify that the clusters generated correspond to an actual regular pattern in the data. It is therefore appropriate to apply other clustering algorithms and to compare the results obtained by different methods. In this way it is also possible to evaluate if the number of identified clusters is robust with respect to the different techniques applied.

At a subsequent phase it is recommended to calculate some performance indicators. Let $\mathcal{C} = \{C_1, C_2, \ldots, C_K\}$ be the set of K clusters generated. An indicator of homogeneity of the observations within each cluster C_h is given by the *cohesion*, defined as

$$\text{coh}(C_h) = \sum_{\substack{\mathbf{x}_i \in C_h \\ \mathbf{x}_k \in C_h}} \text{dist}(\mathbf{x}_i, \mathbf{x}_k). \tag{12.28}$$

The overall cohesion of the partition $\mathcal{C}$ can therefore be defined as

$$\text{coh}(\mathcal{C}) = \sum_{C_h \in \mathcal{C}} \text{coh}(C_h). \tag{12.29}$$

One clustering is preferable over another, in terms of homogeneity within each cluster, if it has a smaller overall cohesion.

An indicator of inhomogeneity between a pair of clusters is given by the *separation*, defined as

$$\text{sep}(C_h, C_f) = \sum_{\substack{\mathbf{x}_i \in C_h \\ \mathbf{x}_k \in C_f}} \text{dist}(\mathbf{x}_i, \mathbf{x}_k). \tag{12.30}$$

Again the overall separation of the partition $\mathcal{C}$ can be defined as

$$\text{sep}(\mathcal{C}) = \sum_{\substack{C_h \in \mathcal{C} \\ C_f \in \mathcal{C}}} \text{sep}(C_h, C_f). \tag{12.31}$$

One clustering is preferable over another, in terms of inhomogeneity among all clusters, if it has a greater overall separation.

A further indicator of the clustering quality is given by the *silhouette coefficient*, which involves a combination of cohesion and separation. To calculate the silhouette coefficient for a single observation $\mathbf{x}_i$, three steps should be followed, as detailed in Procedure 12.3

Procedure 12.3 – Calculation of the silhouette coefficient

1. The mean distance u_i of $\mathbf{x}_i$ from all the remaining observations belonging to the same cluster is computed.
2. For each cluster C_f other than the cluster to which $\mathbf{x}_i$ belongs, the mean distance w_{if} between $\mathbf{x}_i$ and all the observations in C_f is calculated. The minimum v_i among the distances w_{if} is determined by varying the cluster C_f.
3. The silhouette coefficient of $\mathbf{x}_i$ is defined as

$$\text{silh}(\mathbf{x}_i) = \frac{v_i - u_i}{\max(u_i, v_i)}.$$

The silhouette coefficient varies between -1 and 1. A negative value indicates that the mean distance u_i of the observation $\mathbf{x}_i$ from the points of its cluster is greater than the minimum value v_i of the mean distances from the observations of the other clusters, and it is therefore undesirable since the membership of $\mathbf{x}_i$ in its cluster is not well characterized. Ideally the silhouette coefficient should be positive and u_i should be as close as possible to 0. Finally, it should be noticed that the overall silhouette coefficient of a clustering may be computed as the mean of the silhouette coefficients for all the observations in the dataset $\mathcal{D}$.

Silhouette coefficients can be illustrated by *silhouette diagrams*, in which the observations are placed on the vertical axis, subdivided by clusters, and the values of the silhouette coefficient for each observation are shown on the horizontal axis. Usually the mean value of the silhouette coefficient for each cluster is also given in a silhouette diagram, as well as the overall mean for the whole dataset.

Figures 12.7 and 12.8 show the silhouette diagrams corresponding to different clusterings. In particular, Figure 12.7(b) shows the silhouette diagram corresponding to a cut of four clusters in the dendrogram shown in Figure 12.7(a), obtained by applying a divisive hierarchical algorithm to the *mtcars* dataset using the mean Euclidean distance. Figure 12.8(a) shows the silhouette diagram corresponding to a cut of four clusters in the dendrogram shown in Figure 12.3(a), obtained by applying an agglomerative hierarchical algorithm to the *mtcars* dataset using the mean Euclidean distance. Finally, Figure 12.8(b) shows the silhouette diagram corresponding to the clustering with $K = 4$ obtained by applying a medoids partitioning algorithm.

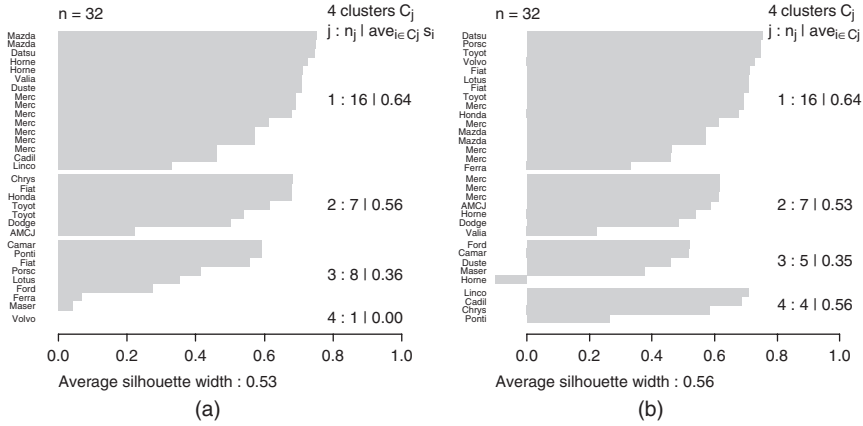

Figure 12.8 Silhouette diagrams with four clusters for an agglomerative hierarchical algorithm, applied to the mtcars dataset (a) with the mean Euclidean distance, and (b) for a medoids partitioning algorithm

12.5 Notes and readings

Among the various texts devoted to clustering, we wish to mention Anderberg (1973), Hartigan (1975), Romesburg (1984), Murtagh (1985), Aldenderfer and Blashfield (1985), Kaufman and Rousseeuw (1990) and Everitt *et al*. (2001). Surveys are also presented in Jain *et al*. (1999) and Mirkin (1996). More specifically, for hierarchical methods the reader is also referred to Fisher (1996) and Jain and Dubes (1988). The evaluation of clustering models is described in Halkidi *et al*. (2002a, b). The interconnections between clustering and classification are discussed in Fraley and Raftery (2002) and Spaeth (1980).