Chapter 10
Performance and Evaluation Measures

10.1 Introduction

In cluster analysis it is important to apply special evaluation and performance measures to assess the quality of clustering solutions and to compare the performance of different clustering algorithms. Here, we differentiate evaluation and performance measures. Evaluation measures are predominantly used to judge the quality of clustering solutions whereas performance measures are applied to compare the efficiency of the algorithms using the computational time and/or the number of the objective and constraint functions evaluations.

A good clustering algorithm is able to find a true number of clusters and to compute well-separated clusters which are compact and connected. Nevertheless, quantifying and measuring these objectives are not a trivial task. In some applications when a data set contains a small number of instances and a very few attributes, it might be possible to intuitively evaluate clustering results. However, such an evaluation is not possible in large and medium sized data sets or even in small data sets with several attributes.

The objectives of clustering—separability, connectivity, and compactness—are defined as follows:

- *separability* of clusters means that they are pairwise separable: that is, for each pair of clusters there exists a hyperplane separating them in the n-dimensional space. Roughly speaking in this case, data points from different clusters are away from each other;
- *connectivity* is the degree to which neighboring data points are placed in the same cluster [137]. This degree is defined by a neighborhood algorithm. The most commonly used neighborhood construction algorithms are the k-nearest

A. M. Bagirov et al., *Partitional Clustering via Nonsmooth Optimization*,
Unsupervised and Semi-Supervised Learning,
https://doi.org/10.1007/978-3-030-37826-4_10

neighbors, the ε-neighborhood [99], and the NC algorithm [151]. In general, the connectivity decreases when the number of clusters increases;
- *compactness* of clusters is characterized by the degree of density of data points around cluster centers. The compactness improves when the number of clusters increases.

The quality of clustering results can be evaluated by the *cluster validity indices*. In general, validation criteria can be divided into two groups: the *internal validation*—that is based on the information intrinsic to data, and the *external validation*—that is based on the previous knowledge of data. For instance, such knowledge can be a class distribution of a data set. In this case, the known class distributions in data sets can be used to compare the quality of solutions obtained by clustering algorithms. More precisely, cluster distributions are matched with class distributions and for example, the notion of *purity* is used to compare clustering algorithms.

In this chapter, we describe some evaluation measures including various cluster validity indices, silhouette coefficients, Rand index, purity, normalized mutual information, and F-score to mention but a few. Among these measures, the last three are external criteria. The Rand index and its modification can be used both as an internal and an external criteria. All other evaluation measures considered in this chapter are internal criteria.

Clustering algorithms can be compared using their accuracy, the required computational time, and the number of distance function evaluations. Using these three measures, we introduce performance profiles for clustering and apply them to compare the clustering algorithms.

10.2 Optimal Number of Clusters

Determining the optimal number of clusters is among the most challenging problems in cluster analysis. Various cluster validity indices can be used to find the optimal number. The values of these indices are calculated for different number of clusters and a curve of the index values with respect to the number of clusters is drawn. With most cluster validity indices, the optimal number corresponds to the global (or local) minimum or maximum of a cluster validity index. The following algorithm describes the necessary steps to find the optimal number of clusters with respect to a given validity index.

Algorithm 10.1 Finding optimal number of clusters

Input: Minimum and maximum number of clusters, $k_{\min}$ and $k_{\max}$, respectively.
Output: An optimal number of clusters $k \in [k_{\min}, k_{\max}]$.

1: *(Initialization)* Set $j = k_{\min}$.
2: Run a clustering algorithm with j clusters.
3: Compute the cluster distribution $\bar{A} = \{A^1, \ldots, A^j\}$ and the corresponding cluster centers $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_j) \in \mathbb{R}^{nj}$. Calculate the index value for the jth partition.
4: *(Stopping criterion)* If $j < k_{\max}$, then set $j = j + 1$ and go to Step 2.
5: Select $j \in [k_{\min}, k_{\max}]$ for which the partition provides the best result according to some criteria (minimum or maximum) as the optimal number of clusters. Set $k = j$ and **stop**.

Remark 10.1 The value of $k_{\min}$ is usually chosen to be 2.

Remark 10.2 In the case of the incremental clustering algorithms, it is sufficient to select only $k_{\max}$ and there is no need to apply this algorithm repeatedly as it calculates clusters incrementally.

As mentioned above, the optimal number of clusters usually corresponds to optimizers of cluster validity indices. However, it is not always the case as some indices may monotonously decrease (or increase) depending on the number of clusters or indices may have several local maximum or minimum values if $k_{\max}$ is very large. Therefore, it may be more appropriate to use "*knee*" points to find the optimal number of clusters. In the univariate case knee is a point where the index curve is best approximated by a pair of lines. Although the knee point on the index curve may indicate the optimal number of clusters, locating it is not an easy task.

One way to find the knee point is to use the difference between successive values of indices. If the difference is significant, then the previous value of the index can be accepted as the knee point. This type of detection uses only local information and does not reflect the global trend of the index curve. Another way is to apply the L-method that examines the boundary between the pair of straight lines in which they most closely fit the index curve in hierarchical/segmentation clustering (see [256] for more details).

10.3 Cluster Validity Indices

Finding the optimal number of clusters usually relies on the notion of the cluster validity index as mentioned in the previous section. In addition, cluster validity indices can be applied to evaluate the quality of cluster solutions and also can be used as objective functions in clustering problems. Cluster validity indices have been widely studied and applied in cluster analysis [134, 135, 155, 214, 303, 314]. Most of them assume certain geometrical shapes for clusters. When these assumptions are not met, then such indices may fail. Therefore, there is no universal cluster

validity index applicable to all data sets and different indices should be tried in order to decide about the true cluster structure of a data set.

Let $k \geq 2$ and $\bar{A} = \{A^1, \ldots, A^k\}$ be the cluster distribution of the data set A and $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k$ be its cluster centers. Let also $m_j = |A^j|$ be the number of points in the cluster A^j, $j = 1, \ldots, k$. Two important numbers are used in most cluster validity indices. The first one is the sum of squares within the clusters *(intra-cluster)* defined as

$$W_k = \sum_{j=1}^{k} \sum_{\boldsymbol{a} \in A^j} d_2(\boldsymbol{x}_j, \boldsymbol{a}). \tag{10.1}$$

Note that W_k is the value of the clustering objective function multiplied by the number of points in the set A. It can be rewritten as

$$W_k = \sum_{\boldsymbol{a} \in A} \min_{j=1,\ldots,k} d_2(\boldsymbol{x}_j, \boldsymbol{a}). \tag{10.2}$$

The second number is the sum of squares between each cluster center and the center of the entire set A—called the *(inter-cluster)*—defined as

$$B_k = \sum_{j=1}^{k} m_j d_2(\boldsymbol{x}_j, \bar{\boldsymbol{x}}), \tag{10.3}$$

where $\bar{\boldsymbol{x}} \in \mathbb{R}^n$ is the center of the set A.

Note that W_k is used to measure the compactness of clusters whereas B_k is considered as a measure of separation of clusters.

10.3.1 Optimal Value of Objective Function

An optimal value of the objective function in the clustering problem can be used to find the optimal number of clusters. There are different optimization models of clustering problems, and the aim in these models is to minimize the objective function by attaining a high intra-cluster and a low inter-cluster similarity. This means that the clustering objective function is an internal criterion for the quality of clustering and we can get compact and in some cases well-separated clusters by minimizing the objective. However, this aim may not be achieved always. The main reason is that clustering is a global optimization problem—it has many local solutions and only global or nearly global ones provide a good quality cluster solutions. Nevertheless, most clustering algorithms are local search methods. They start from any initial solution and find the closest local solution which might be far away from the global one.

The optimal value of the clustering function found by a clustering algorithm decreases usually as the number of clusters increases. However, this is not always the case. For example, optimal values may not decrease when the k-means or the k-medians algorithms are applied. In these cases, the use of such values may lead to an erroneous decision on the number of clusters as these values may be generated by unexpected local minimizers.

The use of the incremental approach helps us to avoid such undesirable situations. The optimal value of the clustering function found by an incremental clustering algorithm monotonously decreases as the number of clusters increases. Monotonicity of these values with respect to the number of clusters means that there are no (local) minimizers and the knee points should be used to estimate the optimal number of clusters. The following approach can be used to estimate the optimal number of clusters. Let $\varepsilon > 0$ be a given tolerance. For the clustering problem, if

$$\frac{W_k - W_{k+1}}{W_1} \leq \varepsilon,$$

then k is accepted as the optimal number of clusters (W_k is defined in (10.1)). The number W_1 is used to define the relative error as this number is a characteristic of the whole data set.

Note that the optimal value of the cluster function alone may not provide an accurate optimal number of clusters and additional measures are needed. In what follows, we introduce cluster validity indices using the similarity measure d_2; however, most of them can also be given for the similarity measures d_1 and d_∞.

10.3.2 Davies–Bouldin Index

Davies–Bouldin (DB) index [75] is a function of the ratio of the sum of within-cluster scatter to between-cluster separation. For the k-partition problem consider

$$S(A^j) = \frac{1}{m_j} \sum_{\boldsymbol{a} \in A^j} d_2(\boldsymbol{x}_j, \boldsymbol{a}),$$

which is the average squared Euclidean distance of all data points from the cluster A^j to their cluster center $\boldsymbol{x}_j$, $j = 1, \ldots, k$. Let $d_2(\boldsymbol{x}_l, \boldsymbol{x}_j)$ be the squared Euclidean distance between cluster centers $\boldsymbol{x}_l$ and $\boldsymbol{x}_j$, $l, j = 1, \ldots, k$, $j \neq l$. Introduce the following two numbers:

$$R_{lj} = \frac{S(A^l) + S(A^j)}{d_2(\boldsymbol{x}_l, \boldsymbol{x}_j)}, \quad \text{and}$$

$$\bar{R}_j = \max_{l=1,\ldots,k} R_{lj}, \quad j = 1, \ldots, k.$$

The small value of R_{lj} means that the lth and jth clusters are separated, and the small value of $\bar{R}_j$ indicates that the jth cluster is separated from all other clusters. The DB index is defined as

$$DB_k = \frac{1}{k} \sum_{j=1}^{k} \bar{R}_j.$$

Note that the DB index is small if the clusters are compact and well-separated. Consequently, the DB index will have a small value for a good clustering. More specifically, the optimal number of clusters can be identified using local minimizers of the DB index.

10.3.3 Dunn Index

Dunn (Dn) index was introduced in [93, 94]. For $k \geq 2$, consider the k-partition problem. Let

$$\text{dist}(A^j, A^l) = \min_{\boldsymbol{a} \in A^j,\ \boldsymbol{b} \in A^l} d_2(\boldsymbol{a}, \boldsymbol{b})$$

be a squared Euclidean distance between the lth and jth clusters, $j, l = 1, \ldots, k$, $j \neq l$. The squared diameter of the cluster A^j is given by

$$\text{diam}(A^j) = \max_{\boldsymbol{a}, \boldsymbol{b} \in A^j} d_2(\boldsymbol{a}, \boldsymbol{b}).$$

Let

$$\Delta_k = \max_{j=1,\ldots,k} \text{diam}(A^j).$$

Then the Dn index is defined as

$$Dn_k = \frac{1}{\Delta_k} \min_{j=1,\ldots,k} \min_{l=1,\ldots,k,\ l \neq j} \text{dist}(A^j, A^l).$$

The Dn index maximizes the inter-cluster distances and minimizes the intra-cluster distances. This index measures the minimum separation to maximum compactness ratio, so the higher the Dn index value is the better clustering is. Therefore, the number of clusters that maximizes Dn_k can be chosen as the optimal number of clusters. Some generalizations of the Dn index have been proposed, for instance, in [255].

It should be noted that the squared distance between clusters can be defined in many different ways. Here, we define it by calculating pairwise squared distances between points from clusters. However, cluster centers can be used to define this

distance. Note that the similarity measures d_1 and d_∞ can also be utilized to define the *Dn* index.

10.3.4 Hartigan Index

Hartigan (H) index is one of the first cluster validity indices introduced in [143]. The H index is defined as

$$H_k = \Big(\frac{W_k}{W_{k+1}} - 1\Big)(m - k - 1). \tag{10.4}$$

Another expression for this index is

$$H_k = \log \frac{B_k}{W_k},$$

where W_k and B_k are defined in (10.1) and (10.3), respectively.

Let us consider the first definition of the H index. We assume that $W_{k+1} \leq W_k$ for all $k \geq 1$. However, the difference between two successive values of W_k becomes smaller and smaller as the number k of clusters increases. This means that the first term in (10.4) is nonnegative and approaches to 0 as the number of clusters increases. The second term decreases as the number of clusters increases. Therefore, the maximum value of the H index may correspond to the optimal number of clusters.

The second expression for the H index is based on the compactness and the separation of clusters. For a good clustering, the value of B_k is expected to be as large as possible and the value of W_k to be as small as possible. This means that the (local) maximum of the H index corresponds to a good clustering distribution. Note that different similarity measures can be used to define the H index.

10.3.5 Krzanowski–Lai Index

Krzanowski–Lai (KL) index was introduced in [187] and is defined as

$$KL_k = \frac{|v_k|}{|v_{k+1}|}, \quad k > 1.$$

Here

$$v_k = \big(k - 1\big)^{\frac{2}{n}} W_{k-1} - k^{\frac{2}{n}} W_k,$$

and n is the number of attributes in the data set A.

If the set A contains a large number of attributes, then $v_k \approx W_{k-1} - W_k$ and the optimal number of clusters may coincide with a local maximizer of KL_k or with its knee point. On the other hand, when the number of attributes is very small (say for example, less than six), then the optimal number of clusters can be identified using knee points of the KL index curve.

10.3.6 Ball & Hall Index

Ball & Hall (BH) index was introduced in [40]. This index is very simple and can be easily calculated as

$$BH_k = \frac{W_k}{k}.$$

Since one can expect that $W_{k+1} \leq W_k$ for all $k > 1$ it follows that the BH index decreases as k increases. Therefore, in general, it is not expected that BH_k has a local minimizer or maximizer. This is always true for incremental clustering algorithms for which BH_k decreases monotonously. In this case, we can define a tolerance $\varepsilon > 0$. If

$$BH_k - BH_{k+1} \leq \varepsilon \quad \text{for some} \quad k \geq 2,$$

then k can be accepted as the optimal number of clusters. Alternatively, the optimal number of clusters can be determined using knee points on the BH index curve.

10.3.7 Bayesian Information Criterion

Bayesian information criterion (BIC) was introduced in [315]. The BIC is defined as

$$BIC_k = Lm - \frac{1}{2}k(n+1)\sum_{j=1}^{k}\log(m_j),$$

where $L > 0$ is a log-likelihood in the BIC, m_j is the number of data points in the jth cluster, $j = 1, \ldots, k$, and n is the number of attributes in the data set A. Note that knee points of the BIC can be considered as a possible optimal number of clusters.

10.3.8 WB Index

The sum-of-squares based index, called the WB index was introduced in [316] and its modifications are studied in [314]. The WB index is defined as

$$WB_k = \frac{kW_k}{B_k},$$

where W_k and B_k are defined in (10.1) and (10.3), respectively. The name of this index originates from notations used for these two numbers.

As mentioned before, the smaller the value of W_k is the better compactness of clusters and the larger value of B_k is the better separated clusters. Therefore, the minimum of WB_k corresponds to the optimal number of clusters.

10.3.9 Xu Index

Xu index [304] is defined as

$$Xu_k = \frac{n}{2}\log\left(\frac{W_k}{nm^2}\right) + \log(k).$$

Since for a good clustering the values of W_k are expected to be as small as possible it follows that the optimal number of clusters can be identified using local minimizers of the Xu index.

10.3.10 Xie-Beni Index

Xie-Beni (XB) index [303] is applicable to fuzzy clustering problems. Nevertheless, it can also be applied to hard clustering problems with some slight modifications.

Let w_{ij} be a membership degree of the ith data point $\boldsymbol{a}_i$ to the cluster A^j, $i = 1, \ldots, m$, $j = 1, \ldots, k$. Introduce the following numbers:

$$U_k = \sum_{i=1}^{m}\sum_{j=1}^{k} w_{ij}^2 d_2(\boldsymbol{x}_j, \boldsymbol{a}_i), \quad \text{and}$$

$$\tilde{U}_k = \min_{j=1,\ldots,k} \min_{t=j+1,\ldots,k} d_2(\boldsymbol{x}_j, \boldsymbol{x}_t).$$

Here, U_k is the clustering objective function value on the fuzzy clustering problem multiplied by the number of data points. The value of $\tilde{U}_k$ indicates how far cluster centers lie from each other. Then the XB index is defined as

$$XB_k = \frac{U_k}{m\tilde{U}_k}.$$

It is clear that the smaller value of U_k means more compact clusters. The larger value of $\tilde{U}_k$ indicates well-separated clusters. Therefore, the minimum value of the XB index corresponds to the optimal number of clusters.

To make the XB index applicable to hard clustering problems, U_k needs to be modified as follows:

$$U_k = \sum_{j=1}^{k} \sum_{\boldsymbol{a} \in A^j} d_2(\boldsymbol{x}_j, \boldsymbol{a}), \quad \text{or}$$

$$U_k = \sum_{i=1}^{m} \min_{j=1,\ldots,k} d_2(\boldsymbol{x}_j, \boldsymbol{a}_i).$$

In this case, U_k coincides with W_k defined in (10.1). The similarity measures d_1 and d_∞ can also be used to define the XB index.

10.3.11 Sym Index

Sym (*Sm*) index [42] is used to measure the overall average symmetry with respect to cluster centers. Take any cluster $\hat{A}$ from the k-partition $\bar{A} = \{A^1, \ldots, A^k\}$ and denote its center by $\hat{\boldsymbol{x}}$. Let $\bar{\boldsymbol{y}} \in \hat{A}$. Compute $2\hat{\boldsymbol{x}} - \bar{\boldsymbol{y}}$ which reflects the point $\bar{\boldsymbol{y}}$ with respect to the center $\hat{\boldsymbol{x}}$. Denote the reflected point by $\hat{\boldsymbol{y}}$. Assume that k_N nearest neighbors $\boldsymbol{y}_i,\ i = 1, \ldots, k_N$ of $\hat{\boldsymbol{y}}$ are at the squared Euclidean distances $d_2(\hat{\boldsymbol{y}}, \boldsymbol{y}_i)$. Compute the symmetry measure d_s of $\bar{\boldsymbol{y}}$ with respect to $\hat{\boldsymbol{x}}$ as

$$d_s(\bar{\boldsymbol{y}}, \hat{\boldsymbol{x}}) = \frac{\sum_{i=1}^{k_N} d_2(\hat{\boldsymbol{y}}, \boldsymbol{y}_i)}{k_N}.$$

Then the point symmetry based distance $d_{ps}(\bar{\boldsymbol{y}}, \hat{\boldsymbol{x}})$ between $\bar{\boldsymbol{y}}$ and $\hat{\boldsymbol{x}}$ is computed as

$$d_{ps}(\bar{\boldsymbol{y}}, \hat{\boldsymbol{x}}) = d_s(\bar{\boldsymbol{y}}, \hat{\boldsymbol{x}}) \cdot d_2(\bar{\boldsymbol{y}}, \hat{\boldsymbol{x}}).$$

Note that the number of neighbor points k_N cannot be 1. Otherwise, the point $\hat{\boldsymbol{y}}$ is in a data set and $d_{ps}(\bar{\boldsymbol{y}}, \hat{\boldsymbol{x}}) = 0$ and therefore, the impact of the Euclidean distance is ignored. On the other hand, large values of k_N may reduce the symmetry property of a point with respect to a particular cluster center. In practice, k_N is a user defined number and can be chosen as 2.

The maximum separation between a pair of clusters over all possible pairs of clusters is defined by

$$D_k = \max_{i=1,\ldots,k} \max_{j=1,\ldots,k} \|\hat{\boldsymbol{x}}_i - \hat{\boldsymbol{x}}_j\|. \tag{10.5}$$

For each cluster A^j, $j = 1, \ldots, k$ compute

$$E_j = \sum_{\boldsymbol{a} \in A^j} d_{ps}(\boldsymbol{a}, \hat{\boldsymbol{x}}_j), \quad \text{and}$$

$$\mathcal{E}_k = \sum_{j=1}^{k} E_j.$$

Then the *Sm* index is defined as

$$Sm_k = \frac{D_k}{k\mathcal{E}_k}.$$

The *Sm* index should be maximized in order to obtain the optimal number of clusters. This index can also be extended to use the similarity measures d_1 and d_∞.

10.3.12 I Index

I index [8, 208] is defined as

$$I_k = \left(\frac{1}{k} \times \frac{F_1}{F_k} \times D_k\right)^q,$$

where D_k is given in (10.5) and q is any positive integer—usually $q = 2$. The number F_k is defined as the value of clustering function multiplied by the number of data points. That is

$$F_k = \sum_{\boldsymbol{a} \in A} \min_{j=1,\ldots,k} d_2(\boldsymbol{a}, \boldsymbol{x}_j).$$

It is obvious that F_1 corresponds to F_k when $k = 1$.

There are three factors in the definition of the I index: the first one is the reciprocal of the number of clusters and it decreases as the number of clusters increases; the second factor is the ratio of F_1 and F_k. Here, F_1 is a constant for a given data set. Since, in general, with an increase of k the value of F_k will be decreased this factor ensures the formation of more compact clusters; the third factor, D_k generally increases as k increases. Due to the complementary nature of these three factors, it is guaranteed that the I index is able to determine the optimal partitioning. It is clear that the I index is maximized to obtain the optimal number of clusters.

10.3.13 Calinski–Harabasz Index

Calinski–Harabasz (CH) index [57] is defined using the sum of squares within the clusters and the sum of squares between the clusters. The CH index for the k-partition problem with $k \geq 2$ is given by

$$CH_k = \frac{(m-k)B_k}{(k-1)W_k}, \tag{10.6}$$

where W_k and B_k are defined in (10.1) and (10.3), respectively.

The CH index can be expressed with a different formulation. Let d_A be the general mean of all squared distances between points $\boldsymbol{a}_i, \boldsymbol{a}_j \in A$:

$$d_A = \frac{2\sum_{i=1}^{m}\sum_{j=i+1}^{m} d_2(\boldsymbol{a}_i, \boldsymbol{a}_j)}{m(m-1)},$$

and d_j be the mean value for each cluster A^j, $j = 1, \ldots, k$:

$$d_j = \frac{2\sum_{\boldsymbol{a}\in A^j}\sum_{\boldsymbol{b}\in A^j} d_2(\boldsymbol{a}, \boldsymbol{b})}{m_j(m_j-1)}.$$

Then the sum of squares within the clusters can be alternatively computed as (cf. (10.1))

$$W_k = \frac{1}{2}\Big(\sum_{j=1}^{k}(m_j-1)d_j\Big),$$

and the sum of squares between the clusters is alternatively defined as (cf. (10.3))

$$B_k = \frac{1}{2}\Big((k-1)d_A + (m-k)Q_k\Big).$$

Here, Q_k is a weighted mean of the differences between the general and the within-cluster mean squared distances, that is

$$Q_k = \frac{1}{m-k}\sum_{j=1}^{k}(m_j-1)(d_A - d_j).$$

Then the CH index, given in (10.6), can be reformulated as

$$CH_k = \frac{d_A + \left(\frac{m-k}{k-1}\right)Q_k}{d_A - Q_k}.$$

If the distances between all pairs of points are equal, then $Q_k = 0$ and we get $CH_k = 1$. Since d_A is a constant for a data set A it follows that the minimum value of W_k maximizes Q_k for a given k.

The parameter Q_k can also be used to compare partitions obtained for different number of clusters: the difference $Q_k - Q_{k-1}$ indicates an average gain in the compactness of clusters resulting from the change from $k-1$ to k clusters. Hence, the behavior of Q_k depending on k may be sensitive to the existence of such clusters. Let

$$q_k = \frac{Q_k}{d_A}.$$

It is clear that $q_k \in [0, 1]$. The case $q_k = 0$ means that all distances between pairs of data points are equal while $q_k = 1$ implies $k = m$.

Then the CH index can be rewritten as

$$CH_k = \frac{1 + \left(\frac{m-k}{k-1}\right) q_k}{1 - q_k}.$$

If data points are grouped into k clusters with a small within-cluster variation, then the change from $k-1$ to k causes a considerable increase in q_k. This in turn leads to a rapid increase of the CH index. Therefore, this index can be applied to identify the optimal number of clusters. It is suggested in [57] to choose the value of k for which the CH index has a (local) maximum or at least a comparatively rapid increase. The latter point can be considered as a knee point on the CH index curve. If there are several such local maxima or knee points, then the smallest corresponding value of k can be chosen. In practice, this means that the computation can be stopped when the first local maximum or the knee point is found.

10.4 Silhouette Coefficients and Plots

The aim of the *silhouette plot* is to identify compact and well-separated clusters [251] (see, also [174]). More precisely, the silhouette plot is used to interpret and validate consistency within clusters. It provides a concise graphical representation of how well each data point lies within its cluster.

Silhouettes are constructed using the k-partition $\bar{A} = \{A^1, \ldots, A^k\}$, its cluster centers $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k$, and the collection of all proximities between data points. Take any point $\boldsymbol{a} \in A$ and denote by A^j, $j \in \{1, \ldots, k\}$ the cluster to which this point belongs. Assuming that this cluster contains other data points apart from $\boldsymbol{a}$, compute

$$\bar{d}_{\boldsymbol{a}} = \frac{1}{m_j - 1} \sum_{\boldsymbol{b} \in A^j, \boldsymbol{b} \neq \boldsymbol{a}} d_2(\boldsymbol{a}, \boldsymbol{b}).$$

Here, $\bar{d}_{\boldsymbol{a}}$ is the average dissimilarity of the point $\boldsymbol{a}$ to all other points of the cluster A^j. Choose any $t \in \{1, \ldots, k\},\ t \neq j$ and compute

$$\bar{d}_{t,\boldsymbol{a}} = \frac{1}{m_t} \sum_{\boldsymbol{b} \in A^t} d_2(\boldsymbol{a}, \boldsymbol{b}),$$

which is the average dissimilarity of the point $\boldsymbol{a}$ to points from the cluster A^t. Calculate the minimum average dissimilarity

$$\hat{d}_{\boldsymbol{a}} = \min_{t=1,\ldots,k,\ t \neq j} \bar{d}_{t,\boldsymbol{a}}.$$

Let $\bar{d}_{t_0,\boldsymbol{a}} = \hat{d}_{\boldsymbol{a}},\ t_0 \in \{1, \ldots, k\},\ t_0 \neq j$. The cluster A^{t_0} is called the *neighbor cluster* of the data point $\boldsymbol{a}$. For each point $\boldsymbol{a} \in A$, calculate the *silhouette coefficient* $s(\boldsymbol{a})$ using the following formula:

$$s(\boldsymbol{a}) = \begin{cases} 1 - \bar{d}_{\boldsymbol{a}}/\hat{d}_{\boldsymbol{a}}, & \text{if} \quad \bar{d}_{\boldsymbol{a}} < \hat{d}_{\boldsymbol{a}}, \\ 0, & \text{if} \quad \bar{d}_{\boldsymbol{a}} = \hat{d}_{\boldsymbol{a}}, \\ \hat{d}_{\boldsymbol{a}}/\bar{d}_{\boldsymbol{a}} - 1, & \text{if} \quad \bar{d}_{\boldsymbol{a}} > \hat{d}_{\boldsymbol{a}}. \end{cases}$$

This formula can be rewritten as

$$s(\boldsymbol{a}) = \frac{\hat{d}_{\boldsymbol{a}} - \bar{d}_{\boldsymbol{a}}}{\max\{\bar{d}_{\boldsymbol{a}},\ \hat{d}_{\boldsymbol{a}}\}}.$$

It is clear that $s(\boldsymbol{a}) \in [-1, 1]$ for all $\boldsymbol{a} \in A$. When $s(\boldsymbol{a})$ is close to 1, the *within dissimilarity* $\bar{d}_{\boldsymbol{a}}$ is much smaller than the smallest *between dissimilarity* $\hat{d}_{\boldsymbol{a}}$. Therefore, we can say that $\boldsymbol{a}$ is *well-clustered* as it seems that $\boldsymbol{a}$ is assigned to an appropriate cluster and the second best choice A^{t_0} is not nearly as close as the actual choice A^j. If $s(\boldsymbol{a}) = 0$ or it is close to 0, then $\bar{d}_{\boldsymbol{a}}$ and $\hat{d}_{\boldsymbol{a}}$ are approximately equal. This means that the point $\boldsymbol{a}$ lies equally far away from A^j and A^{t_0}, and it is not clear which cluster it should be assigned. In this case, the data point can be considered as an *intermediate case*. When $s(\boldsymbol{a})$ is close to -1, $\bar{d}_{\boldsymbol{a}}$ is much larger than $\hat{d}_{\boldsymbol{a}}$. This means that on the average $\boldsymbol{a}$ is much closer to A^{t_0} than A^j. Hence, it would be more natural to assign $\boldsymbol{a}$ to A^{t_0} than to A^j and we can conclude that $\boldsymbol{a}$ is *misclassified*. To conclude, $s(\boldsymbol{a})$ measures how well the data point $\boldsymbol{a}$ matches with the clustering at hand, that is, how well the point has been assigned.

The numbers $s(\boldsymbol{a})$ can be used to draw the silhouette plot. The silhouette of the cluster A^j is a plot of $s(\boldsymbol{a})$ ranked in a decreasing order for all points $\boldsymbol{a}$ in A^j. The silhouette plot shows which data points lie well within the cluster and which ones are merely somewhere in between clusters. A wide silhouette indicates large $s(\boldsymbol{a})$ values and hence a pronounced cluster. The other dimension of the silhouette plot is its height which simply equals the number of objects in A^j.

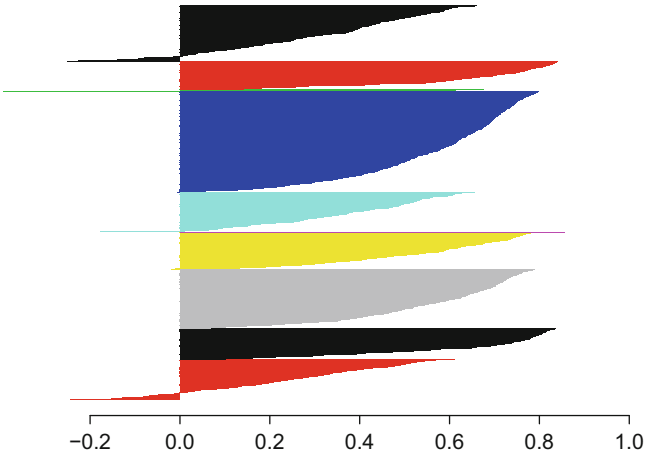

Fig. 10.1 Illustration of silhouette plot for ten clusters

Combining the silhouette of different clusters we can draw a single silhouette plot for the whole data set A. This allows us to distinguish *clear-cut* clusters from the *weak* (not well-separated) ones. In addition, the number of objects in each cluster A^j can be detected by using the height of the silhouette of the cluster A^j.

To illustrate silhouette plots we consider the 10-partition of image segmentation data set obtained using the MGKM. The silhouette plot of the 10-partition is given in Fig. 10.1. It is depicted using the R package available from https://cran.r-project.org/web/packages/clues/clues.pdf (see [290], for details). In this figure, clusters are shown using different colors. The height and area of the cluster depend on the number of its data points. The very narrow "pink" cluster is the smallest and the "blue" cluster is the largest one. If any cluster has data points with negative silhouettes, then this cluster is not well-separated. The absence of such points means that the cluster is considered as well-separated. The top "black," "green," "light blue," "yellow" and the bottom "red" clusters are not well-separated from other clusters. The "blue" cluster contains a very few "misclassified" points. The remaining four clusters are considered as separable from other clusters. The right hand side of clusters reflects the number of data points well lying inside their own clusters. The top "red," "blue," and the bottom "black" clusters contain more such points than any other cluster.

10.5 Rand Index

Rand (*Rn*) index [243] measures similarity between two different cluster distributions of the same data set (see, also [215, 257]). Let $\bar{A}_1$ and $\bar{A}_2$ be two cluster distributions of the data set A:

Table 10.1 Contingency table for comparing partitions $\bar{A}_1$ and $\bar{A}_2$

Partition	Clusters	$\bar{A}_2$				Total
		A_2^1	A_2^2	...	$A_2^{k_2}$	
$\bar{A}_1$	A_1^1	n_{11}	n_{12}	...	n_{1k_2}	s_1
	A_1^2	n_{21}	n_{22}	...	n_{2k_2}	s_2
	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
	$A_1^{k_1}$	$n_{k_1 1}$	$n_{k_1 2}$	...	$n_{k_1 k_2}$	s_{k_1}
Total		r_1	r_2	...	r_{k_2}	m

$$\bar{A}_1 = \{A_1^1, \ldots, A_1^{k_1}\}, \quad k_1 > 1, \quad \text{and} \tag{10.7}$$

$$\bar{A}_2 = \{A_2^1, \ldots, A_2^{k_2}\}, \quad k_2 > 1. \tag{10.8}$$

Denote the elements of the $k_1 \times k_2$ matrix by n_{ij}, where n_{ij} represents the number of data points belonging to the ith cluster A_1^i of the cluster partition $\bar{A}_1$ and the jth cluster of the cluster partition $\bar{A}_2$, $i = 1, \ldots, k_1$, $j = 1, \ldots, k_2$. Then the contingency table for distributions $\bar{A}_1$ and $\bar{A}_2$ can be formed as in Table 10.1. In this table, the entry s_i indicates the number of data points in the ith cluster of the cluster partition $\bar{A}_1$ and r_j shows the number of data points in the jth cluster of the cluster partition $\bar{A}_2$.

Let $\mathcal{C}(m, 2)$ be the number of 2-combinations from m data points. Among all possible combinations of pairs $\mathcal{C}(m, 2)$, there are the following different types of pairs:

- data points in a pair belong to the same cluster in $\bar{A}_1$ and to the same cluster in $\bar{A}_2$. Denote the number of such pairs by N_1;
- data points in a pair belong to the same cluster in $\bar{A}_1$ and to different clusters in $\bar{A}_2$. Denote the number of such pairs by N_2;
- data points in a pair belong to the same cluster in $\bar{A}_2$ and to different clusters in $\bar{A}_1$. Denote the number of such pairs by N_3; and
- data points in a pair belong to different clusters in $\bar{A}_1$ and to different clusters in $\bar{A}_2$. Denote the number of such pairs by N_4.

The values of N_1, N_2, N_3, and N_4 can be calculated using entries from Table 10.1:

$$N_1 = \sum_{i=1}^{k_1} \sum_{j=1}^{k_2} \mathcal{C}(n_{ij}, 2) = \frac{1}{2} \left(\sum_{i=1}^{k_1} \sum_{j=1}^{k_2} n_{ij}^2 - m \right),$$

$$N_2 = \sum_{i=1}^{k_1} \mathcal{C}(s_i, 2) - N_1 = \frac{1}{2} \left(\sum_{i=1}^{k_1} s_i^2 - \sum_{i=1}^{k_1} \sum_{j=1}^{k_2} n_{ij}^2 \right),$$

$$N_3 = \sum_{j=1}^{k_2} \mathcal{C}(r_j, 2) - N_1 = \frac{1}{2} \left(\sum_{j=1}^{k_2} r_j^2 - \sum_{i=1}^{k_1} \sum_{j=1}^{k_2} n_{ij}^2 \right), \quad \text{and}$$

$$N_4 = \mathcal{C}(m,2) - N_1 - N_2 - N_3 = \frac{1}{2}\left(\sum_{i=1}^{k_1}\sum_{j=1}^{k_2} n_{ij}^2 + m^2 - \sum_{i=1}^{k_1} s_i^2 - \sum_{j=1}^{k_2} r_j^2\right).$$

Then the Rn index can be calculated as

$$Rn = \frac{N_1 + N_4}{N_1 + N_2 + N_3 + N_4}.$$

Note that the Rn index is ranging from 0 to 1. If $Rn = 0$, then two cluster distributions have no similarity, and if $Rn = 1$, then they are identical.

The Rn index can also be used to measure the similarity between a cluster distribution and a class distribution in data sets with class outputs. In this case, we can replace one of cluster distributions above by the class distribution. Therefore, the Rn index is both an internal and an external index.

10.6 Adjusted Rand Index

The Rn index of two random cluster distributions may not have a constant value (say zero) or it may approach its upper limit of unity as the number of clusters increases. To overcome these limitations several other measures have been introduced, for example, the Fowlkes–Mallows index proposed in [109]. Another example of such measures is the *Adjusted Rand* (ARn) index which is an improvement of the Rn index [150]. This index is considered as one of the successful cluster validation indices [215].

Recall the cluster distributions $\bar{A}_1$ and $\bar{A}_2$, given in (10.7) and (10.8), for the data set A. Then the ARn index is

$$ARn = \frac{\mathcal{C}(m,2)(N_1 + N_4) - \Big((N_1 + N_2)(N_1 + N_3) + (N_2 + N_4)(N_3 + N_4)\Big)}{\Big(\mathcal{C}(m,2)\Big)^2 - \Big((N_1 + N_2)(N_1 + N_3) + (N_2 + N_4)(N_3 + N_4)\Big)},$$

or

$$ARn = \frac{\mathcal{C}(m,2)\sum_i\sum_j \mathcal{C}(n_{ij},2) - \Big(\sum_i \mathcal{C}(s_i,2)\sum_j \mathcal{C}(r_j,2)\Big)}{\frac{1}{2}\mathcal{C}(m,2)\Big(\sum_i \mathcal{C}(s_i,2) + \sum_j \mathcal{C}(r_j,2)\Big) - \Big(\sum_i \mathcal{C}(s_i,2)\sum_j \mathcal{C}(r_j,2)\Big)},$$

where, n_{ij}, s_i, r_j are values from Table 10.1.

The ARn index can be used both as an internal and an external index. In the latter case it is assumed that the data set A has class outputs and its cluster distribution is compared with the class distribution.

10.7 Purity

Purity measures the quality of clustering in data sets with class outputs [83]. More precisely, it shows how well the cluster distribution obtained by a certain clustering algorithm reflects the existing class structure of a data set. Therefore, the purity is applicable when a data set has a class label, that is, the purity is an external criterion. The purity can be computed for each cluster $A^j,\ j = 1, \ldots, k$ and the whole data set A.

Let $\bar{A} = \{A^1, \ldots, A^k\}$ be the cluster distribution of the set A obtained by a clustering algorithm. It is assumed that $k \geq 2$ and $A^j \neq \emptyset,\ j = 1, \ldots, k$. Let also $\bar{C} = \{C_1, \ldots, C_l\}$ be the set of true classes of the set A. The cluster A^j may contain different number of points from classes $C_t,\ t = 1, \ldots, l$. Denote by n_{tj} the number of points from the tth class belonging to the jth cluster. For this cluster compute

$$\bar{m}_j = \max_{t=1,\ldots,l} n_{tj}, \quad j = 1, \ldots, k.$$

Then the purity of the cluster A^j is defined as

$$Pu(A^j) = \frac{\bar{m}_j}{m_j},$$

where m_j is the number of points in the cluster A^j. Usually the purity is expressed in percentage, hence this formula can be rewritten as

$$Pu(A^j) = \frac{\bar{m}_j}{m_j} \times 100\%.$$

The purity for the cluster distribution $\bar{A}$ is

$$Pu(\bar{A}) = \frac{\sum_{j=1}^{k} \bar{m}_j}{m} \times 100\%.$$

The purity is 1 (100%) if each cluster contains data points only from one class. Note that increasing the number of clusters, in general, will increase the purity. In particular, if each cluster has only one data point, then the purity is equal to 1 (100%). This implies that the purity cannot be used to evaluate the trade-off between the quality and the number of clusters.

10.8 Normalized Mutual Information

Normalized mutual information (NMI) is a combination of the mutual information and the entropy [189]. The mutual information is used to measure how well the computed clusters and the true classes predict one another. The entropy is used to measure the amount of information inherent in both the cluster distribution and the true classes. The entropy is also used to normalize the mutual information which allows us to evaluate the trade-off between the quality and the number of clusters. The NMI is an external criterion to evaluate the quality of clusters.

Recall that $\bar{A}$ is the cluster distribution obtained by a clustering algorithm for the data set A and $\bar{C}$ is the set of its true classes. Let n_{tj} be the number of points from the tth class belonging to the jth cluster, n_t be the number of points in the class C_t, and m_j be the number of points in the cluster A^j.

The estimation $\bar{P}_j$ of probability for a data point being in the jth cluster is $\bar{P}_j = m_j/m,\ j = 1, \ldots, k$, the estimation $\hat{P}_t$ of probability for a data point being in the tth class is $\hat{P}_t = n_t/m$ and finally, the estimation $\bar{P}_{tj}$ of probability for a data point being in the intersection $A^j \cap C_t$ is $\bar{P}_{tj} = n_{tj}/m$. Then the mutual information $I(\bar{A}, \bar{C})$ between the cluster distribution $\bar{A}$ and the class distribution $\bar{C}$ is defined as

$$\begin{aligned} I(\bar{A}, \bar{C}) &= \sum_{j=1}^{k} \sum_{t=1}^{l} \bar{P}_{tj} \log \left(\frac{\bar{P}_{tj}}{\bar{P}_j \hat{P}_t} \right) \\ &= \sum_{j=1}^{k} \sum_{t=1}^{l} \frac{n_{tj}}{m} \log \left(\frac{m n_{tj}}{m_j n_t} \right). \end{aligned}$$

The estimation $H(\bar{A})$ for the entropy of the cluster distribution $\bar{A}$ is computed as

$$\begin{aligned} H(\bar{A}) &= -\sum_{j=1}^{k} \bar{P}_j \log \bar{P}_j \\ &= -\sum_{j=1}^{k} \frac{m_j}{m} \log \left(\frac{m_j}{m} \right) \end{aligned}$$

and the estimation $H(\bar{C})$ for the entropy of the class distribution $\bar{C}$ is

$$\begin{aligned} H(\bar{C}) &= -\sum_{t=1}^{l} \hat{P}_t \log \hat{P}_t \\ &= -\sum_{t=1}^{l} \frac{n_t}{m} \log \left(\frac{n_t}{m} \right). \end{aligned}$$

Then the NMI is calculated by

$$NMI(\bar{A}, \bar{C}) = \frac{2I(\bar{A}, \bar{C})}{H(\bar{A}) + H(\bar{C})}.$$

It is clear that $NMI(\bar{A}, \bar{C}) \in [0, 1]$. $NMI(\bar{A}, \bar{C}) = 1$ means that the cluster distribution $\bar{A}$ and the class distribution $\bar{C}$ of the data set A are identical, that is the cluster distribution obtained by a clustering algorithm has a high quality.

10.9 *F*-Score

F-score (known also as *F-measure*) is an external validity index to evaluate the quality of clustering solutions [191]. The introduction of this index is inspired by the information retrieval metric known as the F-measure. We describe the F-score using notations from the previous section.

The F-score combines the concepts of *precision* (Pr) and *recall* (Re). For the cluster $A^j,\ j = 1, \ldots, k$ and the class $C_t,\ t = 1, \ldots, l$, Pr and Re are defined as

$$Pr(A^j, C_t) = \frac{n_{tj}}{n_t}, \quad \text{and}$$
$$Re(A^j, C_t) = \frac{n_{tj}}{m_j}.$$

Then the F-score of the cluster A^j and the class C_t is given by

$$F(A^j, C_t) = \frac{2Pr(A^j, C_t) \times Re(A^j, C_t)}{Pr(A^j, C_t) + Re(A^j, C_t)}.$$

Note that $F(A^j, C_t) \in [0, 1],\ t = 1, \ldots, l,\ j = 1, \ldots, k$. The total F-score F_t is computed as

$$F_t = \frac{1}{m} \sum_{j=1}^{k} m_j \max_{t=1,\ldots,l} F(A^j, C_t).$$

It is obvious that $F_t \in [0, 1]$. The higher the F-score is, the better the clustering solution is. This measure has an advantage over the purity since it measures both the homogeneity and the completeness of a clustering solution. The *homogeneity* of a clustering solution means that all its clusters contain only data points which are members of a single class. The *completeness* of a clustering solution means that data points that are members of a given class are elements of the same cluster.

10.10 Performance Profiles in Cluster Analysis

Performance profiles, introduced in [86], are widely used to compare optimization algorithms. Such profiles have been introduced for comparison of (sub)gradient-based and derivative free optimization algorithms. The number of function (and gradient) evaluations and the computational time are usually used to compute these profiles.

Clustering is a global optimization problem and the ability of a clustering algorithm to find global or nearly global solutions is important as such solutions provide the best cluster structure of a data set with the least number of clusters. Therefore, here we introduce performance profiles for comparison of clustering algorithms. The profiles are defined using three parameters: the accuracy, the number of distance function evaluations, and the computational time.

10.10.1 Accuracy

Accuracy of clustering algorithms can be determined using the known global solutions or the best known solutions of clustering problems. Consider the k-partition problem in a data set A. Assume that $f_k^* > 0$ is the best known value of the objective function in the k-partition problem and $\bar{f}_k$ is the lowest value of this function obtained by a clustering algorithm Υ. Then the accuracy (or error) E_Υ of the algorithm Υ for solving the k-partition problem is defined as

$$E_\Upsilon^k = \frac{\bar{f}_k - f_k^*}{f_k^*}. \tag{10.9}$$

In some cases, it is convenient to present the error (or accuracy) in %, therefore the error can be defined as

$$E_\Upsilon^k = \frac{\bar{f}_k - f_k^*}{f_k^*} \times 100\%. \tag{10.10}$$

We describe performance profiles using the percentage representation of the error. Assume that the algorithm Υ is applied to solve the l-clustering problems for $l = 2, \ldots, k$ in t data sets. Then the total number of clustering problems is $t(k-1)$. Denote by $E_\Upsilon^{l,q}$ the error of the solution obtained by the algorithm Υ for solving the l-clustering problem in the qth data set, where $l \in \{2, \ldots, k\}$ and $q \in \{1, \ldots, t\}$. Let $\tau \geq 0$ be any given number. For the algorithm Υ, define the set

$$\sigma_\Upsilon(\tau) = \left\{(l, q) : E_\Upsilon^{l,q} \leq \tau\right\}.$$

It is clear that the set $\sigma_\Upsilon(0)$ contains only those indices (l, q) in which the algorithm Υ finds the best known solutions. Furthermore, for sufficiently large τ we have

$$\sigma_\Upsilon(\tau) = \big\{(l, q) : l \in \{2, \ldots, k\},\ q \in \{1, \ldots, t\}\big\}.$$

Then the probability that the algorithm Υ solves the collection of clustering problems with the accuracy $\tau \geq 0$ is given as

$$P_\Upsilon(\tau) = \frac{|\sigma_\Upsilon(\tau)|}{t(k-1)}.$$

Define a threshold $\tau_0 > 0$. An algorithm Υ is unsuccessful in solving the l-clustering problem in the qth data set if $E_\Upsilon^{l,q} > \tau_0$ ($l \in \{2, \ldots, k\},\ q \in \{1, \ldots, t\}$). This allows us to draw the graph of the function $P_\Upsilon(\tau)$ in the interval $[0, \tau_0]$.

Note that the higher the graph on the left hand side is more accurate the algorithm is in finding the best known solutions than other algorithms. The higher on the right hand side means that the algorithm is more robust in finding the nearly best known solutions than the other algorithms.

10.10.2 Number of Distance Function Evaluations

Most optimization based clustering algorithms use values and subgradients (or gradients) of the cluster function to solve clustering problems. To compute them, we need to calculate distance functions and also apply minimum operations for each data point. Recall that the data set A has m data points and n attributes. It is expected that, in average, the number of distance function evaluations in a clustering algorithm depends linearly or almost linearly on the number of clusters. At each iteration, the number of such evaluations is $O(mk)$, where k is the number of clusters.

Assume that a clustering algorithm uses M iterations to solve the k-clustering problem. Then the expected value for the number of distance function evaluations required by the algorithm is

$$N(m, k) = cMmk. \tag{10.11}$$

Here, $c > 0$ is a given constant. The number N can be used as a benchmark to evaluate the performance of clustering algorithms. Note that unlike the performance profiles usually used in optimization [86], this benchmark does not depend on any solver. Therefore, we can use it to rank clustering algorithms both in the sense of efficiency and robustness.

First, we define when one can consider the performance of an algorithm as success or failure. Since clustering is a global optimization problem we consider any run of a clustering algorithm Υ as success if it finds either global (or best known) or

nearly global (or nearly best known) solution. Thus, we can define some threshold $\tau > 0$ (in %). If the error $E^k_\Upsilon \leq \tau$, then the algorithm Υ is considered to be successful otherwise, it fails to solve the clustering problem.

Assume that v clustering solvers $\Upsilon_1, \ldots, \Upsilon_v$ are applied to solve the l-clustering problems for $l = 2, \ldots, k$ in t data sets. Let $N(l, q)$ be a benchmark number computed using (10.11) for the qth data set with l clusters. For each solver $\Upsilon_s,\ s = 1, \ldots, v$, denote by Q_s the set of clustering problems successfully solved applying this solver:

$$Q_s \subseteq \big\{(l, q) : q \in \{1, \ldots, t\};\ l \in \{2, \ldots, k\}\big\}. \tag{10.12}$$

Let

$$V = \big\{s \in \{1, \ldots, v\} : Q_s \neq \emptyset\big\}, \tag{10.13}$$

and take any $s \in V$. Define the number

$$\sigma_s(l, q) = \frac{N^s_{lq}}{N(l, q)},$$

where N^s_{lq} is the number of distance function evaluations used by the solver Υ_s for solving the l-clustering problem in the qth data set.

Compute the number

$$\tau_{\max} = \max_{s \in V} \max_{(l,q) \in Q_s} \sigma_s(l, q).$$

For any $s \in V$ and a number $\tau \in (0, \tau_{\max}]$ consider the set

$$X_s(\tau) = \big\{(l, q) \in Q_s : \sigma_s(l, q) \leq \tau\big\}.$$

Then the performance profiles for the solver $\Upsilon_s,\ s \in V$ can be defined as

$$\rho_s(\tau) = \frac{|X_s(\tau)|}{t(k-1)}.$$

For other solvers $\Upsilon_s,\ s \notin V$ we set $\rho_s(\tau) = 0$ for all $\tau \geq 0$.

10.10.3 Computational Time

Performance profiles using the computational time can be obtained in a similar way to those using the distance function evaluations. The number of operations for one evaluation of the squared Euclidean distance function is $3n - 1$, where n is

the number of attributes in the data set A. Consider the k-clustering problem. The number of distance function evaluations for one iteration of an algorithm and the total number of operations are estimated by $O(mk)$ and $O(nmk)$, respectively.

Assume that a clustering algorithm uses about M iterations to solve the k-clustering problem. Then we get a number

$$T(m,k) = \bar{c}Mnmk \tag{10.14}$$

as an expected value for the number of operations, where $\bar{c} > 0$ is a given constant. Dividing T by 10^9 (this number depends on characteristics of a computer), we get the expected value $\overline{T}$ for the computational time. This number is used as a benchmark to evaluate the performance of clustering algorithms based on the computational time.

As in the case of the number of distance function evaluations, we introduce a threshold $\tau > 0$ (in %) to define whether an algorithm fails or succeeds to solve the clustering problem. Assume that v clustering solvers $\Upsilon_1, \ldots, \Upsilon_v$ are applied to solve the l-clustering problems for $l = 2, \ldots, k$ in t data sets. For each solver $\Upsilon_s,\ s = 1, \ldots, v$ define the set Q_s using (10.12) and the set V by applying (10.13).

Take any $s \in V$ and compute

$$\beta_s(l,q) = \frac{T_{lq}^s}{\overline{T}(l,q)},$$

where T_{lq}^s is the computational time used by the solver Υ_s for solving the l-clustering problem in the qth data set. Calculate the number

$$\tau_{\max} = \max_{s \in V} \max_{(l,q) \in Q_s} \beta_s(l,q).$$

For any $s \in V$ and a number $\tau \in (0, \tau_{\max}]$, consider the set

$$X_s(\tau) = \left\{(l,q) \in Q_s : \beta_s(l,q) \leq \tau\right\}.$$

For each solver $\Upsilon_s,\ s \in V$, we can define the performance profiles as follows:

$$\rho_s(\tau) = \frac{|X_s(\tau)|}{t(k-1)}.$$

For other solvers $\Upsilon_s,\ s \notin V$ we set $\rho_s(\tau) = 0$ for all $\tau \geq 0$.