How much *true* structure has been discovered?
Validating Explorative Clustering on a Hold-Out Test Set

F. Höppner

University of Applied Sciences Braunschweig/Wolfenbüttel
D-38440 Wolfsburg, Germany
f.hoeppner@fh-wolfenbuettel.de

Abstract. Comparing clustering algorithms is much more difficult than comparing classification algorithms, which is due to the unsupervised nature of the task and the lack of a precisely stated objective. We consider explorative cluster analysis as a predictive task (predict regions where data lumps together) and propose a measure to evaluate the performance on an hold-out test set. The performance is discussed for typical situations and results on artificial and real world datasets are presented for partitional, hierarchical, and density-based clustering algorithms. The proposed S-measure successfully senses the individual strengths and weaknesses of each algorithm.

1 Introduction

Cluster analysis is about finding groups in data. The most prominent conception is that clusters are *clouds of data objects* that should be compact and well separated from each other [1–3] or an agglomeration of such clouds [4]. Alternatively "*clusters may be described as connected regions of multi-dimensional space containing a relatively high density of points, separated from other such regions by a region containing a relatively low density of points*" [2], which is closer to the density-based view on clustering.

Many clustering algorithms have been proposed, such as prototype-based methods, density-based or hierarchical methods. Each algorithm has its individual strengths, weaknesses, biases and assumptions and usually it is not a priori clear which algorithm is best for the data at hand. It is still an open question how to compare the performance of clustering algorithms. Typically, a *validity measure* [5–11] is used to assess the quality of a result such that the performance can be compared indirectly.

Compared to classification, the variety of performance measures is much larger and less established. Many validity measures make use of the individual outcome of the clustering algorithm and thus cannot be used with a competing

[0] published in: Machine Learning and Data Mining in Pattern Recognition, 6th International Conference, MLDM 2009, LNCS 5632, pp 385–397, Springer

algorithm from a different paradigm (applies to [6–8], for instance). Others do not address the quality of the clustering, but the accordance of two results (such as [5]). A good match, however, can be achieved likewise with pairs of good clusters and pairs of poor clusters. To assess the performance, this measure would require the existence of a reference solution, which is usually not present in unsupervised learning. Yet other measures verify the clusters against local distortions of the data [10] or resampling [11, 12], but they do not allow a comparison of the most stable partitions of two different clustering algorithms.

Why does clustering suffer from this unsatisfying situation but classification does not? A classifier can be seen as a function $f : X \to C \cup \{\perp\}$ providing for a record x from the d-dimensional data space X the predicted class $c \in C$ (or $\perp$ in case no prediction is made). This *common way of using a classifier* makes it easier to define validation procedures that can be used with any classifier. With clustering there is not a single *common usage* – in many applications cluster analysis is used as a means to an end (e.g. in preprocessing tasks, local regression model, etc.), but in others as an end in itself (e.g. descriptive or explorative tasks). Focussing on one kind of application may help us to find a validation procedure more general than existing validity measures.

In this paper, we will follow this idea and focus on explorative clustering (pattern discovery). We discuss the *typical use* in this case, in what respect the known clustering algorithms provide this information and propose a first validity measure that goes into this direction. Results on three different kinds of clustering algorithms (partitional: k-Means [13], density-based: DBScan [14], agglomerative hierarchical clustering (AHC), e.g. [4]) demonstrate the applicability of the measure across different approaches to cluster analysis.

2 Explorative Analysis and Cluster Assignment

Suppose we are interested in finding groups in our database (of customers, for example). Note that we are *not* interested in artificially dividing the customers into similar groups, but we want to know if the data itself supports a partition into different groups of customers. We then want to *understand* the result, i.e., in the explorative setting we need a compact, comprehensive description:

1. *Where do the clusters lie in the data space and how far do they extend?* – Without loss of generality we assume that clusters are represented by a set of geometrical elements or shapes such as hyperballs, -boxes, -planes, -ellipsoids, Voronoi cells, etc. For the sake of simplicity, in this paper we assume that all clusters $c \in C$ are composed out of hyperspherical shapes $B(x, r) = \{y \in X \mid \|x - y\| \leq r\}$.[1]
2. *How large is the gap between the clusters, how well is a cluster separated from the remainder?* – We indicate the gap by a *separation area* around the clusters of width 2ε (such that hyperballs of range ε will fit into this area). This range may vary from cluster to cluster, of course.

[1] Note that we do not assume that all clusters *have* hyperspherical shapes, we only assume that we can *approximate* their shape by the union of several hyperspheres.

DBScan [14]
DBScan represents a cluster already by a set of small hyperspheres with some fixed radius ε around the *core points* of the cluster. This set corresponds exactly to the set of *basic shapes*. The algorithm guarantees only that neighboring clusters have at least a separation of 2ε, therefore this parameter can be taken unaltered.

k-Means [13]
The original k-Means algorithm is a partitional clustering algorithm that assigns every record to a cluster, it does not consider the possibility of noise or outliers. Here, we heuristically define the *extent* of each cluster by means of hyperspheres: if data does not fall into their volume, it does not belong to the cluster. Furthermore, if the hyperspheres of two prototypes intersect, we consider them as representing a single cluster, so the number of obtained clusters is not always identical to k. Let C_i contain all data assigned to cluster $\#i$. For each cluster $\#i$ with prototype $p_i \in X$ we calculate the average distance of all data points assigned to this cluster: $$r_i = \frac{1}{|C_i|} \sum_{x \in C_i} \|x - p_i\|$$ Since this is the average distance, we choose the size of the cluster twice as large, i.e., we define the shape of the cluster as a hypersphere with the prototype as its center and the radius $r_i^\bullet = \sqrt[d]{2 \cdot r_i^d}$ where d is the dimensionality. The distance between the hyperspheres of cluster $\#i$ and $\#j$ is given by $d_{i,j} = \|p_i - p_j\| - r_i^\bullet - r_j^\bullet$. In case $d_{i,j} < 0$ the hyperspheres intersect and we consider them as a single cluster (represented by two (or even more)) prototypes. From the prototype location and the cluster size we can directly calculate the separation margin between the clusters. If the resulting ε gets very small, 10% of the cluster radius is used as a lower bound for ε (if ε is very small, the estimated density may easily become very high because the volume of the neighborhood gets very small).

Agglomerative Hierarchical Clustering (AHC), e.g. [4]
We consider the classical single-linkage AHC algorithm here. The resulting dendrogram is used to come up with the final clusters, by *cutting off* all edges that cross a certain distance level. This *cut-off distance* d_{cut} is often determined by visual inspection of the dendrogram. The leaves of the remaining trees represent the elements of the cluster. By including all data in hyperspheres of radius d_{cut} around each data object in the cluster, the clusters remain disjoint and are represented by simple shapes. Thereby a separation of $\varepsilon = d_{\text{cut}}$ is guaranteed.

Table 1. Providing the necessary information for explorative analysis.

Thus, a clustering algorithm may deliver a set $\mathcal{S} \subseteq \mathbb{N} \times X \times \mathbb{R}_+ \times \mathbb{R}_+$ where $(c, x, r, \varepsilon) \in \mathcal{S}$ denotes a hyperball $B(x, r)$ whose elements belong to cluster $\#c \in \mathbb{N}$ with a separation of at least 2ε (with $\varepsilon \leq r$, cf. Sect. 3). To avoid ambiguities when clusters are composed out of multiple shapes, we require $\forall (c, x, r, \varepsilon), (c', x', r', \varepsilon') \in \mathcal{S} : B(x, r) \cap B(x', r') \neq \emptyset \Rightarrow c = c'$ (that is, overlapping hyperspheres belong to the same cluster). This information gives a concise impression of the reported clusters and also suits an *assignment task*, where new data has to be associated with the identified clusters (or regarded as noise).

Although probably all clustering algorithms claim that they are (more or less) suited for explorative analysis and the assignment task, we can already see at this point that prominent clustering algorithms do not provide all of the above-mentioned information. For instance, k-Means provides no information about the size or separation of the clusters[2]; agglomerative hierarchical clustering delivers the points belonging to the clusters, but no extension of the cluster. Nevertheless, such algorithms *are* used for such tasks – and typically they are heuristically extended to provide the missing properties. But these extensions are usually not part of the original algorithm and often not subject to the evaluation procedure. Our proposition is that for explorative purposes a clustering algorithm must deliver the above-mentioned information and if a heuristic is involved in obtaining this information then it must also be part of the evaluation. While the validity measure proposed in the next section is not applicable to standard k-Means alone (because separation information is missing), it applies very well to numerous variants of k-Means with different heuristic extensions. For this paper, we canonically extend three common clustering algorithms in a straightforward fashion as it is often done in the literature (cf. Table 1).

We intentionally selected three clustering algorithms from different paradigms to show the wide applicability of the approach. These algorithms (k-Means, DBScan, AHC) are widely known and we refer to the literature for a detailed description. We restrict ourself to a brief summary of their outcomes. The k-Means algorithms delivers (an a priori specified number of) k prototypical data objects that represent a whole cluster. All data is associated with its closest prototype. The DBScan algorithm delivers for each cluster a subset of the dataset, the set of *core points*, which offer some minimal data density. The core points as well as all data within some epsilon-range constitute the cluster. The AHC algorithm returns a binary tree with the data at the leaves (dendrogram). Each inner node represent a union of its descendents at some specific distance, representing the distance between all data in the respective subtrees. By cutting the tree at some distance d, it resolves into several subtrees whose set of leaves represent the individual clusters.

[2] During k-Means iterations, all data is associated with the closest prototype, but this does not necessarily mean that once the final prototype locations are found *all new data* will also belong to one cluster – in this case there would be no 'void space' between the clusters.

3 Measuring Recovered Structure

Given that we have the desired information, we now consider how to measure how well the dataset-inherent structure has been recovered. In all definitions of a cluster, it must distinguish itself from its neighborhood by an increased data density. To verify the existence of a cluster, we have to approve a change in the data density: we expect the data density *within a cluster c* (interior density $\varrho_c^\bullet$) to be larger than the data density *outside the cluster c* (exterior density ϱ_c°). The larger the difference $\varrho_c^\bullet - \varrho_c^\circ$, the more distinctive is the discovered structure. If there is no difference at all or the exterior density is even larger than the interior density, the algorithm has made an error in its prediction (false positive). This gives us a very intuitive understanding of the *amount of structure correctly discovered by a clustering algorithm*: We define the S-measure (structure measure) as the sum of differences in the data densities as observed in a hold out test set:

$$S = \sum_{c \in C} (\varrho_c^\bullet - \varrho_c^\circ) \tag{1}$$

Since the recognized structure should be substantial, that is, generalize to the population from which the data was sampled, we measure this difference in the data density on a hold-out test set. While the evaluation of a classifier is done *per record* (correct class prediction), we evaluate a partition *per cluster* (correctly predicted existence of a cluster). Rather than using the data from the hold-out test set one by one to verify the results, it is used to get estimates of $\varrho^\bullet$ and ϱ°.

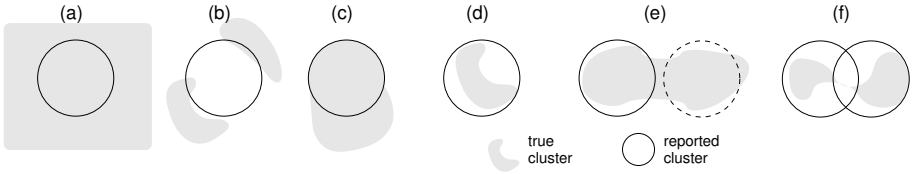


Fig. 1. Problems that occur with some clustering algorithms.

The data densities within and outside the cluster are derived from multiple estimates. In a first run, we use the test data as seeds for *probing points* at which we measure the data densities. In a second run, we estimate the data density at these locations and calculate interior and exterior densities. To *critically evaluate* the cluster c, we are particularly interested in (cf. Fig. 1) ...

... regions of high data density *outside* the cluster, because this may indicate false positives (Fig. 1a), poor shape recognition (Fig. 1b,1c) or cluster splitting (Fig. 1e). Therefore we use the data from the hold out test set as seeds for a set P_c° of exterior probing points. We thereby ignore regions without any test data, but the estimated density would be zero anyway.

... regions of low data density *inside* the cluster, because this indicates a poor shape recognition (Fig. 1b,1d) or a noise-bridge between clusters (Fig. 1f). So we do not use the test data as seed but draw random samples $x' \in B(x,r)$ from a cluster-defining hyperball $B(x,r)$ to obtain a set of interior probing points $P_c^\bullet$.

Finally, for each $x' \in P_c^\circ \cup P_c^\bullet$ we estimate the data density $\varrho_{x'}$. For a set of neighborhoods P_c^* (either P_c° or $P_c^\bullet$), let $\varrho(P_c^*) = \{\varrho_{x'} \mid x' \in P_c^*\}$. Then we define for each cluster c

$$\begin{aligned} \varrho_c^\bullet &\quad \text{as the } \textit{33rd percentile} \text{ of } \varrho(P_c^\bullet) \text{ and} \\ \varrho_c^\circ &\quad \text{as the } \textit{66th percentile} \text{ of } \varrho(P_c^\circ) \end{aligned}$$

The rationale for using different percentiles (rather than the median) is again the critical evaluation of the reported clusters as discussed above (bias to low densities within cluster and to high densities outside the cluster).

With this definition of $\varrho_c^\bullet$ and ϱ_c° the S-measure penalizes false positives because the summand becomes even negative in case there is no difference in the (average) data density within and outside the cluster (Fig. 1a). Due to the variance in the measurements, ϱ_c° (66th percentile) will yield higher values than $\varrho_c^\bullet$ (33rd percentile) and thus $\varrho_c^\bullet - \varrho_c^\circ < 0$. There is no penalty for false negatives (clusters that were not discovered) because we have no knowledge about such clusters[3]. However, since S is not bounded, other clustering algorithms that discover the missed clusters will outperform the former algorithm in terms of recovered structure. Note that we intentionally have removed any factor for the size of the cluster in order to not discriminate against small but substantive clusters (cf. [15]).

We use hyperballs $B(\cdot,\varepsilon)$ of the same size for estimating interior and exterior density of a shape (c,z,r,ε). The rationale behind this decision is that interior and exterior of a cluster should be analyzed at the same *resolution*: if there is only a small margin between two clusters, we have to look at the data carefully to distinguish the clusters. In this case we do not expect the clusters themselves to have gaps of similar size, because this would make the current partition questionable. Therefore, the interior density should be measured at the same resolution as the separation area. Similar arguments apply for the case of well separated clusters.

Implementation and Examples for Probing Point Selection

Fig. 2 illustrates how the probing points are determined in our experiments. For a given x the closest shape element (c,z,r,ε) is identified. If x lies within the hyperball $B(z,r)$, it belongs to the cluster c. We randomly determine an x' within $B(z,r-\varepsilon)$, such that the ε-neighborhood around x' is completely

[3] Penalizing false negatives would require their identification by, e.g., a clustering algorithm. But it is exactly the performance of these clustering algorithms that we seek to measure, so using them during the evaluation would be circular reasoning.

contained in $B(z, r)$ (case A). Otherwise, if there is no hyperball containing x, x does not belong to any cluster and we identify the closest separation area. We avoid border effects by shifting x towards z such that $N = B(x', \varepsilon)$ lies completely in the separation area (case B). In all other cases, x does neither belong to a cluster nor a separation area. Once the probing points have been determined, the densities at these spots are estmated during a second run over the test set.

in: set of shapes S, x from hold-out test set
out: returns probing point

1 $(c, z, r, \varepsilon) = \text{argmin}_{(c,z,r,\varepsilon)\in S} \|z - x\| - r$
2 if $\|z - x\| \leq r$ (case A: x lies within cluster c)
3 randomly sample $x' \in B(z, r - \varepsilon)$
4 return interior probing point x' for cluster c
5 endif
6 $(c, z, r, \varepsilon) = \text{argmin}_{(c,z,r,\varepsilon)\in S} \|z - x\| - r - 2\varepsilon$
7 if $\|z - x\| \leq r + 2\varepsilon$ (x in separation area)
8 let $x' = x$
9 if $\|z - x'\| \leq r + \varepsilon$ (case B: adjust x')
10 $x' = x' + \frac{r+\varepsilon}{\|z-x'\|}(z - x')$
11 return exterior probing point x' for cluster c
12 endif
13 return $\perp$ (case C: x is not used as a seed)

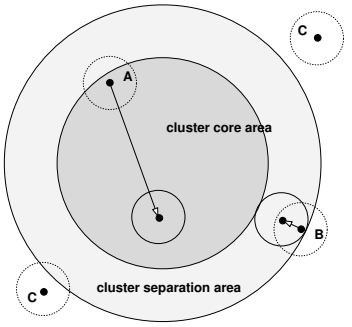


Fig. 2. Determining probing points for $\varrho^\bullet$ and ϱ°.

To illustrate the probing point selection, Fig. 3 shows the probing points for some exemplary cases. In the top row, three clusters identified by DBScan are shown. One can clearly see how the probing points for the separation area are aligned around the respective cluster. The space between the exterior probing points and the interior probing points within the cluster corresponds to the diameter of the neighborhood. One half of this space is covered by the neighborhoods of cluster core and separation area, resp. The first two examples from the second row illustrate the case for AHC, which is quite similar to the DBScan results. The second example shows the case where the single linkage algorithm has resulted in a joint cluster for the two data agglomerations at the bottom. Since the space between the three data agglomerations is sparsely populated, there are fewer exterior probing points in this case and their alignment is less clearly visible. (But introducing more probing points does not make sense – as there is no data in our hold out test set in this area, the estimated density will be zero.) Finally, the last example shows the case of k-Means. Two cases (b, d) illustrate the report of *false positives*, which will be penalized by the S-measure, because the exterior density is equally high or higher than the interior density.

Regarding runtime complexity, the complete evaluation consists of the selection of k probing points (which is $O(n)$ with a test set of size n), the density

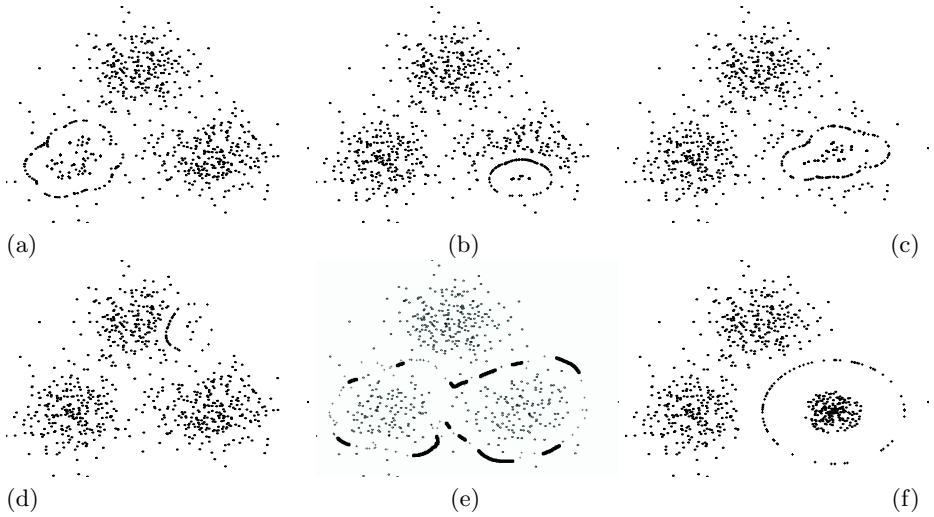


Fig. 3. Location of probing points for selected clusters found by DBScan (a-c), AHC (d, e) and k-Means (f). Data too far away from the selected cluster is shown unaltered. In case (e) artificial lines have been added to support the visibility of the contour.

estimation (which is $O(n \cdot k)$) and the determination of the respective percentiles (which is $O(k \log k)$ in a naïve implementation). For our datasets of moderate size we used $k \approx n$, but for large datasets it is sufficient to consider a random sample of fixed size s per cluster, that is, $k = s \cdot |C|$. Thus, the approach is also suited for large data sets.

4 Experimental Evaluation

Various artificial and real datasets have been used to test the S-measure. The tables show the averaged S-values over a 10-fold cross validation. Since the density estimation in the validation phase also requires sufficient data, the training and test sets were of the same size (50% each), that is, the training/test datasets consisted of subsets #1-5 (test)/#6-10 (train), #2-6 (test)/#1,7-10 (train) etc. Note that the S-values in the tables evaluate the clustering as a whole; a poor S-value does not necessarily mean that none of the clusters has been recovered successfully, but may also be caused by penalizing a number of false positives. A more detailed "per cluster"-inspection can be carried out by looking at the individual summands. For each algorithm, we tried a range of settings for the main parameter, but there was no exhaustive search for the best setting. As already mentioned in Sect. 1, a comparison with competitive measures is not

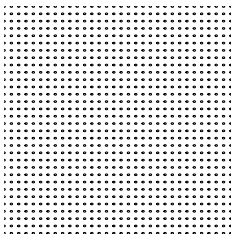
possible, because to the best of our knowledge none of the known measures can be used for all three algorithms.[4]

4.1 Typical Situations (Artificially Generated)

The design of the following artificially generated datasets is driven by the known weaknesses of clustering algorithms, such as the reporting of false positives (k-Means), difficulties with slowly changing data densities (DBScan), poor shape recognition (k-Means), sensitivity to outliers (AHC), etc. The datasets provoke such situations and our aim is not to compare the performance of the clustering algorithms in the first place (we know in advance which will perform best), but to see if the judgment of the S-measure corresponds to our expectations, that is, if it is sensitive enough to recognize the poor performance caused by the weaknesses of the respective algorithms.

No clusters present: Some clustering algorithms (e.g. k-Means) report a cluster in some area, where the data density is high, but there is no difference between the data density within and outside the reported cluster (cf. Fig. 1a). Such arbitrary selected clusters are incidental and the S-measure should penalize the reporting such clusters (cf. discussion in Sect. 3).

The dataset in Fig. 4 consists of a uniform grid without any internal structure. The table shows, for each algorithm, the selection of the main parameter, the number of obtained clusters (minimum/maximum number in 10-fold cross validation) and mean and standard deviation of the S-measure. Note that each clustering algorithms gets only half of this dataset (random sample) and therefore it looks not quite as regular to the clustering algorithm as the full dataset shown in Fig. 4. The omitted data introduces a light perturbance in the data density.



k-Means				DBScan (eps=0.2)				AHC			
k	$\|C\|$	μ	σ	MinPts	$\|C\|$	μ	σ	cut	$\|C\|$	μ	σ
2	1	-0.8	22.1	7	1	128.1	5.8	0.10	1-5	-57.2	44.5
3	1	-2.7	22.1	10	1-2	114.5	26.5	0.13	1	58.3	42.0
4	2-4	-81.4	60.5	13	1	97.8	28.8	0.16	1	114.3	7.8
5	3-4	-162.1	46.0	16	1	56.5	23.8	0.19	1	111.0	6.1
6	4	-182.3	38.7	19	1-3	-26.2	50.3	0.22	1	106.5	5.1

Fig. 4. 1024 records regularly distributed within $[-1, 1]^2$.

This dataset represents an uncomfortable situation for k-Means, since the k prototypes have to be placed somewhere. For small values of k, the prototypes

[4] Some approaches are applicable to all three algorithms, but measure the robustness or sensitivity, but not the overall quality of the clustering (see also introduction).

are united to one cluster (cf. Table 1), but still the union of 2-3 spheres poorly approximates a rectangle. On the test set, there are no differences in the average interior and exterior densities and the S-measure correctly indicates the poor performance. For small values of MinPts, DBScan correctly identifies a single cluster that contains (almost) all data (for MinPts=4 and 7 we obtain the true density of 128 (half of 1024 records within $[-1,1]^2$)). As MinPts is increased, more and more data is required to become a core point. This is achieved by chance in regions where most of the data from the full set is contained in the training set. But during validation against the test set, the reported cluster shape cannot be confirmed and the S-value decreases. For AHC, if the cut-off distance is chosen too small (0.1), the small perturbations induce up to 5 clusters that cannot be justified by the test set. For larger values all records are included in a single cluster, but AHC does not achieve the correct (maximal) S-measure of 128 because in hierarchical clustering there is no distinction between *core* and *border* points of a cluster (as in DBScan). Therefore, the density estimation near the border includes empty areas which damps the obtained interior densities.

(Invalid) Assumptions on Shape. Another problem with many clustering algorithms is that assumptions on the shape of the clusters do not hold in a given dataset. As a result, the models fit the true cluster poorly or multiple models are used to approximate a single cluster in the data. If such a case occurs, there is either (a) some part of the separation area with a data density as high as the interior area or (b) some part of the interior area with a data density as low as the separation area. The probing point selection method as well as the definition of $\varrho_c^\bullet$ and ϱ_c° as the 33rd and 66th percentiles were designed to detect such situations (cf. Sect. 3).

The dataset in Fig. 4 is closely related to this problem, because there is a single cluster of rectangular shape and the assumption on hyperspherical clusters does not hold for k-Means. We have seen already, that the poor performance of k-Means has been detected by the S-measure. Another example is shown in Fig. 5, consisting of three types of clusters: a Gaussian cluster, a box and a ring. Although the size of the ring-shaped cluster is recognized by the k-Means clusters quite well (the data-to-cluster association is correct), it receives small interior densities since most of the neighborhoods of interior probing points are empty. Again, AHC and DBScan detect the correct shapes and receive higher S-values.

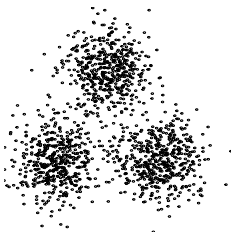
Separation and Noise. If clusters of the Gaussian type are close together, it becomes more difficult to sharply distinguish them. The examples in Fig. 6c and 7d consist of three clusters each, but with different separation. The larger the separation area (with a reduced data density) is, the better the cluster can be distinguished from its surrounding. The k-Means algorithm has an *overview* about the (relative) position of each cluster and can actively adjust the width of the separation area (cf. Table 1). Accordingly, the S-values of k-Means are very good for all cases but $k = 2$. It is also remarkable that the correct number of

k	k-Means $\|C\|$	μ	σ	DBScan (eps=0.12) MinPts	$\|C\|$	μ	σ	AHC cut	$\|C\|$	μ	σ
2	1	-158.6	146.0	4	3	1427.9	39.2	0.10	3-5	1314.6	155.9
3	3	638.4	12.0	7	3	1439.0	45.9	0.13	3-5	1243.1	144.5
4	3	598.2	58.0	10	3	1430.1	45.9	0.16	3-4	1100.4	92.7
5	3	638.0	42.7	13	3	1430.1	54.1	0.19	3	1066.9	19.0
6	3	573.7	63.4	16	3	1423.5	73.8	0.22	3	971.3	14.8

Fig. 5. Results on dataset *mixed* (500 records per cluster).

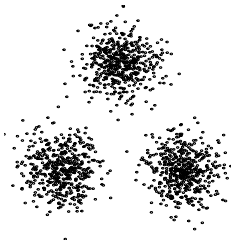
clusters is determined in all cases for $k = 3, 4, 5, 6$ due to the prototype-merging heuristic in Table 1. For $k > 3$ at least one cluster is composed out of two prototypes and this influences the shape of the cluster – it is no longer spherical. The largest S-measure, however, is obtained for $k = 3$ where each cluster is correctly approximated by a single hypersphere.

In contrast to k-Means, DBScan and AHC take a fine-grained look at the clusters and do not provide much information about their true separation. DBScan has difficulties with clusters that level out smoothly, because incidental data agglomerations can influence the shape of the cluster dramatically: As shown in Fig. 3c), although the position of the cluster has been determined correctly, we have a flat ellipsoid rather than a spherical shape. This shape was justified by the training set, but was not completely verified by the test set, which is why the difference $\varrho_c^\bullet - \varrho_c^\circ$ and therefore its contribution to the S-measure is rather small. AHC suffers from single linkage distance: As shown in figure 3d), the clusters get easily connected by chance due to single data points. A consistently high data density cannot be verified in the test set, which is a poor recovery is penalized by the S-measure. The bias of k-Means is best suited for these datasets and this is reflected by the S-values. Again, the expected performance of the algorithm is well recovered by the S-measure.



k	k-Means $\|C\|$	μ	σ	DBScan (eps=0.12) MinPts	$\|C\|$	μ	σ	AHC cut	$\|C\|$	μ	σ
2	1	-76.2	84.6	4	2-5	285.1	67.8	0.10	6-12	-474.2	361.8
3	3	748.7	51.3	7	3-4	369.1	73.7	0.13	2-5	43.3	123.0
4	3	541.1	65.8	10	3-4	406.7	76.0	0.16	1-2	46.0	142.7
5	3	600.1	78.8	13	3-4	296.2	118.4	0.19	1	140.1	8.7
6	3	567.1	51.4	16	2-3	137.0	155.8	0.22	1	143.3	8.6

Fig. 6. Results on dataset *three-3-7*.



k-Means				DBScan (eps=0.12)				AHC			
k	$\|C\|$	μ	σ	MinPts	$\|C\|$	μ	σ	cut	$\|C\|$	μ	σ
2	1	-48.8	113.7	7	3-4	355.8	67.8	0.10	4-8	-98.6	241.2
3	3	691.9	21.5	10	3	391.2	45.4	0.13	3-6	210.9	247.9
4	3	679.9	52.3	13	3	444.3	54.5	0.16	3-4	391.6	75.5
5	3	572.4	32.0	16	3	457.5	66.7	0.19	3-4	388.8	86.5
6	3	521.2	44.6	19	3-4	419.9	138.2	0.22	3	391.6	75.5

Fig. 7. Results on dataset *three-3-10*.

4.2 Real data

We show the results on three real datasets, namely the *iris* data, dimensions 4 and 8 of the *cloud* dataset, and an extract of the *wine* dataset (dimensions 2, 7, 10 and 11).

The overall best performance for iris (Fig. 8) is achieved by DBScan due to the good shape adaption. The extremely high variance in the AHC results again indicate the deficiencies of the single linkage method (for such a small dataset as *iris* the existence of a single record may be crucial for AHC). DBScan has an overall tendency to 2 clusters, whereas with k-Means the majority of runs ended with 3 clusters.

The 2-dimensional subset of the *cloud* dataset is comparable to the dataset in Fig. 5 with respect to the variability in the cluster shapes. Similar to earlier results, the k-Means results are inferior to DBScan and AHC, although the variances of DBScan and AHC are at least in some cases extremely high again (compared to k-Means), which is due to their high sensitivity to noise and the occassional induction of small local clusters that cannot be verified in the test set.

The 4-dimensional subset of the *wine* dataset is roughly comparable to the dataset in Fig. 6 in terms of cluster shape and noise level. Similar to earlier results, k-Means performs best here.

5 Conclusions

We have investigated into a validity measure that offers some imporant new properties. Firstly, it allows the direct comparison of results of clustering algorithms from different paradigms. Secondly, focussing on explorative analysis, we have proposed to validate the results of a clustering algorithm by *verifying* the obtained clusters on a hold-out test set: The larger the difference of interior and exterior density of the cluster, the more pronounced is the cluster. Thirdly, the measure penalizes the detection of false positives (reported clusters that are actually none).

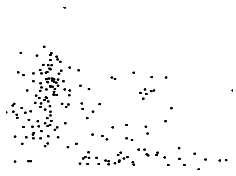
Experiments have been carried out on clustering algorithms from three different paradigms (partitional, hierarchical, density-based). The experiments on

IRIS

k-Means				DBScan (eps=0.5)				AHC			
k	$\|C\|$	μ	σ	MinPts	$\|C\|$	μ	σ	cut	$\|C\|$	μ	σ
2	2	11.6	3.3	4	2-3	68.7	13.3				
3	2-3	27.4	10.6	7	2	78.7	15.2	0.19	1-3	-9.6	366.5
4	2-3	17.8	13.7	10	2	76.8	10.3	0.22	2-4	-69.2	215.0
5	2-3	10.8	29.1	13	1-2	68.4	7.4	0.25	1-4	-41.5	21.1
6	2-4	37.2	32.2	16	1-2	60.3	6.3				

CLOUD (dimensions 4 and 8)

k-Means				DBScan (eps=0.16)				AHC			
k	$\|C\|$	μ	σ	MinPts	$\|C\|$	μ	σ	cut	$\|C\|$	μ	σ
2	2	677.9	66.0	4	2-3	1449.8	868.8	0.13	2-7	1817.5	848.1
3	3	181.3	42.4	7	1-4	1965.8	635.7	0.16	2-5	1203.6	771.2
4	3	-93.7	78.3	10	2-4	2320.1	190.0	0.19	1-3	1176.2	654.1
5	4	614.8	256.9	13	4	2392.2	220.1	0.22	1-2	957.5	516.3
6	5-6	814.1	202.9	16	3-5	2386.0	215.2	0.25	1	217.9	22.3



WINE (dimensions 2, 7, 10 and 11)

k-Means				DBScan (eps=0.75)				AHC			
k	$\|C\|$	μ	σ	MinPts	$\|C\|$	μ	σ	cut	$\|C\|$	μ	σ
2	2	16.0	3.5	4	3-5	-4.3	7.3	0.60	5-8	-19.2	13.1
3	2	16.1	2.6	7	2	8.8	2.4	0.65	4-6	-7.7	10.4
4	2	17.5	2.7	10	2	13.0	2.0	0.70	3-5	-4.5	11.5
5	2	15.4	4.6	13	2	14.4	2.9	0.75	3-5	-10.3	7.6
6	2	17.9	6.4	16	1-2	14.4	4.1	1.10	1	2.3	0.2

Fig. 8. Results on real datasets (IRIS, WINE, CLOUD).

various datasets have shown that it rewards strengths and penalizes weaknesses of clustering algorithms from different paradigms. Therefore we consider the S-measure as a promising new direction.

In some experiments the observed variance of the S-values was quite high. There are multiple reasons for this, but the most influential aspect is that for each of the three algorithms there are certain situations to which they respond very sensitive, thereby inducing large variance in the results. This is well-known in the literature and the experiments just underline that it remains an important research problem to reduce the sensitivity of the clustering algorithms such that they deliver clusters more robustly.

References

1. Kaufman, L., Rousseeuw, P.J.: Finding Groups in Data – An Introduction to Cluster Analysis. Wiley (1989)
2. Everitt, B.S.: Cluster Analysis. Wiley (1974)
3. Hartigan, J.A.: Clustering Algorithms. John Wiley & Sons (1975)

4. Jain, A.K., Dubes, R.C.: Algorithms for Clustering Data. Prentice-Hall (1988)
5. Rand, W.M.: Objective Criteria for the Evaluation of Clustering Methods. Journal of the American Statistical Association **66**(336) (1971) 846–850
6. Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York (1981)
7. Xie, X.L., Beni, G.: A Validity Measure for Fuzzy Clustering. IEEE Trans. on Pattern Analysis and Machine Intelligence **13**(8) (August 1991) 841–847
8. Fisher, D.H.: Knowledge Acquisition Via Incremental Conceptual Clustering. Machine Learning **2**(2) (1987) 139–172
9. Halkidi, M., Batistakis, Y., Vazirgiannis, M.: Clustering Validity Methods: Part I. **31**(2) (2002) 40–45
10. Möller, U., Radke, D.: A Cluster Validity Approach based on Nearest Neighbour Resampling. In: Proc. 18th Int. Conf. Pattern Recognition. (2006) 892–895
11. Levine, E., Domany, E.: Resampling Methods for Unsupervised Estimation of Cluster Validity. Neural Computation **13** (2001) 2573–2595
12. Borgelt, C., Kruse, R.: Finding the Number of Fuzzy Clusters by Resampling. In: IEEE Int. Conf. on Fuzzy Systems. (2006) 48–54
13. McQueen, J.B.: Some methods of classification and analysis of multivariate observations. In: Proc. of 5th Berkeley Symp. on Mathematical Statistics and Probability. (1967) 281–297
14. Ester, M., Kriegel, H.P., Sander, J., Xiaowei, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proc. of the 2nd ACM SIGKDD Int. Conf. on Knowl. Discovery and Data Mining, Portland, Oregon (1996) 226–331
15. Höppner, F.: Local pattern detection and clustering – are there substantive differences? In: Local Pattern Detection. Number 3539 in LNAI, Springer (2005) 53–70