

Subspace Clustering for Uncertain Data

Stephan Günnemann Hardy Kremer Thomas Seidl

Data management and exploration group
RWTH Aachen University, Germany
{günnemann, kremer, seidl}@cs.rwth-aachen.de

Abstract

Analyzing uncertain databases is a challenge in data mining research. Usually, data mining methods rely on precise values. In scenarios where uncertain values occur, e.g. due to noisy sensor readings, these algorithms cannot deliver high-quality patterns. Beside uncertainty, data mining methods face another problem: high dimensional data. For finding object groupings with locally relevant dimensions in this data, subspace clustering was introduced. For high dimensional uncertain data, however, deciding whether dimensions are relevant for a subspace cluster is even more challenging; thus, approaches for effective subspace clustering on uncertain databases are needed.

In this paper, we develop a method for subspace clustering for uncertain data that delivers high-quality patterns; the information provided by the individual distributions of objects is used in an effective manner. Because in uncertain scenarios a strict assignment of objects to single clusters is not appropriate, we enrich our model with the concept of membership degree. Subspace clustering for uncertain data is computationally expensive; thus, we propose an efficient algorithm. In thorough experiments we show the effectiveness and efficiency of our new subspace clustering method.

1 Introduction

Uncertainty is ubiquitous in application domains where data is processed [2]. Examples are sensor networks or data that is perturbed because of security reasons. For databases, two definitions of uncertainty are co-existent: tuple uncertainty [9] and attribute uncertainty [7]. Tuple uncertainty refers to whole database tuples, e.g. the probability of a tuple's validity; attribute uncertainty expresses imprecise information about attribute values of a tuple, i.e. no exact values are given. This paper deals with attribute uncertainty.

Uncertain data can be divided into two categories: data with inherent uncertainty and data with artificially induced uncertainty. Data of the first category is attributable to imprecise real world measurements, e.g. due to noisy sensor readings, transmission inaccuracies, or hardware failures. A concrete example are local sensing applications such as GPS, where coordinates become

invalid after very short time; exact tracking is infeasible. Another example are sensor networks, where sensor readings of different nodes can be inconsistent; combining these readings yields uncertainty. Artificial uncertainty is mostly based on two motivations: First, security reasons as anonymization of money transactions for subsequent data mining tasks. Second, deliberate information loss: data is sampled, aggregated, or rounded to satisfy memory or bandwidth constraints.

Uncertain objects are represented by probability density functions (*pdf*). In this paper, we introduce a novel approach to mine interesting object groupings from uncertain databases.

In general, data mining on uncertain databases is critical since it is prone to errors in the data, that is, some value deviation can induce completely different results. Furthermore, attribute values with a large error are less reliable for data mining purposes than ones with small errors. This has to be considered.

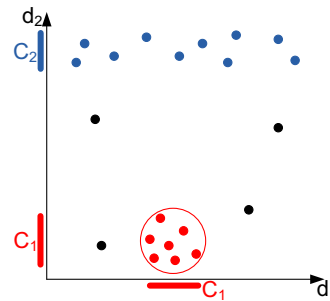


Figure 1: Example for subspace clustering

The data mining task of clustering is used for automatic grouping of similar objects, i.e. clusters. A cluster of similar objects is defined by a low distance between the corresponding objects; distances between objects of different clusters are high. In modern applications, e.g. data warehouses or gene expression analysis, objects are described by many dimensions. For higher dimensionalities, however, distances become increasingly similar [11]. This effect (curse of dimensionality) makes tra-

ditional clustering algorithms inappropriate for mining meaningful clusters. To mitigate the effect, dimensionality can be reduced by global techniques like PCA. The dimensionality is reduced, but the achievable results are questionable: First, it is often unclear how to interpret the new dimensions of the transformed reduced space. Second, and more severe, in high dimensional spaces dimensions can have locally varying relevance for different groups of objects. That is, different groups have different relevant dimensions. This local relevance cannot be detected through global analysis. Accordingly, recent research introduced subspace clustering, aiming at detecting locally relevant dimensions per cluster. Figure 1 shows a simple example; C_1 denotes a full space cluster, i.e. both dimensions are relevant, while C_2 represents a cluster in the projection to dimension d_2 . In d_1 , the object values of C_2 are highly scattered, making d_1 irrelevant for C_2 .

Existing work on subspace clustering is focused on precise databases. While there exist extensions of traditional full-space clustering algorithms to handle uncertainty, approaches for subspace clustering on uncertain data are non-existent to the best of our knowledge. The relevance of this research area can be pointed out by an example: Consider the data warehouse of a financial institution. Data warehouses usually contain noisy data and many dimensions. Thus, interesting patterns are mostly hidden in subspaces, i.e. subspace clustering is well fitted for mining this kind of data. To ensure security of costumers, however, sensitive data is perturbed before transferring it to the data warehouse, creating a need for subspace clustering on these uncertain datasets.

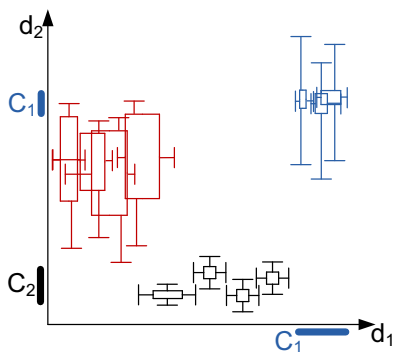


Figure 2: Example for uncertain data

For subspace clustering on uncertain databases, several problems need to be solved.

First, all clustering algorithms are based on object similarity. But how to measure similarity between uncertain objects? Simple solutions that remove uncertainty before using a standard clustering algorithm

waste important information; the clustering process could benefit from that information so that the clustering quality is increased.

Second, under uncertainty the decision whether a specific dimension is relevant for a subspace cluster is challenging. See Figure 2 for an illustration. In the plot, uncertain objects are represented by percentiles of their distribution. For each dimension, the 5th, 25th, 75th, and 95th percentiles are given. Comparing the groupings of blue and red objects, their individual uncertain objects have similar 5th and 95th percentiles, but the 25th and 75th percentiles differ significantly; i.e. for the red objects the values are distributed over a much higher range. Whether the groupings constitute clusters and what their relevant dimensions are cannot be answered easily. Intuitively, the blue objects correspond to a cluster with two relevant dimensions d_1, d_2 : in d_1 the individual object values have overall a low dispersion and in d_2 the inner percentiles (25% , 75%) show that the distributions are concentrated on a short range in a similar position. In contrast, the inner percentiles of the red objects have a high extent in both dimensions; no cluster should be found here. A third case is illustrated by the black objects: for the individual objects the dispersion is low for both dimensions. In d_1 , however, the objects themselves are highly scattered over the whole dimension, making d_1 irrelevant for cluster C_2 .

Third, the negative effects of high dimensionality, e.g. sparsity of the data space and alike distances between objects, are even more amplified by uncertainty.

And finally, the question of whether an object can belong to several clusters has to be resolved. Partitioning clustering approaches that find disjoint clusters are obviously out of place in an uncertain setting. For instance, an uncertain object can belong to different clusters with the same probability.

In this paper, we introduce a novel approach for subspace clustering of uncertain data that tackles the issues mentioned above. To improve the clustering results, maximal usage of available information is achieved. We present our approach in three stages; they differ in their degree of information usage. Furthermore, besides presenting a partitioning version of our approach that may be desired by some applications, we introduce novel non-partitioning variants. They are also designed to make maximal usage of the information provided. Especially, we augment clusters with membership degrees of their assigned objects. This improves the quality of clusterings and enables users to extract more useful information out of them. Since our proposed model is computationally expensive, we present an efficient solution that uses Apriori-based pruning and heuristic sampling while still providing high quality results.

Summarized, our contributions are

- Novel subspace clustering for uncertain data.
- Non-trivial handling of uncertainty, i.e. not just dissolving of uncertainty before clustering.
- A novel approach to non-partitioning clustering based on membership degrees.
- Efficient computation that uses Apriori-based pruning and heuristic sampling.

This paper is structured as follows: Section 2 discusses related work on clustering and uncertainty. Section 3 introduces our novel model for subspace clustering on uncertain data. In Section 4 we present an algorithmic solution for the model. Section 5 shows the results of the experimental evaluation and Section 6 concludes the paper.

2 Related Work

Several clustering paradigms have been proposed in the literature. Traditional full-space clustering models use all dimensions in the data space [10, 17]. Attributable to the effects of high dimensionality [11], i.e. distances between objects grow alike, these approaches are missing meaningful clusters. Therefore, for finding clusters with locally relevant dimensions, subspace clustering was introduced. An overview of different subspace clustering approaches can be found in [14, 21]. In [19], the differences between recent subspace clustering approaches are analyzed and thoroughly evaluated. To foster this analysis, the approaches are categorized into three classes: density-based approaches [4, 12] have shown to be inefficient, i.e. they do not scale to high dimensional data; clustering oriented approaches [1] are affected by noise resulting in low clustering quality; cell-based approaches [22, 24] have shown to be very efficient and subspace clusterings of high quality are generated. None of the above mentioned approaches, however, can handle uncertain data.

Uncertainty became ubiquitous in several application domains, i.e. similarity search, indexing, data integration, skyline queries, stream processing, or data mining. A thorough summary can be found in [2]. For clustering, recent publications extend existing traditional clustering approaches to support uncertain data: k-Means [17] was adapted in [6, 8, 20]; density-based clustering, i.e. DBSCAN [10], was extended for handling uncertainty in [15, 16, 23]. All of these algorithms, however, conduct full-space clustering. In this paper, we address subspace clustering for uncertain data.

3 Subspace Clustering for Uncertain Data

In the following Section 3.1 we introduce the basic notations and we present our general subspace cluster definition. First we describe how a single (best) subspace cluster is identified within the database. In Section 3.2-3.4 we instantiate this cluster selection method. We progressively increase the information usage obtained from the uncertain objects. In Section 3.5 we present how a overall clustering, i.e. a set of clusters, can be determined and how overlapping clusters are regarded.

3.1 Subspace cluster identification. In contrast to the traditional subspace clustering task, where each object is represented by a single high-dimensional vector, we have to consider uncertain objects. In our general approach each uncertain object is described by a probability density function (*pdf*). The *pdf* indicates the likelihood for an object to be located in a specified region. For a database DB with n uncertain objects in a d -dimensional space, we have n *pdfs* $p_i(x)$ with $x \in \mathbb{R}^d$ and for all $i = 1, \dots, n$ the following condition is fulfilled:

$$\int_{x \in \mathbb{R}^d} p_i(x) = 1$$

The terms object and *pdf* of an object are used interchangeably in this work. If we need to address the components of a vector x , we use the notation $p_i(x_1, \dots, x_d)$ instead of $p_i(x)$.

Subspace clustering analyzes the clustering of objects in projections of the dataspace. For precise data individual components of the object vectors are skipped to obtain the projections. In our case, however, we have to consider the *pdfs* of uncertain objects. Formally, the projection of an uncertain object p_i to a subspace S is defined by:

DEFINITION 3.1. *Projection of an uncertain object*
 Given a *pdf* p_i and a subspace $S \subseteq Dim = \{1, \dots, d\}$, the projection of p_i to S is the marginal distribution of p_i for S . The obtained *pdf* is called

$$p_i^S(x) \text{ with } x \in \mathbb{R}^{|S|}$$

For example and w.l.o.g. $S = \{1, \dots, s\}$, then

$$p_i^S(x) = p_i^S(x_1, \dots, x_s) = \int \cdots \int_{x_{s+1}, \dots, x_d \in \mathbb{R}} p_i(x_1, \dots, x_d)$$

i.e., we marginalize over the dimensions $\{s + 1, \dots, d\}$.

After defining the objects and their projections, we now introduce our basic subspace cluster definition. In general, a subspace cluster is a set of objects O along with a set of relevant dimensions S in which the

objects group together. In the remaining non-relevant dimensions the objects show no good grouping.

DEFINITION 3.2. *Subspace cluster*

Given a set of dimensions Dim and a database DB of uncertain objects within these dimensions, a subspace cluster C is defined by

$$C = (O, S) \text{ with } O \subseteq DB \text{ and } S \subseteq Dim$$

Our subspace cluster definition adapts the cell-based clustering paradigm [19, 22, 24]. As mentioned in the related work section, it has been shown that approaches from this category are very efficient and that they generate clusterings of high quality [19]. Efficiency is important because we increased the complexity by considering uncertainty. The basic idea of our cluster definition is to approximate clusters via hypercubes. In the relevant dimensions of the subspace cluster the extent of the hypercube is limited by a maximal width w , while in the non-relevant dimensions the extent is unlimited. For example, the black objects in Figure 2 show a high extent in dimension d_1 while in d_2 the grouping is compact. Keep in mind that the shape of a cluster is actually not restricted by the hypercube because we consider *pdfs*. Informally, the uncertain object is very likely to be located within this hypercube. If the support of a hypercube (e.g. the number of contained objects) is high enough ($\geq minSup$), then it is a cluster. The objects correspond to a good grouping because their attribute values vary to at most w within the relevant dimensions.

Algorithm 1 Identification of a subspace cluster

```

1: medoids = selectRandomMedoids(numMeds);
2: bestQuality =  $-\infty$ ;
3: bestMedoid = null; bestSubspace = null;
4: FOREACH( $m$  of medoids)
5:   FOREACH( $S \subseteq Dim$ )
6:     IF( $support(m, S) < minSup$ ) continue;
7:     IF( $quality(m, S) > bestQuality$ )
8:       bestQuality =  $quality(m, S)$ ;
9:       bestMedoid =  $m$ ; bestSubspace =  $S$ ;
10: generateCluster(bestMedoid, bestSubspace);
```

Our approach, presented in Algorithm 1 and similar to Monte Carlo methods [22, 24], tries to identify a single good subspace cluster within the database. This is accomplished by randomly selecting $numMeds$ uncertain objects out of the database that should represent the medoids of different clusters (line 1). For each of these medoids we identify the subspace for which the corresponding subspace cluster around the medoid results in the highest quality (line 5-9). In the end, we

will have identified the medoid that is the best among all randomly selected medoids and we generate the corresponding final subspace cluster (line 10).

To measure the quality of a subspace cluster we need a quality function that accounts for the dimensionality of the cluster and its support. For this purpose we adapt the quality function of [22].

DEFINITION 3.3. *Quality function*

Given a medoid m and a subspace S , let $support(m, S)$ be the support of the cluster C around m in the subspace S . The quality of C is defined by

$$quality(m, S) = support(m, S) \cdot (1/\beta)^{|S|}$$

where $\beta \in (0, 1]$ permits to trade off the support against the dimensionality of the cluster.

As can be seen, the support of the cluster around the medoid is the important aspect that we need to define for uncertain data. For traditional, i.e. certain, data the support can easily be determined by simply counting the number of objects that are near to the medoid. In [22], all objects whose Chebyshev distance within the relevant subspace is smaller than the threshold w are included. Formally, for a medoid m and a subspace S the support is calculated by

$$support_{certain}(m, S) = |\{o \in DB \mid d_{\infty}^S(m^S, o^S) \leq w\}|$$

where x^S is the projection of the point x to the corresponding subspace. The distance is defined by:

DEFINITION 3.4. *Chebyshev distance*

Given a subspace S and two $|S|$ -dimensional points $x, y \in \mathbb{R}^{|S|}$, the Chebyshev distance for this subspace is defined by

$$d_{\infty}^S(x, y) = \max_{i=1, \dots, |S|} \{|x_i - y_i|\}$$

For precise data a final cluster (line 10) can easily be generated: it is a binary decision whether an object exceeds the maximal distance.

The question in the following sections is how to determine the support of a subspace cluster for uncertain objects. We want to use as much information as possible to determine the quality of a cluster (based on the support, cf. Definition 3.3). Additionally, we have to consider which objects belong to the final cluster. For this purpose, it has to be discussed if a binary decision as in the certain case is useful.

3.2 Expectation based support. In a first naive approach each *pdf* is transformed to a single vector that represents the uncertain object and that can therefore

be used to calculate the support of a cluster. A common representative is the expected position $E(p_i, S) \in \mathbb{R}^{|S|}$ of the object p_i in the subspace S . This expectation value is calculated via

$$E(p_i, S) = \int_{x \in \mathbb{R}^{|S|}} x \cdot p_i^S(x) dx$$

With these representations we can directly use our distance function to measure if an object is near the medoid, i.e. if the distance is smaller than w . Thus, the support of a cluster around the medoid m in the subspace S is

$$\text{support}_{exp}(m, S) = |\{p_i \in DB \mid d_\infty^S(E(m, S), E(p_i, S)) \leq w\}|$$

As in the certain case, it is easy to decide which objects belong to the final cluster because it is a binary decision whether p_i contributes to the support or not.

Beside being simple, this first approach is not sufficient to cluster uncertain objects. Let us consider the uncertain objects in the left region of Figure 2. In dimension d_2 the expected positions of the objects do not scatter and thus a relevant dimension is assumed. The uncertain objects, however, show a high variance in this dimension and hence the dimension is not a good candidate for clustering; that is, non-relevant dimensions are wrongly identified as relevant by this first approach. At the same time it is possible to miss some relevant dimensions of a subspace cluster, especially if skewed or multi-modal distributions are present. Similar problems can occur if we use other expectation based methods.

Specifically for subspace clustering we have to solve these problems. If we did not, we would generate low quality groupings and we would identify the wrong relevant dimensions for our subspace cluster. Misinterpretation of the clustering result is possible.

3.3 Minimal probability based support. In our next method we retain the uncertain objects. We do not calculate an actual distance value but we calculate the probability that two objects are near to each other. Formally, the probability that the distance between two independent objects p_i and p_j in a subspace S is smaller than a maximal distance w is

$$(3.1) \quad P_{\leq w}(p_i, p_j, S) = \int_{\substack{x, y \in \mathbb{R}^{|S|} \\ d_\infty^S(x, y) \leq w}} p_i^S(x) \cdot p_j^S(y) dx dy$$

We have to integrate over all possible vectors whose distance to each other is small enough and multiply

the corresponding densities of the underlying *pdfs*. The support of a subspace cluster is now determined by all objects whose probability for this event is high enough. Thus, beside the *maximal* distance w we introduce the *minimal* probability *minProb* that an object has to fulfill to increase the support. The support of a cluster in subspace S around the medoid m is thus defined by

$$\text{support}_{minProb}(m, S) = |\{p_i \in DB \mid P_{\leq w}(m, p_i, S) \geq \text{minProb}\}|$$

Again we have a binary decision which objects belong to the cluster. In this new variant, however, the relevant dimensions of a cluster are better detected. For example, in Figure 2 the dimension d_2 is not selected as relevant for the red objects: due to the large scatter the probability for a small distance is not high enough.

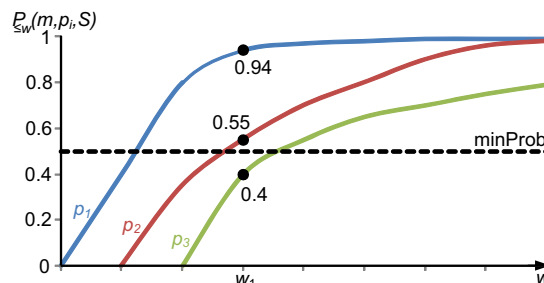


Figure 3: Probability distribution with varying w

Problems of this second method are presented in Figure 3. Given a fixed medoid m , the maximal distance w is plotted on the x-axis. The y-axis indicates the probability that an object is nearer to m than the current w . Three different objects are analyzed in the figure. Given the distance threshold w_1 and a minimal probability of $\text{minProb} = 0.5$, the object p_3 does not contribute to the support of the cluster. If many objects marginally miss the threshold, the calculated support will be underrated. The probability threshold is a hard bound for the objects to be included in the cluster.

Furthermore, the two other objects increase the support of the cluster because the probability is above the required threshold. Both objects, however, are considered equally important for the cluster, i.e. each object increases the support by 1, even if their probability values differ. In reality, the object p_1 is more likely clustered with the medoid than the object p_2 . Thus, the *minProb*-based method does not distinguish between clusters that with a high probability occur in the database and clusters that are improbable. This is related to the fact that dense clusters are usually preferred over sparse clusters. In Figure 4, both clusters would get the same support value with this method; the cluster

on the right should be preferred because the uncertainty within this clustering structure is very low.

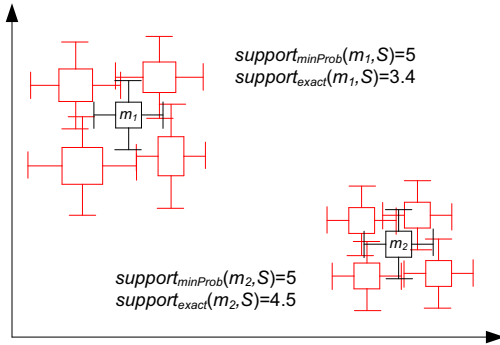


Figure 4: Minimal probability vs. exact probability

3.4 Exact probability based support. In this last approach we prefer clusters that appear with a high probability in the feature space. Until now each object in a cluster has the same weight, i.e. importance, for that cluster if the thresholds are exceeded (binary decision). In our following idea we use the probabilities in a more advanced way by weighting the objects according to their likelihood; accordingly, a possibility to calculate the support is $\sum_{p_i \in DB} P_{\leq w}(m, p_i, S)$ for given m and S . In contrast to $support_{minProb}$, objects with a higher probability increase the support and thus the quality of the cluster to a higher degree. At the same time, we do not exclude objects via a hard bound of a threshold; that is, we can separate clusters based on the magnitude of their supporting objects and not only based on their size. With this support definition, the cluster on the right in Figure 4 has a higher support than the other cluster. An interesting aspect is that the support can be any positive real number and is not restricted to natural numbers. Hence a more fine-grained indication of the support is possible.

The definition of the support, as proposed above, includes all objects, even if their probability to be near the medoid is very low. These objects might be undesirable for the user and might be removed from the cluster. To optionally prohibit objects with a probability near to zero, we include a probability constraint $\epsilon Prob$ that each object has to exceed. Thus, our final support definition is

$$support_{exact}(m, S) = \sum_{\substack{p_i \in DB \\ P_{\leq w}(m, p_i, S) \geq \epsilon Prob}} P_{\leq w}(m, p_i, S)$$

As before, we can calculate the support and the quality of a subspace cluster with this formula. Thus,

in Algorithm 1 we can determine the best medoid m and the best subspace S for which we generate the surrounding cluster. Our novel support criterion, however, no longer corresponds to a binary decision whether an object is included in the cluster or not. What is the final cluster presented to the user? Obviously, we could select all objects that account to the sum but this would contradict to our assumption that each object has a different weight for the support.

We extend the notion of subspace cluster to subspace cluster with membership degree. Each object within a cluster is associated to a membership degree that indicates the importance of the object for the cluster or the degree of this object belonging to the cluster.

DEFINITION 3.5. *Subspace cluster with membership degree*

Given a set of dimensions Dim and a database DB of uncertain objects within these dimensions, a subspace cluster C with membership degree is defined by

$$C = (O, S, deg) \text{ with } O \subseteq DB \text{ and } S \subseteq Dim$$

and $deg : O \rightarrow (0, 1]$ is a function that maps each object of the cluster to a specific membership degree.

Let m and S be the medoid and subspace calculated in Algorithm 1 based on the support function $support_{exact}$. The generated cluster is $C = (O, S, deg)$ with $O = \{p_i \in DB \mid P_{\leq w}(m, p_i, S) \geq \epsilon Prob\}$ and $deg(p) = P_{\leq w}(m, p, S)$.

Keep in mind that each membership degree is only with respect to one cluster; thus, we are not allowed to infer information in comparison to other clusters. Especially, the membership degree of one object p_i can exceed the value 1 summed up over several clusters. E.g., for two clusters C_1 and C_2 the degree of object p_i can be 0.6 and 0.7. High membership degrees of a single object for several clusters are possible because in subspace clustering objects can belong to several different clusters due to the individual subspaces.

3.5 Determination of overall clustering. In the previous sections we described how a single (best) cluster is found in an uncertain database. Our overall goal is to identify a set of clusters that describe the characteristics of the whole dataset; the obtained clusters should be discriminable in the sense that the same clustering structure is not identified twice. Thus, we cannot simply run Algorithm 1 several times because the same clusters may result. In this section we present three solutions for this problem; they are based on modifications of the medoid selection in Algorithm 1 (line 1).

In [22, 24] a simple **partitioning** approach is used to cope with the problem of recurring clusters. If a

subspace cluster is identified, the corresponding objects are removed from the database. On this smaller subset of objects we repeat the procedure of cluster generation (e.g. with Algorithm 1) until no more clusters are identified. The termination for this approach is guaranteed because the minimal required size/support of a cluster is fixed to a constant value and in each step one removes objects from the database. By this, each cluster represents a different set of objects. These sets are pairwise disjoint, i.e. a partitioning of the data is realized. Formally, given the set of clusters $\{C_i = (O_i, S_i) \mid i = 1, \dots, k\}$ generated so far in k runs of Algorithm 1, we draw the medoids in the $(k + 1)$ th run uniformly and randomly from

$$candMedoids_{k+1} = DB - \bigcup_{i=1}^k O_i$$

, i.e. the objects already covered by clusters are removed from the database and consequently are not used as medoids anymore. To ensure the partitioning, the clustered objects are also restricted to these subsets. Obviously for the first run we have $candMedoids_1 = DB$. According to [24] we select $numMeds = 2 \cdot |candMedoids_{k+1}|/minSup$ medoids from the current candidate set.

An uncertain object contributes to a cluster only in a specific amount. In a partitioning approach, already clustered objects cannot support another cluster. Thus, it seems reasonable to handle uncertain objects in a different way. Furthermore, in subspace clustering a non-partitioning approach can lead to better groupings because some objects might be clustered in several different dimensions. For these reasons, we introduce two novel non-partitioning clustering methods; they incorporate with our cluster selection procedure but are also transferable to other subspace cluster definitions.

In our **first non-partitioning** approach we do not remove the objects from the database, i.e. the clustered objects are not restricted to the above defined candidate subset; all objects can support a new cluster even if they are already contained in several other clusters. The set of possible medoids, however, is reduced in each step until this set is empty or no more clusters are identified in a run of Algorithm 1. Thus, the termination of the method is again guaranteed.

This first non-partitioning variant is appropriate for our *minProb*-based support definition because this method adds only objects with high probability values to the clusters. However, our exact support definition is able to include further objects also with smaller probabilities; all these objects are excluded from the candidate set for the medoid selection in following runs.

Thus, we introduce our **second variant of non-partitioning** subspace clustering for the support definition *support_{exact}*. This variant uses the additional knowledge about the membership degree of the clustered objects. The basic idea is that an object with a small membership degree for the already known clusters is possibly included in further clusters. Thus, it is reasonable to use this object as a medoid in later runs. However, the probability of this object to be selected is small compared to those objects that are not included in any clusters; the probability is high compared to objects that are in several clusters or to objects that are very likely in clusters. Consequently, we use a weighted sampling to determine our medoids in each run. Let $\{C_i = (O_i, S_i, deg_i) \mid i = 1, \dots, k\}$ be the set of subspace clusters with membership degree detected in the runs 1 to k . The medoids in run $k + 1$ of Algorithm 1 are drawn from DB based on a weighted sampling with the weights

$$\forall o \in DB : weight(o) = \begin{cases} 1, & \text{if } o \in (DB - \bigcup_{i=1}^k O_i) \\ \max\{0, 1 - \sum_{\substack{i=1 \\ o \in O_i}}^k deg_i(o)\}, & \text{else} \end{cases}$$

Initially each object has the weight 1 and hence we draw the medoids uniformly out of the whole database. The procedure terminates because the weights are continuously decreasing. Especially an uncertain object that has been used as a medoid for cluster generation can never be selected twice: its membership degree must be 1 and hence its weights in following runs are 0. The number of medoids $numMeds$ selected in the current run is determined by $2 \cdot \sum_{o \in DB} weight(o)/minSup$, i.e. the cardinality used before is replaced by the sum of weights. With this method we can generate overlapping clusters based on the membership degree of the uncertain objects.

3.6 Summary In the previous section we introduced our model for uncertain subspace clustering. We presented three versions, each with increasing complexity, for the calculation of the support of a cluster. We use information that is based on the probability density functions of the uncertain objects. Furthermore, we developed different methods to identify the overall clustering and we consider the overlap of clusters by our non-partitioning approaches. Table 1 gives an overview of the possible combinations and their corresponding names as used in the experiments.

4 Algorithmic properties

In the following section we discuss some algorithmic properties to ensure an efficient execution of our ap-

	Partitioning	Non-Partitioning
$support_{exp}$	Exp-P	Exp-NP
$support_{minProb}$	MinProb-P	MinProb-NP
$support_{exact}$	Exact-P	Exact-NP

Table 1: Overview of our algorithms

proach. The most important aspect in Algorithm 1 is the determination of the best subspace for a selected medoid, i.e. the subspace that yields the highest quality; in order to do so, we have to calculate the support of the current subspace cluster.

Anti-monotonicity property. Two of our techniques exploit an anti-monotonicity property that is valid for our subspace clustering model. Let O_S, O_T be the set of objects that supports the cluster around the medoid m in subspace S and T respectively; then for all subspaces $T \supseteq S$ we have $O_T \subseteq O_S$. Similar: $support(m, T) \leq support(m, S) \forall T \supseteq S$.

In this paragraph we prove that this property holds for all three variants of our support definition, starting with the expectation based variant. For a medoid m and subspaces $T \supseteq S$ it holds

$$\begin{aligned} O_T &= \{p_i \in DB \mid d_\infty^T(E(m, T), E(p_i, T)) \leq w\} \\ &\subseteq \{p_i \in DB \mid d_\infty^S(E(m, S), E(p_i, S)) \leq w\} = O_S \end{aligned}$$

Accordingly, $support_{exp}(m, T) = |O_T| \leq |O_S| = support_{exp}(m, S)$.

Proof. W.l.o.g. we assume $S = \{1, \dots, s\}$ and $T = S \cup \{s+1\}$. For two points $x, y \in \mathbb{R}^{|T|}$ and their projections to S , denoted as $x^S, y^S \in \mathbb{R}^{|S|}$, it holds that

$$\begin{aligned} d_\infty^S(x^S, y^S) &= \max_{i=1, \dots, s} \{|x_i - y_i|\} \\ &\leq \max_{i=1, \dots, s, s+1} \{|x_i - y_i|\} = d_\infty^T(x, y) \end{aligned}$$

and

$$(4.2) \quad d_\infty^T(x, y) \leq w \Rightarrow d_\infty^S(x^S, y^S) \leq w$$

Using the corresponding expectation values for x, y, x^S , and y^S concludes the proof. \square

The remaining two support definitions do not use the distance function directly but the probability $P_{\leq w}(p_i, p_j, S)$. We show the following property: $P_{\leq w}(p_i, p_j, T) \leq P_{\leq w}(p_i, p_j, S) \forall T \supseteq S$.

Proof. W.l.o.g. we assume $S = \{1, \dots, s\}$ and $T = S \cup \{s+1\}$. Then, for $x, y \in \mathbb{R}^{|T|}$ and their projections

to S $x^S, y^S \in \mathbb{R}^{|S|}$, it holds

$$\begin{aligned} P_{\leq w}(p_i, p_j, T) &= \int_{\substack{x, y \in \mathbb{R}^{|T|} \\ d_\infty^T(x, y) \leq w}} p_i^T(x) \cdot p_j^T(y) dx dy \\ &= \int_{\substack{x, y \in \mathbb{R}^{|T|} \\ \max_{i=1, \dots, s+1} \{|x_i - y_i|\} \leq w}} p_i^T(x) \cdot p_j^T(y) dx dy \\ &= \int_{\substack{x^S, y^S \in \mathbb{R}^{|S|} \\ \max_{i=1, \dots, s} \{|x_i - y_i|\} \leq w}} \int_{\substack{x_{s+1}, y_{s+1} \in \mathbb{R} \\ |x_{s+1} - y_{s+1}| \leq w}} p_i^T(x) \cdot p_j^T(y) dx dy \\ &\leq \int_{\substack{x^S, y^S \in \mathbb{R}^{|S|} \\ d_\infty^S(x^S, y^S) \leq w}} \int_{x_{s+1} \in \mathbb{R}} \int_{y_{s+1} \in \mathbb{R}} p_i^T(x) \cdot p_j^T(y) dx dy \\ &= \int_{\substack{x^S, y^S \in \mathbb{R}^{|S|} \\ d_\infty^S(x^S, y^S) \leq w}} p_i^S(x^S) \cdot p_j^S(y^S) dx^S dy^S \\ &= P_{\leq w}(p_i, p_j, S) \quad \square \end{aligned}$$

Now we can infer the condition $P_{\leq w}(p_i, p_j, T) \geq const \Rightarrow P_{\leq w}(p_i, p_j, S) \geq const$ for $T \supseteq S$ and therefore the objects $O_T = \{p_i \in DB \mid P_{\leq w}(m, p_i, T) \geq const\}$, supporting the cluster in the subspace T , are a subset of the objects in subspace S , i.e. $O_T \subseteq O_S = \{p_i \in DB \mid P_{\leq w}(m, p_i, S) \geq const\}$. Then it simply holds

$$\begin{aligned} support_{minProb}(m, T) &= |O_T| \\ &\leq |O_S| = support_{minProb}(m, S) \end{aligned}$$

for $const = minProb$. Furthermore, because of $P_{\leq w}(m, p_i, T) \leq P_{\leq w}(m, p_i, S)$ for $p_i \in O_T$, with $const = \varepsilon Prob$ we have that

$$support_{exact}(m, T) \leq support_{exact}(m, S)$$

Candidate generation and pruning. Based on the anti-monotonicity property we can use the Apriori principle known from association rule mining [3] to enhance the performance of our algorithm. We efficiently generate and prune cluster candidates of higher dimensionality based on the lower dimensional projections for one medoid.

We start with the 1-dimensional subspaces and calculate the support of the resulting clusters around a medoid. All subspaces that do not exceed the $minSup$ threshold are discarded (cf. Alg. 1). Afterwards we can obtain candidates for the $(x+1)$ -dimensional subspace clusters based on the retained x -dimensional ones. This procedure stops if the $minSup$ threshold is not exceeded by any of the generated candidates. With the Apriori method we prevent to analyze all exponential many subspaces for one medoid (line 5 of Algorithm 1) by early pruning of several subspaces.

Pruning based on quality estimation. While our first method prunes candidates w.r.t. a single medoid, our second approach makes use of already detected clusters, which could also be obtained from already processed medoids. We estimate the quality of all possible higher-dimensional projections of a cluster yet to come. We know the maximal support (because it can only decrease, cf. anti-monotonicity) and the maximal dimensionality (union of the retained dimensions) and hence an upper bound for the quality can be determined. This allows an additional pruning if the quality of the currently best found cluster, potentially based on another medoid, is already above this upper bound. In this case, we do not need to generate the high-dimensional projections.

Sampling method. In some scenarios it might be possible to compute the exact values for $P_{\leq w}(m, p_i, S)$ by solving the integral in Equation (3.1). Assuming any possible *pdf*, however, this is an infeasible task. In our implementation we use Monte-Carlo sampling to approximately calculate the integrals and hence we realize an efficient method.

Each *pdf* p_i is represented by m objects $I = \{o_i^1, o_i^2, \dots, o_i^m\}$ drawn from the distribution obtained by p_i . Given two *pdfs* p_i and p_j and their sampled objects I and J respectively, we denote the projections of the objects in I and J to the subspace S with I^S and J^S . Then we can approximate $P_{\leq w}(p_i, p_j, S)$ by

$$\frac{1}{|I^S| \cdot |J^S|} \cdot |\{(a, b) \in I^S \times J^S \mid d_\infty^S(a, b) \leq w\}|$$

, i.e. we compute the pairwise distances between the sampled objects and use the percentage of the objects pairs that have a smaller distance than w .

Summarized, we use three techniques to increase the efficiency of our algorithm. The Apriori principle is used to compute the set of uncertain objects that support the cluster in the current subspace. With the quality estimation we are able to prune further subspaces based on already detected clusters. Finally, the efficient sampling approach approximates the expensive integral computation.

5 Experiments

5.1 Setup. Because there exist no direct competitors in the domain of subspace clustering for uncertain data, we compare our approaches with UK-Means [6] and Proclus [1]. UK-Means is chosen as a representative for fullspace clustering on *uncertain* data while Proclus identifies *subspace* clusters on certain data. The number of clusters k , required for both algorithms, is set to the true number of clusters hidden in the data. The average dimensionality for Proclus is also obtained by

the hidden clusters. According to [24], we use $\beta = 0.25$ and we calculate the minimal support based on the database size, i.e. $minSup = \alpha \cdot |DB|$ with $\alpha = 0.07$. Furthermore, $w = 6$, $minProb = 0.5$, and $\epsilon Prob = 0.1$.

We generate synthetic data by adapting the methods from [1, 22, 24] to obtain overlapping clusters with uncertain objects. From each probability density function 10 samples are drawn. Unless stated otherwise, we generate data with 16 dimensions, 1000 objects, 10 clusters, an average cluster dimensionality of 25% from all dimensions, 15% overlapping objects, and 10% uniformly distributed noise objects. The *pdfs* are Gaussian distributions with a variance of 1% of the data range in the relevant dimensions and up to 50% of the data range in the non-relevant ones. To study the performance on real world datasets we use data that is available at the UCI archive [5] (pendigits, glass, breast cancer). In addition, we use features extracted from sequence data in [13] (shape). These datasets are also used in the evaluation study [19]. The data is enriched with uncertainty by replacing each object with a Gaussian *pdf* at the same position. Different variances are used in the experiments.

We measure the quality of the subspace clustering results via the F1 value, which is similar used in [18, 19]. We map each hidden cluster of a synthetic dataset to one identified cluster that matches the objects and the corresponding subspaces best. The harmonic mean of recall ('are all objects and dimensions detected?') and precision ('how accurate is the cluster detected?') is the F1 value of the cluster. The average of all clusters' F1 values is the quality value of the entire clustering. With this measure we account for the object groupings and the identified subspaces simultaneously. For real world data we use the class labels as indicators for the natural grouping of objects. No hidden subspaces are available in this case, and therefore the quality accounts only for the object groupings. The results of all algorithms are averaged over 5 runs.

All experiments were conducted on computers with Opteron 2.3GHz CPUs and Java6 64 bit.

5.2 Evaluation of different variants. In the following, we evaluate the different variants of our approach. That is, the support definitions and the differences between partitioning and non-partitioning clusterings based on these definitions.

For partitioning clustering, Figure 5 compares the quality of the three support definitions for different underlying distributions of the uncertain objects, i.e. we used Gaussian, Exponential (skewed), or Uniform distributions for each object. As it was expected from our model, the qualities of the resulting clusterings

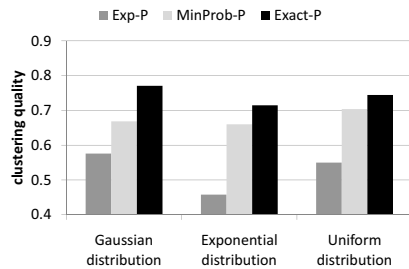


Figure 5: Comparison of support definitions (partitioning)

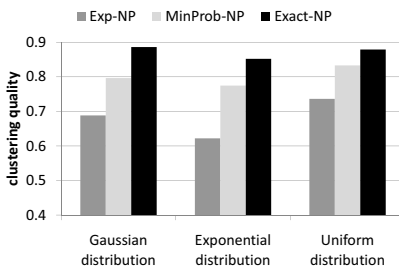


Figure 6: Comparison of support definitions (non-partitioning)

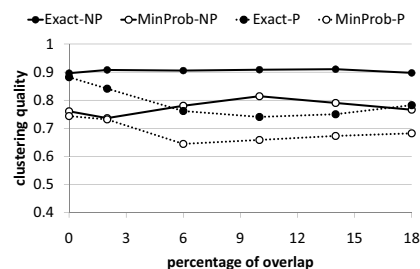


Figure 7: Varying overlap

improve with increasing usage of information. This is true for all three distributions. In comparison to the other two methods, Exp-P shows inferior performance and it is very sensitive to the chosen distribution; this is especially true for the skewed distribution. In Figure 6 the same experiment was repeated for the non-partitioning variants and confirms the conclusions drawn from Figure 5. As a result, we exclude Exp-P and Exp-NP in the rest of the experiments. A direct comparison of Figures 5 and 6 shows that overall the non-partitioning approaches generate clusterings of higher quality. This is also illustrated in Figure 7, where a varying overlap in the dataset, i.e. the percentage of objects that are assigned to more than one cluster, is analyzed. For no overlap (0%) the results of the partitioning (dotted lines) and non-partitioning (solid lines) variants are nearly the same. With increasing overlap both MinProb-NP and Exact-NP exceed their partitioning-based counterparts by more than 10%, and therefore we will only use the non-partitioning variants in the following experiments.

5.3 Comparison with competing approaches.

In the following we compare our methods to UK-Means and Proclus in several experiments on synthetic and real world datasets.

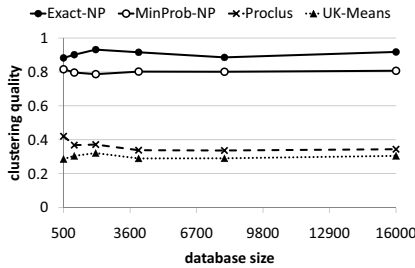
Database size is analyzed in Figure 8. The quality experiment shows that our approaches MinProb-NP and Exact-NP significantly outperform the competing algorithms by 45% and 55% respectively. For all algorithms the quality of clustering results are very stable, that is, the database size has no influence. Figure 8(b) compares the runtimes for the same experiment (Please note the logarithmic scale). The slopes of all 4 algorithms are similar, however, both UK-Means and Proclus are faster than our approaches, which is easily explainable: UK-Means only operates in the full dimensional space, i.e. no subspace projections are examined as in the other algorithms. Proclus uses certain values, but our approaches have to consider all the sample points of uncertain objects. Considering the bad quality

of UK-Means and Proclus, their high efficiency is of no benefit. If we compare MinProb-NP to Exact-NP, an interesting observation can be made: Exact-NP takes more time but delivers better results.

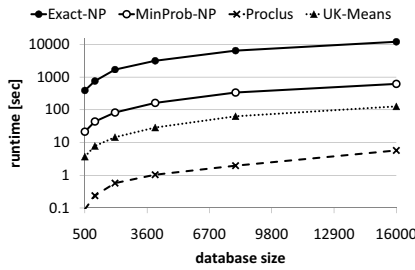
Varying *dimensionality of the data space* is evaluated in Figure 9. The resulting clustering qualities are similar to the experiment in Figure 8(a), i.e. the competing algorithms are clearly outperformed by both of our methods by more than 50%. The runtimes also show a similar behavior as for database scalability. MinProb-NP and Exact-NP have higher qualities, and thus their runtime performances are again unproblematic.

The influence of a varying *average dimensionality of subspace clusters* in a 16-dimensional dataspace is examined in Figure 10. As before, the clustering qualities of Exact-NP and MinProb-NP exceed the results of the competing approaches significantly. While the results of our algorithms are very stable, the results of UK-Means and Proclus improve with a higher average dimensionality. Since UK-Means is a full-space clustering method, this is reasonable; with increasing dimensionality, more near-fullspace clusters are hidden in the data. For Proclus the results are rather surprising: because it is a subspace clustering algorithm, better results for lower average dimensionalities were expected. In contrast, our approaches are not influenced by the average dimensionality; the correct clusters are detected on all settings. The runtime evaluation in Figure 10(b) shows that the runtimes of the UK-Means and Proclus are relatively stable, while the runtimes of our algorithms increase with higher dimensionalities. This can be explained by the Apriori-based (cf. Section 4) procedure: for higher average dimensionalities, much more subspace candidates for each medoid have to be generated; for lower dimensionalities subspaces are pruned much earlier.

The influence of a varying *percentage of noise* in the dataspace, by increasing the number of objects that do not belong to any cluster, is analyzed in Figure 11. As in the other experiments, our methods show a much higher quality than UK-Means or Proclus.

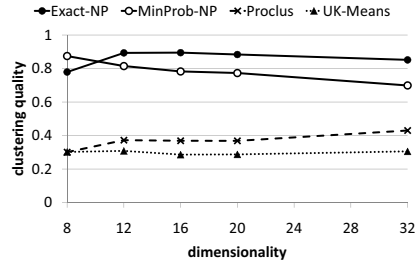


(a) quality

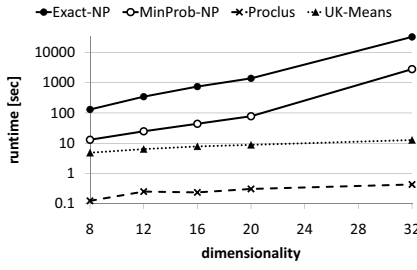


(b) efficiency

Figure 8: Varying size of the database

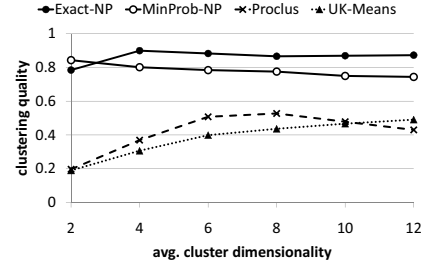


(a) quality

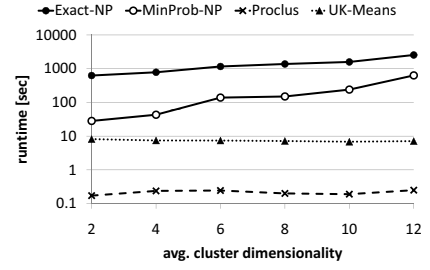


(b) efficiency

Figure 9: Varying dimensionality of the dataspace



(a) quality



(b) efficiency

Figure 10: Varying average dimensionality of clusters

While our methods are very robust to noise, UK-Means and Proclus respond very sensitive: the quality decreases constantly with an increasing noise ratio of the dataspace. In Figure 11(b) the runtimes of all algorithms increase with more noise. The performance of our methods are more prone to it because with an increasing noise ratio, more medoids are randomly selected and discarded afterwards.

Performance of our model on *real world data* is analyzed in Figure 12. We present the results for the 4 datasets pendigits, glass, breast cancer, and shape. Note that we have optimized parameters for each algorithm on each dataset; for our model, we chose the best result from all methods. Proclus is executed on the original precise data. Our model and UK-Means use the uncertain variants of the data; the variance of the underlying Gaussian distributions is set to 1%, 10%, and 25% of the data range.

The results on the pendigits dataset are presented in Figure 12(a). We can see that also on real world data our model outperforms the competing algorithms. Interestingly, the results of Proclus, operated on precise data, are worse than the results of the approaches that have to cope with uncertain information. For higher variances, however, we can see a decrease in quality; the clustering structure is obfuscated by the high variance and hence a detection of clusters is difficult.

On the remaining datasets similar results are obtained. Only the shape dataset (Figure 12(d)) is an exception: the quality of Proclus is slightly better than the

quality of UK-Means. Nevertheless, for every dataset the effectivity of our model is higher compared to the competing methods.

6 Conclusion

In this paper, we proposed a model that enables subspace clustering of uncertain data. We presented three variants that can be distinguished by the amount of information extracted from the probability density functions. Furthermore, we introduced a non-partitioning algorithm that is based on the concept of membership degree. We provided an efficient solution that uses Apriori-based pruning and heuristic sampling. Thorough experimental evaluation demonstrates that high-quality clusterings are generated and that competing algorithms are substantially outperformed.

Acknowledgment

This work has been supported by the UMIC Research Centre, RWTH Aachen University, Germany.

References

- [1] C. C. Aggarwal, J. L. Wolf, P. S. Yu, C. Procopiuc, and J. S. Park. Fast algorithms for projected clustering. In *SIGMOD*, pages 61–72, 1999.
- [2] C. C. Aggarwal and P. S. Yu. A survey of uncertain data algorithms and applications. *TKDE*, 21(5):609–623, 2009.

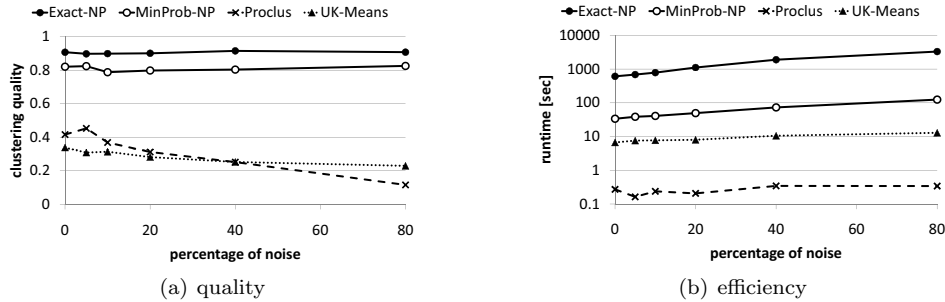


Figure 11: Varying percentage of noise in the database

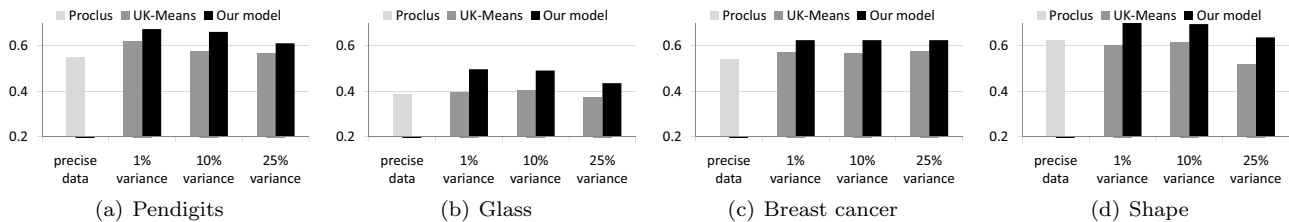


Figure 12: Clustering quality on real world data

- [3] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *VLDB*, pages 487–499, 1994.
- [4] I. Assent, R. Krieger, E. Müller, and T. Seidl. EDSC: Efficient density-based subspace clustering. In *CIKM*, pages 1093–1102, 2008.
- [5] A. Asuncion and D. Newman. UCI machine learning repository, <http://www.ics.uci.edu/~mllearn/mlrepository.html>, 2007.
- [6] M. Chau, R. Cheng, B. Kao, and J. Ng. Uncertain data mining: An example in clustering location data. In *PAKDD*, pages 199–204, 2006.
- [7] R. Cheng, D. V. Kalashnikov, and S. Prabhakar. Evaluating probabilistic queries over imprecise data. In *SIGMOD*, pages 551–562, 2003.
- [8] G. Cormode and A. McGregor. Approximation algorithms for clustering uncertain data. In *PODS*, pages 191–200, 2008.
- [9] N. N. Dalvi and D. Suciu. Efficient query evaluation on probabilistic databases. *VLDBJ*, 16(4):523–544, 2007.
- [10] M. Ester, H.-P. Kriegel, J. S, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, pages 226–231, 1996.
- [11] A. Hinneburg, C. C. Aggarwal, and D. A. Keim. What is the nearest neighbor in high dimensional spaces? In *VLDB*, pages 506–515, 2000.
- [12] K. Kailing, H.-P. Kriegel, and P. Kroeger. Density-connected subspace clustering for high-dimensional data. In *SDM*, pages 246–257, 2004.
- [13] E. Keogh, L. Wei, X. Xi, S.-H. Lee, and M. Vlachos. LB.Keogh supports exact indexing of shapes under rotation invariance with arbitrary representations and distance measures. In *VLDB*, pages 882–893, 2006.
- [14] H.-P. Kriegel, P. Kröger, and A. Zimek. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *TKDD*, 3(1):1–58, 2009.
- [15] H.-P. Kriegel and M. Pfeifle. Density-based clustering of uncertain data. In *KDD*, pages 672–677, 2005.
- [16] H.-P. Kriegel and M. Pfeifle. Hierarchical density-based clustering of uncertain data. In *ICDM*, pages 689–692, 2005.
- [17] S. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.
- [18] G. Moise, J. Sander, and M. Ester. P3C: A robust projected clustering algorithm. In *ICDM*, pages 414–425, 2006.
- [19] E. Müller, S. Günemann, I. Assent, and T. Seidl. Evaluating clustering in subspace projections of high dimensional data. In *VLDB*, pages 1270–1281, 2009.
- [20] W. K. Ngai, B. Kao, C. K. Chui, R. Cheng, M. Chau, and K. Y. Yip. Efficient clustering of uncertain data. In *ICDM*, pages 436–445, 2006.
- [21] L. Parsons, E. Haque, and H. Liu. Subspace clustering for high dimensional data: a review. *SIGKDD Explorations*, 6(1):90–105, 2004.
- [22] C. M. Procopiuc, M. Jones, P. K. Agarwal, and T. M. Murali. A monte carlo algorithm for fast projective clustering. In *SIGMOD*, pages 418–427, 2002.
- [23] H. Xu and G. Li. Density-based probabilistic clustering of uncertain data. *CSSE*, 4:474–477, 2008.
- [24] M. L. Yiu and N. Mamoulis. Frequent-pattern based iterative projected clustering. In *ICDM*, pages 689–692, 2003.