Visual Interactive Neighborhood Mining on High Dimensional Data

Emin Aksehirli[•] Bart Goethals[•] Emmanuel Müller[•○]

[•] University of Antwerp, Belgium
{firstname.lastname}@uantwerpen.be

[○] Karlsruhe Institute of Technology, Germany
emmanuel.mueller@kit.edu

ABSTRACT

Cluster analysis is widely used for explorative data analysis, however, it is not trivial to select the right method and optimal parameters. Moreover, not all clustering methods can work with raw or dirty data. In this paper, we introduce an interactive data exploration tool, VINeM, which combines interactive mining with unsupervised tools by exploiting an intuitive neighborhood-based visualization technique. Local neighborhood based visualization is useful not only for analyzing multiple (dis-)similarity measures but also for effectively discarding noise. VINeM works well with high dimensional data and can be used to find subspace clusters.

Keywords

subspace clustering, clustering, interactive data mining, high dimensional data, subspace selection, data visualization

1. INTRODUCTION

Explorative data analysis is an important tool that is used in both academic and industrial research areas. In recent years computational discoveries have become more affordable than the actual experiments, thanks to Moore's law. Therefore, researchers try to infer as much as they can from a limited set of data and refer to experiments only for conclusive results [14].

In most data exploration scenarios, little is known about the data, which means that the data is not annotated, and hence, not a good fit for supervised learning tasks, e.g. classification. Unsupervised methods are more suitable for data exploration tasks since they can work with limited initial knowledge about the data.

Unsupervised methods have their own challenges, foremost of which is the selection of the method that best fits the data at hand. This is not easy considering the thousands of available clustering methods in the literature [17]. Moreover, the selection of the algorithm is the first step of the process. Finding its optimal parameters is the next challenge which requires an understanding of the data.

To get a grasp of the data, one can rely on the most general and the least complicated techniques, such as descriptive statistics or visualizations. Descriptive statistics depend heavily on assumptions about the data. Not only determining them is not easy, but also, statistics can be misleading if those assumptions do not hold. Data visualization techniques, however, are easy to use by non-expert users and provide significant information. Therefore, data visualisation has always been an attractive research area [19]. Furthermore, using suitable visualizations, the user can be involved in the critical steps to improve the discovery process [11, 7]. On the other hand, existing tools either (1) are not capable of visualizing different views on the data [26], (2) do not integrate interactive and unsupervised mining methods [13, 23, 8], (3) are not aware of clusters that exist in subsets of the dimensions [27], (4) or are not designed to find the clusters [29].

Data from observations are rarely usable as they are. Before starting the actual analysis, analysts should deal with additional data cleaning steps, such as removing noise and dealing with missing values. Methods that are robust enough to cope with dirty or unstructured data can significantly shorten this tedious process.

In this paper, we introduce an interactive high-dimensional data analysis tool that can visualise different views on the data in a unifying framework while seamlessly integrates unsupervised and interactive mining tasks. In summary, the contributions of this paper are as follows:

- Neighborhood-based unifying data visualization,
- Micro cluster-based relevant dimension detection,
- GUI application that combines interactive data exploration with automated tools,
- Use case scenarios for various data exploration tasks.

In Section 2, we discuss the properties of high-dimensional data space in terms of clustering. Then, we introduce a neighborhood-based data representation that can cope with high-dimensional and noisy data while being easily visualizable (Section 3). In Section 4, we present our software tool that exploits this representation to make the data more available to the user, both for understanding and for wrangling. We explore two use-case scenarios in Section 5 and conclude in Section 6.

2. CLUSTERING HIGH DIMENSIONAL DATA

As a result of the advances in the data gathering and data storing technologies, we can associate many attributes with a single data object. Although more data may provide us with new insights, it may also hinder the discovery process by cluttering the interesting relations with redundant information. Furthermore, since the data objects become more and more alike with an increasing number of dimensions [6], the traditional definition of similarity becomes meaningless in high-dimensional data, and hence, clustering methods that depend on the similarity between objects fail to cope with high-dimensional data.

KDD 2015 Workshop on Interactive Data Exploration and Analytics (IDEA'15) August 10th, 2015, Sydney, Australia.

On the other hand, similarities of the objects according to a subset of attributes can still be meaningful. *For example, if a group of people have similar values for a specific bio-marker and they show a tendency for drug abuse, then other attributes such as the eye color, weight or sex are probably irrelevant.* Therefore, meaningful knowledge in high-dimensional data is often extracted as a tuple of similar objects and the attributes in which they are similar. Such information is called a *subspace cluster*.

Formally, a data object $\mathbf{o}$ is defined as a vector over a set of attributes $\mathscr{A}$. A dataset $\mathscr{DB}$ is a collection of data objects. A cluster is a set of objects $C \subset \mathscr{DB}$. A subspace cluster is defined as a tuple of an object set and an attribute set, $SC = (C, A)$ where $A \subseteq \mathscr{A}$.

High dimensionality poses a problem during the visualization as well. Feature selection techniques, such as PCA [24] and MDS [10], are used in the literature to represent data in a lower dimensional space. However, feature selection is done on the whole dataset and can therefore easily miss the subspace clusters [23]. Reducing the dimensions while keeping the cluster structures is also proposed [25, 30, 21], but requires a computationally expensive preprocessing which also depends on the assumptions on the data. Therefore, they are not suitable for interactive data exploration settings.

Scatter plot matrices show a 2D matrix of scatter plots for each pair of dimensions. Although they are useful to get a grasp of the relatively lower dimensional data, it gets harder to interpret them for the high-dimensional data because the number of charts is a combinatorial function of the number of dimensions. Parallel coordinates represents the relations of objects in different projections. They provide an interactive exploration environment but they cannot be used for non-univariate projections, i.e., they can represent only 1D projections [16].

3. NEIGHBORHOOD DATABASE

3.1 Object Neighborhoods

"Tell me who your friends are, and I will tell you who you are." The core concept of this famous phrase is successfully applied to many cases of data analysis. Neighborhoods, i.e. *friends*, of data objects provide robust assessment of the similarity. They can even be more accurate than the actual features in some cases [18]. Neighborhoods are a good estimator for determining class labels [12], or whether an object is an outlier [9]. As they are good at preserving the local relations, they are used to overcome the problems of high dimensionality [3].

DEFINITION 1 (NEIGHBORHOOD). *Neighborhood of an object* $\mathbf{o}$*, denoted by* $N(\mathbf{o})$*, is a set of objects that are similar to* $\mathbf{o}$.

DEFINITION 2 (ε-NEIGHBORHOOD). *The radius-based neighborhood,* ε*-Neighborhood, of an object* $\mathbf{o}$ *is defined as the set of all objects that are more similar to* $\mathbf{o}$ *than a certain scalar value. Formally, let* $\delta : \mathscr{DB}^2 \to \mathbb{R}_+$ *be a dissimilarity measure,* $\varepsilon \in \mathbb{R}_+$ *and* $\mathbf{o}, \mathbf{p} \in \mathscr{DB}$*,*

$$\varepsilon\text{-}N(\mathbf{o}) = \{\mathbf{p} | \delta(\mathbf{o}, \mathbf{p}) < \varepsilon\}$$

DEFINITION 3 (k-NEAREST NEIGHBORHOOD). *Let* $NN_k(\mathbf{o})$ *represent the* k^{th} *closest object to* $\mathbf{o}$*, the k-nearest neighborhood of* $\mathbf{o}$ *is:*

$$k\text{-}NN(\mathbf{o}) = \{\mathbf{p} | \delta(\mathbf{o}, \mathbf{p}) \leq \delta(\mathbf{o}, NN_k(\mathbf{o}))\}$$

Since the k-nearest neighborhood uses a relative similarity threshold, it is more robust for assessing the similarities in heterogeneous data than ε-neighborhood. Note that we use the concept of generic *neighborhood* (Definition 1) if the type of the neighborhood is irrelevant.

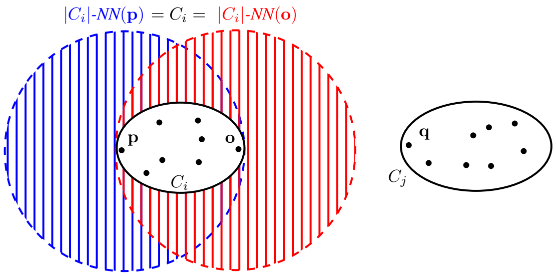


Figure 1: Case of separable clusters

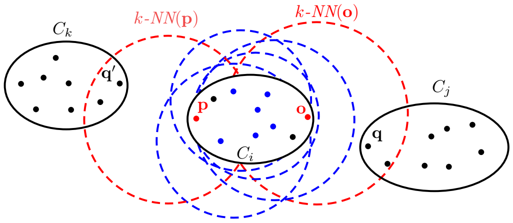


Figure 2: Case of non-separable clusters

DEFINITION 4 (NEIGHBORHOOD DATABASE). *The Neighborhood Database (*$\mathscr{ND}$*) is the collection of neighborhoods of all objects. Figure 3b shows the neighborhood database of the dataset in Figure 3a.*

Since *a cluster is defined as a group of similar data objects*, there is a clear connection between the cluster structures in the data and the neighborhoods. Figure 1 shows two clusters that are clearly separated, i.e., each object in a cluster is more similar to objects in the same cluster than to objects in the other clusters. For each object $\mathbf{o} \in C_i$, its k-nearest neighborhood with k equal to the size of its cluster, $|C_i|$-$NN(\mathbf{o})$, is equal to the cluster itself. Formally, if cluster C_i is clearly separated, then $|C_i|\text{-}NN(\mathbf{o}) = C_i, \forall \mathbf{o} \in C_i$. If the clusters are not separated, e.g., as shown in Figure 2, the most of the objects in C_i still include the whole cluster, provided that the neighborhoods are large enough.

Neighborhoods of the objects that are in the same cluster are either the same or share a large set of objects. Therefore, we can find

	A_1	A_2
$\mathbf{o}_1$	4000	3200
$\mathbf{o}_2$	5	6
$\mathbf{o}_3$	7	6
$\mathbf{o}_4$	3000	4000
$\mathbf{o}_5$	6	8
$\mathbf{o}_6$	5000	4200

(a) Data

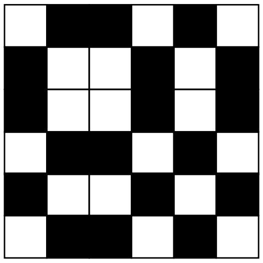
3-$NN(\mathbf{o}_1)$	$\{\mathbf{o}_1, \mathbf{o}_4, \mathbf{o}_6\}$
3-$NN(\mathbf{o}_2)$	$\{\mathbf{o}_2, \mathbf{o}_3, \mathbf{o}_5\}$
3-$NN(\mathbf{o}_3)$	$\{\mathbf{o}_2, \mathbf{o}_3, \mathbf{o}_5\}$
3-$NN(\mathbf{o}_4)$	$\{\mathbf{o}_1, \mathbf{o}_4, \mathbf{o}_6\}$
3-$NN(\mathbf{o}_5)$	$\{\mathbf{o}_2, \mathbf{o}_3, \mathbf{o}_5\}$
3-$NN(\mathbf{o}_6)$	$\{\mathbf{o}_1, \mathbf{o}_4, \mathbf{o}_6\}$

(b) Neighborhood database

Figure 3: An example dataset and its neighborhood database

	$\mathbf{o}_1$	$\mathbf{o}_2$	$\mathbf{o}_3$	$\mathbf{o}_4$	$\mathbf{o}_5$	$\mathbf{o}_6$
3-$NN(\mathbf{o}_1)$	1	0	0	1	0	1
3-$NN(\mathbf{o}_2)$	0	1	1	0	1	0
3-$NN(\mathbf{o}_3)$	0	1	1	0	1	0
3-$NN(\mathbf{o}_4)$	1	0	0	1	0	1
3-$NN(\mathbf{o}_5)$	0	1	1	0	1	0
3-$NN(\mathbf{o}_6)$	1	0	0	1	0	1

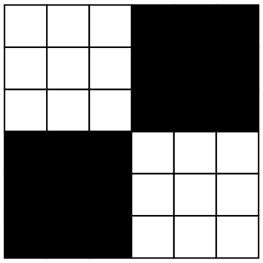
(a) Neighborhood matrix



(b) Visual Figure 4a

	$\mathbf{o}_2$	$\mathbf{o}_5$	$\mathbf{o}_3$	$\mathbf{o}_4$	$\mathbf{o}_1$	$\mathbf{o}_6$
3-$NN(\mathbf{o}_2)$	1	1	1	0	0	0
3-$NN(\mathbf{o}_5)$	1	1	1	0	0	0
3-$NN(\mathbf{o}_3)$	1	1	1	0	0	0
3-$NN(\mathbf{o}_4)$	0	0	0	1	1	1
3-$NN(\mathbf{o}_1)$	0	0	0	1	1	1
3-$NN(\mathbf{o}_6)$	0	0	0	1	1	1

(c) Neighborhood matrix of A_1



(d) Visual Figure 4c

Figure 4: Neighborhood matrices of the example dataset

the cluster structures by finding the repetitive patterns in the neighborhoods [2, 3]. Although the repetitive patterns can be detected by unsupervised tools, selecting the correct parameters may not be trivial. On the other hand, through proper representation, a human can spot repetitive patterns even in noisy settings and provide the intuition that would substantially improve the accuracy and the speed.

3.2 Representation

DEFINITION 5 (NEIGHBORHOOD MATRIX). *A neighborhood matrix is a binary adjacency matrix where columns and rows respectively represent objects and their neighborhoods. If the object in column j is in the neighborhood of object i, then the corresponding cell at i^{th} row and j^{th} column is 1, and 0 otherwise.*

Figure 3a shows a small data set of 6 data objects with 2 attributes. We can identify two cluster structures: $\{\mathbf{o}_1, \mathbf{o}_4, \mathbf{o}_6\}$ and $\{\mathbf{o}_2, \mathbf{o}_3, \mathbf{o}_5\}$. Figure 4a shows the neighborhood matrix for the 3 nearest neighborhoods of the dataset. While the repetitive neighborhoods are present, it is hard to see and mine. On the other hand, by using the order in the data, we can create a better representation that is easier to interpret and compute. For example, Figure 4c shows the same neighborhoods, where objects and neighborhoods are ordered according to A_1.

The neighborhood representation is compatible with the pixel-based visualization [19]. A *neighborhood matrix* can be represented graphically as an $n \times n$ square, where a pixel is white if the value of the corresponding cell is 1 and black otherwise. Figures 4b and 4d are the pixel-based representation of the neighborhood matrices in figures 4a and 4c, respectively. Cluster structures in the data are stunningly visible in Figure 4d.

Advantages of this representation include: (1) The representation is intuitive. (2) Individual objects are visible, i.e., they can be differentiated from their surroundings, regardless of the local density. (3) Since the repetitive structures stand out, cluster structures are visible. (4) It allows interactive mining since the individual objects are visible. (5) The relations in the representation are explainable because the original attributes are not distorted, e.g., the matrix is not translated as in PCA. (6) Creating the representation is not computationally expensive compared to dimensionality reduction techniques [21]. (7) Compared to graph representation, it provides a scalable and extendible solution, e.g., pixel sizes can be determined by the screen and the data size.

3.3 Mining the Neighborhoods

The evaluation of clusters is an important aspect when finding meaningful clusters. As we discussed in Section 3, objects in the same cluster repetitively co-occur in the neighborhoods of other objects. In this regard, we use the *support* of a cluster as a measure of repetitiveness of an object set in the neighborhood database. Therefore, clusters can be detected by finding the object sets that have a high *support*.

DEFINITION 6 (SUPPORT(σ)). *The support of an object set, i.e. cluster, is the number of neighborhoods in which the objects occur together. Formally,*

$$\sigma(C) = |\{\mathbf{o} \mid C \subseteq N(\mathbf{o}), N(\mathbf{o}) \in \mathscr{N}\mathscr{D}\}|$$

Support is a useful measure also for interactive mining. As shown in Section 4, supports of an object set in different neighborhood databases give an overview of the cluster formations. Moreover, this relation between the clusters and the support in the neighborhoods, can directly be mapped to frequent itemset mining [1], so that the whole literature of frequent itemset mining methods becomes available for cluster analysis [3].

One clear advantage of using neighborhood databases is their unifying representation. All of the $\mathscr{N}\mathscr{D}$s, regardless of the underlying (dis-)similarity measure, share the same properties, which means less context switches and more clarity for the user. For example, consider these measures: Euclidian distances on subsets of attributes, cosine similarity on all numeric attributes, a measure for the boolean attributes, two different measures for the same categorical attribute. $\mathscr{N}\mathscr{D}$s for each of them can be mined with the same set of tools because we are looking for the same kind of information: the objects that frequently co-occur in the neighborhoods.

As we discuss in Section 2, for high-dimensional data, local similarities are more meaningful than similarities in the whole data space. Therefore, bottom-up search is a widely used strategy to find subspace clusterings [22, 20]. In this regard, we propose to start the interactive analysis with the neighborhoods in 1 dimensional projections and find the object sets that repetitively co-occur in the neighborhoods in different projections.

Our tool, cf. Section 4, includes two methods for unsupervised mining of neighborhood databases, both of which satisfy the time constraints of an interactive setting. ***Sampling Miner*** mines the dataset for a subset of all the frequently co-occurring object sets. It is based on a Monte Carlo process, and as such, it mines a predetermined number of maximally large object sets that has more support than a certain threshold. These object sets are counterparts of *maximal frequent itemset*s [5]. Although it is not guaranteed to be complete, it produces satisfactory results [3]. ***Fast Miner*** exploits the orders in an attribute to find the complete cluster structure [2]. It starts by mining the individual $\mathscr{N}\mathscr{D}$s for a complete set of 1 dimensional clusters. Then, each cluster is refined further by checking whether any of its subsets are clusters in other dimensions. *Fast Miner* can only be used to mine neighborhoods of univariate measures.

Typically, subspace clusters overlap with each other both object-wise and attribute-wise. For example, a set of objects can form a cluster in dimensions 1 and 2 while another, but not necessarily disjoint, set of objects can form a cluster in dimensions 2 and 3. Therefore, the relation between the attributes should be investigated by assessing localities instead of the whole domain. We propose to use micro clusters to find the similarities between attributes. These kind of micro clusters are used to detect cluster structures [4, 15]. We mine a sample of micro clusters of size 5 in each attribute. The number of micro clusters shared by a pair of attributes becomes their similarity score. If a set of attributes are similar to each other, it can be worthwhile to investigate the neighborhoods of the combination of these dimensions. Even though the complete cluster structures are not visible in projections, micro clusters can effectively catch and aggregate local similarities, as we will show in Section 5.2.

Although sorted and non-sorted neighborhoods essentially contain the same information; in a visual setting, it is often easier to work with sorted neighborhoods. Unfortunately, sorted neighborhoods are possible only for univariate (1 dimensional) projections. For the interactive setting in VINeM, we approach the problem by partially sorting the $\mathscr{N}\mathscr{D}$ by focusing on a subset of the objects. The partial sorting is done by selecting an object as a reference and sorting the remaining objects and neighborhoods according to their similarities to the reference object. As we show in Section 5.2, partial sorting provides enough visual information for the identification of cluster structures.

Even if there are no cluster structures in the data, there is a minimum amount of repetition in the neighborhoods. In the case of uniformly distributed data, each individual object appears in exactly k neighborhoods, while the consecutive object sets of size $\frac{k}{2}$ appear in exactly $\frac{k}{2}$ neighborhoods. Neighborhoods for a uniform dataset are shown in Figure 5. This observation is used as a key indicator to discard the projections that do not have cluster structures.

Outliers and noise objects exist in relatively sparse areas and do not occur frequently in neighborhoods of objects [28]. Therefore, noise can be easily identified in the neighborhood database as the objects with low support.

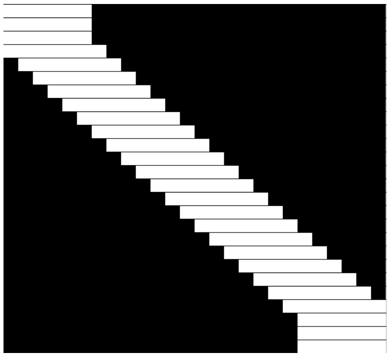


Figure 5: Neighborhoods of uniform data

If a set of objects form cluster structures in multiple attributes, then their values are similar to each other in these attributes. Therefore, we can use the dispersion of an object set in an attribute as an heuristic for cluster formation. Standard deviation and median absolute deviation (MAD) are widely used measures for dispersion. According to our observation, MAD gives more accurate results than the standard deviation.

4. VISUALIZATION

In this section we introduce VINeM, a data exploration tool that exploits the neighborhood database representation. An important design goal of VINeM is to provide a user-friendly interface that allows a user to easily blend unsupervised tools with instant decisions. Therefore, all of the features are designed for human interaction while they can be manipulated by the unsupervised tools.

VINeM is implemented in Java using Swing as GUI toolkit. It is accessible along with a detailed user manual on the supplementary website.[1]

4.1 Neighborhood

The main interaction area of VINeM is the neighborhood panel where the neighborhood matrix is shown, cf. Figure 6. As discussed in Section 3.2, columns represent the objects and rows represent the neighborhoods.

Currently, two kinds of neighborhoods are supported: ε-neighborhood and k-nearest neighborhood. The *kNN* representation is the default because of its robustness. VINeM starts by showing the *kNN* neighborhood databases for each of the individual attributes. Initially, the matrix and the objects are sorted according to the shown attribute. The order of the objects can be observed in the selection list where the ids, or the names, of the objects are shown, cf. 4 in Figure 7.

The matrix representation can be manipulated using the following means:

Dissimilarity measures for the projections. In the initial setting, there is one neighborhood matrix per attribute, each of which represents the neighborhoods according to Euclidian distances per dimension. Which $\mathscr{N}\mathscr{D}$ to show can be selected by using dropdown, cf. 10 in Figure 7.

Object and neighborhood order. The order of the objects is not necessarily dependent on the dissimilarity measure. For example, it is possible to sort the neighborhoods in attribute 1 according to attribute 2. On the other hand, the order and the measure are synchronized by default for convenience. The dimension that is used

[1] http://adrem.uantwerpen.be/vinem

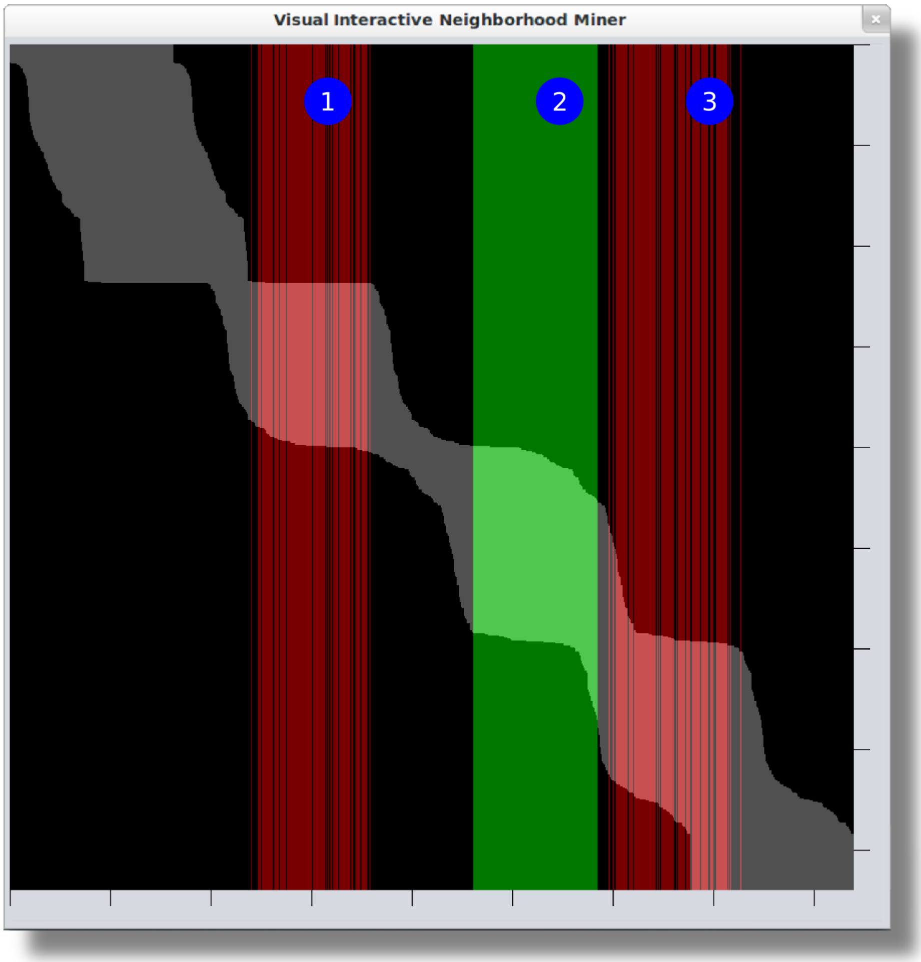


Figure 6: Neighborhood matrix

to order the objects can be selected using a slider, cf. 8 in Figure 7. If the synchronization is enabled, this will also change the dissimilarity measure.

The type of the neighborhood assessment, ε vs. kNN. The parameter for neighborhood size (k) can be set for *kNN*, while the parameter for neighborhood radius (ε) is required for the ε-Neighborhood. For convenience, possible radius sizes are pre-computed per similarity measure and user selects among them. The type of the neighborhood and its parameters can be selected on the control panel, cf. 7 in Figure 7.

4.2 Interactivity

Interactivity of VINeM starts with parameter selection. The interface and the representation are updated instantly after each parameter change, allowing the user to experiment with parameters on a responsive interface.

The selection of objects is the first step of the interactive analysis. There are two intuitive ways to select objects: (1) Dragging the left mouse button on the neighborhood matrix highlights the columns in green and selects the corresponding objects, cf. 2 in Figure 6. (2) Selecting objects from the selection list combo box by using standard list selection techniques, cf. 4 in Figure 7. The main selection method is dragging on the matrix, the list selection is for fine tuning. There are three modes for the selection on matrix, cf. 5 in Figure 7. In “Select” mode, only the objects under the mouse are selected. The “And” mode selects the objects if they are already selected while the “Or” mode adds the new selection to the already selected objects.

A separate frame, cf. Figure 8, shows the information about the selected objects, such as the size of the selection, their support and dispersion in other $\mathcal{N}\mathcal{D}$s. This information is used to determine the next $\mathcal{N}\mathcal{D}$ to investigate. Selection of the objects is an essential part of the data analysis in VINeM. During the analysis, the selection is iteratively refined by removing the objects that do not belong to the cluster in the additional attributes, which results in a selection of similar objects according to some attributes, i.e., a subspace cluster.

Selected objects can be identified as a cluster by clicking the button “Cluster Selected”, cf. 5 in Figure 7. The clusters that are identified either by the user or by an unsupervised tool are shown in the cluster list window, cf. Figure 9. Any of the identified clusters can be visualised on the neighborhood matrix along with the selected objects, so that the selection can be compared with the known clusters. Clusters are highlighted in red on the neighborhood matrix, cf. 1 and 3 in Figure 6. Clusters can be manipulated by adding or removing objects. If a substantial refinement is

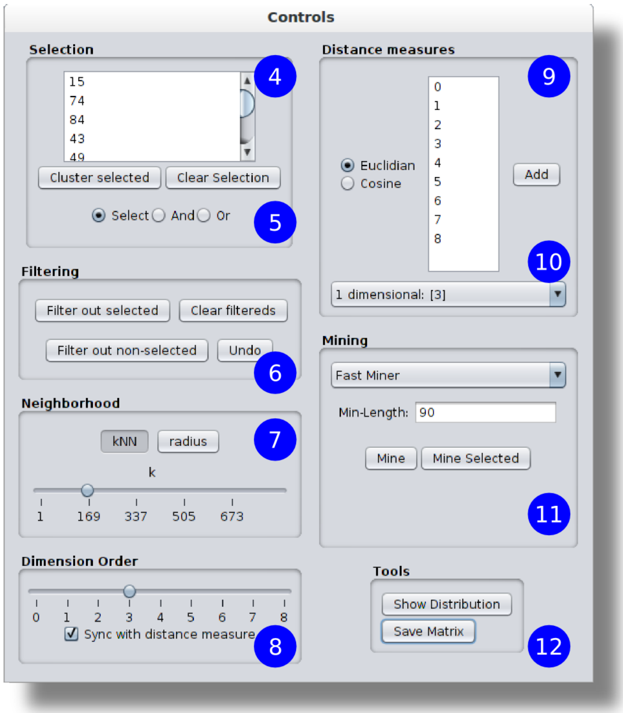
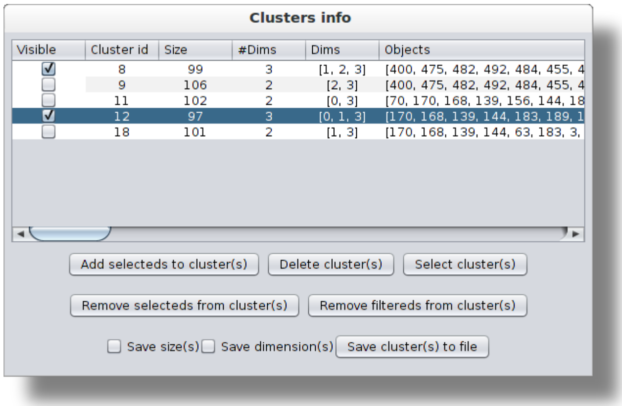


Figure 7: Control Panel

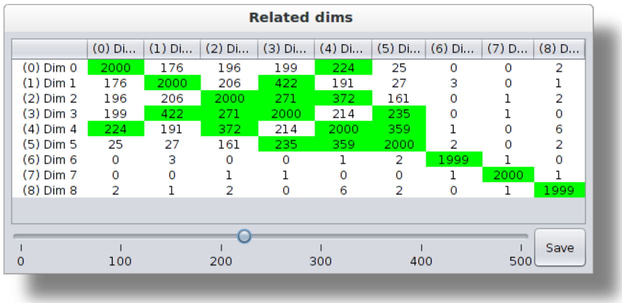


Clusters info

Visible	Cluster id	Size	#Dims	Dims	Objects
☑	8	99	3	[1, 2, 3]	[400, 475, 482, 492, 484, 455, 4
□	9	106	2	[2, 3]	[400, 475, 482, 492, 484, 455, 4
□	11	102	2	[0, 3]	[70, 170, 168, 139, 156, 144, 18
☑	12	97	3	[0, 1, 3]	[170, 168, 139, 144, 183, 189, 1
□	18	101	2	[1, 3]	[170, 168, 139, 144, 63, 183, 3,

Add selecteds to cluster(s)
Delete cluster(s)
Select cluster(s)
Remove selecteds from cluster(s)
Remove filtereds from cluster(s)
□ Save size(s)
□ Save dimension(s)
Save cluster(s) to file

Figure 9: List of detected clusters



Related dims

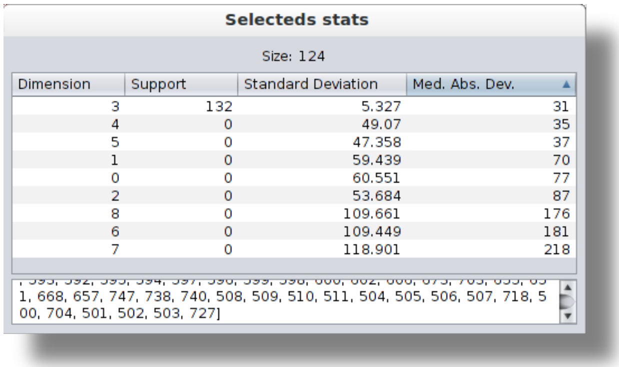
	(0) Di...	(1) Di...	(2) Di...	(3) Di...	(4) Di...	(5) Di...	(6) Di...	(7) D...	(8) D...
(0) Dim 0	2000	176	196	199	224	25	0	0	2
(1) Dim 1	176	2000	206	422	191	27	3	0	1
(2) Dim 2	196	206	2000	271	372	161	0	1	2
(3) Dim 3	199	422	271	2000	214	235	0	1	0
(4) Dim 4	224	191	372	214	2000	359	1	0	6
(5) Dim 5	25	27	161	235	359	2000	2	0	2
(6) Dim 6	0	3	0	0	1	2	1999	1	0
(7) Dim 7	0	0	1	1	0	0	1	2000	1
(8) Dim 8	2	1	2	0	6	2	0	1	1999

Figure 10: Relevancy score of dimensions

required, they can always be converted into selections. Any subset of clusters can be saved to a file for further study.

A set of objects can be filtered out, i.e. removed, from the view to focus on the objects under consideration, cf. ⑥ in Figure 7. Note that filtering removes the objects and their neighborhoods just from the view, i.e., the neighborhoods are not re-computed. Therefore, it is fast and it does not change the data on the fly. Filtering comes in handy while performing a cross measure analysis since it hides the noise caused by the projection.

Neighborhood databases for new measures can be added by just a few mouse clicks using the GUI. To add an $\mathcal{ND}$, a similarity measure is selected along with the dimensions on which it will be applied, cf. ⑨ in Figure 7. After adding the new $\mathcal{ND}$, it can be viewed by selecting from the drop-down menu, cf. ⑩ in Figure 7.



Selecteds stats

Size: 124

Dimension	Support	Standard Deviation	Med. Abs. Dev.
3	132	5.327	31
4	0	49.07	35
5	0	47.358	37
1	0	59.439	70
0	0	60.551	77
2	0	53.684	87
8	0	109.661	176
6	0	109.449	181
7	0	118.901	218

1, 668, 657, 747, 738, 740, 508, 509, 510, 511, 504, 505, 506, 507, 718, 5
00, 704, 501, 502, 503, 727]

Figure 8: Basic information about the selection

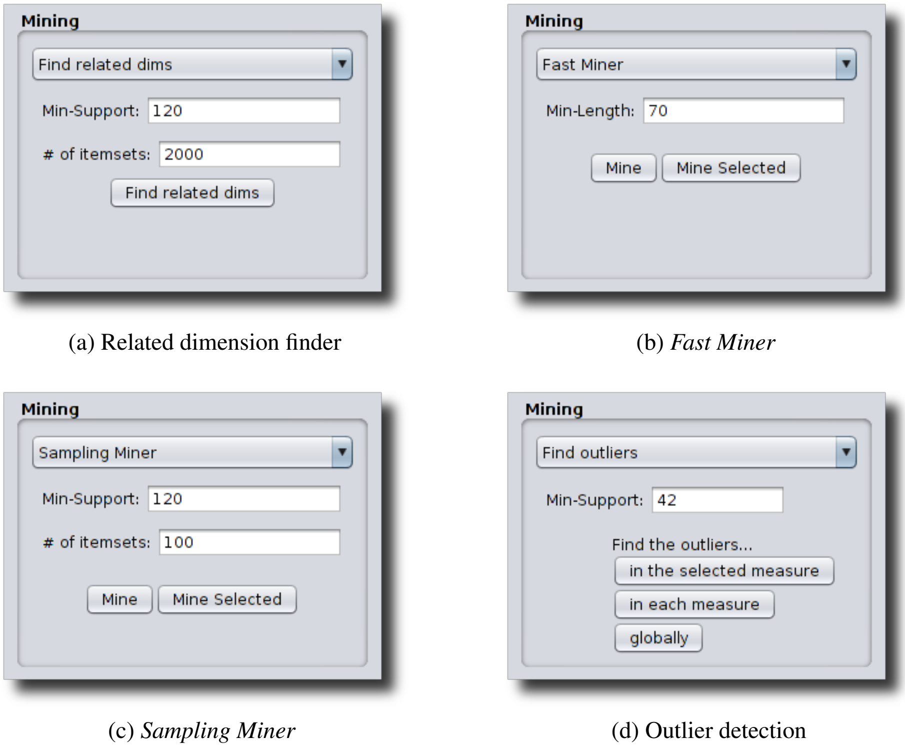
$\mathcal{ND}$s of non-univariate similarity measures can be partially sorted. Right clicking on a column selects the corresponding object as the reference and sorts the other objects according to their similarities to this object, cf. Section 3.3.

4.3 Unsupervised Tools

VINeM is bundled with unsupervised tools to support the user during the data analysis, cf. ⑪ in Figure 7. They are seamlessly integrated with the interactive tools where applicable.

Related dimension finder mines the $\mathcal{ND}$ of each individual attribute for micro clusters as explained in Section 3.3. Parameters are updated with suggested values when the $\mathcal{ND}$ is modified, cf. Figure 11a. Relevancy scores between pairs of attributes, i.e. the number of shared micro clusters, are shown as a table. A thresholding slider is provided for visual assistance on detecting the highly related dimensions, cf. Figure 10.

The two miners that are introduced in the Section 3.3, namely *Fast Miner* and *Sampling Miner*, can be used for unsupervised mining. While *Fast Miner* requires only one parameter which is the minimum length of a cluster, cf. Figure 11b; *Sampling Miner* requires two parameters: required minimum support of an object set to be identified as a cluster and number of samples, cf. Figure 11c. Both of the miners can be run either on the whole dataset or only on the selected objects, so that the objects that are not under consideration can be left out. The suggested values for the parameters are



Min-Support: 120
# of itemsets: 2000

(a) Related dimension finder

Min-Length: 70

(b) *Fast Miner*

Min-Support: 120
# of itemsets: 100

(c) *Sampling Miner*

Min-Support: 42

(d) Outlier detection

Figure 11: Parameters for miners

provided for a quick start. After the run, all of the found clusters are added to the cluster list for further investigation.

Automated noise/outlier detection is also provided as a miner. The objects that have a support less than the threshold are identified as noise. The support of objects can be evaluated (1) *in the selected measure*, (2) *in each measure*, or (3) in *all measures*, cf. Figure 11d. Noise objects are added as a cluster. It is up to the user to filter them out of the view.

5. APPLICATION

5.1 Finding clusters in subspaces

Figure 12 shows the steps of an interactive mining process. We start by finding a neighborhood size that makes the clusters visible in one of the $\mathcal{N}\mathcal{D}$s. In this example, we can see the cluster structures in Dim(ension) 1 for a certain k value (Step 1). We identify a one dimensional cluster by selecting its objects (Step 2). The next step is to check other dimensions to find out whether any subsets forms a cluster there. When we switch to the $\mathcal{N}\mathcal{D}$ of the Dim 2, objects in the cluster are still marked with red (Step 3).

Since we are looking for subsets of our cluster, we remove the remaining objects by filtering them out (Step 4). We decide that the objects do not form cluster structures in Dims 2 and 3 because their filtered $\mathcal{N}\mathcal{D}$s look like the $\mathcal{N}\mathcal{D}$ of a uniform distribution, which is shown in Figure 5 (Steps 4 and 5). Note that, although there are some repetitive neighborhoods in the filtered $\mathcal{N}\mathcal{D}$ of Dim 2, they are too small to be identified as clusters. Filtered $\mathcal{N}\mathcal{D}$ of Dim 4 looks interesting (Step 6), there is a large set of objects that co-occur in neighborhoods. We select these objects and identify them as a cluster in Dimensions 1 and 4.

5.2 Finding relevant dimensions

Figure 13 shows the steps for an exploratory analysis on a 10 dimensional dataset. In this dataset, the clusters are not immediately visible in individual dimension projections. Step 1 of the figure shows 4 of the $\mathcal{N}\mathcal{D}$s. There are signs of structures in Dims 0 and 2, which can probably be enhanced by modifying the neighborhoods size, while Dims 4 and 6 look like they lack any kind of structure. It is possible that the whole cluster structures are not visible in one dimensional projections. Running the *related dimension finder* gives us the scores for each pair of dimensions. Then, we can interactively decide which of the dimensions are related to each other by examining the high scores (Step 2). It looks like the two sets of dimensions {0, 1, 2, 3} and {4, 5, 6, 7} share common structures.

With a few clicks, we add a new dissimilarity measure that assess the neighborhoods in the combination of dimensions 4, 5, 6, and 7 (Step 3). Although it is almost impossible to spot the structures in the $\mathcal{N}\mathcal{D}$ of the combined dimensions (Step 4), sorting the neighborhoods according to a reference object, i.e. partial sorting in Sec. 3.3, helps us to see the structure in the data (Step 5). The blob on the top left of the sorted $\mathcal{N}\mathcal{D}$ represents a large set of objects that co-occur in the neighborhoods. We select these objects for further analysis (Step 6). Note that, since this sorting is according to only one object, some objects that are not inside the cluster can be in the blob by mistake, and they can be removed by using the partial orderings of the objects in the blob. We investigate further by sorting the $\mathcal{N}\mathcal{D}$ according to an object that is not in the cluster candidate (Step 7), and now we can see some other structures in the $\mathcal{N}\mathcal{D}$. We identify the selection as a cluster (Step 8), and then we continue our analysis with selecting a new cluster candidate. We can still modify the neighborhood parameters to improve the view. For example, cluster structures are more visible in Step 8 compared to Step 7, because of using ε-neighborhoods instead of *kNN*.

6. CONCLUSION

While visualisation and interactiveness are very important for exploratory data analysis, available tools fall short to satisfy all of its challenges. In this paper we show that local neighborhoods provide the means for both intuitive visualization and interactive mining of subspace clusters. We introduce VINeM, a platform-independent, visual and interactive data analysis tool that exploits the intuitive neighborhood-based representation to seamlessly combine a user friendly interactive interface with the unsupervised tools. We introduce a micro cluster based tool to find relevant dimensions and we provide example scenarios of exploratory data analysis to show the usefulness of our application.

7. REFERENCES

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, volume 1215, pages 487–499, 1994.

[2] E. Aksehirli, B. Goethals, and E. Müller. Efficient cluster detection by ordered neighborhoods. In *2015 17th International Conference on Big Data Analytics and Knowledge Discovery (DaWaK) (accepted)*, 2015.

[3] E. Aksehirli, B. Goethals, E. Müller, and J. Vreeken. Cartification: A neighborhood preserving transformation for mining high dimensional data. In *2013 IEEE 13th International Conference on Data Mining (ICDM)*, pages 937–942, Dec. 2013.

[4] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning a mahalanobis metric from equivalence constraints. *Journal of Machine Learning Research*, 6(6):937–965, 2005.

[5] R. J. Bayardo, Jr. Efficiently mining long patterns from databases. *SIGMOD Rec.*, 27(2):85–93, June 1998.

[6] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When is "nearest neighbor" meaningful? In *Database Theory–ICDT'99*, pages 217–235. Springer, 1999.

[7] M. Boley, M. Mampaey, B. Kang, P. Tokmakov, and S. Wrobel. One click mining: Interactive local pattern discovery through implicit preference and performance learning. In *Proceedings of the ACM SIGKDD*, IDEA '13, pages 27–35, New York, NY, USA, 2013. ACM.

[8] S. Bremm, T. von Landesberger, M. He\s s, T. Schreck, P. Weil, and K. Hamacherk. Interactive visual comparison of multiple trees. In *Visual Analytics Science and Technology (VAST), 2011 IEEE Conference on*, pages 31–40. IEEE, 2011.

[9] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. Lof: identifying density-based local outliers. In *ACM Sigmod Record*, volume 29, pages 93–104. ACM, 2000.

[10] T. F. Cox and M. A. Cox. *Multidimensional scaling*. CRC Press, 2010.

[11] B. Goethals, S. Moens, and J. Vreeken. MIME: a framework for interactive visual pattern mining. In *Proceedings of the 17th ACM SIGKDD*, pages 757–760. ACM, 2011.

[12] M. Goldstein. k_n-nearest neighbor classification. *Information Theory, IEEE Transactions on*, 18(5):627–630, 1972.

[13] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.

[14] T. Hey, S. Tansley, and K. Tolle, editors. *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Microsoft Research, Redmond, Wash., 1 edition edition, Oct. 2009.

[15] P. Hu, C. Vens, B. Verstrynge, and H. Blockeel. Generalizing from Example Clusters. In J. Fürnkranz, E. Hüllermeier, and T. Higuchi, editors, *Discovery Science*, number 8140 in Lecture Notes in Computer Science, pages 64–78. Springer Berlin Heidelberg, Jan. 2013.

[16] A. Inselberg and B. Dimsdale. Parallel coordinates. In *Human-Machine Interactive Systems*, pages 199–233. Springer, 1991.

[17] A. K. Jain. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31(8):651–666, June 2010.

[18] R. Jarvis and E. Patrick. Clustering using a similarity measure based on shared near neighbors. *IEEE Transactions on Computers*, C-22(11):1025 – 1034, Nov. 1973.

[19] D. Keim. Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):1–8, Jan. 2002.

[20] H.-P. Kriegel, P. Kröger, M. Renz, and S. Wurst. A generic framework for efficient subspace clustering of high-dimensional data. In *Fifth IEEE International Conference on Data Mining*, pages 8 pp.–, Nov. 2005.

[21] L. Maaten. Learning a parametric embedding by preserving local structure. In *International Conference on Artificial Intelligence and Statistics*, pages 384–391, 2009.

[22] G. Moise and J. Sander. Finding non-redundant, statistically significant regions in high dimensional data: a novel approach to projected and subspace clustering. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '08, pages 533–541, New York, NY, USA, 2008. ACM.

[23] E. Müller, S. Günnemann, I. Assent, and T. Seidl. Evaluating clustering in subspace projections of high dimensional data. *Proceedings of the VLDB Endowment*, 2(1):1270–1281, 2009.

[24] K. Pearson. Liii. on lines and planes of closest fit to systems of points in space. *Philosophical Magazine Series 6*, 2(11):559–572, 1901.

[25] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, Dec. 2000. PMID: 11125150.

[26] J. Seo and B. Shneiderman. Interactively exploring hierarchical clustering results [gene identification]. *Computer*, 35(7):80–86, July 2002.

[27] J. Seo and B. Shneiderman. A rank-by-feature framework for unsupervised multidimensional data exploration using low dimensional projections. In *Information Visualization, 2004. INFOVIS 2004. IEEE Symposium on*, pages 65–72. IEEE, 2004.

[28] B. Sluban, M. Juršič, B. Cestnik, and N. Lavrač. Exploring the power of outliers for cross-domain literature mining. In *Bisociative Knowledge Discovery*, pages 325–337. Springer, 2012.

[29] A. Tatu, F. Maas, I. Farber, E. Bertini, T. Schreck, T. Seidl, and D. Keim. Subspace search and visualization to make sense of alternative clusterings in high-dimensional data. In *Visual Analytics Science and Technology (VAST), 2012 IEEE Conference on*, pages 63–72. IEEE, 2012.

[30] J. B. Tenenbaum, V. d. Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, Dec. 2000. PMID: 11125149.

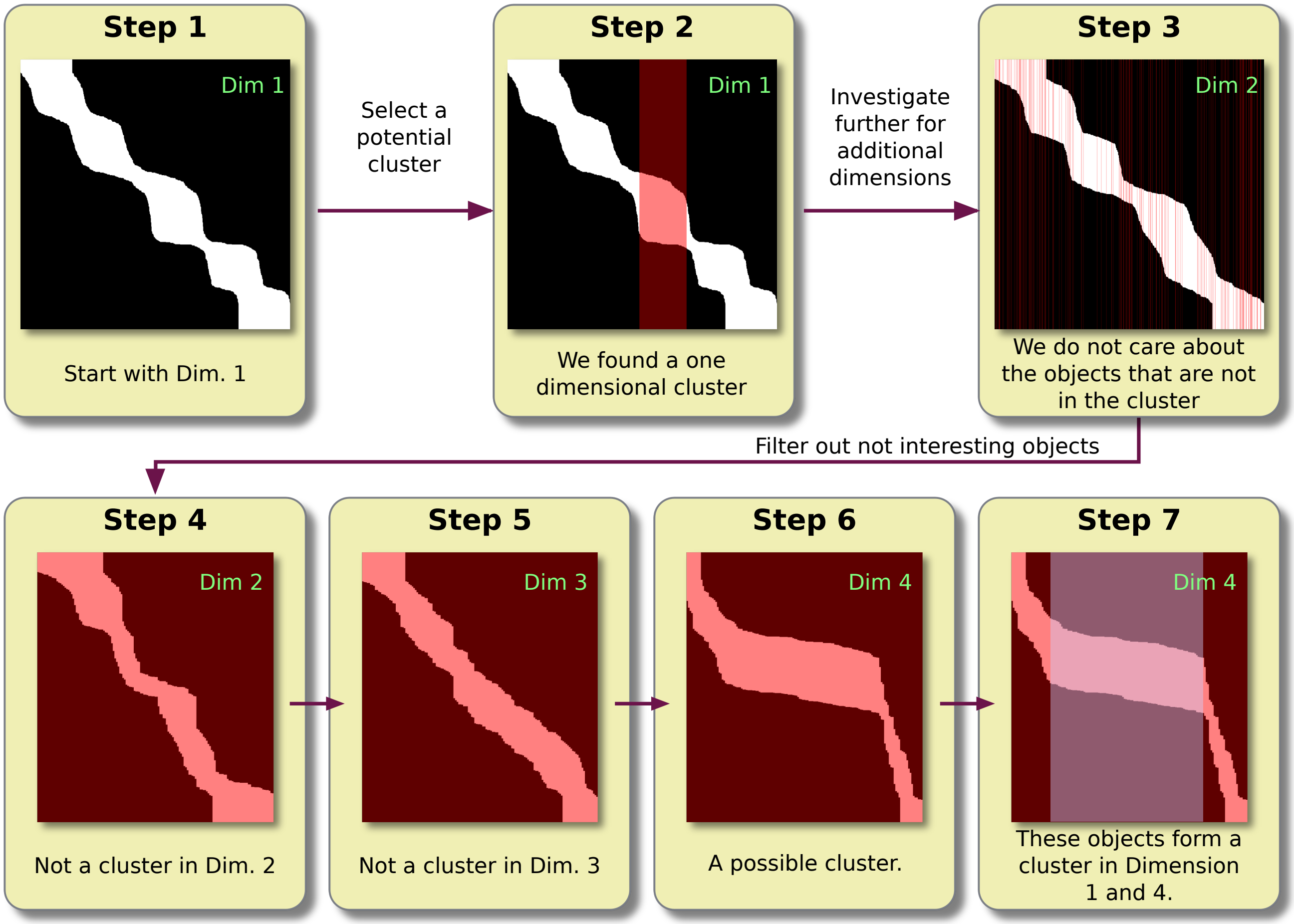
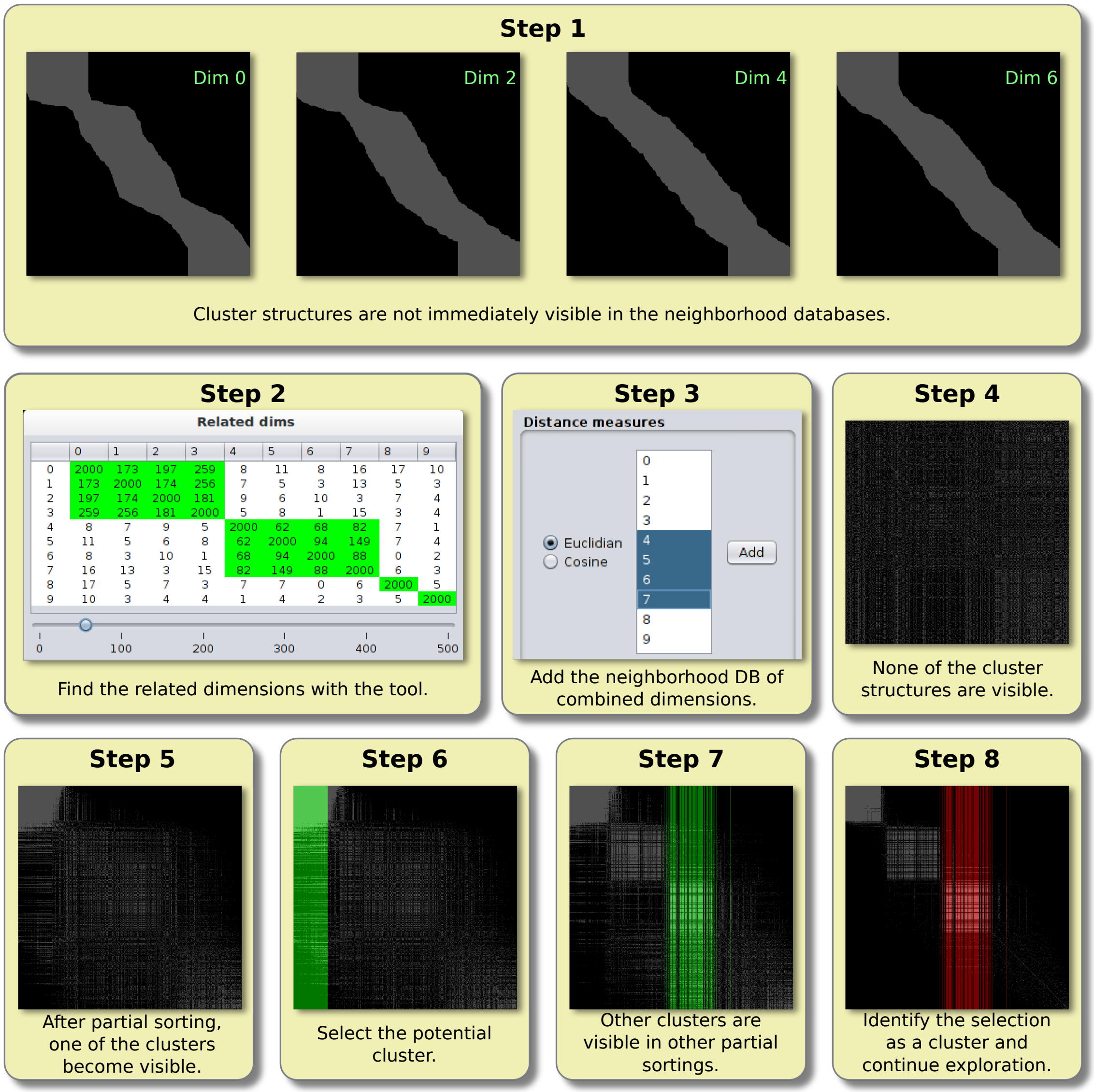

Step 1
Dim 1
Start with Dim. 1
Select a potential cluster
Step 2
Dim 1
We found a one dimensional cluster
Investigate further for additional dimensions
Step 3
Dim 2
We do not care about the objects that are not in the cluster
Filter out not interesting objects
Step 4
Dim 2
Not a cluster in Dim. 2
Step 5
Dim 3
Not a cluster in Dim. 3
Step 6
Dim 4
A possible cluster.
Step 7
Dim 4
These objects form a cluster in Dimension 1 and 4.

Figure 12: Subspace cluster detection



Step 1

Cluster structures are not immediately visible in the neighborhood databases.

Step 2

Related dims

	0	1	2	3	4	5	6	7	8	9
0	2000	173	197	259	8	11	8	16	17	10
1	173	2000	174	256	7	5	3	13	5	3
2	197	174	2000	181	9	6	10	3	7	4
3	259	256	181	2000	5	8	1	15	3	4
4	8	7	9	5	2000	62	68	82	7	1
5	11	5	6	8	62	2000	94	149	7	4
6	8	3	10	1	68	94	2000	88	0	2
7	16	13	3	15	82	149	88	2000	6	3
8	17	5	7	3	7	7	0	6	2000	5
9	10	3	4	4	1	4	2	3	5	2000

Find the related dimensions with the tool.

Step 3

Distance measures

● Euclidian
○ Cosine

Add the neighborhood DB of combined dimensions.

Step 4

None of the cluster structures are visible.

Step 5

After partial sorting, one of the clusters become visible.

Step 6

Select the potential cluster.

Step 7

Other clusters are visible in other partial sortings.

Step 8

Identify the selection as a cluster and continue exploration.

Figure 13: Finding relevant dimensions for cluster detection