A Subspace Filter Supporting the Discovery of Small Clusters in Very Noisy Datasets

Frank Höppner
Ostfalia University of Applied Sciences
Dept. Computer Science
Wolfenbüttel, Germany
f.hoeppner@ostfalia.de

ABSTRACT
Feature selection becomes crucial when exploring high-dimensional datasets via clustering, because it is unlikely that the data groups jointly in all dimensions but clustering algorithms treat all attributes equally. A new subspace filter approach is presented that is capable of coping with the difficult situation of finding small clusters embedded in a very noisy environment (more noise than clustering data), which is not mislead by dense, high-dimensional spots caused by density fluctuations of single attributes. Experimental evaluation on artificial and real datasets demonstrate good performance and high efficiency.

Keywords
feature selection, subspace filter, subspace clustering, cluster analysis

1. INTRODUCTION
Having learned over the past years that (business) data may contain extremely valuable information, people seem to collect yet more data, even if no clear connection to a specific analysis goal is given. Cluster analysis may then give valuable insights by summarising this data and pointing out groups of similar data objects. However, in this situation we would not expect large and well-separated clusters to suggest themselves. Instead, especially when the data was collected without a specific purpose, many attributes will appear potentially irrelevant and clusters may just appear as somewhat more dense areas than the surroundings.

The identification of clusters in such data requires to firstly determine the attributes relevant to the cluster. Standard clustering algorithms assume that all provided attributes are equally important, so either an attribute selection has to take place during preprocessing or the attribute selection has to be incorporated into the clustering algorithm itself (subspace clustering).

SSDBM '14 June 30 - July 02 2014, Aalborg, Denmark
Copyright 2014 ACM 978-1-4503-2722-0/14/06 ...$15.00.
http://dx.doi.org/10.1145/2618243.2618260

There are many clustering paradigms, such as hierarchical clustering (e.g. single-linkage clustering), partitional clustering (e.g. k-means, Gaussian mixture decomposition), grid-based (e.g. CLIQUE), density-based clustering (e.g. DBSCAN, OPTICS). All these algorithms have been applied to various applications and have proven their usefulness. As new clustering algorithms are continuously proposed there seems to be no evidence of convergence to one or a few general purpose clustering algorithms. Clustering is often also a means to an end rather than the end itself, which is why it is difficult to converge to a single definition of a *good cluster* let alone a single best clustering algorithm. Therefore we assume that the selection of the best-suited method is application- and/or goal-driven and there is no such thing as the single best clustering algorithm. Therefore we are interested in selecting promising subspaces independently of the applied clustering method, that is, we focus on subspace selection rather than subspace clustering.

Typical approaches to feature selection (mainly used for classification task) can be divided into wrapper and filter approaches. A wrapper approach selects subspaces randomly or according to some heuristics, runs the selected clustering algorithm for each subspace, evaluates the results and picks the best. Given that the number of subspaces grows exponentially with the dimensionality of the dataset and that the clustering algorithm may have a high complexity itself, this path becomes computationally unfeasible in most cases. Although the effort may be reduced by interlocking subspace search and clustering, it would again be limited to a specific clustering algorithm. A filter approach, on the other hand, ranks individual attributes (or subspaces), which is potentially useful for any kind of clustering algorithm. In this paper, we follow the filter approach, in particular we propose a new filter for subspaces (rather than single attributes).

Assumptions. We assume a dataset $\mathcal{D}$ where n objects of interest (rows) are characterised by d dimensions or attributes (columns) and denote the set of all attributes by $\mathcal{A} = \{A_i \mid 1 \leq i \leq d\}$. By $\mathrm{dom}(A)$, $A \in \mathcal{A}$, we refer to the domain of attribute A. We assume a user is interested in finding clusters in this dataset, which are not necessarily defined over all attributes. Our intention is to suggest and rank subspaces worth further inspection and clustering. The proposed approach has been developed with the following restriction and two assumptions:

- We restrict our proposal to data that is truly numerical, that is, $\mathrm{dom}(A) \subseteq \mathbb{R}$ and rather than only a few "numerical codes" we assume data with many different values.
- We assume that the user does not know if there are any clusters at all. In case there are, the user does not want to miss clusters, but wants to understand dependencies in the lowest possible dimensionality.
- We assume that the data is not naturally divided into well-separated clusters, but there might be more noise than grouping data. We expect clusters to distinguish from their neighbourhood by a somewhat higher data density rather than a void space.

Contributions. The major contributions of the proposed subspace filter are:

- New simple but effective approach to the identification of subspaces that have some clustering tendency.
- A new efficient algorithm that identifies interesting subspaces (not only, but also) under the above-mentioned assumptions (small clusters, not well-separated, much noise).
- Compared to competing approaches, the threshold selection is better justified and does not consist of empirical rules of thumb only.
- Extensive evaluation.

The outline of the paper is as follows. In the next section we review earlier work on filter approaches to feature selection for clustering and discuss the impact of the above-mentioned assumptions on the problem. We then propose a new algorithm for subspace filtering in Sect. 3, which is experimentally evaluated on artificial and real data in Sect. 4. We conclude the paper in Sect. 5.

2. DISCUSSION OF RELATED WORK

2.1 Related Work

There are numerous approaches to clustering in general and subspace clustering in particular. The focus of the paper will be on subspace selection, therefore we refer the reader to [11, 19, 22] for reviews on subspace clustering. While feature selection for classification tasks received much attention, only few publications on feature selection for clustering tasks are available. The comprehensive review of [14] lists 45 feature selection approaches for classification tasks, but only 7 for clustering (50% of them being wrapper approaches).

The wrapper approaches evaluate the validity of the obtained clusters (e.g., compactness and separation) to decide which subspace partitions best. Wrappers (e.g. [6, 12]) are typically used for prototype-based clustering algorithms (like k-means, Gaussian mixture decomposition), where the number of clusters is traditionally determined by similar validity measures. Filter approaches also employ measures of quality or interestingness to judge the importance of an attribute (or a subspace) for clustering. The better the clusters are separated and pronounced, the more the density deviates from a uniform distribution. Thus the amount of disorder introduced by a feature can be used for ranking (e.g. [3, 5]). In [3] a grid-based approach is used to estimate the data density per grid cell and then calculate the entropy of the data distribution over grid cells. The entropy-based quality measure to evaluate the interestingness of a subspace is

$$Q_{\mathrm{ENTROPY}}(S) = \left(\sum_{A \in S} H(A) \right) - H(S)$$

where $H(A)$ is the entropy of a single variable and $H(S)$ of the full subspace (calculated on the grid-cells). In [5] the (parameter-sensitive) definition of a grid is avoided but distances are used to achieve similar results. It was observed in [4] that the histogram of all pairwise distances varies considerably between datasets that do not have cluster substructure and those that do have. A (parameterized) measure is proposed that exploits the shape of the histogram for feature ranking.

Two filter approaches that rank *subspaces* rather than attributes are RIS [8] and SURFING [2]. Let

$$N_\varepsilon^S(x) = \{ y \in \mathcal{D} \,|\, d^S(x,y) \le \varepsilon \} \tag{1}$$

be the ε-neighbourhood of $x \in \mathcal{D}$ where the superscript $S \subseteq \mathcal{A}$ of the distance function d denotes the considered subspace in which the distance is calculated. With RIS the quality of a subspace is defined locally at point x by

$$Q_{\mathrm{RIS}}(S,x) = \frac{|N_\varepsilon^S(x)|}{\text{expected-count in } N_\varepsilon^S(x)}$$

where the denominator denotes the number of data objects expected to fall into a sphere in S with radius ε in case of a uniform distribution. With SURFING the radius ε is adjusted for each $x \in \mathcal{D}$ such that $|N_\varepsilon^S(x)| = k$ remains constant. Somewhat similar to [4], SURFING examines the variation in the distance to the k^{th} nearest neighbour ε (here denoted by $d_{kNN}^S(x)$). The interestingness measure is defined as

$$Q_{\mathrm{SURFING}}(S) = \frac{0.5 \sum_{x \in \mathcal{D}} |\mu_S - d_{kNN}^S(x)|}{\mu_s |Below_S|}$$

where μ_S is the average d_{kNN}^S-distance of all data objects and $Below_S$ is the number of objects falling below μ_S. (In case $Below_S = \emptyset$, the quality is defined to be 0.)

The main advantage of SURFING over RIS is that the quality measure adapts to the mean d_{kNN}^S-distance automatically (cf. denominator). On the other hand, only RIS provides a (heuristic) compensation for border effects: as the subspace dimensionality increases, more and more data points lie close to the boundary of the data (curse of dimensionality).

Most approaches explore subspaces in a level-wise manner (all subspaces of dimensionality n are examined before subspaces of dimensionality $n+1$ are considered). To be interesting, the respective quality measure has to exceed some threshold τ. Two important aspects have to be considered then: Suppose S is an interesting n-dimensional subspace with dimensions A_S. (1) A subspace S' of S may appear interesting simply because it shares its dimensions with S.

Ideally, only the original subspace S gets a good quality assigned, but not its subspace S'. Many approaches thus take care that their quality measure Q yields $Q(S) > Q(S')$ in this case. (2) A superspace S' of S, which is an extension of S with an irrelevant attribute A, may still exceed the quality threshold – not because A contributed to the interestingness of S', but because the density of S was so extraordinary high that it still looks dense after adding a random attribute. A heuristic decision procedure is proposed with RIS to identify such cases. SURFING requires an upward trend in the quality measure for interesting subspaces.

Most of the approaches mentioned above share the property that they evaluate attributes globally over the full dataset. It is, however, well possible that an attribute is useful for one cluster and useless for another; so the utility of an attribute should be evaluated *locally*, which is called *localised feature selection* in [12]. They propose a wrapper approach where the search for the best subspace is performed *for each cluster* separately. While the quality assessment is done *per data object* by RIS (and thus clearly local in the sense of [12]), this is not the case with SURFING (or the entropy-based measure) which evaluate the subspace as a whole.

Although we are interested in subspace filtering rather than subspace clustering, such algorithms face similar problems as they need to explore all subspaces, too. Among the approaches in the literature, the SCHISM [21] comes closest to our own proposal wrt. how interestingness is measured. The grid-based approach assumes independent attributes and derives a bound for the number of hits per grid cell from the Chernoff-Hoeffding bounds of a binomial distribution. (We will comment on SCHISM more detailed later in the paper.) Hypothesis testing is also used in [15] to identify single clusters but not to rank subspaces.

Also somewhat related is the work on outlier detection in [9]. The authors seek for subspaces that exhibit outliers, that is, data objects with a low density compared to their local neighbourhood. With clustering tasks we are interested in data objects of high density. The used notions of interestingness are thus somewhat similar, but the goals and resulting algorithmic approaches differ considerably (and outliers tend to stand out, whereas small clusters are embedded in the noise).

2.2 Discussion

In this section we briefly discuss the problems we face with existing approaches in the light of the assumptions mentioned in the introduction.

Interestingness. What makes a subspace interesting for clustering? Consider the two datasets in Figure 1: which subspaces are interesting? In the 2D case with attributes X and Y one can think of only three subspaces: $\{X\}$, $\{Y\}$ and $\{X, Y\}$. When applied to any of the two datasets, most feature selection algorithms (including RIS and SURFING) would find the full space $\{X, Y\}$ interesting and conjecture clusters in the full space.

However, we argue that only the example on the right of Fig. 1 is interesting. Despite of the similarity of both datasets,

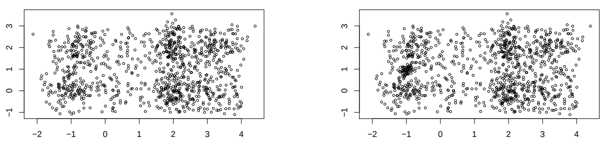
Figure 1: Two 2D-datasets. Which subspaces are interesting?

we consider the dataset depicted on the left as uninteresting and justify this by the process that generated the data: Both datasets differ only by 40 points which were added to the right example. The remaining 1000 points were generated from two independent variables with bimodal distributions. The density peaks in the univariate distributions are responsible for the 2D-peaks, so we can understand Figure 1(left) from the marginal distributions alone – a 2D-view on the data does not contribute anything new. On the right, the additional 40 data points deviate from what we expect by observing the marginal distributions, the additional peak at $(-1, 1)$ can only be explained in the full space $\{X, Y\}$, which makes this subspace worth clustering in 2D and thus interesting.

To carry this example to the extreme, suppose we have d attributes, all with a bimodal distribution without any interaction between any of the attributes. Nevertheless, in the d-dimensional space we will observe 2^d different spots of increased density (combination of the two modes for each dimension). Inspecting the d-dimensional space rather than the univariate attributes would certainly be poor advice. Similar arguments have been made by [6, 9, 10]. Neither RIS [8] nor SURFING [2] or [4] have countermeasures for this case. While [3] emphasises that the attributes of an interesting subspace should correlate and propose to use some threshold on their entropy measure to detect it, it does not work for Fig. 1: *Some* fluctuation is always present, so any correlation measure may provide (insignificant) evidence for minor correlation. The entropy measure operates globally, that is, the few additional points will not really change the (global) correlation between X and Y for the two cases of Fig. 1. There may exist a threshold that is actually only exceeded for the right dataset in Fig. 1, but there is no constructive way to find it beforehand.

Density estimation. Any clustering algorithm uses some kind of density estimation. With k-means a good cluster has a small average distance to its cluster members (minimisation of within-cluster variance), grid-based approaches search for highly populated grid cells, density-based approaches often perform density estimation using the neighbourhood $N_\varepsilon(x)$ for each data point x. The grid- and density-based approaches require some grid resolution or neighbourhood radius ε to be fixed a priori (cf. equation (1)).

It has been observed that a global density threshold, as used by early subspace clustering algorithms, leads to a bias towards a certain dimensionality [11]. As dimensionality increases, so does the distance between data objects and a

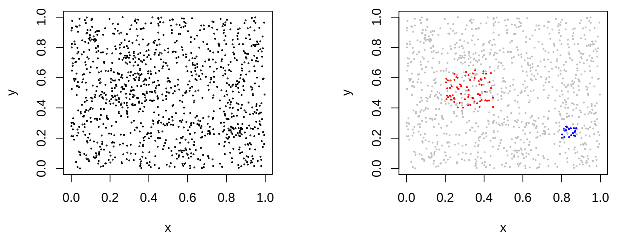

Figure 2: Finding clusters in the raw data (left), consisting of uniform noise plus small clusters (right). A neighbourhood size capable of detecting the blue cluster may fail in detecting the red cluster.

fixed neighbourhood size may become too small to robustly estimate the data density [1]. But as we expect clusters that are neither well-separated nor have a pronounced increase in data-density, we are concerned about the neighbourhood size for another reason. Consider the dataset in Fig. 2(left). It is difficult to spot clusters without having a look at the right figure: the data consists of uniformly distributed noise (gray, 1000 points) with a cluster of size 70 (red) and 20 (blue). The data density in the area of the red (resp. blue) points is roughly two (resp. three) times higher than expected from the background noise. A density estimate in a square area of 0.1^2 may be well-suited to detect the blue cluster, however, it may be too small to robustly detect the red area due to random fluctuation in such a small area. One can easily spot areas in the noise where a recognisable increase in the local density (to a similar density as in the red area) can be observed. To robustly detect the increase in density near the red points, the sampling area should be increased.

Blunt Weapons. Finally, when (comparatively small) clusters are embedded in (lots of) noisy data, some techniques used by the discussed approaches will not work any longer. Subsampling, used by RIS for instance, is likely to miss the few relevant data points that assemble the cluster. Secondly, grid-based approaches try to reduce the computational effort by discretizing the numerical variables, but a small cluster may become undetectable if the few points that raise the data density above the level of the background noise are distributed among multiple grid cells. Thirdly, lavish thresholds (e.g. a large minimum support with frequent itemset mining) may help us to prune many subspaces early and thus reduce run-time, but when clusters exhibit only a moderately increased density, we can't employ too generous thresholds without missing interesting subspaces.

3. APPROACH TO SUBSPACE FILTERING

Our guideline for suggesting subspaces that have a clustering tendency is the following:

Definition 1. A subspace $S \subseteq \mathcal{A}$ is considered interesting, if it discloses some objects for which their *local density* in S is more unlikely to observe than for any other subspace (given the density fluctuations of single attributes).

In line with this notion of interestingness, we propose a subspace filter approach that ranks subspaces according to their interestingness called ROSMULD (**r**anking **o**f **s**ubspaces with **m**ost **u**nlikely **l**ocal **d**ensity). On an abstract level, the proposed subspace filtering algorithm (Alg. 1) consists of the following steps:

1. Based on a range of dimensions where clusters are suspected, and a factor denoting the density increase the user does not want to miss, the neighbourhood size for density estimation is defined. (Sect. 3.4)
2. The attributes are transformed to rank-order and instead of an L_p-norm neighbourhood $N_\varepsilon(x)$ in the original attributes we consider the maximum norm on the ranks. Attributes from an arbitrary distribution are thereby transformed to a uniform distribution. (Sect. 3.1)
3. If individual attributes A, B follow a uniform distribution, so will subspaces $A \times B$ as long as A and B are uncorrelated. Under the assumption of a uniform multivariate distribution (which is now guaranteed for uncorrelated attributes), for every data point we identify the subspace with the density so high, that it is most unlikely to occur by chance. (Sect. 3.2)
4. Subspaces are ranked by how often they have been flagged as surprisingly dense in the previous step.

Each of these steps will be discussed in the subsequent sections.

Algorithm 1 ROSMULD($\mathcal{D}, f, \alpha, \beta$)

```
Require: dataset $\mathcal{D}$ with attributes $A \in \mathcal{A}$, type I/II error
  rates $\alpha$, $\beta$, density factor $f$
Ensure: returns most surprising subspaces of $\mathcal{D}$
  determine $e$ from $f, \alpha, \beta$ acc. to Sect. 3.4
  transform all attributes to rank-order
  for $x \in \mathcal{D}$ do
      determine $p_A$ for $x$ (eq. (2))
      bestS $= \emptyset$              ▷ global var: best subspace for $x$
      bestsofar $= \alpha$       ▷ global var: best quality score so far
      findMostSurprising($x, \emptyset, \mathcal{A}$)
      increment vote-counter for bestS (only if bestS $\neq \emptyset$)
  end for
  drop subspaces that received only a few votes (e.g. 5)
  return interesting subspaces (ordered by votes)
```

3.1 Rank-Ordered Neighbourhood

Instead of analysing the original data, we perform a rank-order transformation first. A rank-order transformation is easily accomplished by ordering the values of an attribute and replacing the original measurements by their rank. Analysing rank-order instead of original data has a long tradition in statistics and is known to be more robust to outliers. The main reason for utilising it here is, however, that it naturally transforms univariate data from an arbitrary to a uniform distribution. According to the first assumption (cf. introduction), we expect identical values to occur only occasionally and use ordinal ranking (that is, all items receive ordinal numbers, ties are broken arbitrarily). For the remainder of this paper, we assume that the original feature vectors $x \in D \subseteq \mathbb{R}^d$ have been transformed to a rank-order feature vector $x \in D_R \subseteq [1,n]^d \subseteq \mathbb{N}^d$.

We define the neighbourhood of some x in a subspace S of the rank-ordered dataset D_R using the maximum norm (L_∞):

$$N_e^S(x) = \{y \in D_R \mid \forall A \in S : |y_A - x_A| \leq e\}$$

That is, the neighbourhood consists of data objects whose difference in the rank in any attribute is at most e. Here, $e \in \mathbb{N}$ takes the role of ε used in $N_\varepsilon^S(x)$. The effect is illustrated in Fig. 3 for the interesting dataset of the introductory example of Fig. 1(right). On the left, the original (interesting) dataset is shown again, where $N_e^{\{X\}}$ and $N_e^{\{Y\}}$ are marked. In the rank-ordered version on the right, this area becomes uniformly distributed. But the relevant dense spot (difficult to distinguish from irrelevant dense spots in Fig. 1(right)) remains dense (marked by a box in Figure 3(right)), because this is a true 2D-cluster and not just a side-effect of co-occurring high densities in univariate distributions.

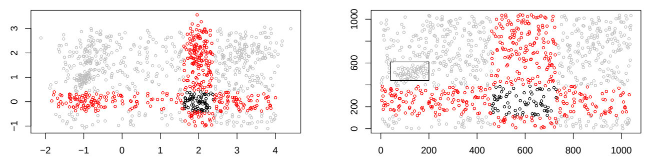

Figure 3: Effect of rank-order transformation on dataset in Fig. 1(right). The neighbourhood $N_\varepsilon(x)$ for original attributes and $N_e(x)$ rank-ordered attributes is shown for one data point.

Given some $e \in \mathbb{N}$ and $x \in D_R$, there are 2^d subspaces $S \subseteq \mathcal{A}$ and to find interesting subspaces we need the number of data points for each neighbourhood $N_e^S(x)$. For a given subspace S, for $y \in D$ to be a neighbour of x it must lie within a e-range of x, that is, $\forall A \in S : |x_A - y_A| \leq e$. The belongingness to $N_e^S(x)$ can be expressed by d binary flags, one for each attribute A: $b_A = \texttt{true} \Leftrightarrow |x_A - y_A| \leq e$. Conceptually, we may convert the dataset D into a d-dimensional 0/1-dataset, where each dichotomous variable $b_A = 1$ indicates if the respective data object is *within e ranks* in attribute A. The neighbour count of x with respect to subspace S may thus be measured by the number of rows having 1's in all columns b_A with $A \in S$. This is exactly the problem of finding the support in itemset mining where the dimensions play the role of items and is used by RIS [8]. Now the computational benefit of the rank-order transformation becomes obvious: In general the effort of frequent pattern mining strongly depends on the largest frequent itemset (here: largest = highest dimensionality), because all subsets will also be frequent. Even without any interesting clusters but mutually independent attributes, a considerable variation in the density may occur as we have seen in Fig. 1. Thus, even for independent attributes it may happen that high-dimensional subspaces contain spots of (relatively) high data density (but no cluster worth considering). Having applied a rank-order transformation, we no longer expect such variations for independent attributes which has a strong impact on any threshold-based subspace enumeration.

3.2 Interestingness of Subspaces

Thanks to the rank-order transformation we can safely assume uniform distributions along each of the dimensions that span an arbitrary subspace. The 1D-neighbourhood of $x \in \mathcal{D}_R$ for any attribute A consists of $2 \cdot e$ neighbours (e to the left and to the right, x itself is excluded). The probability of $y \in D_R \backslash \{x\}$ being a neighbour of x in subspace $\{A\}$ is thus[1]

$$p_A(x) := P\left(y \in N_e^{\{A\}}(x)\right) = \frac{2e}{n-1}. \tag{2}$$

Our initial hypothesis is that subspace S is *not interesting*, that is, all attributes are independent. The rank-order transformation assures uniform univariate distributions and the independence assumption yields

$$p_S(x) := P\left(y \in N_e^S(x)\right) = \prod_{A \in S} p_A(x) = \left(\frac{2e}{n-1}\right)^k \tag{3}$$

as the probability of an arbitrary point lying in $N_e^S(x)$. The number of objects falling into the neighbourhood is thus a random variable and follows a binomial distribution $B(n-1, p_S)$. Formally, we test $H_0 : p_S(x) = \prod_{A \in S} p_A(x)$ against $H_A : p_S(x) > \prod_{A \in S} p_A(x)$. We use a one-sided test because clusters are supposed to have an increased data density. The test statistic is the number of observed elements in the neighbourhood and we reject H_0 if the count exceeds the $(1-\alpha)$-quantile of $B(n-1, p_S)$ (for some significance level α).

Accounting for border effects. With an increasing dimensionality, more and more data lies at the boundary or border of the hypercube, therefore it is important to cope with this effect. Fortunately, this is easily accomplished: If some $x \in \mathcal{D}_R$ lies close to a border, we won't be able to find e neighbours to both sides, but only e_r to the right and e_l to the left. This changes the probability p_A for the respective attribute A to

$$p_A(x) = \frac{e_l + e_r}{n-1}. \tag{4}$$

A change in probability $p_A(x)$ takes an immediate effect on the $(1-\alpha)$-quantile, so the interestingness threshold does vary considerably with the position of x relative to the bounds of the dataset.

Interestingness measure. In the ROSMULD algorithm, we want every data point to vote for the subspace exhibiting the most surprising dense neighbourhood. So we use the p-value of the (one-sided) binomial test, that is, the probability of observing a neighbourhood count at least as high as the observed one, as the quality measure. Let $f(k; n, p)$ be the probability mass function of $B(n,p)$ and $F(k; n, p)$ its

[1]At this point, we assume x is an *inner* point, not near the border or boundary of the dataset. We will deal with border points shortly.

cumulative distribution function. We define interestingness of $N_e^S(x)$ as

$$Q_e^S(x) = 1 - F(|N_e^S(x)|; n-1, \prod_{A \in S} p_A(x)) \tag{5}$$

Data object x votes for subspace S that minimises $Q_e^S(x)$. The more votes a subspace receives, the more interesting the subspace. In the following we may drop indices and argument of $Q_e^S(x)$ if they are clear from the context.

Checking sub- and superspaces. For each $x \in \mathcal{D}_R$ we seek the most surprising subspace and let the data point vote for this subspace. How does this approach compare with other quality measures, in particular with those effects discussed in Sect. 2.2 for which competing approaches installed (heuristic) upward- and downward-pruning as countermeasures? Suppose there is a cluster in subspace $S \subset \mathcal{A}$ that leads to an unexpected high count $k = |N_e^S(x)|$.

Downward pruning: Any subspace $S' \subseteq S$, say $S' \cup \{A\} = S$, will also contain (at least) k elements. If the *true* cluster subspace is S, we want S' to receive a worse score. Under H_0 we have $p_S = p_{S'} \cdot p_A$. If the count k remains roughly constant, its realisation is more unlikely for S because of the smaller probability $p_S < p_{S'}$. Thus, we receive a smaller p-value (higher quality) for S. A data object votes only for the most surprising subspace, so S' does not receive a vote from x.

Upward pruning: Now suppose the opposite case where $S' = S \cup \{A\}$ extends S by an irrelevant attribute A. We would like the quality measure to assign S a better score than S'. A threshold-based technique may still recognise S' as interesting, not because A contributed to the interestingness of S', but because the density of S was so extraordinary high that it still looks dense after adding a random attribute. As already mentioned, competing methods usually utilise heuristic decision procedures to exclude S' from further consideration, but nothing needs to be done using the p-value as quality measure.

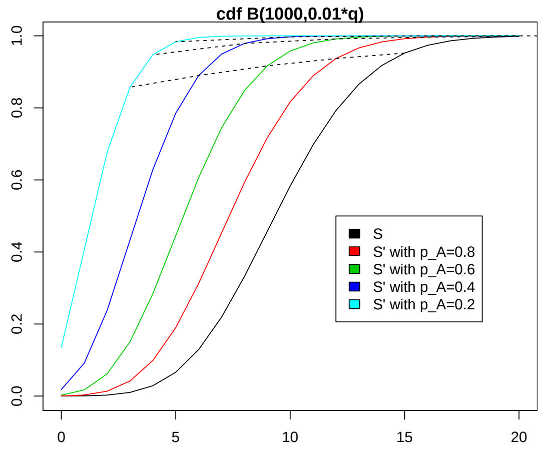

Figure 4: When extending interesting subspaces with an irrelevant attribute, its p-value rises (making it less interesting).

Figure 4 illustrates the behaviour of $Q^S(x)$. The inclusion of an irrelevant attribute A in S' changes the probability $p_{S'}$ of a random point being included in $N_e^{S'}$ to $p_{S'} = p_S \cdot p_A$ and at the same time lowers the expected count to $k \cdot p_A$. The figure shows cumulative distribution functions for $F(k; n = 1000, p = 0.01 \cdot p_A)$, where four different values of p_A are shown. The right-most (black) line corresponds to subspace S, the four left-most curves correspond to S' (with varying p_A). Given we have observed a high count $k > n \cdot p$ in subspace S, the dashed lines connects $F(k)$ with $F(k \cdot p_A)$ of resp. subspaces S' (coloured curves). Looking from right (S) to left (S'), the values $F(k \cdot p_A)$ decrease and thus the p-value $(1-F)$ increases (dashed lines). Thus, S receives again the vote from x, there is no need for any further pruning heuristic.

There is no closed form of $F(k; n, p)$, but this behaviour can also be observed for the upper bounds of $F(k; n, p)$ obtained from the Hoeffding inequality

$$F(k; n, p) \leq \exp(-2(k - np)^2/n) \tag{6}$$

leading us to

$$\begin{aligned} \log F(k \cdot p_A; n, p_{S'}) &\leq -2(k \cdot p_A - n \cdot p_{S'})^2/n \\ &= p_A^2 \cdot (-2(k - np_S)^2/n) \\ &< \underbrace{(-2(k - np_S)^2/n)}_{\log F(k;n,p_S) \leq} \end{aligned}$$

However, the Hoeffding-bounds are known to not work well for extreme values of p. Therefore, in contrast to e.g. SCHISM [21], we are not going to use them to derive any bounds for pruning.

3.3 Finding the Most Surprising Subspace

As already pointed out, frequent pattern mining might be used to examine $N_e^S(x)$ for all subspaces $S \subseteq \mathcal{A}$. The rank transformation would prevent us from wasting time on frequent neighbourhoods caused by variations in univariate densities alone. However, we are only interested in the single best subspace and enumerating all subspaces seems pointless. We propose a recursive depth-first search (Alg. 2) using a bound on the interestingness measure to prune parts of the search space (Alg. 3).

Algorithm 2 findMostSurprising(x,S,R)

Require: current $x \in \mathcal{D}$, subspace $S \subseteq \mathcal{A}$, attributes to examine $R \subseteq \mathcal{A}$, for simplicity bestsofar (smallest p-value) and bestS (best subspace) are global variables

```
 1: $QV = Q_e^S(x)$; let $c_A = |N_e^{S\cup\{A\}}(x)|$ for $A \in R$
 2: if ($QV <$ bestsofar) then                    ▷ better subspace?
 3:     bestsofar=$Q_e^S(x)$; bestS=$S$;             ▷ store subspace
 4: end if
 5: if ($QV -$ bestsofar $> \Delta$) then            ▷ worth to check?
 6:     if (safeToPrune($x$,$S$,$R$)) then          ▷ pruning possible?
 7:         return                               ▷ prune search branch
 8:     end if
 9: end if
10: for $A \in \mathcal{A}$ with $c_A - np_{S\cup\{A\}} > 0$ (in decr. order) do
11:     $R = R \backslash \{A\}$
12:     findMostUnlikely($x$,$S \cup \{A\}$,$R$)   ▷ recursive descend
13: end for
```

Algorithm 3 safeToPrune(x,S,R)

```
Require: current $x \in \mathcal{D}$, subspace $S \subseteq \mathcal{A}$, attributes to examine $R \subseteq \mathcal{A}$, for simplicity bestsofar (smallest p-value) and bestS (best subspace) are global variables
Ensure: true if pruning does not miss a better solution
 1: Order $c_A = |N_e^{S\cup\{A\}}(x)|$, $A \in R$, decr.: $c_1 \geq ... \geq c_{|R|}$
 2: Order $p_A$, $A \in R$, increasingly: $p_1 \leq ... \leq p_{|R|}$
 3: Let $n_i$ be the number of $y \in N_e^S(x)$ that are additionally found in exactly $i$ neighbourhoods $N_e^{S\cup\{A\}}(x)$, $A \in R$
 4: $p = p_S$; prune=false
 5: for i=1..|R| do
 6:     $p = p \cdot p_i$
 7:     if $(1 - F(\min\{c_i, \sum_{j=1}^{|R|} n_j\}; n-1, p) <$ bestsofar then
 8:         prune=false
 9:     end if
10: end for
11: return prune
```

During the recursive depth-first search, suppose we arrive at some $S \subseteq \mathcal{A}$ (for some $x \in \mathcal{D}$) and determined $k = |N_e^S(x)|$. Then, $F(k; n, p)$ is compared with the currently best subspace to see if a more interesting subspace has been found. To decide if a further descend (to higher-dimensional subspaces) is worth the effort, we need a lower bound of all p-values achievable in any subspaces reachable from the current subspace.

Lemma 1. Given a dataset $\mathcal{D}_R$ with d attributes $A \in \mathcal{A}$, an $x \in \mathcal{D}_R$, a subspace $S \subseteq \mathcal{A}$ and $R = \mathcal{A}\backslash S$, $|R| = r$. Let $c_i = |N_e^{S\cup\{A_i\}}(x)|$ be the neighbourhood count in subspace $S \cup \{A_i\}$, $A_i \in R$, ordered decreasingly $c_1 \geq c_2 \geq \ldots \geq c_r$. Let $p_i = p_{A_i}(x)$, $A_i \in R$, ordered increasingly $p_1 \leq p_2 \leq \ldots \leq p_r$. Let n_i be the number of $y \in N_e^S(x)$ that are additionally found in exactly i neighbourhoods $N_e^{S\cup\{A\}}(x)$, $A \in R$. Then for any $S' \subseteq \mathcal{A}$ with $S \subset S'$ its quality is bounded by

$$Q^{S'} \geq 1 - \max_{m=1..r} F(\min\{c_m, \sum_{i=m}^{r} n_i\}; n-1, \prod_{j=1}^{m} p_j) \qquad (7)$$

Proof. To find a lower bound of the p-value $Q^S = 1 - F(k; ...)$ we seek an upper bound of $F(k; ...)$. For fixed p the cumulative distribution function $F(k; n-1, p)$ is strictly increasing in k. An upper bound is thus obtained from $F(k^*; n-1, p)$ where k^* is an upper bound of the achievable count in any subspace S. We find such a bound individually for all subspace dimensionalities $|S| + m$, $m = 1..r$, and get the overall upper bound by taking their maximum (max-term in (7)).

So let $S' \subseteq \mathcal{A}$ with $S \subset S'$ with $|S'| = |S| + m$, $m \in \{1, \ldots, r\}$. For a subspace S' of dimensionality $|S| + m$ and an upper bound on $k^* = |N_e^{S'}(x)|$ we have $k^* \leq c_m$: Apparently we need m attributes $A \in R$ having $|N_e^{S\cup\{A\}}(x)| \geq k^*$ in order to reach the same count in the full subspace S'. Because of $c_1 \geq c_2 \geq \ldots \geq c_m$ this holds for c_m. Secondly, for an y to be element of $N_e^{S'}(x)$ it must be contained in at least m neighbourhoods $N_e^{S\cup\{A\}}(x)$. An upper bound for the number of cases contained in that many neighbourhoods is $\sum_{i=m}^{r} n_i$. Altogether, it is guaranteed that $k^* \leq \min\{c_m, \sum_{i=m}^{r} n_i\}$, so we can use this min-term as an upper bound the number of neighbours.

The probability p_S of a random point belonging to $N_e^S(x)$ varies with the participating attributes: $p_S(x) = \prod_{A\in S} p_A(x)$. With $f(k; n-1, p)$ being unimodal with a single peak at $(n-1)p$, for fixed k the cdf $F(k; n-1, p)$ is strictly decreasing in p (the dataset size n is constant). Therefore we find the maximal F (or minimal p-value) by minimising p. The smallest p for a subspace consisting of m dimensions is obtained from $\prod_{j=1}^{m} p_j$ as the p_i are ordered increasingly. $\square$

The bound provided by the Lemma can be used to cut a whole branch of the search tree.[2] It uses information that is easily calculated (n_i, c_i, p_i), as we will see below. But the bound relies on multiple $F(k; ..)$ values, which is an expensive operation (we used the algorithm from [20] to determine $F(k; n, p)$). To speed up these calculations, we initially build a tabulated approximation to first check if the exact calculation has realistic chances of beating the best value found so far before. (As the calculated p-values may become very small, we use log-transformed p-values in the program.)

To find c_i, n_i and p_i we need the neighbours of x in individual attributes and subspaces (which can be derived from the neighbourhood of single dimensions due to the maximum norm). For every $x \in D_R$ the 1D-neighbourhoods are easily determined from the ordering of individual attribute values (which has already been done for the rank-order transformation). As illustrated in Fig. 5, when sorting by individual attributes, we store the record ID with the attribute value, such that we can easily identify the IDs of the e nearest neighbours from the previous and next e neighbours in the sorted array. Note that the complexity of neighbourhood identification becomes $O(d \cdot e)$, which may be considerably smaller than the usually mentioned $O(n^2)$ for naive neighbourhood search or $O(n \log n)$ for indexed search.

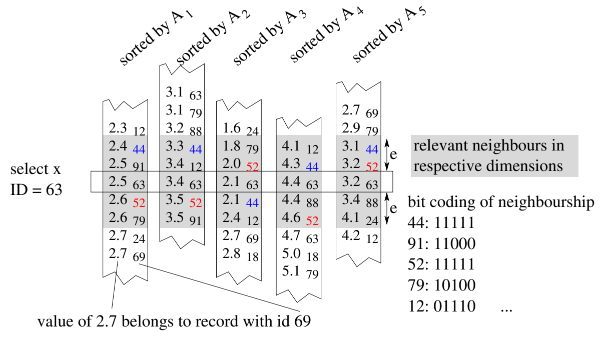

Figure 5: Using the sorted attributes to find neighbours in $N_e^S(x)$.

For all records whose ID appear in at least one 1D-neighbourhood, for subsequent calls we bit-encode the belongingness to all neighbourhoods and operate on these bitcodes rather

[2]The possibility of pruning is only checked if the best-so-far value is considerably better than the current value. For all experiments we have chosen a threshold on $\log(QV) - \log(\text{bestsofar}) > 100$ in line 5 of Alg. 2.

than the full dataset. The bitcode resembles all necessary information to determine c_i and n_i. Bit i is set if and only if $y \in N_e^{\{A_i\}}(x)$: (1) for c_{A_i} just count how often bit i is set, (2) increase n_i when i bits are set in total. When an attribute A has been selected for the subsequent recursive call, the number of data objects that lie in the higher dimensional subspace decreases quickly, so we copy the bitsets into a new array to avoid the traversal of the full array in deeper branches of the search tree.

The goal of the depth-first search is to find those subspaces with a high quality (small p-value) as fast as possible, such that the pruning mechanism of Lemma 1 can become most effective. We have already argued that the best score is achieved for subspaces S with high counts k and small p_S. In both cases, the term $k - n \cdot p_S$ becomes larger, so it is used as a heuristic guideline which attribute we should try first to arrive at highly scored subspaces. (That this term is also the core in the Hoeffding bound (6) may be seen as another justification.) We thus choose the attribute A that maximises $(c_A - np_{S\cup\{A\}})$ (cf. line 10 of Alg. 2; c_A corresponds to the count k we achieve in $S \cup \{A\}$, cf. line 1).

The algorithm is thus prepared for both scenarios: Firstly, with uninteresting datasets (without any local dependency between attributes) all attributes are uniformly distributed (after rank-order transformation). Initially, we have to consider all data objects that are at least within the neighbourhood of x in one dimension. As we proceed during the search, some attributes A are included in the subspace S under consideration. But due to the independence of variables, the inclusion of any attribute A in S reduces the data objects by a factor of p_A, so their number decreases quickly with each recursive call. Secondly, with well pronounced clusters, the maximisation of $c_A - np_S$ leads us quickly to the best scored subspaces S and from then on, the pruning of Lemma 1 prevents us from wasting time on the all subspaces of S.

3.4 Threshold Selection

We require the user to specify a significance level α: we count only votes from subspaces which are guaranteed to significantly deviate from the expected count. The neighbourhood size e remains to be specified and we propose to use statistical power to choose e. By choosing the significance level α small enough we control the type I error rate, that is, we avoid flagging a subspaces as interesting if it is not. A type II error denotes the case that H_0 is actually wrong but the test does not recognise this. Limiting the type II error will lead us to a minimal neighbourhood size e.

From H_0 (independence of attributes) we assume that $p_S = \prod_{A\in S} p_A$ holds. H_0 is already violated if p_S is only slightly higher, but a user probably wouldn't care about a 1% higher density. So we ask the user to provide a density factor f, say $f = 1.5$, she actually does care about and is willing to miss only in $\beta = 1\%$ of all cases. To justify the participation of any attribute A in the exceptional subspace, this factor should apply to all attributes of the subspace. Therefore, we define

$$p_S^+ := \prod_{A\in S} (f \cdot p_A) \tag{8}$$

n	2	3	4	5	6	7	8	9	10
2000	61	99	131	160	187	205	215	236	255
4000	88	157	220	279	332	372	394	437	475
8000	126	256	381	485	591	673	722	809	885
16000	181	405	641	875	1053	1219	1414	1497	1652
32000	261	658	1108	1523	1949	2308	2593	2772	3082
64000	374	1067	1864	2652	3472	4181	4755	5133	5751

Table 1: Necessary neighbourhood sizes for $f = 2.0$, $\alpha = 0.01$ (+Bonferroni adjustment), $\beta = 0.1$ for various dimensionalities and dataset sizes.

to be the relevant threshold for type II errors: by specifying a bound β on the probability of type II errors we limit the probability of failing to reject H_0 in case $p_S = p_S^+$.

For the binomial test we determine a critical value q (the $(1-\alpha)$-quantile of $B(n-1, p_S)$) and reject H_0 in case the observed count $C \geq q$. The power of the test is given by the probability of observing a count $C < q$ (or $C \leq q-1$ as counts are integer) in case of $p_S = p_S^+$:

$$1 - power(p_S^+) = \beta = F(q-1; n-1, p_S^+)$$

The critical value q as well as the power of the test is eventually determined by e (because e defines p_A, which define p_S, cf. eq. (2) and (4)). By varying e we control not only q but also the power of the test. Besides $\beta \geq 1 - power(p_S^+)$ we additionally require $q > 1$ because we want $q - 1$ (used for the determination of the power) to remain positive.

To find the appropriate values of e, we may simply increment e until the desired level of β is reached. The reference value p_S^+ depends on the dataset size as well as the dimensionality of the subspace S. Table 1 shows the required minimum neighbourhood sizes e for various dimensions and dataset sizes at $f = 2.0$ ($\alpha = 0.01$ with Bonferroni adjustment for the dataset size and $\beta = 0.1$). If the user is confident that clusters have approx. dimensionality d, choosing e from the respective columns guarantees a detection of subspaces with density increased by a factor f (per dimension) within the limits of type I and type II error α and β, resp. If the user does not want to focus on a specific dimensionality (from which a global e is chosen), the ROSMULD algorithm is easily modified to use a different value of e for each subspace dimensionality.

From the neighbourhood sizes in Table 1 we recognise that for higher-dimensional subspaces e may cover around 20% of the dataset size (e to the left and to the right). To achieve this in grid-based approaches (like SCHISM), the discretization of attributes must not exceed 5 bins. While ROSMULD centers such a neighbourhood around each and every data object, the subdivision in grid-based algorithms is fixed once and forever. With such a coarse grid the locality of the search is very poor and it becomes extremely unlikely to robustly recognise small clusters.

3.5 Voting

Although we use only simple voting, that is, every data object has the same weight when voting for a subspace, one can think of arbitrary weights. A data object does not vote for a subspace if the deviation is not significant and the thresholds are determined such that it is equally likely to exceed the critical value for a 2-dimensional or a 5-dimensional subspace. However, the degree of exceeding the α-determined critical value may be quite different, so the quality score (p-value) may be used directly as a voting weight (e.g. $-\log(Q_e^S(x))$).

4. EXPERIMENTAL EVALUATION

All datasets used are publicly available or were made available at `http://public.ostfalia.de/~hoeppnef/rosmuld.html`. If not otherwise stated, we used $\alpha = 0.01$ with Bonferroni adjustment for multiple testing (for the size of the dataset), $\beta = 0.01$ and a factor $f = 2.0$ throughout the experiments.

Globally irrelevant attributes may be identified by calculating a (global) correlation matrix (entropy, Pearson correlation, rank correlation, etc.) Due to their global nature, however, they are not sensitive to a local correlation that affects only a few points. The first two subsections will show that the proposed algorithm succeeds in this task. We then consider the case of well-separated clusters and real data sets, and finally discuss the efficiency.

4.1 No Clusters

This is an important scenario especially for prototype-based clustering algorithms like k-means, which always generate an a priori specified number of clusters even if there are no clusters at all in the data. Determining a clustering tendency is an important step in cluster analysis (see e.g. [7, 18]). Ideally the subspace filter should not propose subspace which have no clustering tendency.

A 10-dimensional test data set of size 10.000 consisted of uniformly distributed attributes, Gaussian distributed attributes and attributes being a mixture of two Gaussians. There is a considerable fluctuation in the data density and k-NN-distance in this dataset, as can be seen from Fig. 6. Both, Q_{RIS} and Q_{SURFING} therefore flag various subspaces as interesting, although it has been generated from independent attributes alone. The ROSMULD algorithm was executed with $e \in \{300, 500, 700\}$ and the result was always the same: no subspaces were flagged as being interesting.

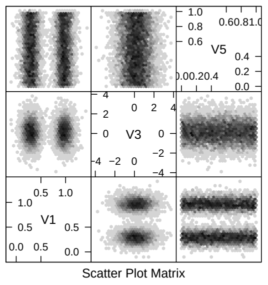

Figure 6: Three dimensions of dataset from Sect. 4.1.

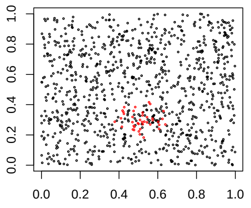

Figure 7: Two relevant dimensions of dataset from Sect. 4.2.

4.2 Small Clusters Hidden in Noise

When ROSMULD is applied to the two datasets from the introduction (Fig. 1), the 2-dimensional x/y-space was flagged as interesting only for the dataset on the right. Next, we generated a six-dimensional dataset with all attributes sampled from a uniform distribution (unit interval). Among the 1000 objects, only 40 where drawn from a 3-dimensional Gaussian distribution with standard deviation $\sigma = 0.06$. It is difficult to recover the 3D-cluster visually from a 2D-scatter plot matrix, because the density increases only slightly and dense spots occur also by chance. Therefore the 40 additional points are coloured in red in Fig. 7. As we look for a low-density cluster, we lower f to $f = 1.5$. ROSMULD successfully identified the relevant subspace, only the 3D-subset that contains the hidden cluster received votes.

When applying the quality measure of SURFING to this dataset, we obtain the following values (we denote the interesting attributes as A_2, A_3 and A_6): $Q(\{A_2\} = 0.269$, $Q(\{A_2, A_3\}) = 0.195$ and $Q(\{A_2, A_3, A_6\}) = 0.163$. The SURFING algorithm recognises interesting subspaces by an increasing trend in the quality measure, however, in this dataset the quality Q_{SURFING} decreases with dimensionality, so SURFING is not able to recover the relevant 3D subspace.

Finally, a 10-dimensional dataset with a 5-dimensional small cluster was randomly generated (uniformly distributed within [0, 1], cluster edge length 0.07). The cluster is very sparsely populated compared to the size of the dataset (5000 records). Table 2 shows how often (among 10 runs) a cluster has been selected using different thresholds for e from different columns in Table 1. The actually detected subspaces did not always correspond to the original 5D-subspace (but only subspaces thereof), however, this was not to be expected given that we have implanted only 5-30 data objects in a 5-dimensional space. Due to the random generation of a few data objects only, it is very likely that the (estimated) variance per dimension varies considerably. If most objects are (by chance) close together in one dimension, only four relevant dimensions will be discovered. Altogether, despite the constant $f = 2.0$ and varying e-value, the small cluster has been discovered in most of the trials.

cluster size	e selected for dimensionality:						
(# objects)	2	3	4	5	6	7	8
5	0	6	5	0	3	3	5
10	0	10	9	6	10	10	8
15	4	10	10	10	10	10	10
20	10	10	10	10	10	10	10
25	10	10	10	10	10	10	10
30	10	10	10	10	10	10	10

Table 2: Detection of a 5D-cluster (5-30 points) in a random 10D-dataset of size 5000 (of out 10 runs).

4.3 Well-Separated Clusters

This work was motivated by the problem of finding small clusters in the presence of many irrelevant attributes. We nevertheless present results on well-separated clusters to demonstrate ROSMULDs broad applicability. We use the datasets from [17], where they were used to compare various subspace clustering algorithms. (The artificially generated data is available from their web site.) The examples consists of 20-dimensional datasets each with 11 clusters of dimensionality 50%, 60% and 80% of the dataset dimensionality. From the attached cluster labels we investigated the within-cluster variance and identified the relevant attributes for each clus-

dataset	size	dim	sec.	most interest. subspaces
bike	731	5	≤ 1	$\{A_1, A_4, A_5\}$
wine qual.	6497	11	9.3	$\{A_4, A_8, A_{11}\}$, $\{A_6, A_7\}$
pendigits	7494	16	111	$\{A_6, A_8, A_{10}\}$, $\{A_8, A_{10}, A_{12}\}$, $\{A_8, A_{14}, A_{16}\}$
vowel	990	10	≤ 1	$\{A_1, A_2, A_4\}$

Table 3: Results on some UCI real world datasets.

ter. In the five datasets there are always 1 small cluster, 2 medium clusters (twice as large as the small one) and 8 large clusters (thrice as large as the small cluster).

We selected e for dimensionality 10 (column 10 in Table 1) and 20 (not shown in Table 1) (and $f = 1.5$). In the latter case, for each of the five datasets only 8-14 subspaces received votes. For the largest example (S5500) 10 subspaces were found corresponding exactly to 10 (out of 11) cluster subspaces, only the smallest cluster was not found. The results for other examples were very similar; in another dataset (S2500) 9 subspaces were detected, again the smallest one was missed and the subspace of a second cluster was fully contained in the subspace of another, the smaller subspace received votes from both clusters. When assuming dimensionality 10 a few more subspaces received a few additional votes, but the top-ranked subspaces were not affected. For another set of examples with an increasing level of noise the performance did not degrade, the top-ranked subspaces did not change (for run-times see Sect. 4.5).

The number of votes received corresponds very well to the size of the included clusters. In these datasets there was only one cluster per subspace, so the data objects that voted for the subspace actually form the cluster. This does not hold in general, of course, but even though the subspace filter was not meant as a clustering algorithm, its results could be used to initialise clustering algorithms (initial clusters for k-means, density thresholds for density-based clustering).

4.4 Real Datasets

We investigate some real datasets (shown in Table 3) from the UCI repository [13]. No cluster-class relationship was examined (as done, e.g., by [17]) to evaluate the subspaces: firstly, we want to evaluate the discovered subspaces and do not (yet) have clusters, and secondly, we cannot see any convincing reason[3] why clusters should generally follow some class labels that may have been attached for reasons that have nothing to do with the grouping tendency of the data (see also the pendigits example below). We briefly discuss the meaningfulness of the discovered subspaces instead.

The first dataset from the UCI repository is the wine quality dataset. It consists of 4898 instances and 11 attributes (plus class information). The most interesting subspaces consisted of the attributes alcohol, residual sugar and density. Its relevance is confirmed by winemakers, who get reasonably-accurate estimates of alcohol content and residual sugar levels of wines from their specific gravity measurements.

[3][17] argue: "*Therefore the idea is to use labelled data with the assumption that the natural grouping of the objects is somehow reflected by the class labels.*"

The algorithm was also applied to the pendigits dataset for pen-based recognition of handwritten digits. It consists of 10992 instances and 16 attributes. The most interesting subspaces consisted of attributes 6, 8, 10 and 8, 10, 12. The 16 attributes denote eight pairs of (x, y) pen-coordinates, thus the subspaces denote a sequence of three successive y-coordinates. This is easily explained by noting that many digits start and end differently, but the mid-part consists of a downward motion (decreasing y-positions at time 3, 4, 5). Another interesting subspace consisted of the 4th, 7th and 8th y-coordinates (cf. Fig. 8). For various handwritten numbers in this dataset, these y-positions are close together: With digits 0, 5, 6 and 8 the 4th position has low y-values and the 7th and 8th y-position have high y-coordinates, while the digits 2, 3 and (some types of) 5 have intermediate values at the 4th y-position and low values at 7th and 8th y-position.

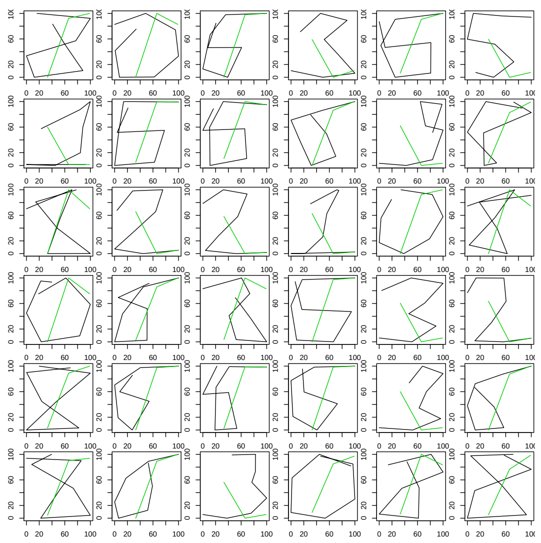

Figure 8: Excerpt of the pendigits dataset. The eight (x,y)-pairs were used to draw the original digits. The green overlaid lines connect the 4th, 7th and 8th y-positions; note that the shape of the green lines is similar across different digits (= classes).

The bike sharing dataset summarises the number of daily bike rentals. From the five numerical dimensions (feeling temperature, humidity, windspeed, count of casual users, count of registered users) the most interesting subspace consisted of three attributes: temperature, casual and registered users. The data objects that vote for this subspace resemble similar days of low temperature and few (casual/regular) users and days of high temperatures with some casual and some or many regular users, which is a meaningful dependency.

4.5 Efficiency

For the experiments we used an Intel quad core mobile i7, 2.4 Ghz, with 700 MB heap space assigned (which was not nearly used, ROSMULD has low memory consumption). Note that the ROSMULD algorithm lends itself for parallelization because every data object can be handled independently. However, the run-times reported here come from a *single-threaded* Java program. We compare the run-times with [17], where a similar setting was used (quad core Opteron 2.3 GHz with RAM limited to 1.5 GB, nothing is said about the number of cores that were actually used).

First we report run-times for the datasets used in Sect. 4.3. For the neighbourhood size e we (under-) estimated the dimensionality of clusters to be half of the dataset dimensionality (that is, 10 in the 20-dimensional datasets). Figure 9 shows the run-time of the ROSMULD algorithm versus the dataset size and noise percentage in comparison to the run-times of subspace clustering algorithms reported in [17]. Note that a log-scale is used for all run-times. As already mentioned, the clusters contained 50%, 70% and 80% of the available 20 attributes, which is not the intended scenario of the ROSMULD algorithm. Nevertheless, the performance in Fig. 9(left) is very competitive. By increasing the noise level we approach the intended application of the ROSMULD algorithm and the performance in Fig. 9(right) is only beaten by Proclus (prototype-based approach) and partially by Mineclus (cell-based clustering). Note that the neighbourhood size e is not constant, but increases with the dataset size (cf. Table 1).

Figure 10 shows the scalability of the ROSMULD algorithm for three clusters of different density in a 5-dimensional subspace for various dataset sizes and dimensionalities. Figure 10(left) shows run-times for datasets twice as large as those in Fig. 9, different dimensionality as well as for two different assumption on cluster dimensionality (lower curve: 5, upper curve: 10). On the right of Fig. 10 the scalability wrt. the dimensionality of the dataset is shown. As the numbers show, ROSMULD scales well.

The competing approaches RIS and SURFING have been reported to require much larger run-times. The Apriori approach embedded in RIS leads to extremely large run-times for high-dimensional clusters, as the detection of a 12-dimensional interesting subspace requires all $2^{12} - 1$ subspaces to be explored beforehand (see for instance the results in [9]). SURFING seems to suffer from similar problems, the only case of 12-dimensional cluster (Table 1 of [2]) took almost 4.5 hours to find (on a 2.8 GHz CPU). And also the grid-based SCHISM algorithm is outperformed (cf. Fig. 9).

5. CONCLUSIONS

We have proposed a subspace filter for clustering tasks, which is capable of identifying subspaces in case of both, well-separated clusters and small clusters in a very noisy environment. This is challenging, because small clusters hidden in heavy noise are difficult to robustly distinguish from the background. Locally dense areas are not reported if they are caused by fluctuation in the univariate distributions only, such that the reported subspaces contain only attributes that truly contribute to the abnormally high density. In the experiments the proposed ROSMULD algorithm was very robust against noise and scaled well with the dataset size and the dataset dimensionality.

There are many directions for future work. First, the ROSMULD algorithm does not only provide the relevant subspaces, but may also provide the individual data objects that *voted* for this subspace. These data objects are likely to belong to the clusters of the respective subspace, so they may be an excellent initialisation for, say, efficient prototype-based clustering algorithms. Second, the run-time can still be improved: Up to now, the data objects are processed independently, which is good for parallelization, but once a data object x has voted for subspace S, many objects of $N_e^S(x)$ may also vote for the same subspace. If we attach S as a label to $y \in N_e^S(x)$, we may evaluate this subspace first for y and gain an advantage for pruning. Density estimation techniques as proposed in [16] may be useful to improve the run-time complexity further.

References

[1] I. Assent, R. Krieger, E. Müller, and T. Seidl. DUSC: Dimensionality Unbiased Subspace Clustering. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pages 409–414. Ieee, Oct. 2007.

[2] C. Baumgartner, K. Kailing, H.-P. Kriegel, P. Krüger, and C. Plant. Subspace Selection for Clustering High-Dimensional Data. In *ICDM*, 2004.

[3] C.-H. Cheng, A. W. Fu, and Y. Zhang. Entropy-based subspace clustering for mining numerical data. *Proc. SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pages 84–93, 1999.

[4] M. Dash, K. Choi, P. Scheuermann, and H. Liu. Feature selection for clustering - A filter solution. *Data Mining, 2002. ICDM*, 2002.

[5] M. Dash and H. Liu. Feature selection for clustering. *PAKDD*, 2000.

[6] J. Dy and C. Brodley. Feature selection for unsupervised learning. *The Journal of Machine Learning Research*, 5:845–889, 2004.

[7] B. Hopkins and J. Skellam. A new method for determining the type of distribution of plant individuals. *Annals of Botany*, XVIII(70), 1954.

[8] K. Kailing, H. Kriegel, P. Kröger, and S. Wanka. Ranking interesting subspaces for clustering high dimensional data. In *PKDD*, volume 2838, pages 241–252, 2003.

[9] F. Keller, E. Muller, and K. Bohm. HiCS: High Contrast Subspaces for Density-Based Outlier Ranking. In *2012 IEEE 28th International Conference on Data Engineering*, pages 1037–1048. Ieee, Apr. 2012.

[10] H. Kriegel, P. Kroger, M. Renz, and S. Wurst. A Generic Framework for Efficient Subspace Clustering of High-Dimensional Data. In *Fifth IEEE International Conference on Data Mining (ICDM'05)*, pages 250–257. Ieee, 2005.

[11] H.-P. Kriegel, P. Kröger, and A. Zimek. Clustering high-dimensional data. *ACM Transactions on Knowledge Discovery from Data*, 3(1):1–58, Mar. 2009.

[12] Y. Li, M. Dong, and J. Hua. Localized feature selection for clustering. *Pattern Recognition Letters*, 29(1):10–18, Jan. 2008.

[13] K. B. Lichman and M. *UCI Machine Learning Repository.* PhD thesis, University of California, Irvine, 2013.

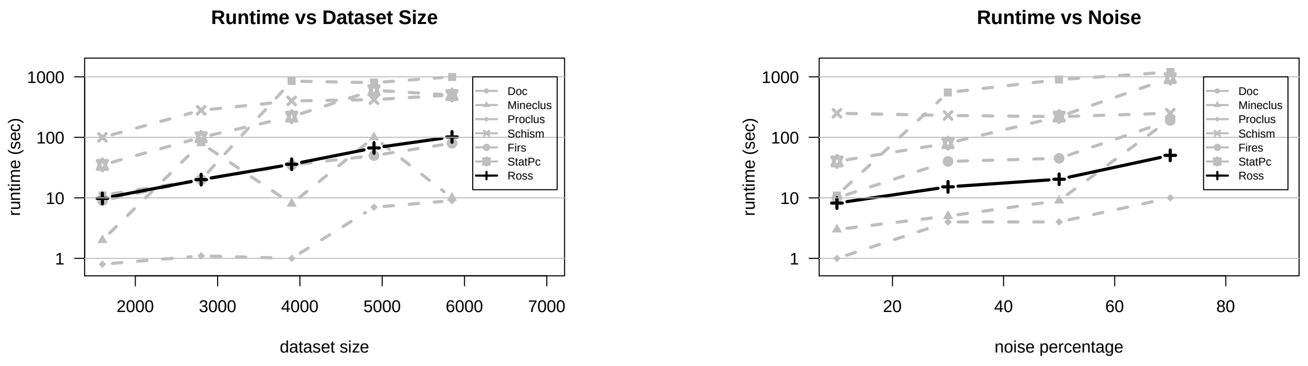

Figure 9: Run-times for datasets from Sect. 4.3.

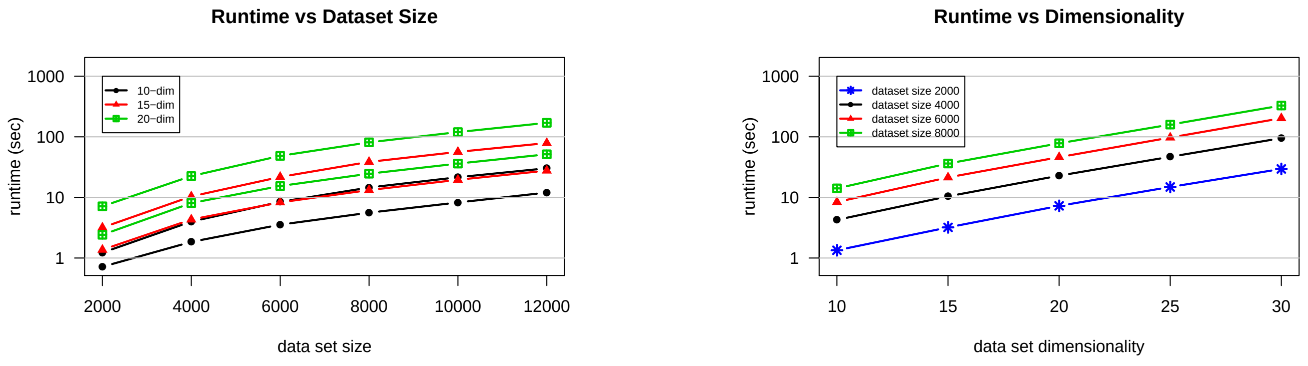

Figure 10: Run-time depending on size of the dataset (left) and dimensionality (right).

[14] H. Liu and L. Yu. Toward integrating feature selection algorithms for classification and clustering. *... and Data Engineering, IEEE Transactions on*, 17(4):491–502, 2005.

[15] G. Moise and J. Sander. Finding non-redundant, statistically significant regions in high dimensional data: a novel approach to projected and subspace clustering. In *ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pages 533–541, 2008.

[16] E. Müller, I. Assent, R. Krieger, S. Günnemann, and T. Seidl. DensEst: Density Estimation for Data Mining in High Dimensional Spaces. *SDM*, pages 175–186, 2009.

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

[18] E. Panayirci and R. Dubes. A test for multidimensional clustering tendency. *Pattern Recognition*, 16(4):433–444, 1983.

[19] L. Parsons, E. Haque, H. Liu, and L. Parson. Subspace Clustering for High Dimensional Data - A Review. *ACM SIGKDD Explorations Newsletter*, 6:90–105, 2004.

[20] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes*. 2007.

[21] K. Sequeira and M. Zaki. SCHISM: A New Approach to Interesting Subspace Mining. *Int. J. of Business Intelligence and Data Mining*, 1(2):137—-155, 2005.

[22] K. Sim, V. Gopalkrishnan, A. Zimek, and G. Cong. A survey on enhanced subspace clustering. *Data Mining and Knowledge Discovery*, 26(2):332–397, Feb. 2012.