Int J Data Sci Anal (2017) 4:173–189
DOI 10.1007/s41060-017-0068-8

REGULAR PAPER

Hiding outliers in high-dimensional data spaces

Georg Steinbuss[1] · **Klemens Böhm**[1]

Received: 22 March 2017 / Accepted: 28 August 2017 / Published online: 12 September 2017

Abstract Detecting outliers in very high-dimensional data is crucial in many domains. Due to the curse of dimensionality, one typically does not detect outliers in the full space, but in subspaces of it. More specifically, since the number of subspaces is huge, the detection takes place in only some subspaces. In consequence, one might miss *hidden outliers*, i.e. outliers only detectable in certain subspaces. In this paper, we take the opposite perspective, which is of practical relevance as well, and study how to hide outliers in high-dimensional data spaces. We formally prove characteristics of hidden outliers. We also propose an algorithm to place them in the data. It focuses on the regions close to existing data objects and is more efficient than an exhaustive approach. In experiments, we both evaluate our formal results and show the usefulness of our algorithm using different subspace selection schemes, outlier detection methods and datasets.

Keywords Outlier detection · High dimensionality · Subspaces · Hidden outliers

1 Introduction

Many applications in different domains, e.g. fraud detection, depend on the effective and efficient identification of outliers [5]. Due to the curse of dimensionality, outliers often occur in attribute subspaces. Such outliers are referred to as subspace outliers. In high-dimensional spaces, it is not feasible to inspect all subspaces for outliers, since their number grows exponentially with the dimensionality. Thus, most approaches only inspect a subset of the set of all subspaces. Depending on the subspaces inspected, the outlier detection method used and the distribution of the data, so-called *hidden outliers* may occur. A hidden outlier exhibits its outlier behaviour only in subspaces where no outlier detection takes place. Hence, the characteristic whether an outlier is hidden or not depends on the subspaces where one is looking for outliers. Figure 1 displays a low-dimensional illustrative example. The outlier in the figure is hidden when looking at each one-dimensional subspace in isolation. It can only be detected when looking at the two-dimensional subspace.

1.1 Motivation

In this article, we examine how to place hidden outliers in high-dimensional data spaces, and we quantify the risk of the data owner that such outliers can be placed in the data. We see three reasons why studying the issue is necessary, namely (1) increasing the reliability of critical infrastructures, (2) coping with attacks, and (3) systematic evaluation of outlier detection algorithms. We now elaborate on these points one by one.

1.1.1 Reliability of critical infrastructures

Think of data objects each representing a state of a critical infrastructure. Outliers are unusual system states which may represent any kind of fault or a state preceding a fault. Since faults of infrastructures that are critical may be catastrophic [15], any action preventing such faults pays off. However, data objects representing these states usually do not exist or are extremely rare. Hidden outliers represent combinations of values that remain undetected with existing models. If

✉ Georg Steinbuss
georg.steinbuss@kit.edu

Klemens Böhm
klemens.boehm@kit.edu

[1] Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany

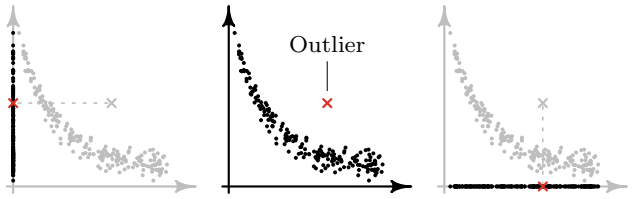

Fig. 1 Example showing a hidden outlier

hidden outliers were detected, a domain expert could inspect them and assess how detrimental they are.

1.1.2 Attacks with hidden outliers

Research on classifier evasion [19] studies the behaviour of an adversary attempting to 'vanquish' a learner. Example 1 shows that the situation here is analogous, including the motivation, i.e. studying the adverse behaviour in order to shield against it. While Example 1 is extreme for the sake of illustration, it is our running example due to its intuitiveness. The example also shows that hidden outliers pose a risk, and it is worthwhile to quantify this risk.

Example 1 Think of a criminal intending to commit credit-card fraud. He has inside information, i.e. he knows that the bank checks for fraud by means of an outlier-detection method on a high-dimensional representation of the credit-card transactions. More specifically, the bank uses a subspace-search method that is confined to subspaces consisting of few attributes only. The attacker then identifies regions of the data space where outliers are hidden from the detection method and designs fraudulent transactions whose representation falls into these regions.

1.1.3 Evaluation of subspace outlier detection

Being able to hide outliers is expected to help evaluate subspace outlier detection methods. Current schemes for evaluation often either use an already existing minority class as outlying or downsample the data to one rare class [3]. However, these outliers are not necessarily subspace outliers, in contrast to hidden outliers generated with our approach. Placed hidden outliers are known to be outliers in certain subspaces, and one can quantify how well they are found. An evaluation scheme based on our approach would also allow differentiating between different kinds of hidden outliers and might be more systematic than one depending on the outliers which have been found so far. The design and assessment of such evaluation schemes is one of our long-term research efforts and is beyond the scope of this current article. Here, our concern is the effective and comprehensive placement of subspace outliers.

1.2 Challenges and contributions

To our knowledge, the general question of how to place outliers in datasets has not been studied explicitly so far. Placing outliers such that they are hidden is even more unclear. Various challenges arise when designing such a placement, as follows.

Hidden outliers are inliers in certain subspaces. This means that we cannot just use extreme values to obtain hidden outliers. Compared to Fig. 1, the situation may also be more complex. For instance, one is not confined to just 'high-dimensional versus low-dimensional subspace'. Instead, a mix of the two is feasible as well. The variety of outlier definitions is another, orthogonal challenge. Some rely on statistical models, others on spatial proximity [11] or angles between objects [12]. We hypothesise that different outlier definitions lead to different regions where hidden outliers can be placed. When designing a general algorithm that hides outliers we cannot assume much on the outlier detection method used. Another challenge relates to the relative size of the region where hidden outliers can be placed. In some scenarios, this region may be small, while it may be huge in others, e.g. close to the full data space. Placing many hidden outliers that are diverse requires methods that adapt to the size of this region. That is, the placement should cover a broad range of positions if the region is huge. When the region is small in turn, the placement must be more fine-grained. A last challenge is that assessing the probability of success of attackers (the ones who hide the outliers) is not trivial. In contrast to Example 1, attackers may not have full access to the data. Hence, an attack is more likely to be successful if hiding is feasible without knowing much on the data. Any risk assessment should take into account the extent of knowledge which is necessary for the hiding.

In our work we start by deriving important characteristics of hidden outliers analytically, focusing on multivariate data following a normal distribution. One result is that hidden outliers do exist in this setting. Another one is that correlation within subspaces can reduce the size of the region of hidden outliers. Another contribution of ours is an algorithm that places hidden outliers. It relies on only one mild assumption regarding outlier detection, but without the assumptions behind our theoretical analyses, e.g. the normal distribution assumption. Its design is based on the hypothesis that hidden outliers tend to be close to real data objects. This is based on the property that hidden outliers must be inliers in some subspaces. Hence, our algorithm concentrates on placing hidden outliers in regions close to existing data objects, with adjustable tightness. This allows for a placement that concentrates on a small region, close to the data, or a rather large one. The algorithm does not rely on any assumption regarding the outlier detection method used, except for a nonrestrictive one: Namely, the detection method must flag points as out-

liers or not. The output of any method we are aware of can be transformed without difficulty to have this characteristic (see e.g. [14]). Our algorithm also gives way to a rigid definition of the risk of an attacker being able to hide outliers. Finally, we have carried out various experiments. They confirm that some of our theoretical findings also hold in the absence of the underlying model assumptions, e.g. for other outlier detection methods and datasets. They also demonstrate that our algorithm is much better in hiding than a baseline. In particular, this holds for high-dimensional datasets. The main part of the paper is structured as follows:

1. Definition of hidden outliers. Section 4.1
2. Analytical derivations of characteristics of hidden outliers. Section 4.3
3. Algorithm to place hidden outliers. Section 4.4
4. Evaluation of concept and algorithm. Section 5

All our code and datasets used are publicly available.[1]

2 Related work

We are not aware of any comprehensive study of hidden outliers. [27] however describes the notion of *masked* outliers. *Masked* means that irrelevant attributes within a dataset can hide the outlier behaviour to some extent. Our work is of course related to the various methods for outlier detection and subspace search. Some schemes exist solely for subspace search [6,16], some with integrated outlier detection [13,17] and numerous methods merely for outlier detection [2,8,11,12]. All outlier detection methods compute whether existing data objects are outliers or not. This is different from our approach: We study how to place outliers in datasets.

Related to detecting outliers with the help of subspace search is detection with the help of *feature selection*. [1] is one of the early pieces of work in this discipline. They select features where the density of inliers is high and the density of outliers is low. This is inspired by the well-known outlier detection algorithm LOF [2]. [21,22] both work with categorical data and consider so-called value couplings to select features. These couplings determine that features are only selected when they agree on the outlierness of objects. Feature selection and subspace search are related. However, there is a profound difference on the methodology level. Subspace search aims at finding different subspaces, each regarding other aspects of outlierness. Feature selection on the other hand aims at finding a single subspace that exhibits all or at least many of these outlier aspects. Because of this difference, subspaces found and methods developed for subspace search differ significantly from those for feature selection.

To illustrate *classifier evasion* mentioned before explicitly, think of a spam filter. The idea now is that a spammer wants to send emails that are *as close as possible* to spam, but are classified as regular. However, existing approaches to find such positions [20,26] rely on at least one instance of spam email, which we do not rely on in outlier detection. Secondly, we are not aware of any approach considering the effects of using subspaces. *Adversarial examples* [25] that were recently introduced in the neural network community follow the concept of classifier evasion. An adversarial example is an object that is classified wrongly due to some small changes to it. [25] formalises the crafting of such adversarial examples as an optimisation problem, to find the smallest modification of an object of some class so that it is classified differently. We in turn want to maximise the number of hidden outliers placed successfully. Other approaches for crafting adversarial examples also focus on the case that the classifier they invade is a neural network [4,23].

Protecting privacy is another area in data analysis that is related. This is because some approaches for privacy protection add objects to the data. For example, [7] proposes an algorithm to add dummy objects to position data of individuals, in order to have better privacy. Clearly, the objective is different: privacy protection approaches attempt to add data that behaves like the original data. Hence, the true data are hidden, while relevant information is still available. We in turn hide data objects which contradict the general structure of the data. Another difference is that such privacy approaches so far are global, i.e. not based upon subspaces.

Another related notion is *robust statistics*. It deals with the fact that assumptions in statistics often are only approximations of reality. Violations of these assumptions are often interpreted as outliers. Hence, robust approaches take such possible violations into account to stabilise statistical models. [24] for instance proposes a modification of a subclass of Gauss–Markov models such that it is free from outlier hiding effects. Without these modifications, outliers might affect the model itself in a way that they are not detectable, i.e. are hidden. However, such approaches are not based upon subspaces, do not compute outlier regions or address the problem of hiding outliers in the data.

3 Notation

Let DB be a database containing n objects, each described by a d-dimensional real-valued data vector $\mathbf{y} = (y^{(1)}, \ldots, y^{(d)})^T$. The set $\mathcal{A} = \{1, \ldots, d\}$ denotes the full attribute space. W.l.o.g., we assume that each attribute lies within $[l, u]$ where $l,\ u \in \mathbb{R}$. An attribute subset $\mathcal{S} = \{a_1, \ldots, a_m\} \subseteq \mathcal{A}$ is called a m-dimensional subspace projection ($1 \leq m \leq d$). A set $Collection = \{\mathcal{S}_1, \ldots, \mathcal{S}_t\} \subseteq P(\mathcal{A})$ is a collection of

[1] Our code and data: http://ipd.kit.edu/mitarbeiter/steinbussg/Experiments_HideOutlier.zip.

t subspace projections ($1 \leq t \leq 2^d - 1$). The set $Full\mathcal{R} = \{\mathbf{y} \in [l, u]^d\}$ is the entire data space. When not stated different explicitly, for any region $\mathcal{R}$, it holds that $\mathcal{R} \subseteq Full\mathcal{R}$. Further, we assume that there exists a function $out^{\mathcal{S}}(\cdot)$ of the form:

$$out^{\mathcal{S}}(\mathbf{y}) := \begin{cases} 1 & \text{if } \mathbf{y} \text{ is outlier in } \mathcal{S}, \\ 0 & \text{if } \mathbf{y} \text{ is inlier in } \mathcal{S}. \end{cases} \quad (1)$$

The function $out^{\mathcal{S}}(\cdot)$ is a generic outlier definition. Different outlier detection methods which typically incorporate different definitions of this generic function are in use. Many such methods output a score instead of a binary value. However, we assume that these scores are transformed to a binary signal, e.g. by applying a threshold.

4 The region of hidden outliers

In this section we formalise the notion of hidden outlier and derive important characteristics. Section 4.2 features some assumptions behind our formal results. In Sect. 4.1, we define hidden outliers and other relevant concepts. In Sect. 4.3, we derive our formal results. Section 4.4 features an algorithm to place hidden outliers. This algorithm also allows to define the *risk* of hidden outliers.

4.1 Definition

We briefly review the well-known multiple view property [18] of subspace outliers before presenting our underlying definitions. It is crucial for the notion of hidden outliers. The property refers to the case that an object can be an outlier in some subspaces while being an inlier in others. Hence, there are several subspaces, each containing outliers that are different in nature. Think for instance of a dataset from a bank. In this data there might be a subspace related to characteristics of the family of customers and another one regarding their employment. The outliers in the family characteristics subspace are of a different nature than those in the employment subspace. With subspace outlier detection, promising subspaces are detected in a first step; in a second step, each of these subspaces is searched for outliers using a conventional outlier detection method.

Bearing the multiple view property in mind, we define the notion of hidden outlier as follows:

Definition 1 Let two disjunct sets of subspace projections $Collection_{outlier}$ and $Collection_{inlier}$ be given. $\mathbf{o} \in [l, u]^d$ is a hidden outlier with respect to subspace collections $Collection_{inlier}$ and $Collection_{outlier}$ if

$$out^{\mathcal{S}}(\mathbf{o}) = 0 \; \forall \, \mathcal{S} \in Collection_{inlier}$$

and $\exists \, \mathcal{S} \in Collection_{outlier} : out^{\mathcal{S}}(\mathbf{o}) = 1$

The number of subspaces not in $Collection_{inlier}$ is usually rather high. Testing a subspace for outliers contained in it is expensive computationally. Thus, we focus on the case that the hidden outliers are outlier in at least one subspace of $Collection_{outlier}$ instead of any subspace not in $Collection_{inlier}$. $Collection_{inlier}$ and $Collection_{outlier}$ must always be disjunct. This is because there cannot be any point being an inlier and outlier in the same subspace. However, there can be overlapping attributes in subspaces of both sets. If there is no attribute within subspaces of both sets, the task of placing hidden outliers is rather simple. One creates an outlier for one of the subspace in $Collection_{outlier}$ and sets the values for the remaining attributes in $\mathcal{A}$ to the ones of any existing inlier object. Thus, in this article we focus on scenarios with such overlap. Based on this definition, we now formulate a hypothesis.

Hypothesis 1 Since hidden outliers are inliers for all subspaces in $Collection_{inlier}$, hidden outliers must be spatially close to the points in DB.

We will return to this hypothesis when designing our algorithm (Sect. 4.4) and in the experiments (Sect. 5.4.6).

A core issue in this study is to identify the positions/region with the following characteristic: if we place a data point there, it is a hidden outlier. We now derive this region and present some characteristics of hidden outliers. To this end, we do not rely on any further assumption regarding $out^{\mathcal{S}}(\cdot)$.

Definition 2 The *region of inliers* $In\mathcal{R}(Collection)$ is defined as

$$\{\mathbf{o} \in [l, u]^d \mid out^{\mathcal{S}}(\mathbf{o}) = 0 \; \forall \, \mathcal{S} \in Collection\}$$

The *region of outliers* $Out\mathcal{R}(Collection)$ is its complement.

Definition 2 formalises the notion of the region fulfilling one property of hidden outliers, i.e. regions with positions that are inlier or outlier for each subspace in $Collection$. This notion is a prerequisite before defining the region of hidden outliers. See Fig. 2a, b for examples using the Mahalanobis distance. We discuss characteristics of it in Sect. 4.3.

Lemma 1 *The region* $In\mathcal{R}(Collection)$ *is the intersection of the regions* $In\mathcal{R}(\{\mathcal{S}\}) \; \forall \, \mathcal{S} \in Collection$.

Lemma 1 states that we can derive $In\mathcal{R}(Collection)$ using only intersections of $In\mathcal{R}(\{\mathcal{S}\})$, i.e. the inlier region in a single subspace. Detecting outliers in one subspace is well defined and has been explored intensively.

Definition 3 Let two sets of subspace projections $Collection_{outlier}$ and $Collection_{inlier}$ be given. The region of hidden outliers $Hidden(Collection_{outlier}, Collection_{inlier})$ is the intersection of the region $In\mathcal{R}(Collection_{inlier})$ and the region $Out\mathcal{R}(Collection_{outlier})$.

Thus, every point in *Hidden* is a hidden outlier. We see that, up to intersections, unions and complements, *Hidden* solely depends upon outlier detection in a single subspace. However, $In\mathcal{R}(\{\mathcal{S}\})$ is of arbitrary shape—depending on $out^{\mathcal{S}}(\cdot)$. Hence, computing these intersections, unions and complements is arbitrarily complex.

The number of possible *Collection*s is huge: having $|P(\mathcal{A})|(= 2^d - 1)$ subspaces yields $2^{|P(\mathcal{A})|} - 1$ possible *Collection*s. The number of possible combinations of two *Collection*s to obtain *Hidden* is even larger. Thus, in the first part of this work we focus on two cases, $Collection_1 = \{\mathcal{A}\}$ (the full space) and $Collection_2 = \{\{1\}, \ldots, \{d\}\}$ (each one-dimensional subspace).

Notation 1 *HiddenIn*$\mathcal{A}$ refers to the setting when $Collection_{inlier} = \{\{1\}, \ldots, \{d\}\}$ and $Collection_{outlier} = \{\mathcal{A}\}$. *HiddenFrom*$\mathcal{A}$ to the setting when vice versa $Collection_{inlier} = \{\mathcal{A}\}$ and $Collection_{outlier} = \{\{1\}, \ldots, \{d\}\}$.

In setting *HiddenIn*$\mathcal{A}$, outliers are detectable in the full space, but not in any one-dimensional projection. Setting *HiddenFrom*$\mathcal{A}$ is the opposite: outliers are not detectable in the full space, but in at least one of the one-dimensional projections.

Example 2 In Fig. 2c, the Mahalanobis distance is used to identify outliers. The red crosses are hidden outliers in the settings just proposed. The square represents the bound for points that are inliers in both subspaces of $Collection = \{\{1\}, \{2\}\}$. The circle represents the inlier bound for points that are inliers in $Collection = \{\mathcal{A}\} = \{\{1, 2\}\}$. Hidden outliers in setting *HiddenIn*$\mathcal{A}$ are points inside the square but outside of the circle. Analogously, hidden outliers in setting *HiddenFrom*$\mathcal{A}$ are points outside of the square but inside the circle.

In some cases, we will refer to a more general form of the settings *HiddenIn*$\mathcal{A}$ and *HiddenFrom*$\mathcal{A}$, i.e. where one collection is an arbitrary partition of $\mathcal{A}$ into subspaces.

When analysing characteristics of *Hidden*, we will make use of the relative volume of a region. More explicitly, we use it to bound the region of hidden outliers.

Definition 4 Let a region $\mathcal{R} \in \mathbb{R}$ be given. The **relative volume of $\mathcal{R}$** is defined as $RelativeVolume(\mathcal{R}) := \text{Volume}(Full\mathcal{R} \cap \mathcal{R}) \div \text{Volume}(Full\mathcal{R})$.

Lemma 2 *An upper bound on* $RelativeVolume(Hidden)$ *is the minimum of* $RelativeVolume(In\mathcal{R}(Collection_{inlier}))$ *and* $RelativeVolume(Out\mathcal{R}(Collection_{outlier}))$.

Thus, if the relative volume of $Out\mathcal{R}(Collection_{outlier})$ or $In\mathcal{R}(Collection_{inlier})$ is very small, e.g. zero, we know that the relative volume of *Hidden* cannot be larger.

In a next step, we investigate specific scenarios with an outlier detection method using the Mahalanobis Distance. Having such a specific outlier notion allows to derive distinct characteristics of *Hidden*.

4.2 Assumptions for formal results

We assume that DB follows a multivariate normal distribution (*MVN*) with zero mean. Of course, Gaussian distributed data points have attribute limits $-\infty$ and $+\infty$. However, we assume that l and u are so large that even outliers will most likely be contained in the range spanned by l and u. With *MVN* data, the Mahalanobis distance [11] yields the likeliness of a data point. We assume data objects to be outliers if they are very unlikely according to that distance. We refer to the Mahalanobis distance of $\mathbf{y}$ in subspace $\mathcal{S}$ as $MDist^{\mathcal{S}}(\mathbf{y})$. $Quantile(\alpha,\ df)$ is the α quantile of a χ^2 distribution with df degrees of freedom. According to [11], we can instantiate our outlier definition as follows:

$$out^{\mathcal{S}}(\mathbf{y}) := \begin{cases} 1 & \text{if } \left[MDist^{\mathcal{S}}(\mathbf{y})\right]^2 > Quantile(0.975,\ |\mathcal{S}|)\,, \\ 0 & \text{otherwise.} \end{cases}$$

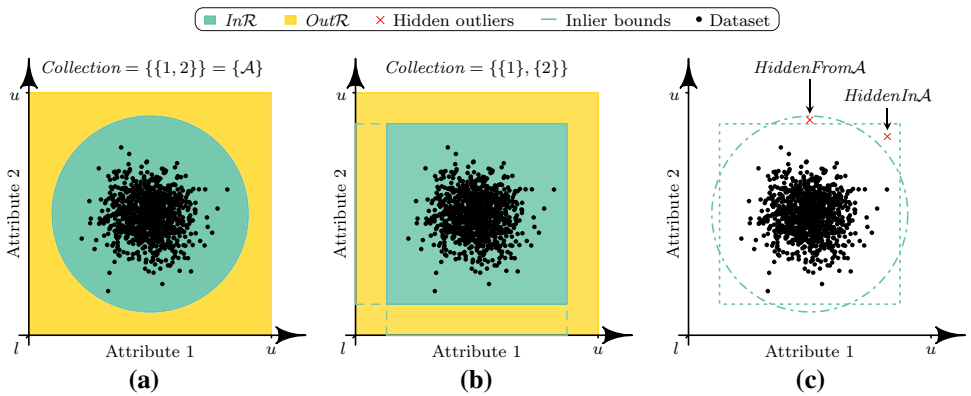

Fig. 2 Example for $In\mathcal{R}(Collection)$, $Out\mathcal{R}(Collection)$ and hidden outliers in *HiddenIn*$\mathcal{A}$ and *HiddenFrom*$\mathcal{A}$

4.3 Formal results

Motivation for Theorem 1 Figure 2c is a two-dimensional example illustrating hidden outliers. The lines are outlier boundaries using the Mahalanobis distance. In this two-dimensional case, hidden outliers can exist for both settings *HiddenIn*$\mathcal{A}$ and *HiddenFrom*$\mathcal{A}$. We wonder whether this eventuality of having hidden outliers extends to higher dimensionalities and more general subspace selections. We answer this question by analysing a more general scenario. In our two sample settings, there are two kinds of subspace selections. We have $Collection_1 = \{\{1\}, \ldots, \{d\}\}$ and $Collection_2 = \{\mathcal{A}\}$. To generalise this, we replace $Collection_1$ with an arbitrary partition of the attribute space.

Theorem 1 *Let $\mathcal{A}$ be the full data space and Collection a nontrivial (i.e. $\neq \mathcal{A}$) partition of $\mathcal{A}$ into subspaces. Let the number of dimensions of $\mathcal{A}$ and of each subspace in Collection converge to infinity. Let the data attributes be i.i.d. with $N(0, 1)$. Then there exists a hidden outlier that is outlier in at least one subspace of Collection and inlier in $\mathcal{A}$. There also exists a hidden outlier that is outlier in $\mathcal{A}$ but inlier in each subspace of Collection.*

All proofs are in appendix. We have assumed that the dimensionality goes to infinity in order to approximate *Quantile* in the proof. However, Fig. 2c shows that the theorem holds even in a two-dimensional case. Our experiments will show that it holds for other datasets as well.

Motivation for Theorem 2: Next, we consider the effect of correlation on the relative volume of $In\mathcal{R}(\{\mathcal{A}\})$. Figure 3 displays $In\mathcal{R}(\{\mathcal{A}\})$ in a two-dimensional example. The circle is for the case that Attributes 1 and 2 are uncorrelated. The ellipse stands for strong correlation. The volume of the ellipse is smaller than the one of the circle. Thus, a higher correlation seems to imply a smaller relative volume of $In\mathcal{R}(\{\mathcal{A}\})$ and a larger relative volume of $Out\mathcal{R}(\{\mathcal{A}\})$. Theorem 2 formalises this for data spaces of arbitrary dimensionality.

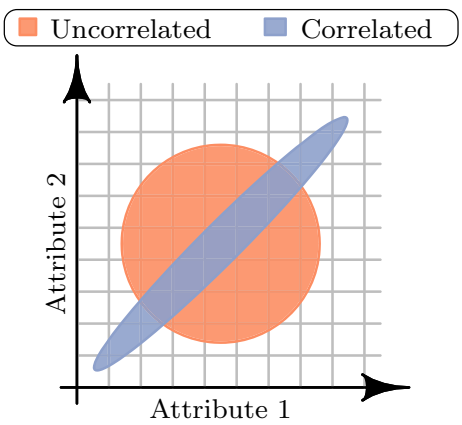

Fig. 3 Motivation for Theorem 2 and Hypothesis 2

Theorem 2 *Let subspaces $\mathcal{S}_1$ and $\mathcal{S}_2$, both of dimensionality d and MVN distributed, be given, and let the attributes in $\mathcal{S}_1$ be i.i.d. with $N(0, \lambda)$. The covariance matrix in $\mathcal{S}_1$ is Σ_1 and in $\mathcal{S}_2$ Σ_2. For these matrices, it holds that $diag(\Sigma_1) = diag(\Sigma_2)$, and that Σ_2 has off-diagonal elements (i.e. covariance). Then we have:*

$$RelativeVolume(In\mathcal{R}(\mathcal{S}_1)) \geq RelativeVolume(In\mathcal{R}(\mathcal{S}_2))$$

This theorem relies on one further technical assumption spelled out in appendix which also contains the proof.

Lemma 3 *From $Out\mathcal{R}(\mathcal{S}) = \overline{In\mathcal{R}(\mathcal{S})}$ it follows that*

$$RelativeVolume(Out\mathcal{R}(\mathcal{S}_1)) \leq RelativeVolume(Out\mathcal{R}(\mathcal{S}_2))$$

Motivation for Hypothesis 2 Theorem 2 reasons on the influence of correlation on inlier and outlier regions. In *HiddenIn*$\mathcal{A}$, the outlier region is $Out\mathcal{R}(\{\mathcal{A}\})$, i.e. involves the full space. In *HiddenFrom*$\mathcal{A}$, the inlier region $In\mathcal{R}(\{\mathcal{A}\})$ involves the full space. In both settings, the respective other region depends on a *Collection* consisting of only one-dimensional subspaces. Correlation does not affect the distribution within these subspaces and hence does not affect the relative volume. Lemma 2 states that the minimum of the relative volumes of inlier and outlier region is an upper bound on the relative volume of *Hidden*. Thus, if one of the relative volumes of inlier and outlier increases or decreases this bound might do so as well.

Hypothesis 2 Correlated data leads to a smaller relative volume of *Hidden* in setting *HiddenFrom*$\mathcal{A}$ than uncorrelated data. In *HiddenIn*$\mathcal{A}$, it is larger.

If this hypothesis holds, it is more difficult to hide outliers in correlated subspaces in *HiddenFrom*$\mathcal{A}$ and less difficult in *HiddenIn*$\mathcal{A}$. We evaluate this assumption using various datasets and outlier detection methods in Sect. 5.2.2.

4.4 Algorithm for *Hidden*

So far, we have studied characteristics of *Hidden* analytically depending on certain assumptions. We now propose an algorithm which places hidden outliers, for any instantiation of $out^{\mathcal{S}}(\cdot)$, subspace selections and dataset.

Our proposed algorithm is randomised, i.e. checks for random points $\mathbf{x} \in [u, l]^d$ whether they are part of *Hidden*. The baseline we propose samples these points according to a uniform distribution with domain $[u, l]^d$. However, inspecting such samples would not only be extremely expensive, it also would not take into account that *Hidden* can be a very small portion of *Full*$\mathcal{R}$. See Fig. 2c. The red crosses indicate some areas of points in *HiddenIn*$\mathcal{A}$ and *HiddenFrom*$\mathcal{A}$. We have computed these areas by detecting outlier positions in each

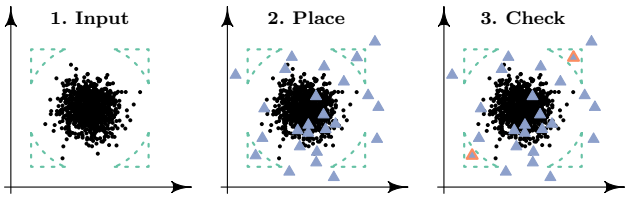

Fig. 4 Exemplary flow of our placement algorithm

attribute in isolation as well as in the full space. While the Region *HiddenIn*$\mathcal{A}$ is rather large, *HiddenFrom*$\mathcal{A}$ is not. If *Hidden* is small, an algorithm which concentrates on this part of the data space is desirable. However, the algorithm should also inspect points exhaustively otherwise. A placement that is always exhaustive however would leave aside Hypothesis 1. It has stated that hidden outliers are close to *DB*. According to it, it is unlikely that an extreme position, a point far from *DB*, is a hidden outlier. To facilitate a placement that is adaptive in this spirit, we specify a parameter to model the probability of positions to be checked. In particular, points next to existing data points $\mathbf{y} \in DB$ will have a higher likelihood.

Our algorithm basically consists of two steps. First it places random points in the data space. Second, it checks which ones are hidden outliers. Figure 4 illustrates the flow. The example resembles the scenario from Fig. 2. The method detecting outliers is based upon the Mahalanobis distance, and our data have two dimensions. In Fig. 4, we place hidden outliers in the *HiddenIn*$\mathcal{A}$ setting. That is, the hidden outliers are outliers in the full space but must not be detectable in any one-dimensional projection. The leftmost plot shows the dataset that is part of the input of the algorithm. The area framed by the dotted lines are positions where objects, when placed there, will be hidden outliers. This area depends on other inputs of the algorithm (e.g. which detection method is used in a subspace). The other two plots display the actual placement. In the second plot, random points are placed in the domain. Here, the baseline placement scheme from our paper, i.e. uniform sampling, has been used. Afterwards these placed points are filtered; only the desired hidden outliers are kept. In the following enumerated list, we discuss the probability of positions being checked and how this check is performed in detail.

4.4.1 Probability of a position

A straightforward approach would be to only check points $\mathbf{x}$ closer than some threshold to existing data objects. However, we should also consider the distance of $\mathbf{x}$ to the attribute bounds. To this end we introduce the parameter $eps \in [0, 1]$. Figure 5 graphs the probability of a position being checked, for a single attribute and data point y. We use the log scale for better illustration. The area of both rectangles is 1. If $eps = 0$

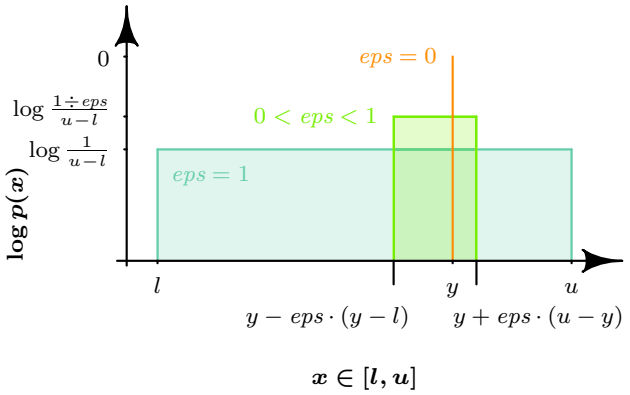

Fig. 5 Exemplary probability density of checking random points regarding a single attribute and data object y

we do not allow for any distance greater than 0 to y. Thus, the probability of checking y is 1 and of any other position 0. If $eps = 1$ we allow for any distance, as long as positions do not exceed the attribute bounds. We set the probability of checking x to be constant and thus to $\frac{1}{u-l}$. If $0 < eps < 1$, a point x between $y - eps \cdot (y - l)$ and $y + eps \cdot (u - y)$ is checked with a probability $\frac{1 \div eps}{u-l}$. Any point outside of these bounds is not checked.

Definition 5 Let $\mathbf{y}_1, \ldots, \mathbf{y}_n \in [l, u]^d$ and $eps \in [0, 1]$ be given. The **surrounding region** of a single observation $\mathbf{y}_j$ is defined as:

$$Surr\mathcal{R}^{eps}(\mathbf{y}_j) := \left\{ \mathbf{x} \in [l, u]^d \mid DistRatio^{\mathbf{y}}(\mathbf{x}) \leq eps \right\}$$

where

$$DistRatio^{\mathbf{y}}(\mathbf{x}) := \max_{i \in \mathcal{A}} \left(\frac{y^{(i)} - x^{(i)}}{u - y^{(i)}}, \frac{x^{(i)} - y^{(i)}}{y^{(i)} - l} \right)$$

The **surrounding region** of several observations $\mathbf{y}_1, \ldots, \mathbf{y}_n$ is $Surr\mathcal{R}^{eps}(\mathbf{y}_1, \ldots, \mathbf{y}_n) := \bigcup_{j=1}^{n} Surr\mathcal{R}^{eps}(\mathbf{y}_j)$.

Thus, $Surr\mathcal{R}^{eps}(\mathbf{y})$ consists of all $\mathbf{x}$ whose probability of being checked is greater than zero. Figure 6 displays the surrounding region for several points in an example where $|DB| = 4$.

The following two lemmas feature useful characteristics of the surrounding region.

Lemma 4 *If* $eps = 0$ *then* $Surr\mathcal{R}^{0}(\mathbf{y}_j) = \{\mathbf{y}_j\}$ *and* $Surr\mathcal{R}^{0}(\mathbf{y}_1, \ldots, \mathbf{y}_n) = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$.

Lemma 5 *If* $eps = 1$ *then* $Surr\mathcal{R}^{1}(\mathbf{y}_j) = Full\mathcal{R}$ *and* $Surr\mathcal{R}^{1}(\mathbf{y}_1, \ldots, \mathbf{y}_n) = Full\mathcal{R}$.

Hence, the surrounding region can consist of solely the observations themselves, the entire data space or a middle ground

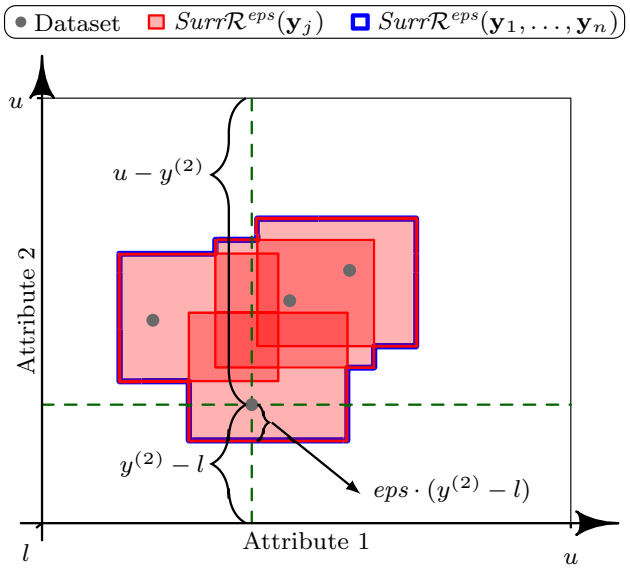

Fig. 6 Example of range distance and surrounding region

between these extremes ($0 < eps < 1$). Placing random points only in $SurrR^{eps}(DB)$ does away with the difficulties of an exhaustive placement. However, our approach so far features a parameter (*eps*), and it is unclear how to choose its value. We will discuss this in Sect. 4.4.4.

Regarding the selection of random points, a surrounding region gives way to a probability distribution from which to draw the samples, as follows

$$p(\mathbf{x}) \propto \frac{1}{n} \sum_{j=1}^{n} \mathbf{1}_{\{\mathbf{x} \in SurrR^{eps}(\mathbf{y}_j)\}} \tag{2}$$

$p(\mathbf{x})$ is the multivariate generalisation of $p(x)$ in Figure 5. We now propose the algorithm in Algorithm 1 to sample points, given the density function. The first step to obtain #*Samples* random samples from $p(\mathbf{x})$ is to randomly draw #*Samples* data points from *DB*. For every attribute value of each such point, the algorithm calculates two new positions, one towards the upper attribute limit u and one towards the lower limit l. Both are scaled by *eps*. The algorithm then determines randomly whether the sample has the position next to u or l. This results in #*Samples* random samples from $p(\mathbf{x})$.

4.4.2 Checking positions

The next step necessary to place hidden outliers is to check if a point is a hidden outlier or not. See Algorithm 2 for our algorithm. For a given point, it checks if it is an inlier in each subspace in $Collection_{inlier}$ and an outlier in at least one subspace of $Collection_{outlier}$. Thus we can filter sampled points for hidden outliers. Armed with these algorithms, it is now possible to hide outliers in $SurrR^{eps}(DB)$, for given a dataset *DB*, *eps* and #*Samples*, as long as $RelativeVolume(Hidden) > 0$. In the following we will discuss an alternative interpretation of *eps* which allows to choose *eps* and define the risk of hidden outliers.

Algorithm 1 Sample points using the pdf from Equation 2

Require: #Samples, *eps*, *DB*
Ensure: Samples $\mathbf{x}_1, \ldots, \mathbf{x}_{\#Samples}$
Sample $\mathbf{y}_1, \ldots, \mathbf{y}_{\#Samples}$ from *DB* (with replacement)
for $\mathbf{y}_j \in \mathbf{y}_1, \ldots, \mathbf{y}_{\#Samples}$ **do**
  for attribute i of $\mathbf{y}_j$ **do**
    lowerChange ← Sample from Unif(0, $y_j^{(i)} - l$)
    upperChange ← Sample from Unif(0, $u - y_j^{(i)}$)
    Select at random if
    $x_j^{(i)} \leftarrow y_j^{(i)} - eps \cdot$ lowerChange
    or
    $x_j^{(i)} \leftarrow y_j^{(i)} + eps \cdot$ upperChange
  end for
end for
return $\mathbf{x}_1, \ldots, \mathbf{x}_{\#Samples}$

Algorithm 2 Filter sample points for hidden outliers

Require: $Collection_{outlier}$, $Collection_{inlier}$, $\mathbf{x}_1, \ldots, \mathbf{x}_{\#Samples}$, *DB*, $out^{\mathcal{S}}(\cdot)$
Ensure: Set of hidden outliers *SamplesHidden*
for $\mathbf{x}_j \in \mathbf{x}_1, \ldots, \mathbf{x}_{\#Samples}$ **do**
  IsInlierInAll ← TRUE
  IsOutlier ← FALSE
  for type ∈ {outlier, inlier} **and for** $\mathcal{S} \in Collection_{\text{type}}$ **do**
    outRes ← $out^{\mathcal{S}}(\mathbf{x}_j)$
    if outRes ≠ type = inlier **then**
      IsInlierInAll ← FALSE
    end if
    if outRes = type = outlier **then**
      IsOutlier ← TRUE
    end if
  end for
  if IsInlierInAll & IsOutlier **then**
    Add $\mathbf{x}_j$ to *SamplesHidden*
  end if
end for
return *SamplesHidden*

4.4.3 Interpreting eps

To motivate our interpretation we revisit Fig. 6. The region $SurrR^{eps}(\mathbf{y}_1, \ldots, \mathbf{y}_n)$ with the blue surrounding is a boundary for the observations. *eps* controls its tightness. Lemma 5 has stated that, if $eps = 0$, $SurrR^1(DB)$ contains each point from *DB*. Thus $p(\mathbf{x})$ from Eq. 2 is the empirical distribution function of *DB*. If $eps \approx 0$, $p(\mathbf{x})$ still is similar to the empirical distribution. However, the closer *eps* is to 1, the less similar the empirical data distribution is to $p(\mathbf{x})$. Their similarity quantifies how much knowledge $p(\mathbf{x})$ reveals on observations in *DB*. Thus, one can interpret *eps* as an indi-

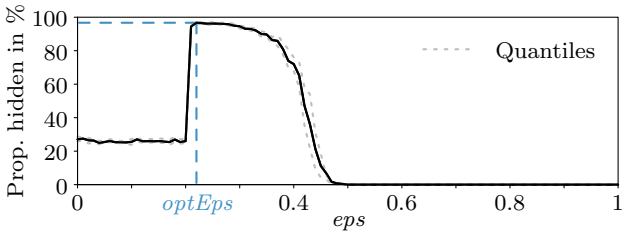

Fig. 7 Demonstration of the connection of the proportion of hidden outliers in placed points and *eps*. Using *HiddenFrom*$\mathcal{A}$, Arrhythmia and DBOut (See Sect. 5.1). The risk is 0.36

cation of how much information on the data (*DB*) is used in the sampling procedure.

4.4.4 Choosing eps

Up to here, *eps* is an exogenous parameter, without the flexibility envisioned. Thus we now add one step to the algorithm. Figure 7 graphs the proportion of sampled points that are hidden outliers in one example setting. The maximum is reached at $eps \approx 0.3$. Hence, in this example, 0.3 is the value that allows for the best placement of hidden outliers. We refer to the *eps* that maximises the proportion of hidden outliers as *optEps*. This is not just the optimum for *eps* but also allows for the computation of the risk of hidden outliers. Usually there is no knowledge on the dependency between *eps* and the proportion of hidden outliers. This is why we propose to use a genetic algorithm to find *optEps*.

4.4.5 The risk of hidden outliers

Definition 6 Let *DB*, $out^{\mathcal{S}}(\cdot)$, $Collection_{inlier}$ and $Collection_{outlier}$ be given. The *risk of attacker success* is the harmonic mean of *optEps* and the proportion of hidden outliers the attacker is able to hide in the surrounding region $Surr\mathcal{R}^{optEps}(DB)$.

This risk has domain [0, 1]. If hiding outliers is difficult, the risk of the data owner is small. Recall our interpretation of *eps* as the amount of information known on the data. $optEps \approx 1$ means that an attacker does not need any information on *DB* to place hidden outliers. If both *optEps* and the maximal proportion of hidden outliers are high, it is easy to hide outliers, and the risk is high. If only one of them or both are low, the risk is also low. Hence, the risk is low if an attacker either needs much knowledge on the data and/or placed points rarely are hidden outliers.

4.4.6 Complexity

The algorithm we propose exclusively targets at high result quality. We deem absolute runtime less important, as long as it is not excessive, since a data owner will conduct the analysis proposed here offline. Having said this, we nevertheless discuss the worst case complexity of our solution. When placing hidden outliers for a given *eps*, the algorithms performs $\#Calc = \#Samples \cdot |Collection_{inlier}| \cdot |Collection_{outlier}|$ calculations. Let $maxFitEval$ be the maximal number of fitness-function evaluations by the genetic algorithm.

Lemma 6 *The worst case complexity of our algorithm is* $O(\#Calc \cdot maxFitEval)$.

4.4.7 Summary of the algorithm

The algorithm needs four inputs: the two subspace collections ($Collection_{inlier}$, $Collection_{outlier}$), an outlier detection method ($out^{\mathcal{S}}(\cdot)$) and the number of samples (#Samples). An additional optional input is *eps*; when not supplied, the algorithm itself determines *eps* (*optEps*). First, the algorithm obtains #Samples points from the probability distribution in Eq. 2 (See Algorithm 1). Then these points are filtered. Only points that are inlier in each subspace of $Collection_{inlier}$ and outlier in at least one subspace of $Collection_{outlier}$ according to $out^{\mathcal{S}}(\cdot)$ remain. See Algorithm 2. They are hidden outliers. When *eps* is not given, the algorithm repeats this procedure for different values of *eps*. A heuristic generates values of *eps*. They target at maximising the share of hidden outliers in each sample.

5 Experiments

In Sect. 4.3, we have derived characteristics of *Hidden* analytically, assuming a specific outlier definition and underlying data distribution. In our experiments we investigate its behaviour in terms of other outlier definitions and datasets using our algorithm. The experiments show the general ability of our algorithm to place hidden outliers and the vulnerability of different detection methods. We also study the role of *optEps*, e.g. whether there exists a unique one.

5.1 Experiment setup

5.1.1 Outlier detection

Additionally to the Mahalanobis distance we investigate three other outlier definitions. One of them, follows the (k, $dmax$)-Outlier, short DBOut, proposed in [9]. A point is an outlier if at most k objects have a distance less than $dmax$. The distance used is the euclidean metric. Hence, solely the dimensionality of a subspace implies different magnitudes of distances [27]. That is why, instead of using a fixed $dmax$ for each subspace, we use an adaptive $dmax$. In particular, we set $dmax = 0.2 \cdot \sqrt{|\mathcal{S}|}$. The factor $\sqrt{|\mathcal{S}|}$ is used to

scale the distances to the size of the subspace. If the subspace is of size 1, the maximal Euclidian distance is $1 = \sqrt{1}$. (Datasets are normalised to [0, 1].) In general, the maximal Euclidian distance between two points is $\sqrt{|\mathcal{S}|}$. Hence, the distance of an inlier to its nearest neighbour can be at most 20% of the maximal distance. The last two methods we use are ABOD [12] and LoOP [10]. ABOD uses angles to obtain the outlierness of a point. These angles are much more stable in higher dimensions than $\mathcal{L}_p$-distances. [12] proposes three different implementations of ABOD which incorporate different tradeoffs between performance and result quality. We use the fastest implementation, FastABOD. LoOP is an adoption of the well-known LOF [2] that incorporates a density based on the neighbourhood of data points. This allows to find density based outliers without assuming a specific distribution. In comparison with LOF, LoOP returns a score that lies in [0, 1] and implies an outlier probability instead of a score in $[0, \infty]$. Except for FastABOD and LoOP all methods already output a binary signal if a data point is an outlier or not. FastABOD and LoOP output scores. Regarding LoOP a low score indicates usual observations, as for FastABOD a high score. In our experiments, we need an automatic threshold that allows to transform that score to a binary signal. Regarding FastABOD we decided to use the empirical 2.5 % quantile of the resulting scores to this end. For LoOP we used a threshold of 0.5. We set the neighbourhood size to $k = 5$.

5.1.2 Datasets

Two datasets we use are artificial and 14 are real-world benchmark datasets, including two high-dimensional datasets from the UCI ML Repository,[2] namely Madelon and Gisette (500 and 5,000 attributes). The remaining real-world datasets are from [3]. We always use the normed data, and when the data have been down-sampled we use version one. The two artificial datasets are produced by sampling from a multivariate Gaussian distribution, each with 500 observations and 30 attributes. One dataset is from a MVN distribution where each attribute is i.i.d N(0,1) , referred to as '*MVN* cor.'. In the second one '*MVN*', where attributes are N(0, 1) distributed as well, each pair of attributes is correlated with a covariance of 0.8. They are mostly used for the experiments studying Hypothesis 2. To obtain comparability across datasets, each datasets has been normalised (i.e. $l = 0$ and $u = 1$).

5.1.3 Subspace selection

To evaluate our theoretical findings, we have a deterministic procedure for subspace selection (*HiddenIn*$\mathcal{A}$ and *HiddenFrom*$\mathcal{A}$). Only for Theorem 1 we need to sample subspace partitions. However, when evaluating the general quality of our approach, instantiations of $Collection_{inlier}$ and $Collection_{outlier}$ are much less obvious. We need *realistic* and *diverse* instantiations. However, the number of possible combinations is daunting. On the other hand, the relationship of subspace size and number of possible subspaces, exemplary displayed in Fig. 8, implies the following: We assume that one normally checks low- and high-dimensional subspaces for outliers (green area). Hence, it is most likely that hidden outliers occur in the subspaces with a medium number of dimensions (red area). Thus, these outlier subspaces are a natural selection. However, even with these restrictions, the number of outlier and even inlier subspaces can still be infeasibly large. Thus, we sample them according to the procedure in Algorithm 3.

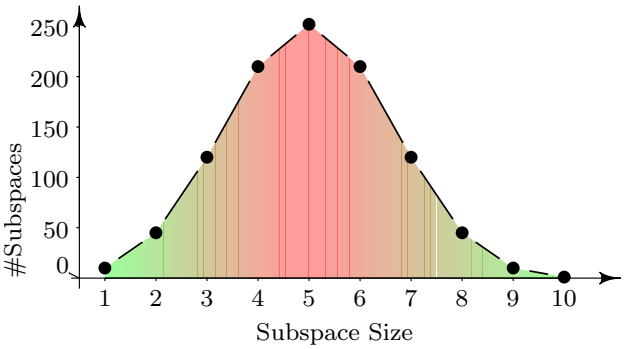

Fig. 8 Number of subspaces versus subspace size (using ten attributes) (colour figure online)

Algorithm 3 Sample inlier and outlier subspaces

Require: #Subspaces, $\mathcal{A} = \{1, \ldots, d\}$, $out^S(\cdot)$
Ensure: Subspace collections for inlier and outlier
1: **if** $out^S(\cdot)$ is FastABOD or LoOP **then**
2: $\text{combs}_{\text{inlier}} \leftarrow \{\mathcal{S} \subseteq \mathcal{A} : |\mathcal{S}| \geq d - 2\}$
3: **else**
4: $\text{combs}_{\text{inlier}} \leftarrow \{\mathcal{S} \subseteq \mathcal{A} : |\mathcal{S}| \leq 2\}$
5: **end if**
6: $Collection_{inlier} \leftarrow$ Sample #Subspaces from $\text{combs}_{\text{inlier}}$
7: $\tilde{\mathcal{A}} \leftarrow \{a \in \mathcal{S} : \exists\, \mathcal{S} \in \text{inlierCombs}\}$
8: $\text{combs}_{\text{outlier}} \leftarrow \{\mathcal{S} \subseteq \tilde{\mathcal{A}} : |\mathcal{S}| = \lfloor \frac{d}{2} \rfloor\}$
9: $Collection_{outlier} \leftarrow$ Sample #Subspaces from $\text{combs}_{\text{outlier}}$
10: **return** $Collection_{inlier}$, $Collection_{outlier}$

First we sample subspaces for $Collection_{inlier}$. For outlier detection methods for high-dimensional spaces (LoOP and FastABOD), we use the large subspaces as inlier subspaces (right green area). For the other outlier detection techniques, we use the smaller subspaces (left green area). Then we obtain the attributes that are contained in the sampled inlier subspaces. From those we sample the outlier subspaces. This guarantees that attributes from inlier and outlier subspaces overlap.

[2] UCI ML Repository: http://archive.ics.uci.edu/ml/.

Table 1 Percentage of runs with more than zero hidden outlier

	MDist	DBOut	LoOP	FastABOD
*	64.29	77.86	87.14	95.36
**	37.86	87.50	96.43	95.71

* $Collection_{outlier} = \mathcal{A}$, ** $Collection_{inlier} = \mathcal{A}$

Table 2 Difference in percentage of hidden outliers

Values in %	*MDist*	DBOut	LoOP	FastABOD
(a) *HiddenIn*$\mathcal{A}$				
MVN	28.14	56.54	12.96	1.12
MVN cor.	69.84	1.32	69.20	1.26
Difference	36.64	13.30	56.24	0.14
Relative	56.56	19.04	81.27	11.11
(b) *HiddenFrom*$\mathcal{A}$				
MVN	0.84	0.62	29.08	21.50
MVN cor.	0.30	19.68	20.76	22.00
Difference	−0.48	−0.32	−9.40	−0.50
Relative	−57.14	−51.61	−32.32	2.27

5.2 Evaluating Our theoretical findings

In the first experiments, we investigate the generalisability of our theoretical findings from Sect. 4.3. The experiments approximate the scenarios described in the theorems and hypotheses using various datasets and outlier detection methods, cf. Sect. 5.1.

5.2.1 Theorem 1

The theorem states that hidden outliers exist when either $Collection_{inlier}$ or $Collection_{outlier}$ is a partition of the full attribute space $\mathcal{A}$ into subspaces, and the other one is $\mathcal{A}$ itself. We investigate the generalisability of this statement by varying the data distribution and the outlier detection method. For all datasets we create a number *Collection*s by randomly dividing $\mathcal{A}$ into partitions. For each outlier detection scheme and *Collection* we compute the maximal proportion of hidden outliers regarding two selections of $Collection_{inlier}$ and $Collection_{outlier}$. With the first one, $Collection_{inlier}$ equals *Collection* and $Collection_{outlier} = \mathcal{A}$. Vice versa in the other case, $Collection_{outlier}$ equals *Collection* and $Collection_{inlier} = \mathcal{A}$. We record how often the proportion of hidden outliers is not 0, i.e. one can hide outliers. If Theorem 1 is generalisable, this should be possible. Table 1 lists the percentages of runs where we have been able to hide outliers.

In most cases, our algorithm is able to place hidden outliers. Surprisingly, regarding *MDist* in particular, the success rate is rather low. This is in some contrast to our formal result that states there exist hidden outliers in these cases. The other detection methods show higher success rates. Thus, we conclude that Theorem 1 is generalisable to some extent.

5.2.2 Hypothesis 2

The hypothesis states that it is more difficult to place inliers in correlated subspaces in setting *HiddenFrom*$\mathcal{A}$ and less difficult in setting *HiddenIn*$\mathcal{A}$. To investigate this we try to hide outliers in both settings using different outlier detection schemes. In one dataset, attributes are correlated (*MVN* corr.). In another one they are not (*MVN*). To focus on the effects of correlation, the data have only 10 attributes. If Hypothesis 2 holds we should see an increase in the proportion of hidden outliers from uncorrelated with correlated data in *HiddenIn*$\mathcal{A}$ and a decrease in *HiddenFrom*$\mathcal{A}$. Table 2 lists the results: the percentage obtained in each dataset, the raw difference between the two results and a relative difference. The last entry is obtained by dividing the raw difference by the maximal percentage the detection algorithm has obtained in any of the two datasets.

All detection methods except for FastABOD meet the expectation. We find it interesting that the magnitude of change of proportion is very different for *HiddenIn*$\mathcal{A}$ and *HiddenFrom*$\mathcal{A}$. However, the proportion of placed hidden outliers on each dataset also varies greatly. In summary, although the extents are different, the experiments confirm the hypothesis to some extent.

5.3 Investigating *optEps*

The next experiments target at a crucial parameter of our algorithm. The proposed algorithm is based on a sampling distribution parametrised by *eps*. The *eps* that maximises the proportion of hidden outliers, defined as *optEps*, is important: it allows to quantify the risk of data owners. We now investigate the dependency between *eps* and that proportion, to analyse if *optEps* usually exists, i.e. if there is a global maximum of the dependency.

Figures 7 and 9 illustrate the dependency between the proportion of hidden outliers and *eps*. In Figs. 7 and 9a we see a very distinct *optEps*. However, Fig. 9b shows that this is not always the case. The risk is highest with 0.45 in Fig. 7. *optEps* as well as the maximal proportion are relatively high. Figure 9b shows the lowest risk with 0.12.

5.4 General quality

In these final experiments, we study the general ability of our algorithm to place hidden outliers. We also compare our approach to a baseline. Since this article is first to study hidden outliers, there is no explicit competitor, and we choose

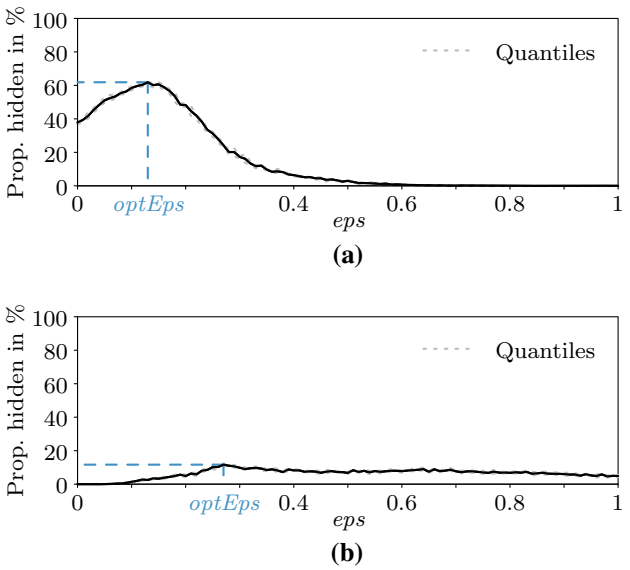

Fig. 9 Proportion of hidden outliers in placed points versus **a** *HiddenFrom*$\mathcal{A}$, Parkinson and LoOP. Risk: 0.21 *eps*. **b** *HiddenIn*$\mathcal{A}$, Lymphography and MDist. Risk: 0.16

uniform full space sampling as baseline, which is equivalent to fixing *eps* to 1. We will declare success if there is an increase in the quality of placing hidden outliers, and this increase is significant, e.g. a factor of at least two or three. Recall that our algorithm requires data, $out^{\mathcal{S}}(\cdot)$, $Collection_{inlier}$, $Collection_{outlier}$ as input. The subspace selection has been derived in Sect. 5.1.3. Regarding the data, we look at all datasets introduced in 5.1.2 and sample different numbers of attributes and observations. However, we only downsample, upsampling would lead to data that is redundant. To vary $out^{\mathcal{S}}(\cdot)$, we use all outlier detection methods introduced in 5.1.1. Additionally, we obtain *eps* candidates by using a fixed sequence of values instead of a heuristic. This allows for further analysis on the effect of *eps* and a straightforward comparison to the baseline. We summarise our results to highlight the effect of the number of attributes or observations, used dataset or detection method and *eps*.

5.4.1 Number of attributes

Figure 10 graphs the effect of the number of attributes. The *y*-axis displays the share of hidden outliers amongst sampled points, i.e. the success of the placement. For algorithm and baseline the figure shows boxplots of the share of hidden outliers. We observe a significant improvement of our algorithm over the baseline—for high-dimensional data in particular. Second, for both alternatives it seems to be more difficult to place hidden outliers in high-dimensional datasets. The shape of the left plot is caused by two effects. One is that it is difficult to hide outliers with few attributes. Any subspace-based outlier detection scheme will most likely detect them. If we

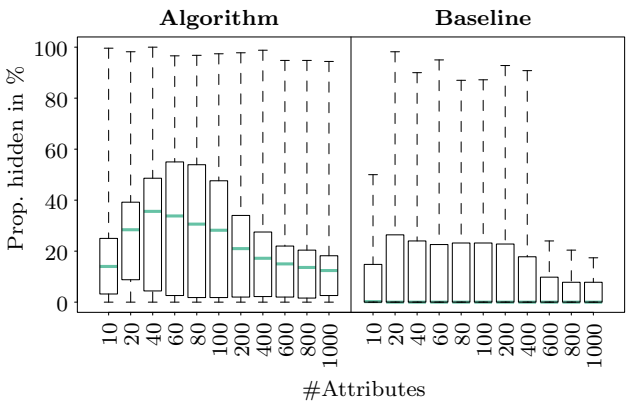

Fig. 10 Effect of number of attributes

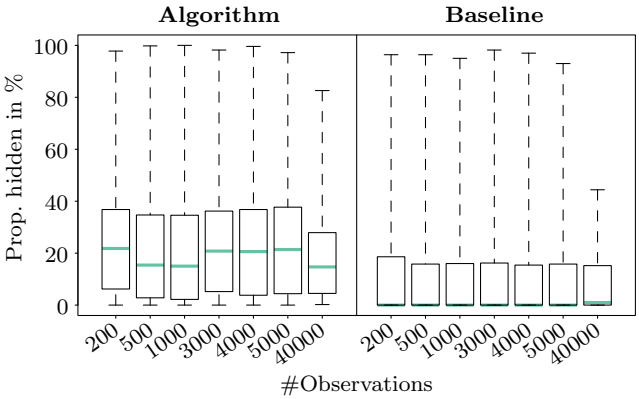

Fig. 11 Effect of number of observations

increase the number of attributes, this effect decreases. However, the second influence is that increasing the number of attributes also increases the number of subspaces searched for outliers. So it is very difficult to find positions that are inliers in each subspace of $Collection_{inlier}$. In consequence, it also is difficult to hide outliers.

5.4.2 Number of observations

Figure 11 plots the number of observations versus the share of hidden outliers. Again, our algorithm is better than the baseline, but with a bit less distinction. The algorithm improves the baseline by a factor of about 5–10. In both cases the effect of the number of observations is not very significant. This is because the number of observations does not change the data distribution much. That is, dense areas will still be dense, and sparse areas will still be sparse.

5.4.3 Dataset used

Table 3 displays the effect of the datset. We summarise the results over all experiments, i.e. with different samplings of observations and attributes. As before, our algorithm outper-

Table 3 Proportion of hidden outliers regarding dataset used

Dataset	Algorithm (in %)	Baseline (in %)
ALOI	18.12	7.08
Annthyroid	13.07	7.13
Arrhythmia	23.64	8.23
Cardiotocography	24.57	7.29
Gisette	33.10	7.72
HeartDisease	29.53	6.85
InternetAds	6.29	3.54
Ionosphere	40.65	15.13
KDDCup99	16.32	10.73
Madelon	33.84	20.03
MVN	30.08	11.17
MVN corr.	18.14	5.49
PenDigits	21.72	5.34
SpamBase	21.27	7.06
Waveform	27.08	13.28
WDBC	22.26	5.93

forms the baseline significantly. We see however that there are drastic differences between datasets. While placing hidden outliers is successful when using Madelon, this is more difficult with, say, InternetAds. We speculate that the different data densities cause this effect. We have seen this effect in our experiments in Sect. 5.2.2 as well.

5.4.4 Measuring dataset characteristics

We now want to further investigate the difference in the proportion of successfully placed hidden outliers regarding the datasets used. Thus, we have conducted an intensive and systematic study, as follows: we have developed some measures that we then have correlated with the characteristics in Table 3. Our measures divide into three types that address the subspaces selected, the amount of irrelevant attributes and the quality of outlier detection. The first type of measure addresses the correlation in subspaces. As part of our formal results, we have already derived that this correlation should have an effect. In particular, we take the average of the Spearman and Pearson correlation in both subspace collections ($Collection_{inlier}$ and $Collection_{outlier}$). When a subspace has more than two dimensions, the correlation measure is the average of each pair of dimensions. Regarding the second type, one measure uses a principal component analysis (PCA) computed on each dataset. Each PC has a score dedicated to its importance in explaining the data objects. If there are only few PCs with very high score and many with low scores, the dataset has a rather low intrinsic dimensionality, i.e. only a few transformed attributes are necessary. A low intrinsic dimensionality means that many irrelevant attributes are present, which might affect our placement: the number of irrelevant features could reduce the share of hidden outliers, by making it difficult to place them. However, it might also be that it increases their proportion, by blurring low-dimensional outliers so that they are inliers in higher dimensions. This is why we have quantified the skewedness of the PCA scores. If this skewedness is high, it is likely that there are only a few PCs with a high score. All our datasets are anomaly detection benchmark datasets, i.e. they include labels for outliers and inliers so that one can use them to evaluate outlier detection algorithms. For the remaining measures, we have fit a random forest to each dataset that distinguishes between the labelled inliers and outliers. We can derive measures from this random forest fit for two of our types: the importance of attributes which belongs to the second type and the quality of the inlier/outlier-classification which is of the third type. In line with the usual definition, the importance of an attribute is the Gini index decrease with this attribute. If it is high, it is important for the classification. From the attribute importance values, we compute the mean value, variance, and skewedness, i.e. how important are the attributes in general, how much does this vary and whether there only are a few attributes that are very important. From the classification, we obtain accuracy, sensitivity (how well are outliers detected) and specificity (how well are inliers detected). We then have computed the correlation between all these measures and the percentage of hidden outliers placed by our algorithm. The results are listed in Table 4. We have separated them by the type of inlier subspaces: low- (up to 2 dimensions) or high-dimensional (at least d-2 dimensions).

Table 4 Correlation of data measurements with percentage of hidden outliers

Measure	Low	High
Average Pearson (Inlier)	0.32	0.30
Average Spearman (Inlier)	0.04	0.14
Average Pearson (Outlier)	0.21	0.45
Average Spearman (Outlier)	−0.06	0.20
Importance Mean	−0.24	0.06
Importance Variance	−0.29	−0.07
Importance Skewness	−0.20	−0.39
PCA Skewness	0.00	0.16
Accuracy	−0.35	−0.45
Specificity	−0.16	−0.42
Sensitivity	−0.41	0.24

We see that many measures influence the percentage of hidden outliers placed. As expected, correlation has an effect. However, the effect in the low-dimensional setting is not very prominent. This might result from the many diverse subspaces that the scores are averaged from, in the low-

dimensional setting in particular. The feature importance has an overall negative effect although the mean is not significant in the high-dimensional setting. This makes sense: if there are only a few important features that are well suited for outlier detection, it is more difficult to hide from these. To illustrate, think of a dataset with four attributes. The first two contain outliers that are easily detectable (see for instance Fig. 1). In the other two attributes, the outlier and inlier class are scattered randomly and are indifferent. As outliers and inliers are indifferent, it is likely that the subspace follows a distribution that makes it difficult for a point to be an outlier. For instance, this is the case with an independent uniform distribution. However, hidden outliers must be outlying in certain subspaces. If this subspace consists of irrelevant attributes (here, the third and the fourth attribute), placing outliers might be difficult. In the high-dimensional setting, the PCA skewedness seems to have an effect. This might be because the attributes that are more than the intrinsic dimensionality allow for a good placement by blurring the result. Remember that in the high-dimensional setting the inlier subspaces ($Collection_{inlier}$) have more dimensions. The accuracy has a negative effect. Clearly, the better the random forest is in detecting outliers, the more difficult it is to hide such. Regarding specificity, the effect is the same. The better the random forest can detect inliers, the less effective is our placement. Regarding the sensitivity, we do not see an obvious interpretation of the results. The effect is opposite for the high- and the low-dimensional settings.

To conclude, we have experimented with various measures to quantify effects that go along with different shares of successfully placed hidden outliers. However, the main takeaway is that there is not one single measure or a few measures in combination correlated with a successful placement.

5.4.5 Outlier detection method used

To determine effects of the method used, we aggregate the percentage of successful runs and the average share of hidden outliers of all experiments. We have done this for our algorithm as well as for the baseline. See Table 5. Some detection methods are very prone to hidden outliers while others are not. Next, the gain in success when using our algorithm varies amongst detection techniques. With *MDist* this gain is high, while it is negligible with FastABOD. However, it is possible to hide outliers with all five detection techniques.

5.4.6 eps used

Figure 12 displays the proportion of hidden outliers versus *eps*. $eps = 1$ is our baseline, i.e. uniform sampling. The median has a peak when *eps* is 0.1. This value results in hidden outliers placed closely to other data points. This confirms Hypothesis 1, i.e. hidden outliers are spatially close to the points in *DB*. The figure also confirms the superiority of our algorithm over the baseline.

Table 5 Effect of the used detection method on the success in hiding outliers and their average proportion

Values in %	*MDist*	DBOut	LoOP	FastABOD
(a) Percentage of runs with placed hidden outliers				
Algorithm	99.32	81.19	97.70	96.06
Baseline	4.46	36.49	21.81	95.78
(b) Average proportion of hidden outliers				
Algorithm	7.67	30.28	30.13	26.41
Baseline	0.03	8.61	3.31	22.26

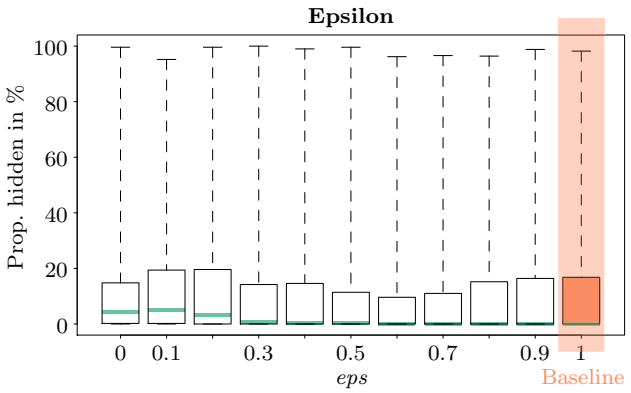

Fig. 12 Effect of *eps*

5.4.7 Summary

The experiments have shown that in many scenarios our approach is able to place hidden outliers irrespective of the dataset or the detection method used. Further, our approach is a significant improvement over the baseline. While not in all cases, our algorithm has improved the result by a factor of three or more in many settings.

6 Conclusions

In this work, we have analysed characteristics of hidden outliers, i.e. outliers that are only detectable in certain attribute subspaces. This includes both formal results based on model assumptions and a proposal for an algorithm that places hidden outliers in data. Regarding the first kind of contribution, we prove the existence of hidden outliers in many scenarios and show that the extent of correlation can have a significant effect on the ease of hiding outliers. The algorithm we have developed places hidden outliers in regions close to existing data objects. We evaluate the generalisabil-

ity of our formal results experimentally with our algorithm. Some of these results do extend to scenarios not covered by the model assumptions. Further we have shown that our algorithm improves the results with a reference baseline significantly.

In the future, we want to further analyse data characteristics that go along with hidden outliers. Having identified such characteristics, one might be able to develop an approach for placing hidden outliers that relies on this information and thus might be more successful. Another future research direction is to improve subspace search using hidden outliers. For instance, this might be done by analysing placed hidden outliers. That is, one could use placed hidden outliers to detect subspaces that should be searched.

Acknowledgements This work was supported by the German Research Foundation (DFG) as part of the Research Training Group GRK 2153: Energy Status Data–Informatics Methods for its Collection, Analysis and Exploitation.

Compliance with ethical standards

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

A Prerequisites for Proofs

We will use the abbreviations $Collection =: \mathcal{SS}, In\mathcal{R}(\{\mathcal{S}\}) =: \mathcal{I}^{\mathcal{S}}$ and $u - l =: range$.

The Mahalanobis distance is defined as: $MDist^{\mathcal{A}}(\mathbf{y}) = \sqrt{(\mathbf{y}-\boldsymbol{\mu})^T \Sigma^{-1}(\mathbf{y}-\boldsymbol{\mu})}$ where $\boldsymbol{\mu}$ is the mean vector and Σ the covariance matrix. W.l.o.g. we assume that $\boldsymbol{\mu} = \mathbf{0}$. As the data are MVN distributed, MDist^2 is χ^2_d distributed. The degrees of freedom are determined by the dimension of the data. We can rewrite:

$$\left[MDist^{\mathcal{A}}(\mathbf{y})\right]^2 = \mathbf{y}^T \Sigma^{-1} \mathbf{y} = \sum_{i\in\mathcal{A}} \sum_{j\in\mathcal{A}} y^{(i)} \cdot \sigma_{-1}^{(i,j)} \cdot y^{(j)}$$

$\sigma_{-1}^{(i,j)}$ denotes the entry in the jth column and ith row of the inverse of the covariance matrix. If the attributes are i.i.d. N(0,1) distributed, this reduces to

$$\left[MDist^{\mathcal{A}}(\mathbf{y})\right]^2 = \sum_{i\in\mathcal{A}} \left[y^{(i)}\right]^2$$

To test a data point for outlier or inlier, we used the outlier initialisation formalised in Sect. 4.2. Although this initialisation uses the 0.975 quantile, here we will use a general α quantile. The quantile function of a χ^2 distribution is not obtainable in closed form. Thus, we will make use of an approximation. Following the central limit theorem, for large degrees of freedom a χ^2_d distribution can be approximated by a $N(d, \sqrt{2d})$ distribution. Thus:

$$\begin{aligned} Quantile(\alpha,\ d) &\approx d + \sqrt{2d}\, z_\alpha \\ \frac{\partial\ Quantile(\alpha,\ d)}{\partial\ d} &\approx 1 + \frac{z_\alpha}{\sqrt{2d}} \end{aligned} \tag{3}$$

where z_α is the α quantile of a standard normal distribution. We can derive that the approximation of the function $Quantile(\alpha,\ d)$ is strictly monotonic increasing. For a fixed distance, e.g. $\left[MDist^{\mathcal{A}}(\mathbf{y})\right]^2 = Quantile(\alpha,\ d)$, the Mahalanobis distance exhibits an ellipsoid form. That is, having $\lambda_1, \ldots, \lambda_d$ eigenvalues and $\mathbf{v}_1, \ldots, \mathbf{v}_d$ eigenvectors of Σ, the ellipsoid has centroid $\boldsymbol{\mu}$, and axes $\mathbf{v}_1, \ldots, \mathbf{v}_d$. Half the length of each axis is determined by $\sqrt{\lambda_i \cdot Quantile(\alpha,\ d)}$.

Introducing subspaces in this setting is quite trivial. We assume that the full data space is $MVN(\mathbf{0}, \Sigma)$ distributed. Hence, any subspace $\mathcal{S}$ is also Gaussian. To obtain its mean and covariance matrix, we only need to drop the irrelevant variables from each parameter of the full space distribution.

B Proofs of Theorems

B.1 Theorem 1

In this proof, we will use the normal approximation given in Eq. 3. From attributes i.i.d $N(0, 1)$ and partitioning $\mathcal{SS}$ follows:

$$\begin{aligned} \left[\text{MDist}^{\mathcal{A}}(\mathbf{y})\right]^2 &= \sum_{i\in\mathcal{A}} \left[y^{(i)}\right]^2 = \sum_{\mathcal{S}\in\mathcal{SS}} \sum_{i\in\mathcal{S}} \left[y^{(i)}\right]^2 \\ &= \sum_{\mathcal{S}\in\mathcal{SS}} \left[\text{MDist}^{\mathcal{S}}(\mathbf{y})\right]^2 \end{aligned}$$

We first prove that a data point $\mathbf{o}_1$ with

$$o_1^{(i)} = \begin{cases} \sqrt{\frac{Quantile(\alpha,\ |\mathcal{A}|)}{|\mathcal{S}|}} & \text{if } i \in \mathcal{S} \\ 0 & \text{otherwise} \end{cases}$$

is an outlier for any $\mathcal{S} \in \mathcal{SS}$ but an inlier for $\mathcal{A}$. We know that $\text{MDist}^{\mathcal{A}}(\mathbf{o}_1) = Quantile(\alpha,\ |\mathcal{A}|)$ and $\text{MDist}^{\mathcal{S}}(\mathbf{o}_1) = Quantile(\alpha,\ |\mathcal{A}|) > Quantile(\alpha,\ |\mathcal{S}|)$. The quantile function is strictly monotonic increasing. Thus, $\mathbf{o}_1$ is an inlier regarding $\mathcal{A}$ but an outlier in $\mathcal{S}$. It is important to note that $\mathbf{o}_1$ is an outlier only regarding subspace $\mathcal{S}$ and not regarding any other subspace in $\mathcal{SS}$.

Moreover, a data point $\mathbf{o}_2$ defined by

$$o_2^{(i)} = \frac{\sqrt{Quantile(\alpha,\ |\mathcal{S}|)}}{|\mathcal{S}|} \quad \text{for } \mathcal{S} \in \mathcal{SS} \ :\ i \in \mathcal{S}$$

is an outlier for $\mathcal{A}$ but an inlier for all $\mathcal{S} \in \mathcal{SS}$. It holds that: $\text{MDist}^{\mathcal{A}}(\mathbf{o}_2) = \sum_{\mathcal{S}\in\mathcal{SS}} \mathit{Quantile}(\alpha,\ |\mathcal{S}|)$ and $\text{MDist}^{\mathcal{S}}(\mathbf{o}_2) = \mathit{Quantile}(\alpha,\ |\mathcal{S}|)$. We know that $\mathbf{o}_2$ is an outlier if

$$\text{MDist}^{\mathcal{A}}(\mathbf{o}_2) > \mathit{Quantile}(\alpha,\ |\mathcal{A}|)$$

We also know that $|\mathcal{A}| = \sum_{\mathcal{S}\in\mathcal{SS}} |\mathcal{S}|$. Hence, in order to show that $\mathbf{o}_2$ is an outlier in $\mathcal{A}$, we have to show:

$$\sum_{\mathcal{S}\in\mathcal{SS}} \mathit{Quantile}(\alpha,\ |\mathcal{S}|) \overset{!}{>} \mathit{Quantile}\left(\alpha,\ \sum_{\mathcal{S}\in\mathcal{SS}} |\mathcal{S}|\right)$$

$$\sum_{\mathcal{S}\in\mathcal{SS}} \left[|\mathcal{S}| + \sqrt{2|\mathcal{S}|}\, z_\alpha\right] > \sum_{\mathcal{S}\in\mathcal{SS}} |\mathcal{S}| + \sqrt{2 \sum_{\mathcal{S}\in\mathcal{SS}} |\mathcal{S}|}\, z_\alpha$$

$$\sum_{\mathcal{S}\in\mathcal{SS}} \sqrt{|\mathcal{S}|} > \sqrt{\sum_{\mathcal{S}\in\mathcal{SS}} |\mathcal{S}|}$$

$$\sum_{\mathcal{S}_1,\mathcal{S}_2\in\mathcal{SS}} \sqrt{|\mathcal{S}_1|}\sqrt{|\mathcal{S}_2|} > \sum_{\mathcal{S}\in\mathcal{SS}} |\mathcal{S}|$$

$$\sum_{\mathcal{S}_1\neq\mathcal{S}_2\in\mathcal{SS}} \sqrt{|\mathcal{S}_1|}\sqrt{|\mathcal{S}_2|} + \sum_{\mathcal{S}\in\mathcal{SS}} |\mathcal{S}| > \sum_{\mathcal{S}\in\mathcal{SS}} |\mathcal{S}|$$

As $\mathcal{SS}$ is a nontrivial partition, i.e. $\mathcal{SS} \neq \mathcal{A}$, the term $\sum_{\mathcal{S}_1\neq\mathcal{S}_2\in\mathcal{SS}} \sqrt{|\mathcal{S}_1|}\sqrt{|\mathcal{S}_2|}$ is greater than 0, and the inequality holds.

B.2 Theorem 2

This theorem relies on an assumption not explicitly listed in the body of the article. Let λ denote the eigenvalue of Σ_1 (algebraic multiplicity of d). Further let $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_d$ denote the eigenvalues of Σ_2. We introduce $\varepsilon_1, \ldots, \varepsilon_d$ which satisfy $\lambda + \varepsilon_i = \tilde{\lambda}_i$. Our assumption is that the ε_i's are symmetrical, i.e. for any $\varepsilon_j > 0$ there exist $\varepsilon_k = -\varepsilon_j$.

We know that, for both subspaces $\mathcal{S}_1$ and $\mathcal{S}_2$, the data full space volume $\text{Volume}(\mathit{Full}\mathcal{R})$ is equal. Using a constant $\text{Volume}(\mathit{Full}\mathcal{R})$ we can write

$$\mathit{RelativeVolume}\left(\mathcal{I}^{\mathcal{S}}\right) \propto \text{Volume}(\mathcal{I}^{\mathcal{S}})$$

This volume is the one of a d-ellipse. Let $\lambda_1, \ldots, \lambda_d$ be the eigenvalues of the covariance matrix within a subspace $\mathcal{S}$, then

$$\mathit{RelativeVolume}\left(\mathcal{I}^{\mathcal{S}}\right) \propto \frac{2\pi^{\frac{d}{2}}}{d\,\Gamma(\frac{d}{2})} \sqrt{\mathit{Quantile}(\alpha,\ |\mathcal{S}|) \prod_{i=1}^{d} \lambda_i} \tag{4}$$

We further know that $\sum_{i=1}^{d} \lambda_i$ is equal to the sum of the trace of the corresponding covariance matrix. The trace of Σ_1 and Σ_2 are the same. We have assumed that each attribute in $\mathcal{S}_1$ is i.i.d. $N(0, \lambda)$. Hence, Σ_1 is a diagonal matrix with λ in each diagonal element. Thus, λ is the variance and each of the d eigenvalues of Σ_1. Σ_2 has off-diagonal elements. Hence, the eigenvalues can differ from the ones in Σ_1. Using the equality of traces we infer that $\sum_{i=1}^{d} \varepsilon_i = 0$. In order to prove our theorem, we need to show that:

$$\prod_{i=1}^{d} \lambda = \lambda^d \geq \prod_{i=1}^{d} \tilde{\lambda}_i = \prod_{i=1}^{d} (\lambda + \varepsilon_i) \tag{5}$$

We introduce $\text{index}_{>0} := \{i \in 1, \ldots, d \mid \varepsilon_i > 0\}$. Similarly, the index of $\varepsilon_i = 0$ as $\text{index}_{=0}$. Let further be $m = |\text{index}_{=0}|$. We can infer that $|\text{index}_{>0}| = \frac{d-m}{2}$. Using this, we can write:

$$\begin{aligned}
\prod_{i=1}^{d} (\lambda + \varepsilon_i) &= \left[\prod_{i\,\in\,\text{index}_{=0}} \lambda\right] \left[\prod_{j\,\in\,\text{index}_{>0}} (\lambda + \varepsilon_j)(\lambda - \varepsilon_j)\right] \\
&= \lambda^m \left[\prod_{j\,\in\,\text{index}_{>0}} (\lambda^2 - \varepsilon_j^2)\right] \\
&= \lambda^m \left(\lambda^2\right)^{\frac{d-m}{2}} - \lambda^m \left[\prod_{j\,\in\,\text{index}_{>0}} \varepsilon_j^2\right] \\
&= \lambda^d \underbrace{- \lambda^m \left[\prod_{j\,\in\,\text{index}_{>0}} \varepsilon_j^2\right]}_{\leq 0}
\end{aligned}$$

Inserting this in Eq. 5 directly proves the theorem. We can also infer that if there is an $\varepsilon_i > 0$, the statement of the theorem extends to

$$\mathit{RelativeVolume}\left(\mathcal{I}^{\mathcal{S}_1}\right) > \mathit{RelativeVolume}\left(\mathcal{I}^{\mathcal{S}_2}\right)$$

References

1. Azmandian, F., et al.: GPU-accelerated feature selection for outlier detection using the local kernel density ratio. In: IEEE 12th International Conference on Data Mining, pp. 51–60 (2012)
2. Breunig, M.M., et al.: LOF: identifying density-based local outliers. ACM Sigmod Rec. **29**(2), 93–104 (2000)
3. Campos, G.O., et al.: On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. Data Min. Knowl. Discov. **30**(4), 1–37 (2016)
4. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572 (2014)
5. Hodge, V.J., Austin, J.: A survey of outlier detection methodologies. Artif. Intell. Rev. **22**(2), 85–126 (2004)
6. Keller, F., Müller, E., Böhm, K.: HiCS: high contrast subspaces for density-based outlier ranking. In: International Conference on Data Engineering, pp. 1037–1048 (2012)
7. Kido, H., Yanagisawa, Y., Satoh, T.: An anonymous communication technique using dummies for location-based services. In:

Proceedings of the International Conference on Pervasive Services, pp. 88–97 (2005)
8. Knorr, E.M., Ng, R.T.: Algorithms for mining distance based outliers in large datasets. In: Proceedings of the International Conference on Very Large Data Bases, pp. 392–403 (1998)
9. Kollios, G., et al.: Efficient biased sampling for approximate clustering and outlier detection in large data sets. Trans. Knowl. Data Eng. **15**(5), 1170–1187 (2003)
10. Kriegel, H.P., Kröger, P., Schubert, E., Zimek, A.: LoOP: local outlier probabilities. In: Proceedings of the ACM Conference on Information and Knowledge Management, pp. 1649–1652 (2009)
11. Kriegel, H.P., Kröger, P., Zimek, A.: Outlier Detection Techniques. Tutorial at Knowledge Discovery and Data Mining (2010)
12. Kriegel, H.P., Schubert, M., Zimek, A.: Angle-based outlier detection in high-dimensional data. In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 444–452 (2008)
13. Kriegel, H.P., et al.: Outlier detection in axis-parallel subspaces of high dimensional data. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 831–838 (2009)
14. Kriegel, H.P., et al.: Interpreting and unifying outlier scores. In: Proceedings of the SIAM International Conference on Data Mining, pp. 13–24 (2011)
15. Kröger, W.: Critical infrastructures at risk: a need for a new conceptual approach and extended analytical tools. Reliab. Eng. Syst. Saf. **93**(12), 1781–1787 (2008)
16. Lazarevic, A., Kumar, V.: Feature bagging for outlier detection. In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 157–166 (2005)
17. Müller, E., Schiffer, M., Seidl, T.: Statistical Selection of Relevant Subspace Projections for Outlier Ranking. International Conference on Data Engineering pp. 434–445 (2011)
18. Müller, E., et al.: Outlier ranking via subspace analysis in multiple views of the data. In: 12th International Conference on Data Mining, pp. 529–538 (2012)
19. Nelson, B., et al.: Classifier evasion: models and open problems. International Workshop on Privacy and Security Issues in Data Mining and Machine Learning, pp. 92–98 (2010)
20. Nelson, B., et al.: Near-optimal evasion of convex-inducing classifiers. AISTATS Artificial Intelligence and Statistics pp. 549–556 (2010)
21. Pang, G., Cao, L., Chen, L.: Outlier detection in complex categorical data by modelling the feature value couplings. In: Proceedings of the 25th International Joint Conference on Artificial Intelligence, pp. 9–15 (2016)
22. Pang, G., et al.: Unsupervised feature selection for outlier detection by modelling hierarchical value-feature couplings. In: IEEE 16th International Conference on Data Mining, pp. 410–419 (2016)
23. Papernot, N., et al.: The limitations of deep learning in adversarial settings. In: IEEE European Symposium on Security and Privacy, pp. 372–387 (2016)
24. Prószyñski, W.: On outlier-hiding effects in specific Gauss–Markov models: geodetic examples. J. Geod. **74**(7–8), 581–589 (2000)
25. Szegedy, C., et al.: Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199 (2013)
26. Xu, W., Qi, Y., Evans, D.: Automatically evading classifiers. In: Proceedings of the Network and Distributed Systems Symposium (2016)
27. Zimek, A., Schubert, E., Kriegel, H.P.: A survey on unsupervised outlier detection in high-dimensional numerical data. Stat.l Anal. Data Min. **5**(5), 363–387 (2012)