Noname manuscript No.
(will be inserted by the editor)

Mining Outlying Aspects on Numeric Data

Lei Duan · Guanting Tang · Jian Pei · James Bailey · Akiko Campbell · Changjie Tang

Received: date / Accepted: date

Abstract When we are investigating an object in a data set, which itself may or may not be an outlier, can we identify unusual (i.e., outlying) aspects of the object? In this paper, we identify the novel problem of *mining outlying aspects on numeric data*. Given a query object o in a multidimensional numeric data set O, in which subspace is o most outlying? Technically, we use the rank of the probability density of an object in a subspace to measure the outlyingness of the object in the subspace. A minimal subspace where the query object is ranked the best is an outlying aspect. Computing the outlying aspects of a query object is far from trivial. A naïve method has to calculate the probability densities of all objects and rank them in every subspace, which is very costly when the dimensionality is high. We systematically develop a heuristic method that is capable of searching data sets with tens of dimensions efficiently. Our empirical study using both real data and synthetic data demonstrates that our method is effective and efficient.

Keywords Outlying aspect · Outlyingness degree · Kernel density estimation · Subspace search

L. Duan, C. Tang
Sichuan University, China.
E-mail: {leiduan, cjtang}@scu.edu.cn

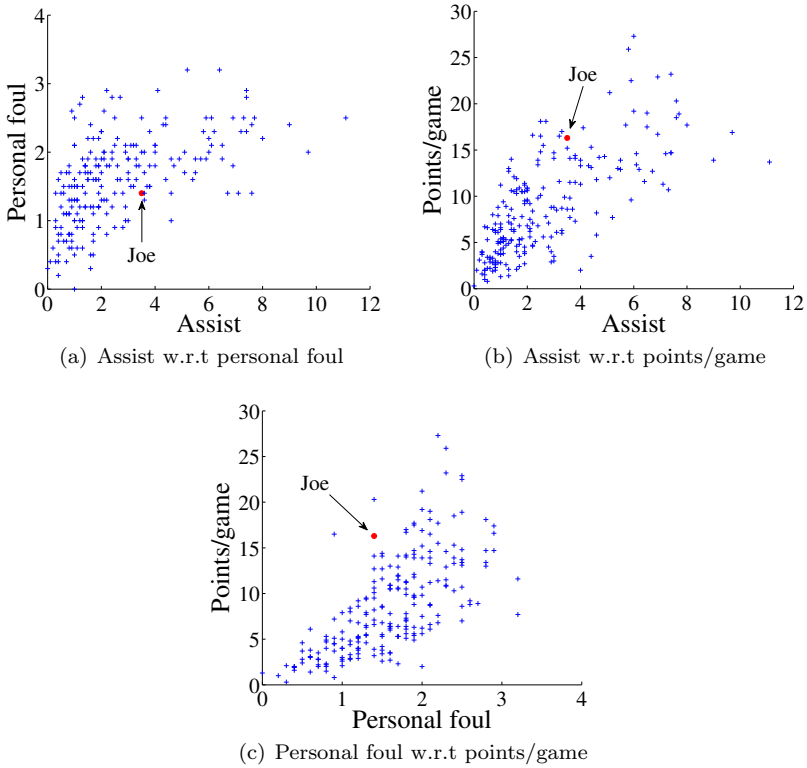
G. Tang, J. Pei
Simon Fraser University, Canada.
E-mail: {gta9, jpei}@cs.sfu.ca

J. Bailey
The University of Melbourne, Australia.
E-mail: baileyj@unimelb.edu.au

A. Campbell
Pacific Blue Cross, Canada.
E-mail: acampbell@pac.bluecross.ca

1 Introduction

In many application scenarios, a user may wish to investigate a specific object, in particular, the aspects where the object is most unusual compared to the rest of the data. For example, when a commentator mentions an NBA player, the commentator may want to name the most distinguishing features of the player, though the player may not be top ranked on those aspects or on any others among all players. Take the technical statistics of the 220 guards on assist, personal foul and points/game in the NBA Season 2012-2013 as an example[1] (Figure 1), an answer for Joe Johnson may be "the most distinguishing feature of Joe Johnson is his scoring ability with respect to his performance on personal foul" (by comparing Figures 1 (a), (b) and (c) based on the notion of density).



(a) Assist w.r.t personal foul

(b) Assist w.r.t points/game

(c) Personal foul w.r.t points/game

Fig. 1 Performance of NBA guards on assist, personal foul and points/game in the 2012-2013 Season (the solid circle (●) represents Joe Johnson)

As another example, when evaluating an applicant to a university program, who herself/himself may not necessarily be outstanding among all applicants,

[1] `http://sports.yahoo.com/nba/stats`

one may want to know the strength or weakness of the applicant, such as "the student's strength is the combination of GPA and volunteer experience, ranking her/him in the top 15% using these combined aspects". Moreover, in an insurance company, a fraud analyst may collect the information about various aspects of a claim, and wonder in which aspects the claim is most unusual. Furthermore, in commercial promotion, when designing an effective advertisement, it may be useful for marketers to know the most distinctive set of features characterizing the product. Similar examples can easily be found in other analytics applications.

The questions illustrated in the above examples are different from traditional outlier detection. Specifically, instead of searching for outliers from a data set, here we are given a query object and want to find the outlying aspects whereby the object is most unusual. The query object itself may or may not be an outlier in the full space or in any specific subspaces. In this problem, we are not interested in other individual outliers or inliers. The outlying aspect finding questions cannot be answered by the existing outlier detection methods directly.

We emphasize that investigating specific objects is a common practice in anomaly and fraud detection and analysis. Specifying query objects is an effective way to explicitly express analysts' background knowledge about data. Moreover, finding outlying aspects extends and generalizes the popular exercise of checking a suspect of anomaly or fraud. Currently, more often than not an analyst has to check the features of an outlying object one by one to find outlying features, but still cannot identify combinations of features where the object is unusual.

Motivated by these interesting applications about analyzing outlying aspects of a query object, in this paper, we model and tackle the problem of mining outlying aspects on numeric data, which is related to, but critically different from traditional outlier detection. Specifically, traditional outlier detection finds *outlier objects* in a set, while the problem of outlying aspect mining studied in this paper finds the *subspaces best manifesting the unusualness of a specified query object*, using the other objects as the background in comparing different subspaces. We address several technical challenges and make solid contributions on several fronts.

First, we identify and formulate the problem of outlying aspect mining on numeric data. Although Angiulli et al (2009, 2013) recently studied detecting outlying properties of exceptional objects, their methods find contextual rule based explanations. We will discuss the differences between our model and theirs in detail in Section 3. As illustrated, outlying aspect mining has immediate applications in data analytics practice.

Second, how can we compare the outlyingness of an object in different subspaces? While comparing the outlyingness of different objects in the same subspace is well studied and straightforward, comparing outlyingness of the same object in different subspaces is subtle, since different subspaces may have different scales and distribution characteristics. We propose a simple yet principled approach. In a subspace, we rank all objects in the ascending order

of probability density. A smaller probability density and thus a better rank indicates that the query object is more outlying in the subspace. Then, we compare the rank statistics of the query object in different subspaces, and return the subspaces of the best rank as the outlying aspects of the query object. To avoid redundancy, we only report minimal subspaces. That is, if a query object is ranked the same in subspaces S and S' such that S is a proper subspace of S' (i.e., $S \subset S'$), then S' is not reported since S is more general. Our model can be extended to many outlyingness measures other than probability density, which we leave for future work.

Third, how can we compute the outlying aspects fast, particularly on high dimensional data sets? A naïve method using the definition of outlying aspects directly has to calculate the probability densities of all objects and rank them in every subspace. This method incurs heavy cost when the dimensionality is high. On a data set of 100 dimensions, $2^{100} - 1 = 1.27 \times 10^{30}$ subspaces have to be examined, which is unfortunately computationally prohibitive using the state-of-the-art technology. To tackle the problem, we systematically develop a heuristic method that is capable of searching data sets with dozens of dimensions efficiently. Specifically, we develop pruning techniques that can avoid computing the probability densities of many objects in many subspaces. These effective pruning techniques enable our method to mine outlying aspects on data sets with tens of dimensions, as demonstrated later in our experiments.

Last, to evaluate outlying aspect mining, we conduct an extensive empirical study on both real and synthetic data sets. We illustrate the characteristics of discovered outlying aspects, and justify the value of outlying aspect mining. Moreover, we examine the effectiveness of our pruning techniques and the efficiency of our methods.

The rest of the paper is organized as follows. We formulate the problem of outlying aspect mining in Section 2, and review related work in Section 3. In Section 4, we recall the basics of kernel density estimation, which is used to estimate the probability density of objects, and present the framework of our method OAMiner (for Outlying Aspect Miner). In Section 5, we discuss the critical techniques in OAMiner. We report a systematic empirical study in Section 6, and conclude the paper in Section 7.

2 Problem Definition

Let $D = \{D_1, \ldots, D_d\}$ be a d-dimensional space, where the domain of D_i is $\mathbb{R}$, the set of real numbers. A *subspace* $S \subseteq D$ ($S \neq \emptyset$) is a subset of D. We also call D the *full space*.

Consider a set O of n objects in space D. For an object $o \in O$, denote by $o.D_i$ the value of o in dimension D_i ($1 \leq i \leq d$). For a subspace $S = \{D_{i_1}, \ldots, D_{i_l}\} \subseteq D$, the *projection* of o in S is $o^S = (o.D_{i_1}, \ldots, o.D_{i_l})$. The *dimensionality* of S, denoted by $|S|$, is the number of dimensions in S.

In a subspace $S \subseteq D$, we assume that we can define a measure of *outlyingness degree* $OutDeg(\cdot)$ such that for each object $o \in O$, $OutDeg(o)$

measures the outlyingness of o. Without loss of generality, we assume that the lower the outlyingness degree $OutDeg(o)$, the more outlying the object o.

In this paper, we assume the generative model. That is, the set of objects O are generated (i.e., sampled) from an often unknown probability distribution. Thus, we can use the probability density of an object o, denoted by $f(o)$, as the outlyingness degree. The smaller the value of $f(o)$, the more outlying the object o. We discuss how to estimate the probability densities in Section 4.1.

How can we compare the outlyingness of an object in different subspaces? Unfortunately, we cannot compare the outlyingness degree or probability density values directly, since the outlyingness degree and the probability density values depend on the properties of specific subspaces, such as their scales. For example, it is well known that probability density tends to be low in subspaces of higher dimensionality, since such subspaces often have a larger "volume" and thus are sparser.

To tackle this issue, we propose to use rank statistics. Specifically, in a subspace S, we rank all objects in O in their outlyingness degree ascending order. For an object $o \in O$, we denote by

$$rank_S(o) = |\{o' \mid o' \in O, OutDeg(o') < OutDeg(o)\}| + 1 \tag{1}$$

the *outlyingness rank* of o in subspace S. The smaller the rank value, the more outlying the object is comparing to the other objects in O in subspace S. We can compare the outlyingness of an object o in two subspaces S_1 and S_2 using $rank_{S_1}(o)$ and $rank_{S_2}(o)$. Object o is more outlying in the subspace where it has the smaller rank. Apparently, in Equation 1, for objects with the same outlyingness degree (probability density value), their outlyingness ranks are the same.

Suppose for object o there are two subspaces S and S' such that $S \subset S'$ and $rank_S(o) = rank_{S'}(o)$. Since S is more general than S', S is more significant in manifesting the outlyingness of o at the rank of $rank_S(o)$ relative to the other objects in the data set. Therefore, S' is redundant given S in terms of outlying aspects. Note that we use rank statistics instead of the absolute outlyingness degree values to compare the outlyingness of an object in different subspaces.

Rank statistics allows us to compare outlyingness in different subspaces, which is an advantage. At the same time, in high dimensional subspaces where the probability density values of objects are very small, comparing the ranks may not be reliable, since the subtle differences in probability density values may be due to noise or sensitivity to parameter settings in the density estimation. Ranking such objects may be misleading. Moreover, more often than not, users do not want to see high dimensional subspaces as answers, since high dimensional subspaces are hard to understand. Thus, we assume a *maximum dimensionality threshold* $\ell > 0$, and consider only subspaces whose dimensionality are not greater than ℓ. Please note that the problem cannot be solved using a minimum density threshold, since the density values are space and dimensionality sensitive, as explained before.

Based on the above discussion, we formalize the problem as follows.

Definition 1 (Problem definition) Given a set of objects O in a multi-dimensional space D, a query object $q \in O$ and a maximum dimensionality threshold $0 < \ell \leq |D|$, a subspace $S \subseteq D$ $(0 < |S| \leq \ell)$ is called a **minimal outlying subspace** of q if

1. (Rank minimality) there does not exist another subspace $S' \subseteq D$ $(S' \neq \emptyset)$, such that $rank_{S'}(q) < rank_S(q)$; and
2. (Subspace minimality) there does not exist another subspace $S'' \subset S$ such that $rank_{S''}(q) = rank_S(q)$.

The problem of **outlying aspect mining** is to find the minimal outlying subspaces of q.

Apparently, given a query object q, there exists at least one, and may be more than one minimal outlying subspace.

3 Related Work

Outlier analysis is a well studied subject in data mining. A comprehensive review of the abundant literature on outlier analysis is clearly beyond the capacity of this paper. Several recent surveys on the topic (Aggarwal, 2013; Chandola et al, 2009; Zimek et al, 2012), as well as dedicated chapters in classical data mining textbooks (Han et al, 2011) provide thorough treatments.

Given a set of objects, traditional outlier detection focuses on finding outlier objects that are significantly different from the rest of the data set. There are different ways to measure the differences between an object and the other objects, such as proximity, distance, and probability density. Many existing methods, such as (Knorr and Ng, 1999; Ramaswamy et al, 2000; Bhaduri et al, 2011), only return outliers, without focusing on explaining why those objects are outlying.

Recently, some studies attempt to explain outlying properties of outliers. The explanation may be a byproduct of outlier detection. For example, Böhm et al (2013) and Keller et al (2012) proposed statistical approaches *CMI* and *HiCS* to select subspaces for a multidimensional database, where there may exist outliers with high deviations. Both *CMI* and *HiCS* are fundamentally different from our method. They choose highly contrasting subspaces for all possible outliers in a data set, while our method chooses subspaces based on the query object.

Kriegel et al (2009) introduced SOD, a method to detect outliers in axis-parallel subspaces. There are two major differences between SOD and our work. First, SOD is still an outlier detection method, and the hyperplane is a byproduct of the detection process. Our method does not detect outliers at all. Second, the models to identify the outlying subspaces in the two methods are very different. When calculating the outlyingness score, SOD only considers the nearest neighbors as references in the full space. Our method considers all objects in the database and their relationship with the query object in subspaces.

Müller et al (2012b) presented a framework, called OutRules, to find explanations for outliers in different contexts. For each outlier, OutRules finds a set of rules $A \rightarrow B$, where A and B are subspaces, and the outlier is normal in subspace A but deviates substantially in subspace B. The deviation degree can be computed using some outlier score, such as LOF (Breunig et al, 2000). Then, a ranked list of rules is output as the explanation of the outlier. Tang et al (2013) proposed a framework to identify contextual outliers in a given multidimensional database. Only categorical data is considered. The methods in (Müller et al, 2012b; Tang et al, 2013) find outliers and their explanations at the same time, and are not designed for finding outlying aspects for an arbitrary query object. Moreover, those two methods focus on finding "conditional outliers", while our method does not assume this constraint.

Müller et al (2012a) computed an outlier score for each object in a database, providing a single global measure of how outlying an object is across different subspaces. The method ranks different outliers instead of the outlying behavior of one query object. In contrast, our approach investigates all possible subspaces for an object and identifies the minimal ones where the object has the lowest density rank (where it appears most unusual), and does not use the notion of subspace clusters.

Given a multidimensional *categorical* database and an object, which is preferably an outlier in the database, Angiulli et al (2009) found the top-k subsets of attributes (i.e., subspaces) from which the outlier receives the highest outlyingness scores. The outlyingness score for a given object in a subspace is calculated based on the frequency of the value that the outlier takes in the subspace. It tries to find subspaces E and S such that the outlier is frequent in one and much less frequent than expected in the other. Searching all such rules is computationally costly. To reduce the cost within a manageable scope, the method takes two parameters, σ and θ, to constrain the frequencies of the given object in subspaces E and S, respectively. Therefore, if a query object is not outlying compared to the other objects, no outlying properties may be detected.

To the best of our knowledge, (Angiulli et al, 2009, 2013) are the only studies on finding explanation of outlying aspects and thus are most relevant to our paper. There are several essential differences between (Angiulli et al, 2009, 2013) and this study. First, (Angiulli et al, 2009, 2013) find contextual rule based explanations, while our method returns individual subspaces where the query object is mostly outlying comparing to the other subspaces. The meaning of the two types of explanation is fundamentally different. Second, (Angiulli et al, 2009) focuses on categorical data, and our method targets on numeric data. Although (Angiulli et al, 2013) considers numeric data, its mining target is substantially different from this work. Specifically, given a set of objects O in a multi-dimensional space D and a query object $q \in O$, (Angiulli et al, 2013) finds the pairs (E, d) satisfying $E \subseteq D$ and $d \in D \setminus E$, such that there exists a subset $O' \subseteq O$, including q, in which objects are similar on E (referred to as *explanation*), while q is essentially different from the other objects in O'

on d (referred to as *property*). Besides one-dimensional attributes, our method can find outlying subspaces with arbitrary dimensionality.

To some extent, outlyingness is related to uniqueness and uniqueness mining. Paravastu et al (2008) discovered the feature-value combinations that make a particular record unique. Their task formulation is reminiscent of infrequent itemset mining, and uses a level-wise Apriori enumeration strategy (Agrawal and Srikant, 1994). It needs a discretization step. Our method is native for continuous data.

Müller et al (2011) proposed the OUTRES approach, which aims to assess the contribution of some selected subspaces where an object deviates from its neighborhood. OUTRES employs kernel density estimation. Different from our approach, OUTRES uses the Epanechnikov kernel rather than the Gaussian kernel. Our approach calibrates densities across subspaces using rank statistics, rather than using an adaptive neighborhood. The emphasis of OUTRES is mainly on finding outliers, rather than exploring subspaces where a query object may or may not be an outlier. Consequently, OUTRES only considers subspaces that satisfy a statistical test for non-uniformity. Moreover, for a chosen object, OUTRES computes an aggregate outlier score that incorporates only the contribution of subspaces where the object has significantly low density.

Our method uses probability density to measure outlying degree in a subspace. There are a few density-based outlier detection methods, such as (Breunig et al, 2000; Kriegel et al, 2008; He et al, 2005; Aggarwal and Yu, 2001). Our method is inherently different from those, since we do not find outlier objects at all.

4 The Framework

In this section, we first review the essentials of kernel density estimation techniques. Then, we present the framework of our OAMiner method.

4.1 Kernel Density Estimation

We use kernel density estimation (Scott, 1992; Silverman, 1986) to estimate the probability density given a set of objects O. Given a random sample $\{o_1, o_2, \ldots, o_n\}$ drawn from some distribution with an unknown probability density f in space $\mathbb{R}$, the probability density f at a point $o \in \mathbb{R}$ can be estimated by

$$\hat{f}_h(o) = \frac{1}{n}\sum_{i=1}^{n} K_h(o - o_i) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{o - o_i}{h}\right)$$

where $K(\cdot)$ is a kernel, and h is a smoothing parameter called the *bandwidth*. A widely adopted approach to estimate the bandwidth is Silverman's rule of thumb (Silverman, 1986), which suggests $h = 1.06\sigma n^{-\frac{1}{5}}$, σ being the standard

deviation of the sample. To further reduce the sensitivity to outliers, in this work, we use a better rule of thumb (Härdle, 1990) and set

$$h = 1.06 \min\{\sigma, \frac{R}{1.34}\} n^{-\frac{1}{5}} \tag{2}$$

where $R = X_{[0.75n]} - X_{[0.25n]}$, and $X_{[0.25n]}$ and $X_{[0.75n]}$, respectively, are the first and the third quartiles.

For the d-dimensional case ($d \geq 2$), $o = (o.D_1, \ldots, o.D_d)^T$, and $o_i = (o_i.D_1, \ldots, o_i.D_d)^T$ $(1 \leq i \leq n)$. Then, the probability density of f at point $o \in \mathbb{R}^d$ can be estimated by

$$\hat{f}_H(o) = \frac{1}{n} \sum_{i=1}^{n} K_H(o - o_i)$$

where H is a bandwidth matrix.

The product kernel, which consists of the product of one-dimensional kernels, is a good choice for multivariate kernel density estimator in practice (Scott, 1992; Härdle et al, 2004). We have

$$\hat{f}_H(o) = \frac{1}{n \prod_{j=1}^{d} h_{D_j}} \sum_{i=1}^{n} \left\{ \prod_{j=1}^{d} K\left(\frac{o.D_j - o_i.D_j}{h_{D_j}} \right) \right\} \tag{3}$$

where h_{D_i} is the bandwidth of dimension D_i $(1 \leq i \leq d)$.

Note that the product kernel does not assume that the dimensions are independent. Otherwise, the density estimation would be

$$\hat{f}_H(o) = \prod_{j=1}^{d} \left(\frac{1}{n \cdot h_{D_j}} \sum_{i=1}^{n} K\left(\frac{o.D_j - o_i.D_j}{h_{D_j}} \right) \right)$$

In this paper, we adopt the Gaussian kernel, which has been popularly used. The distance between two objects is measured by Euclidean distance. The kernel function is

$$K\left(\frac{o - o_i}{h} \right) = \frac{1}{\sqrt{2\pi}} e^{-\frac{(o-o_i)^2}{2h^2}} \tag{4}$$

Note that other kernel functions and distance functions may be used in our framework.

Plugging Equation 4 into Equation 3, the density of a query object $q \in O$ in subspace S can be estimated as

$$\hat{f}_S(q) = \hat{f}_S(q^S) = \frac{1}{n(2\pi)^{\frac{|S|}{2}} \prod_{D_i \in S} h_{D_i}} \sum_{o \in O} e^{-\sum_{D_i \in S} \frac{(q.D_i - o.D_i)^2}{2h_{D_i}^2}} \tag{5}$$

Algorithm 1 $rank_S(q)$ – baseline

Input: a set of objects O, query object $q \in O$, and subspace S
Output: $rank_S(q)$
1: **for** each object $o \in O$ **do**
2: compute $\tilde{f}_S(o)$ using Equation 7
3: **end for**
4: **return** $rank_S(q) = |\{o \mid o \in O, \tilde{f}_S(o) < \tilde{f}_S(q)\}| + 1$

Since we are interested in only the rank of q, that is, $rank_S(q)$, and

$$c = \frac{1}{n(2\pi)^{\frac{|S|}{2}} \prod_{D_i \in S} h_{D_i}} \tag{6}$$

is a factor common to every object in subspace S and thus does not affect the ranking at all, we can rewrite Equation 5 as

$$\hat{f}_S(q) \sim \tilde{f}_S(q) = \sum_{o \in O} e^{-\sum_{D_i \in S} \frac{(q.D_i - o.D_i)^2}{2h_{D_i}^2}} \tag{7}$$

where symbol "$\sim$" means equivalence for ranking.

For the sake of clarity, we call $\tilde{f}_S(q)$ the *quasi-density* of q in S. Please note that, using $\tilde{f}_S(q)$ instead of $\hat{f}_S(q)$ not only simplifies the description, but also saves computational cost for calculating $rank_S(q)$. We will illustrate the details in Section 5.

We can show an interesting property – invariance of ranking under linear transformation. The proof can be found in Appendix A.

Proposition 1 (Invariance) *Given a set of objects O in space $S = \{D_1, \ldots, D_d\}$, define a linear transformation $g(o) = (a_1 o.D_1 + b_1, \ldots, a_d o.D_d + b_d)$ for any $o \in O$, where $a_1, \ldots, a_d$ and $b_1, \ldots, b_d$ are real numbers. Let $O' = \{g(o) | o \in O\}$ be the transformed data set. For any objects $o_1, o_2 \in O$ such that $\tilde{f}_S(o_1) > \tilde{f}_S(o_2)$ in O, $\tilde{f}_S(g(o_1)) > \tilde{f}_S(g(o_2))$ if the product kernel is used and the bandwidths are set using Härdle's rule of thumb (Equation 2).*

Using quasi-density estimation (Equation 7), we can have a baseline algorithm for computing the outlyingness rank in a subspace S, as shown in Algorithm 1. The baseline method estimates the quasi-density of each object in a data set, and ranks them. Let the total number of objects be n. The baseline method essentially has to compute the distance between every pair of objects in every dimension of S. Therefore, the time complexity is $O(n^2|S|)$ in each subspace S.

4.2 The Framework of OAMiner

To reduce the computational cost, we present Algorithm 2, the framework of our method OAMiner (for Outlying Aspect Miner).

Algorithm 2 The framework of OAMiner

Input: a set of objects O and query object $q \in O$
Output: the set of minimal outlying subspaces for q

```
initialize r_best ← |O| and Ans ← ∅;
remove D_i from D if the values of all objects in D_i are identical;
compute rank_{D_i}(q) in each dimension D_i ∈ D;
sort all dimensions in rank_{D_i}(q) ascending order;
for each subspace S searched by traversing the set enumeration tree in a depth-first manner do
    compute rank_S(q);
    if rank_S(q) < r_best then
        r_best ← rank_S(q), Ans ← {S};
    end if
    if rank_S(q) = r_best and S is minimal then
        Ans ← Ans ∪ {S};
    end if
    if a subspace pruning condition is true then
        prune all super-spaces of S
    end if
end for
return Ans
```

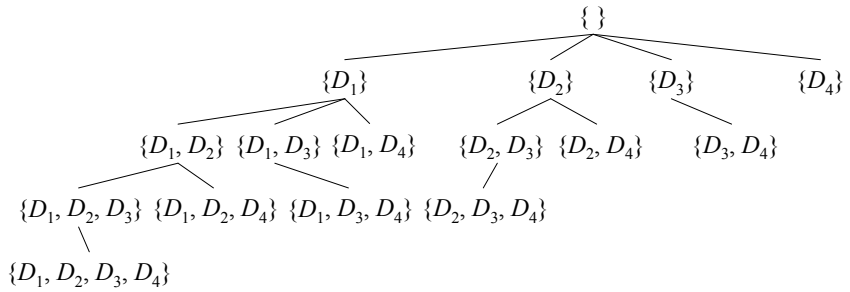


Fig. 2 A set enumeration tree.

First of all, OAMiner removes the dimensions where all values of objects are identical, since no object is outlying in such dimensions. As a result, the standard deviations of all dimensions involved for outlying aspect mining are greater than 0.

In order to ensure that OAMiner can find the most outlying subspaces, we have to enumerate all possible subspaces in a systematic way. Here, we adopt the set enumeration tree approach (Rymon, 1992), which has been popularly used in many data mining methods. Conceptually, a set enumeration tree takes a total order on the set, the dimensions in the context of our problem, and then enumerates all possible combinations systematically. For example, Figure 2 shows a set enumeration tree that enumerates all subspaces of space $D = \{D_1, D_2, D_3, D_4\}$.

OAMiner searches subspaces by traversing the subspace enumeration tree in a depth-first manner. Given a set of objects O, a query object $q \in O$, and a

Table 1 A numeric data set example

object	$o_i.D_1$	$o_i.D_2$
o_1	14.23	1.5
o_2	13.2	1.78
o_3	13.16	2.31
o_4	14.37	1.97

Table 2 Quasi-density values of objects in Table 1

object	$\tilde{f}_{\{D_1\}}(o_i)$	$\tilde{f}_{\{D_2\}}(o_i)$	$\tilde{f}_{\{D_1,D_2\}}(o_i)$
o_1	2.229	1.832	1.305
o_2	2.220	2.529	1.300
o_3	2.187	1.626	1.185
o_4	2.113	2.474	1.314

subspace S, if $rank_S(q) = 1$, then every super-space of S cannot be a minimal outlying subspace and thus can be pruned.

Pruning Rule 1 *If $rank_S(q) = 1$, according to the dimensionality minimality condition in the problem definition (Definition 1), all super-spaces of S can be pruned.*

In the case of $rank_S(q) > 1$, OAMiner prunes subspaces according to the current best rank of q in the search process. More details will be discussed in Section 5.3.

Heuristically, we want to find subspaces early where the query object q has a low rank, so that the pruning techniques take better effect. Motivated by this observation, we compute the outlyingness rank of q in each dimension D_i, and order all dimensions in the ascending order of $rank_{D_i}(q)$.

In general, the outlyingness rank does not have any monotonicity with respect to subspaces. That is, for subspaces $S_1 \subset S_2$, neither $rank_{S_1}(q) \leq rank_{S_2}(q)$ nor $rank_{S_1}(q) \geq rank_{S_2}(q)$ holds in general. Example 1 illustrates this situation with a toy data set.

Example 1 Given a set of objects $O = \{o_1, o_2, o_3, o_4\}$ with 2 numeric attributes D_1 and D_2. The values of each object in O are listed in Table 1. Using Equation 7, we estimate the quasi-density values of each object on different subspaces (Table 2). We can see that $\tilde{f}_{\{D_1\}}(o_2) > \tilde{f}_{\{D_1\}}(o_4)$ and $\tilde{f}_{\{D_2\}}(o_2) > \tilde{f}_{\{D_2\}}(o_4)$, which indicate $rank_{\{D_1\}}(o_2) > rank_{\{D_1\}}(o_4)$ and $rank_{\{D_2\}}(o_2) > rank_{\{D_2\}}(o_4)$. However, for subspace $\{D_1, D_2\}$, since $\tilde{f}_{\{D_1,D_2\}}(o_2) < \tilde{f}_{\{D_1,D_2\}}(o_4)$, $rank_{\{D_1,D_2\}}(o_2) < rank_{\{D_1,D_2\}}(o_4)$.

To make the situation even more challenging, probability density itself does not have any monotonicity with respect to subspaces. Given a query object q, and subspaces $S_1 \subset S_2$. According to Equation 5, we have

$$\frac{\hat{f}_{S_1}(q)}{\hat{f}_{S_2}(q)} = \frac{\sum_{o\in O} e^{-\sum_{D_i\in S_1}\frac{(q.D_i-o.D_i)^2}{2h_{D_i}^2}}}{n(2\pi)^{\frac{|S_1|}{2}}\prod_{D_i\in S_1} h_{D_i}} \Big/ \frac{\sum_{o\in O} e^{-\sum_{D_i\in S_2}\frac{(q.D_i-o.D_i)^2}{2h_{D_i}^2}}}{n(2\pi)^{\frac{|S_2|}{2}}\prod_{D_i\in S_2} h_{D_i}}$$

$$= (2\pi)^{\frac{|S_2|-|S_1|}{2}} \prod_{D_i\in S_2\setminus S_1} h_{D_i} \frac{\sum_{o\in O} e^{-\sum_{D_i\in S_1}\frac{(q.D_i-o.D_i)^2}{2h_{D_i}^2}}}{\sum_{o\in O} e^{-\sum_{D_i\in S_2}\frac{(q.D_i-o.D_i)^2}{2h_{D_i}^2}}}$$

Table 3 Summary of notations

Notation	Description
D	a d-dimensional space
O	a set of objects in space D
h_{D_i}	the bandwidth of dimension D_i
$rank_S(o)$	outlyingness rank of object o in subspace S
$\tilde{f}_S(o)$	quasi-density of object o in subspace S estimated by Equation 7
$\tilde{f}_S^{O'}(o)$	the sum of quasi-density contributions of objects in set O' to object o in subspace S
$TN_S^{\epsilon,o}$	ϵ-tight neighborhood of object o in subspace S
$LN_S^{\epsilon,o}$	ϵ-loose neighborhood of object o in subspace S
$dc_S(o,o')$	the quasi-density contribution of object o' to object o in subspace S
$Comp_S(o)$	a set of objects where OAMiner can determine that their densities are less than the density of o in subspace S and its super-spaces

Since $S_1 \subset S_2$, $\sum_{o\in O} e^{-\sum_{D_i\in S_1}\frac{(q.D_i-o.D_i)^2}{2h_{D_i}^2}} / \sum_{o\in O} e^{-\sum_{D_i\in S_2}\frac{(q.D_i-o.D_i)^2}{2h_{D_i}^2}} \geq 1$ and $(2\pi)^{\frac{|S_2|-|S_1|}{2}} > 1$. However, in the case $\prod_{D_i\in S_2\setminus S_1} h_{D_i} < 1$, there is no guarantee that $\frac{\hat{f}_{S_1}(q)}{\hat{f}_{S_2}(q)} > 1$ always holds. Thus, neither $\hat{f}_{S_1}(q) \leq \hat{f}_{S_2}(q)$ nor $\hat{f}_{S_1}(q) \geq \hat{f}_{S_2}(q)$ holds in general.

5 Critical Techniques in OAMiner

In this section, we present a bounding-pruning-refining algorithm to efficiently compute the outlyingness rank of an object in a subspace, and discuss the critical techniques to prune subspaces.

Table 3 lists the frequently used notations in this section.

5.1 Bounding Probability Density

In order to obtain the rank statistics about outlyingness, OAMiner has to compare the density of the query object with the densities of other objects. To speed up density estimation of objects, we observe that the contributions from remote objects to the density of an object are very small, and the density of an object can be bounded. Technically, we can derive upper and lower bounds of the probability density of an object using a neighborhood. Again, we denote by $\tilde{f}_S(o)$ the quasi-density of object o in subspace S.

For the sake of clarity, we introduce two notations at first. Given objects $o, o' \in O$, a subspace S, and a subset $O' \subseteq O$, we denote by $dc_S(o,o')$ the quasi-density contribution of o' to o in S, and $\tilde{f}_S^{O'}(o)$ the sum of quasi-density contributions of objects in O' to o. That is,

$$dc_S(o,o') = e^{-\sum_{D_i\in S}\frac{(o.D_i-o'.D_i)^2}{2h_{D_i}^2}}$$

$$\tilde{f}_S^{O'}(o) = \sum_{o' \in O'} e^{-\sum\limits_{D_i \in S} \frac{(o.D_i - o'.D_i)^2}{2h_{D_i}^2}}$$

To efficiently estimate the bounds of $\tilde{f}_S(o)$, we define two kinds of neighborhoods. For an object $o \in O$, a subspace S, and $\{\epsilon_{D_i} \mid \epsilon_{D_i} > 0, D_i \in S\}$, the *$\epsilon$-tight neighborhood* of o in S, denoted by $TN_S^{\epsilon,o}$, is $\{o' \in O \mid \forall D_i \in S, |o.D_i - o'.D_i| \leq \epsilon_{D_i}\}$, the *$\epsilon$-loose neighborhood* of o in S, denoted by $LN_S^{\epsilon,o}$, is $\{o' \in O \mid \exists D_i \in S, |o.D_i - o'.D_i| \leq \epsilon_{D_i}\}$. An object is called as an *ϵ-tight (loose) neighbor* if it is in the ϵ-tight (loose) neighborhood. We will illustrate how to efficiently compute $TN_S^{\epsilon,o}$ and $LN_S^{\epsilon,o}$ in Section 5.2.

According to the definitions of $TN_S^{\epsilon,o}$ and $LN_S^{\epsilon,o}$, we obtain the following properties.

Property 1 $TN_S^{\epsilon,o} \subseteq LN_S^{\epsilon,o}$.

Property 2 $TN_S^{\epsilon,o} = LN_S^{\epsilon,o}$ if $|S| = 1$.

Based on $TN_S^{\epsilon,o}$ and $LN_S^{\epsilon,o}$, O can be divided into three disjoint subsets: $TN_S^{\epsilon,o}$, $LN_S^{\epsilon,o} \setminus TN_S^{\epsilon,o}$ and $O \setminus LN_S^{\epsilon,o}$. For any object $o' \in O$, we obtain a lower bound and an upper bound of $dc_S(o, o')$ as follows.

Theorem 1 (Single quasi-density contribution bounds) *Given an object $o \in O$, a subspace S, and a set $\{\epsilon_{D_i} \mid \epsilon_{D_i} > 0, D_i \in S\}$. Then, for any object $o' \in TN_S^{\epsilon,o}$,*

$$dc_S^{\epsilon} \leq dc_S(o, o') \leq dc_S^{max}(o)$$

for any object $o' \in LN_S^{\epsilon,o} \setminus TN_S^{\epsilon,o}$,

$$dc_S^{min}(o) \leq dc_S(o, o') \leq dc_S^{max}(o)$$

for any object $o' \in O \setminus LN_S^{\epsilon,o}$,

$$dc_S^{min}(o) \leq dc_S(o, o') < dc_S^{\epsilon}$$

where

$$dc_S^{\epsilon} = e^{-\sum\limits_{D_i \in S} \frac{\epsilon_{D_i}^2}{2h_{D_i}^2}}$$

$$dc_S^{max}(o) = e^{-\sum\limits_{D_i \in S} \frac{\min\limits_{o' \in O} \{|o.D_i - o'.D_i|\}^2}{2h_{D_i}^2}}$$

$$dc_S^{min}(o) = e^{-\sum\limits_{D_i \in S} \frac{\max\limits_{o' \in O} \{|o.D_i - o'.D_i|\}^2}{2h_{D_i}^2}}$$

The proof of Theorem 1 is given in Appendix B.

Using the size of $TN_S^{\epsilon,o}$ and $LN_S^{\epsilon,o}$, we obtain a lower bound and an upper bound of $\tilde{f}_S(o)$ as follows. The proof can be found in Appendix C.

Corollary 1 (Bounds by neighborhood size) *For any object $o \in O$,*

$$|TN_S^{\epsilon,o}|\ dc_S^{\epsilon} + (|O| - |TN_S^{\epsilon,o}|)\ dc_S^{min}(o) \leq \tilde{f}_S(o)$$

$$\tilde{f}_S(o) \leq |LN_S^{\epsilon,o}|\ dc_S^{max}(o) + (|O| - |LN_S^{\epsilon,o}|)\ dc_S^{\epsilon}$$

Corollary 1 allows us to compute the quasi-density bounds of an object without computing the quasi-density contributions of other objects to it.

Moreover, by Theorem 1, we can obtain following corollaries.

Corollary 2 (Bounds by ϵ-tight neighbors) *For any object $o \in O$ and $O' \subseteq TN_S^{\epsilon,o}$,*

$$\tilde{f}_S^{O'}(o) + (|TN_S^{\epsilon,o}| - |O'|)\ dc_S^{\epsilon} + (|O| - |TN_S^{\epsilon,o}|)\ dc_S^{min}(o) \leq \tilde{f}_S(o)$$

$$\tilde{f}_S(o) \leq \tilde{f}_S^{O'}(o) + (|LN_S^{\epsilon,o}| - |O'|)\ dc_S^{max}(o) + (|O| - |LN_S^{\epsilon,o}|)\ dc_S^{\epsilon}$$

Corollary 3 (Bounds by ϵ-loose neighbors) *For any object $o \in O$ and $TN_S^{\epsilon,o} \subset O' \subseteq LN_S^{\epsilon,o}$,*

$$\tilde{f}_S^{O'}(o) + (|O| - |O'|)\ dc_S^{min}(o) \leq \tilde{f}_S(o)$$

$$\tilde{f}_S(o) < \tilde{f}_S^{O'}(o) + (|LN_S^{\epsilon,o}| - |O'|)\ dc_S^{max}(o) + (|O| - |LN_S^{\epsilon,o}|)\ dc_S^{\epsilon}$$

Corollary 4 (Bounds by a supper set of ϵ-loose neighbors) *For any object $o \in O$ and $LN_S^{\epsilon,o} \subset O' \subseteq O$,*

$$\tilde{f}_S^{O'}(o) + (|O| - |O'|)\ dc_S^{min}(o) \leq \tilde{f}_S(o)$$

$$\tilde{f}_S(o) \leq \tilde{f}_S^{O'}(o) + (|O| - |O'|)\ dc_S^{\epsilon}$$

The proofs of Corollary 2, Corollary 3 and Corollary 4 can be found in Appendix D, Appendix E and Appendix F, respectively.

Since the density of o is the sum of the density contributions of all objects in O, and the density contribution decreases with the distance, OAMiner first computes the quasi-density contributions from the objects in $TN_S^{\epsilon,o}$, then from the objects in $LN_S^{\epsilon,o} \setminus TN_S^{\epsilon,o}$, and last from the objects in $O \setminus LN_S^{\epsilon,o}$.

By computing the bounds of $\tilde{f}_S(o)$, OAMiner takes a bounding-pruning-refining method, shown in Algorithm 3, to efficiently perform density comparison in subspace S. Initially, OAMiner estimates the quasi-density of query object q, which is denoted by $\tilde{f}_S(q)$. Then, for an object o, OAMiner first computes the bounds of $\tilde{f}_S(o)$ by the sizes of $TN_S^{\epsilon,o}$ and $LN_S^{\epsilon,o}$ (Corollary 1), and compares the bounds with $\tilde{f}_S(q)$ (Steps 1-8). If the relation between $\tilde{f}_S(q)$ and the bounds can be determined, that is, either $\tilde{f}_S(q) < \tilde{f}_S(o)$ or $\tilde{f}_S(q) > \tilde{f}_S(o)$, then Algorithm 3 ends. Otherwise, OAMiner updates the lower and upper bounds of $\tilde{f}_S(o)$ by involving the quasi-density contributions of objects in $TN_S^{\epsilon,o}$ (Steps 10-20), in $LN_S^{\epsilon,o} \setminus TN_S^{\epsilon,o}$ (Steps 21-31), and in $O \setminus LN_S^{\epsilon,o}$ (Steps 32-42) one by one, and repeatedly compares the updated bounds with $\tilde{f}_S(q)$, until the relationship between $\tilde{f}_S(q)$ and $\tilde{f}_S(o)$ is fully determined.

Algorithm 3 Density comparison

Input: quasi-density of the query object $\tilde{f}_S(q)$, object $o \in O$, subspace S, the ϵ-tight neighborhood of o $TN_S^{\epsilon,o}$, and the ϵ-loose neighborhood of o $LN_S^{\epsilon,o}$.
Output: a boolean value indicating $\tilde{f}_S(o) < \tilde{f}_S(q)$ is true or not.

$L \leftarrow$ the lower bound of $\tilde{f}_S(o)$ computed by Corollary 1; // bounding
if $L > \tilde{f}_S(q)$ **then**
    return false; // pruning
end if
$U \leftarrow$ the upper bound of $\tilde{f}_S(o)$ computed by Corollary 1; // bounding
if $U < \tilde{f}_S(q)$ **then**
    return true; // pruning
end if
$O' \leftarrow \emptyset$; $\tilde{f}_S^{O'}(o) \leftarrow 0$;
for each $o' \in TN_S^{\epsilon,o}$ **do**
    $\tilde{f}_S^{O'}(o) \leftarrow \tilde{f}_S^{O'}(o) + dc_S(o,o')$; $O' \leftarrow O' \cup \{o'\}$; // refining
    $L \leftarrow$ the lower bound of $\tilde{f}_S(o)$ computed by Corollary 2; // bounding
    if $L > \tilde{f}_S(q)$ **then**
        return false; // pruning
    end if
    $U \leftarrow$ the upper bound of $\tilde{f}_S(o)$ computed by Corollary 2; // bounding
    if $U < \tilde{f}_S(q)$ **then**
        return true; // pruning
    end if
end for
for each $o' \in LN_S^{\epsilon,o} \setminus TN_S^{\epsilon,o}$ **do**
    $\tilde{f}_S^{O'}(o) \leftarrow \tilde{f}_S^{O'}(o) + dc_S(o,o')$; $O' \leftarrow O' \cup \{o'\}$; // refining
    $L \leftarrow$ the lower bound of $\tilde{f}_S(o)$ computed by Corollary 3; // bounding
    if $L > \tilde{f}_S(q)$ **then**
        return false; // pruning
    end if
    $U \leftarrow$ the upper bound of $\tilde{f}_S(o)$ computed by Corollary 3; // bounding
    if $U < \tilde{f}_S(q)$ **then**
        return true; // pruning
    end if
end for
for each $o' \in O \setminus LN_S^{\epsilon,o}$ **do**
    $\tilde{f}_S^{O'}(o) \leftarrow \tilde{f}_S^{O'}(o) + dc_S(o,o')$; $O' \leftarrow O' \cup \{o'\}$; // refining
    $L \leftarrow$ the lower bound of $\tilde{f}_S(o)$ computed by Corollary 4; // bounding
    if $L > \tilde{f}_S(q)$ **then**
        return false; // pruning
    end if
    $U \leftarrow$ the upper bound of $\tilde{f}_S(o)$ computed by Corollary 4; // bounding
    if $U < \tilde{f}_S(q)$ **then**
        return true; // pruning
    end if
end for
return false;

In OAMiner, the neighborhood distance in dimension D_i, denoted by ϵ_{D_i}, is defined as $\alpha\sigma_{D_i}$, where σ_{D_i} is the standard deviation in dimension D_i, and α is a parameter. Our experiments show that α is not sensitive, and can be set in the range of $0.8 \sim 1.2$, by which OAMiner runs efficiently. It is still an open question about how to set the best neighborhood distance for bounding

— this is a future research problem. Theorem 2 guarantees that no matter how to set the neighborhood distance, the ranking results keep unchanged.

Theorem 2 *Given an object $o \in O$, and a subspace S, for any neighborhood distances ϵ_1 and ϵ_2, $rank_S^{\epsilon_1}(o) = rank_S^{\epsilon_2}(o)$, where $rank_S^{\epsilon_1}(o)$ ($rank_S^{\epsilon_2}(o)$) is the outlyingness rank of o in S computed using ϵ_1 (ϵ_2).*

The proof of Theorem 2 can be found in Appendix G.

5.2 Efficiently Estimating Density Bounds

In this subsection, we present strategies in Algorithm 3 that efficiently estimate the lower and upper bounds of quasi-density.

Consider a candidate subspace $S \subseteq D$, and an object $o \in O$. To estimate lower and upper bounds of $\tilde{f}_S(o)$, OAMiner has to compute $TN_S^{\epsilon,o}$, $LN_S^{\epsilon,o}$, dc_S^{ϵ}, $dc_S^{min}(o)$, $dc_S^{max}(o)$ and $dc_S(o,o')$, where $o' \in O$.

In the case $|S| = 1$, we compute $TN_S^{\epsilon,o}$, dc_S^{ϵ}, $dc_S^{min}(o)$, $dc_S^{max}(o)$ and $dc_S(o,o')$ based on their definitions directly. As pointed out in Section 5.1, $TN_S^{\epsilon,o} = LN_S^{\epsilon,o}$ in this case. Moreover, the density contribution is symmetrical, so that the computational cost for $dc_S(o',o)$ can be saved if $dc_S(o,o')$ is available.

Please recall that OAMiner searches subspaces by traversing the subspace enumeration tree in a depth-first manner. For a subspace S satisfying $|S| \geq 2$, denote by $par(S)$ the parent subspace of S. Suppose $S \backslash par(S) = D'$ ($|D'| = 1$). Then, we have

$$TN_S^{\epsilon,o} = TN_{par(S)}^{\epsilon,o} \cap TN_{D'}^{\epsilon,o} \tag{8}$$

$$LN_S^{\epsilon,o} = LN_{par(S)}^{\epsilon,o} \cup LN_{D'}^{\epsilon,o} \tag{9}$$

$$dc_S^{\epsilon} = e^{-\sum\limits_{D_i \in S} \frac{\epsilon_{D_i}^2}{2h_{D_i}^2}} = e^{-(\sum\limits_{D_i \in par(S)} \frac{\epsilon_{D_i}^2}{2h_{D_i}^2} + \frac{\epsilon_{D'}^2}{2h_{D'}^2})} = dc_{par(S)}^{\epsilon} \cdot dc_{D'}^{\epsilon} \tag{10}$$

$$\begin{aligned} dc_S^{min}(o) &= e^{-(\sum\limits_{D_i \in par(S)} \frac{\max\limits_{o' \in O}\{|o.D_i - o'.D_i|\}^2}{2h_{D_i}^2} + \frac{\max\limits_{o' \in O}\{|o.D' - o'.D'|\}^2}{2h_{D'}^2})} \\ &= dc_{S\backslash par(S)}^{min}(o) \cdot dc_{D'}^{min}(o) \end{aligned} \tag{11}$$

$$\begin{aligned} dc_S^{max}(o) &= e^{-(\sum\limits_{D_i \in par(S)} \frac{\min\limits_{o' \in O}\{|o.D_i - o'.D_i|\}^2}{2h_{D_i}^2} + \frac{\min\limits_{o' \in O}\{|o.D' - o'.D'|\}^2}{2h_{D'}^2})} \\ &= dc_{S\backslash par(S)}^{max}(o) \cdot dc_{D'}^{max}(o) \end{aligned} \tag{12}$$

$$\begin{aligned} dc_S(o,o') &= e^{-\sum\limits_{D_i \in S} \frac{(o.D_i - o'.D_i)^2}{2h_{D_i}^2}} = e^{-(\sum\limits_{D_i \in par(S)} \frac{(o.D_i - o'.D_i)^2}{2h_{D_i}^2} + \frac{(o.D' - o'.D')^2}{2h_{D'}^2})} \\ &= dc_{par(S)}(o,o') \cdot dc_{D'}(o,o') \end{aligned} \tag{13}$$

Thus, it is efficient for OAMiner to estimate the bounds of $\tilde{f}_S(o)$ using $par(S)$ and $S \setminus par(s)$.

5.3 Subspace Pruning

Recall that OAMiner searches subspaces by traversing the subspace enumeration tree in a depth-first manner. During the search process, let $\mathcal{S}_1$ be the set of subspaces that OAMiner has searched, and $\mathcal{S}_2$ the set of subspaces that OAMiner has not searched yet. Clearly, $|\mathcal{S}_1 \cup \mathcal{S}_2| = 2^{|D|} - 1$. Given a query object q, let $r_{best} = \min_{S \in \mathcal{S}_1} \{rank_S(q)\}$ be the best rank that q has achieved so far. We can use r_{best} to prune some subspaces not searched yet. Specifically, for a subspace $S \in \mathcal{S}_2$, once we can determine that $rank_S(q) > r_{best}$, then S cannot be an outlying aspect, and thus can be pruned.

Observation 1 *When subspace S is met in a depth-first search of the subspace set enumeration tree, let r_{best} be the best rank of q in all the subspaces searched so far. Given object q with $rank_S(q) \geq 1$, if for every proper super-space $S' \supset S$, $rank_{S'}(q) > r_{best}$, then all proper super-spaces of S can be pruned.*

For the case that $rank_S(q) = 1$, all super-spaces of S can be pruned directly due to the dimensionality minimality condition in the problem definition (Pruning Rule 1). Thus, we only consider the case $rank_S(q) > 1$ here.

To implement Observation 1, in a subspace S where $rank_S(q) > 1$, we check whether there are at least r_{best} objects that are ranked better than q in every super-space of S. If so, all the super-spaces of S can be pruned. Please note that the condition OAMiner checks is sufficient, but not necessary.

Recall that the common factor c (Equation 6) does not affect the outlyingness rank. For simplicity, OAMiner computes the quasi-density $\tilde{f}_S(o)$ (Equation 7) instead of probability density $\hat{f}_S(o)$ (Equation 5) for ranking. Then, we have the following monotonicity of $\tilde{f}_S(o)$ with respect to subspaces.

Lemma 1 *Consider a set of objects O, and two subspaces S and S' satisfying $S' \supset S$. Let $D_i \in S' \setminus S$. If the standard deviation of O in D_i is greater than 0, then for any object $o \in O$, $\tilde{f}_S(o) > \tilde{f}_{S'}(o)$.*

Proof *Consider $D_i \in S' \setminus S$, for any object $o' \in O$, we have $\frac{(o.D_i - o'.D_i)^2}{2h_{D_i}^2} \geq 0$. Since the standard deviation of O in D_i is greater than 0, there exists at least one object $o'' \in O$, such that $\frac{(o.D_i - o''.D_i)^2}{2h_{D_i}^2} > 0$, that is, $e^{-\frac{(o.D_i - o''.D_i)^2}{2h_{D_i}^2}} < 1$. Thus,*

$$\begin{aligned}\tilde{f}_S(o) &= \sum_{o' \in O} e^{-\sum_{D_i \in S} \frac{(o.D_i - o'.D_i)^2}{2h_{D_i}^2}} \\ &> \sum_{o' \in O} e^{-\left(\sum_{D_i \in S} \frac{(o.D_i - o'.D_i)^2}{2h_{D_i}^2} + \sum_{D_i \in S' \setminus S} \frac{(o.D_i - o'.D_i)^2}{2h_{D_i}^2}\right)} = \tilde{f}_{S'}(o)\end{aligned}$$

■

Recall that OAMiner removes the dimensions with standard deviation 0 in the preprocessing step (Step 2 in Algorithm 2). Thus, the standard deviation of any dimension $D_i \in S' \setminus S$ is greater than 0.

OAMiner sorts all dimensions in D in the ascending order of $rank_{D_i}(q)$ ($D_i \in D$), and traverses the subspace set enumeration tree in the depth-first manner. Denote by R the ascending order of $rank_{D_i}(q)$. For a subspace $S = \{D_{i_1}, ..., D_{i_m}\}$, listing in R, let $R(S) = \{D_j \mid D_j \text{ is behind } D_{i_m} \text{ in } R\}$. By Lemma 1, for any subspace S' such that $S \subset S' \subseteq S \cup R(S)$, the minimum quasi-density of q, denoted by $\tilde{f}^{min}_{sup(S)}(q)$, is $\tilde{f}_{S \cup R(S)}(q)$. An object $o \in O$ is called a *competitor* of q in S if $\tilde{f}_S(o) < \tilde{f}^{min}_{sup(S)}(q)$. The set of competitors of q in S is denoted by $Comp_S(q)$. Clearly, for any $o \in Comp_S(q)$, by Lemma 1 we have $\tilde{f}_{S'}(o) < \tilde{f}_S(o) < \tilde{f}^{min}_{sup(S)}(q) \leq \tilde{f}_{S'}(q)$. Thus, $rank_{S'}(o) < rank_{S'}(q)$. Moreover, we have the following property of $Comp_S(q)$.

Property 3 Given a query object q and a subspace S, for any subspace S' such that $S \subset S'$, $Comp_S(q) \subseteq Comp_{S'}(q)$.

Proof Since $S \subset S'$, by Lemma 1, for any $o \in Comp_S(q)$,

$$\tilde{f}_{S'}(o) < \tilde{f}_S(o) < \tilde{f}^{min}_{sup(S)}(q).$$

Since $\tilde{f}^{min}_{sup(S)}(q) \leq \tilde{f}^{min}_{sup(S')}(q)$, we have

$$\tilde{f}_{S'}(o) < \tilde{f}_S(o) < \tilde{f}^{min}_{sup(S)}(q) \leq \tilde{f}^{min}_{sup(S')}(q).$$

Thus, $o \in Comp_{S'}(q)$. That is, $Comp_S(q) \subseteq Comp_{S'}(q)$. ∎

Correspondingly, OAMiner performs subspace pruning based on the number of competitors.

Pruning Rule 2 *When S is met in a depth-first search of the subspace set enumeration tree, let r_{best} be the best rank of q in all the subspaces searched so far. If there are at least r_{best} competitors of q in S, i.e., $|Comp_S(q)| \geq r_{best}$, then all proper super-spaces of S can be pruned.*

Next, we discuss how to compute $\tilde{f}^{min}_{sup(S)}(q)$ when the maximum dimensionality threshold of an outlying aspect, ℓ, is less than $|S| + |R(S)|$. In this situation, $|S| < |S'| \leq \ell < |S| + |R(S)|$. Clearly, it is unsuitable to use $\tilde{f}_{S \cup R(S)}(q)$ as $\tilde{f}^{min}_{sup(S)}(q)$. Intuitively, we can set $\tilde{f}^{min}_{sup(S)}(q)$ to $\min\{\tilde{f}_{S'}(q) \mid |S'| = \ell, S \subset S' \subset S \cup R(S)\}$. However, the computational cost may be high, since the number of candidates is $\binom{|R(S)|}{\ell - |S|}$. Alternatively, we suggest a method to efficiently compute $\tilde{f}^{min}_{sup(S)}(q)$, which uses a lower bound of $\tilde{f}_{S'}(q)$.

For object o', the quasi-density contribution of o' to q in S, denoted by $\tilde{f}_S(q, o')$, is $e^{-\sum_{D_i \in S} \frac{(q.D_i - o'.D_i)^2}{2h^2_{D_i}}}$. Let $R(S, o')$ be the set of $(\ell - |S|)$ dimensions in $R(S)$ with the largest values of $\frac{|q.D_j - o'.D_j|}{h_{D_j}}$ ($D_j \in R(S)$). Then, the minimum

Algorithm 4 $rank_S(q)$ – OAMiner

Input: query object $q \in O$, subspace S, the set of competitors of q discovered in the parent-subspace of S $Comp$ ($Comp$ is empty if $|S| = 1$), and the best rank of q in the subspaces searched so far r_{best}
Output: $rank_S(q)$

```
1: compute f̃_S(q) using Equation 7;
2: rank_S(q) ← |Comp| + 1;
3: for each object o ∈ O \ Comp do
4:     if f̃_S(o) < f̃_S(q) then
5:         rank_S(q) ← rank_S(q) + 1;
6:         if f̃_S(o) < f̃^min_sup(S)(q) then
7:             Comp ← Comp ∪ {o};
8:             if |Comp| = r_best then
9:                 prune super-spaces of S and return; // pruning rule 2
10:            end if
11:        end if
12:        if rank_S(q) > r_best then
13:            return;
14:        end if
15:    end if
16: end for
17: return rank_S(q);
```

quasi-density contribution of o' to q in S' ($S \subset S'$) is $\tilde{f}_{S \cup R(S,o')}(q, o')$. Since $\tilde{f}_{S'}(q) = \sum_{o' \in O} \tilde{f}_{S'}(q, o')$, we have $\tilde{f}^{min}_{sup(S)}(q) = \sum_{o' \in O} \tilde{f}_{S \cup R(S,o')}(q, o') \leq \tilde{f}_{S'}(q)$.

Please note that if we compare $\tilde{f}^{min}_{sup(S)}(q)$ with the quasi-density values of all objects in O, the computational cost for density estimation is considerably high. Especially, when the size of O is large, for the sake of efficiency, we make a tradeoff between subspace pruning and object pruning. Specifically, when we are searching a subspace S, once we can determine that $rank_S(q) > r_{best}$, then we terminate the search of S immediately.

Algorithm 4 gives the pseudo-code of computing outlyingness rank and pruning subspaces in OAMiner. Theorem 3 guarantees that Algorithm 4 can find all minimal outlying subspaces.

Theorem 3 (Completeness of OAMiner) *Given a set of objects O in a multi-dimensional space D, a query object $q \in O$ and a maximum dimensionality threshold $0 < \ell \leq |D|$, OAMiner finds all minimal outlying subspaces of q.*

The proof of Theorem 3 is given in Appendix H.

6 Empirical Evaluation

In this section, we report a systematic empirical study using several real data sets and synthetic data sets to verify the effectiveness and efficiency of our method. All experiments were conducted on a PC with an Intel Core i7-3770 3.40 GHz CPU and 8 GB main memory, running the Windows 7 operating

Table 4 The 20 technical statistics

1: Game played	6: 3-Points (M)	11: Free throw (Pct)	16: Turnover
2: Minutes	7: 3-Points (A)	12: Rebounds (Off)	17: Steal
3: Field goal (M)	8: 3-Points (Pct)	13: Rebounds (Def)	18: Block
4: Field goal (A)	9: Free throw (M)	14: Rebounds (Tot)	19: Personal foul
5: Field goal (Pct)	10: Free throw (A)	15: Assist	20: Points/game

Table 5 Data set characteristics

Data set	# objects	# attributes	Data set	# objects	# attributes
Guards	220	20	Climate model	540	18
Forwards	160	20	Concrete slump	103	10
Centers	46	17	Parkinsons	195	22
Breast cancer	194	33	Wine	178	13

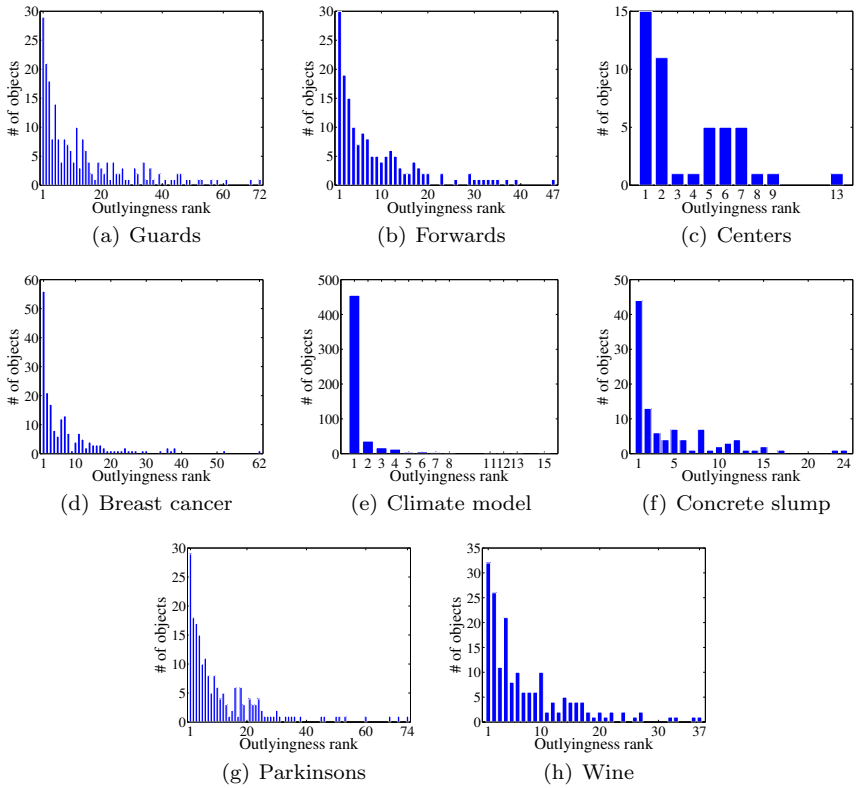
system. The algorithms were implemented in Java and compiled by JDK 7. Since it may likely be too hard for the user to understand the meaning of subspaces with dimensionality more than 5, we set $\ell = 5$ and $\alpha = 1.0$ as default in OAMiner.

6.1 Mining Outlying Aspects on Real Data Sets

NBA coaches, sport agents, and commentators may want to know in which aspects a player is most unusual. Using this application scenario as a case study, we first investigate the outlying aspects of all NBA guards, forwards and centers in the 2012-2013 Season. We collect the technical statistics on 20 numerical attributes from `http://sports.yahoo.com/nba/stats`. Table 4 shows the names of dimensions. The statistics for centers on 3-points (items 6, 7 and 8) are removed since the statistics for most centers are 0. Besides, we apply OAMiner to several real world data sets from the UCI repository (Bache and Lichman, 2013). In our experiments, we remove non-numerical attributes and all instances containing missing values. Table 5 shows the data characteristics.

For each data set, we take each record as a query object q, and apply OAMiner to discover the outlying aspects of q. Figure 3 shows the distributions of the best outlyingness ranks of objects on the data sets. Surprisingly, the best outlyingness ranks of most objects are small, that is, most objects are ranked very good in outlyingness in some subspaces. For example, 90 guards (40.9%), 81 forwards (50.6%) and 32 centers (69.6%) have an outlyingness rank of 5 or better. Most players have some subspaces where they are substantially different from the others. The observation justifies the need for outlying aspect mining.

Figure 4 shows the distributions of the number of the minimal outlying subspaces where the objects achieve the best outlyingness rank on the data sets. For most objects, the number of outlying aspects is small, which is also surprising. As shown in Figure 4(a), 150 (68.2%) objects in Guards have only 1 outlying aspect. This indicates that most objects can be distinguished from the others using a small number of factors.



(a) Guards (b) Forwards (c) Centers

(d) Breast cancer (e) Climate model (f) Concrete slump

(g) Parkinsons (h) Wine

Fig. 3 Distributions of outlyingness ranks ($\ell = 5$)

Table 6 summarizes the mining results of OAMiner on real data sets when $\ell = 4, 5, 6$, respectively. Not surprisingly, the smallest values of outlyingness rank, number of outlying aspects, dimensionality are 1. With larger value of ℓ, the average outlyingness rank decreases, while the average number of outlying aspects and the average dimensionality increase. In addition, we can see that more outlying aspects with a higher dimensionality can be found on data sets with more attributes and more instances. For example, the average number of outlying aspects discovered from Breast cancer is the largest.

6.2 Outlying Aspects Discovery on Synthetic Data Sets

Keller et al (2012) provided a collection of synthetic data sets, each consisting 1000 data objects. Each data set contains some subspace outliers, which deviate from all clusters in at least one 2-5 dimensional subspace. As stated in Keller et al (2012), an object can be an outlier in multiple subspaces independently. We perform test on the data sets of 10, 20, 30, 40, 50 dimensions,

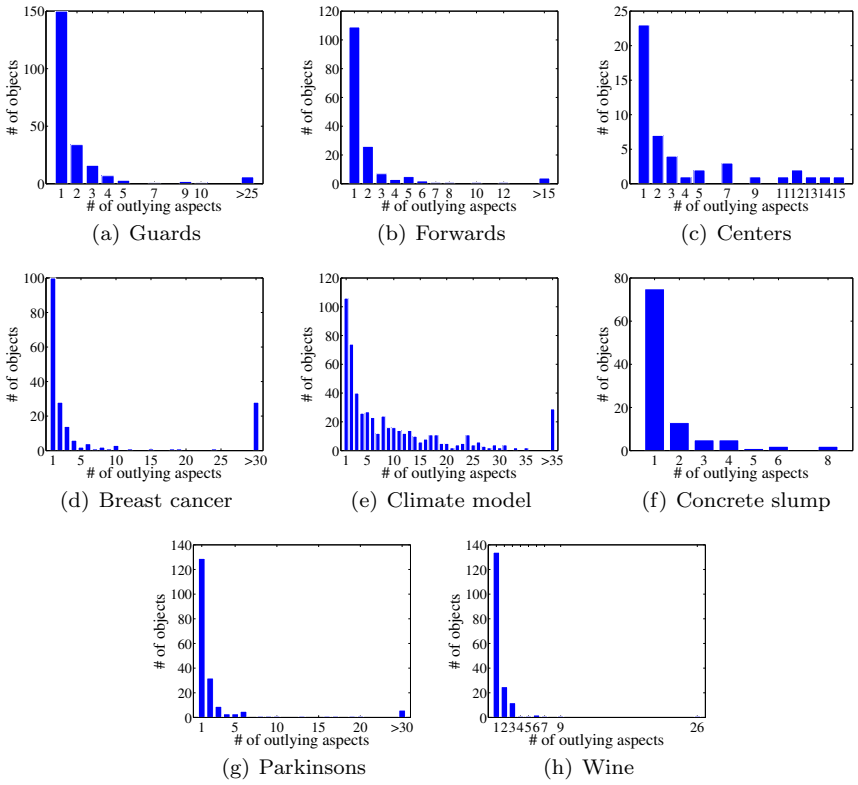


Fig. 4 Distributions of total number of outlying aspects ($\ell = 5$)

and denote the data sets by $Synth\_10D$, $Synth\_20D$, $Synth\_30D$, $Synth\_40D$, $Synth\_50D$, respectively.

For an outlier q in a data set, let S be the ground truth about outlying subspace of q. Please note that S may not be an outlying aspect of q if there exists another outlier more outlying than q in S, since OAMiner finds the subspaces whereby the query object is most outlying. To verify the effectiveness of OAMiner using the known ground truth about outlying subspaces, in the case of multiple implanted outliers in S, we keep q and remove the other outliers, and take q as the query object. Since q is the only implanted strong outlier in subspace S, OAMiner is expected to find the ground truth outlying subspace S where q takes rank 1 in outlyingness, that is, $rank_S(q) = 1$.

We divide the mining results of OAMiner into the following 3 cases:

- Case 1: only the ground truth outlying subspace is discovered by OAMiner with outlyingness rank 1.
- Case 2: besides the ground truth outlying subspace, OAMiner finds other outlying aspects with outlyingness rank 1.

Table 6 Sensitivity of OAMiner's effectiveness w.r.t. parameter ℓ

Data set	ℓ	Outlyingness rank			# of outlying aspects			Dimensionality		
		Min.	Max.	Avg.	Min.	Max.	Avg.	Min.	Max.	Avg.
Guards	4	1	72	13.94	1	49	2.02	1	4	2.79
	5	1	72	13.70	1	111	3.05	1	5	3.68
	6	1	72	13.50	1	359	5.67	1	6	4.83
Forwards	4	1	48	8.79	1	40	2.24	1	4	2.77
	5	1	47	8.54	1	41	2.37	1	5	3.13
	6	1	46	8.43	1	71	2.93	1	6	3.77
Centers	4	1	13	3.70	1	15	3.28	1	4	2.74
	5	1	13	3.57	1	15	3.65	1	5	3.08
	6	1	13	3.54	1	18	3.61	1	6	3.23
Breast cancer	4	1	70	8.04	1	232	9.57	1	4	3.47
	5	1	62	7.74	1	2478	43.37	1	5	4.67
	6	1	56	7.57	1	11681	243.10	1	6	5.77
Climate model	4	1	33	1.97	1	30	4.57	1	4	3.65
	5	1	15	1.45	1	78	10.18	1	5	4.43
	6	1	15	1.28	1	149	16.97	1	6	5.07
Concrete slump	4	1	27	4.67	1	8	1.56	1	4	2.38
	5	1	24	4.44	1	8	1.64	1	5	2.59
	6	1	24	4.41	1	8	1.65	1	6	2.66
Parkinsons	4	1	74	12.13	1	156	4.20	1	4	3.25
	5	1	74	11.51	1	400	7.63	1	5	4.09
	6	1	74	11.33	1	889	14.30	1	6	5.01
Wine	4	1	37	7.65	1	26	1.49	1	4	2.66
	5	1	37	7.47	1	26	1.59	1	5	2.96
	6	1	37	7.46	1	26	1.66	1	6	3.09

– Case 3: instead of the ground truth outlying subspace, OAMiner finds a subset of the ground truth as an outlying aspect with outlyingness rank 1.

Table 7 lists the mining results[2] on $Synth\_10D$. For all outliers (query objects), outlying aspects with outlyingness rank 1 are discovered. Moreover, we can see that for objects 183, 315, 577, 704, 754, 765 and 975, OAMiner finds not only the ground truth outlying subspace, but also some other outlying subspaces (Case 2). For object 245, the outlying aspect discovered by OAMiner is a subset of the ground truth outlying subspace (Case 3). For the other 11 objects, the outlying aspects discovered by OAMiner are identical with the ground truth outlying subspaces (Case 1).

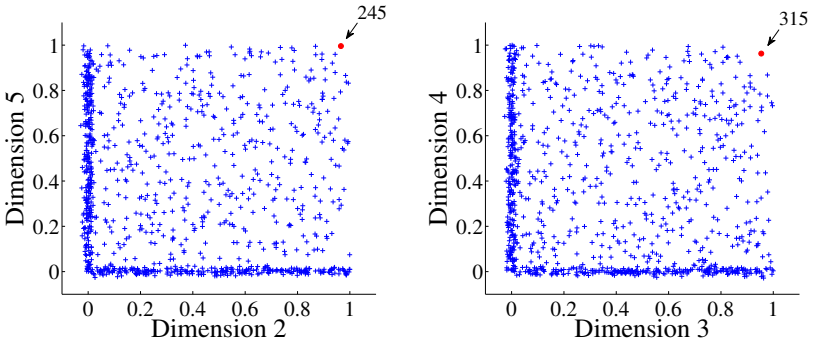
To further demonstrate the effectiveness of OAMiner, for object 245 in Case 2, we illustrate the outlying aspect $\{2, 5\}$ in Figure 5(a), and for object 315 in Case 3, we illustrate the outlying aspect $\{3, 4\}$ in Figure 5(b). Visually, the objects show outlying characteristics in the corresponding outlying aspects.

Table 8 summarizes the mining results of OAMiner on the synthetic data sets of 10, 20, 30, 40, 50 dimensions. As OAMiner finds all subspaces in which the outlyingness rank of the query object are the minimum, we can see that the number of Case 2 increases with higher dimensionality. In other words, more outlying aspects can be found on data sets with more attributes. Please

[2] The object id and dimension id in Tables 7 and 8 are consistent with the original data sets in Keller et al (2012).

Table 7 Outlying aspects on $Synth\_10D$

Query object	Ground truth outlying subspace	Outlying aspect with outlyingness rank 1	Description
172	$\{8,9\}$	$\{8,9\}$	Case 1
183	$\{0,1\}$	$\{0,1\}, \{0,6,8\}$	Case 2
184	$\{6,7\}$	$\{6,7\}$	Case 1
207	$\{0,1\}$	$\{0,1\}$	Case 1
220	$\{2,3,4,5\}$	$\{2,3,4,5\}$	Case 1
245	$\{2,3,4,5\}$	$\{2,5\}$	Case 3
315	$\{0,1\}, \{6,7\}$	$\{0,1\}, \{6,7\}, \{3,4\}, \{3,5,9\}, \{4,6,9\}$	Case 2
323	$\{8,9\}$	$\{8,9\}$	Case 1
477	$\{0,1\}$	$\{0,1\}$	Case 1
510	$\{0,1\}$	$\{0,1\}$	Case 1
577	$\{2,3,4,5\}$	$\{2,3,4,5\}, \{0,3,7\}$	Case 2
654	$\{2,3,4,5\}$	$\{2,3,4,5\}$	Case 1
704	$\{8,9\}$	$\{8,9\}, \{0,2,3,4\}$	Case 2
723	$\{2,3,4,5\}$	$\{2,3,4,5\}$	Case 1
754	$\{6,7\}$	$\{6,7\}, \{2,4,8\}, \{2,6,8\}, \{4,6,8\}$	Case 2
765	$\{6,7\}$	$\{6,7\}, \{1,4,6\}, \{3,4,5,6\}$	Case 2
781	$\{6,7\}$	$\{6,7\}$	Case 1
824	$\{8,9\}$	$\{8,9\}$	Case 1
975	$\{8,9\}$	$\{8,9\}, \{2,5,9\}, \{5,6,8\}, \{2,3,5,8\}$	Case 2



(a) Object 245 (the solid circle •) (b) Object 315 (the solid circle •)

Fig. 5 Outlying aspects for objects 245 and 315 in $Synth\_10D$

note that this observation is consistent with the experimental observations in real data sets (Section 6.1). In addition, the number of Case 3 increases a bit, since OAMiner applies the dimensionality minimality condition to outlying aspect mining.

6.3 Outlying Aspects Discovery on NBA Data Sets

As a real case study, we verified the usefulness of outlying aspect mining by analyzing the outlying aspects of some NBA players.

Please note that "outlying" is different from "outstanding". A player receives a good outlyingness rank in a subspace if very few other players

Table 8 Statistics on the mining results of OAMiner on synthetic data sets

Data set	# of outliers	# of Case 1	# of Case 2	# of Case 3
$Synth\_10D$	19	11	7	1
$Synth\_20D$	25	1	23	1
$Synth\_30D$	44	0	40	4
$Synth\_40D$	53	0	52	1
$Synth\_50D$	68	0	65	3

are close to him in the subspace, regardless of whether the performance is "good" or not. Table 9 lists 10 guards who have the largest number of rank-1 outlying aspects, where the dimensions are represented by their serial numbers in Table 4. (Due to space limits, Table 9 only lists the outlying aspects whose dimensionality are not greater than 3.)

In Table 9, the first several players are not well-known. Their low outlyingness ranks arise due to no other players having similar statistics. For example, Quentin Richardson, who has 18 outlying aspects, just played one game in which he played very well at rebounds, but poor at field goal. Will Conroy played four games and his performance on shooting is poor. Brandon Rush played two games, and his number of personal fouls is large. Ricky Rubio performs well at stealing. Rajon Rondo's ability to assist is impressive, but his statistics for turnover is large. Scott Machado did not make any personal foul in the six games he played. The last four players in Table 9 are famous. Their overall performance on every aspect is much better than most of the other guards. For example, Kobe Bryant is a great scorer, Jamal Crawford's personal fouls are very low, James Harden is excellent at the free throw, and Stephen Curry leads in 3-points scoring.

Please note that different objects may share some outlying aspects with the same outlyingness rank. For example, both Quentin Richardson and Will Conroy are ranked number 1 in $\{5, 8\}$. There are two reasons for this situation. First, the values of objects are identical in these subspaces. Second, the difference between the outlyingness degrees is so tiny that it is beyond the precision of the program.

Table 10 lists the guards who have poor outlyingness ranks overall (i.e. there are not any subspaces where they are ranked particularly well). Their performance statistics is in the middle of the road, and do not have any obvious shortcomings. They may be important to be included in a team as "the sixth man", even though they are not star performers.

As mentioned in Section 3, subspace outlier detection is fundamentally different from outlying aspect mining, since subspace outlier detection finds contrast subspaces for all possible outliers. However, we can make use of the results of subspace outlier ranking to verify to some extent our discovered outlying aspects. Specifically, we look at the objects that are ranked the best by either HiCS (Keller et al, 2012) or SOD (Kriegel et al, 2009), and check their outlyingness ranks. As HiCS randomly selects subspace slices, we run it 3 times independently on each data set with the default parameters. The parameter for the number of nearest neighbors in both LOF and SOD was varied across

Table 9 The guards having the most rank-1 outlying aspects

Name	Outlying aspects ($\ell = 3$)
Quentin Richardson	$\{1\}$, $\{12\}$, $\{14\}$, $\{2,17\}$, $\{3,4\}$, $\{3,13\}$, $\{4,17\}$, $\{5,8\}$, $\{5,11\}$, $\{5,13\}$, $\{13,17\}$, $\{13,20\}$, $\{2,3,16\}$, $\{2,4,5\}$, $\{2,5,6\}$, $\{2,5,7\}$, $\{2,5,9\}$, $\{4,5,7\}$
Will Conroy	$\{2,5\}$, $\{5,8\}$, $\{5,11\}$, $\{5,12\}$, $\{5,13\}$, $\{5,14\}$, $\{5,16\}$, $\{4,5,6\}$, $\{4,5,9\}$, $\{4,5,10\}$, $\{4,5,7\}$, $\{4,5,19\}$, $\{5,6,7\}$, $\{5,7,9\}$
Brandon Rush	$\{5\}$, $\{1,19\}$, $\{2,19\}$, $\{17,19\}$
Ricky Rubio	$\{3,17\}$, $\{7,17\}$, $\{16,17\}$, $\{17,20\}$
Rajon Rondo	$\{15\}$, $\{16\}$, $\{1,17\}$, $\{1,2,20\}$
Scott Machado	$\{19\}$, $\{2,16\}$, $\{5,8,18\}$
Kobe Bryant	$\{3\}$, $\{4\}$, $\{20\}$
Jamal Crawford	$\{19,20\}$, $\{4,19\}$, $\{2,3,19\}$
James Harden	$\{9\}$, $\{10\}$
Stephen Curry	$\{6\}$, $\{7\}$

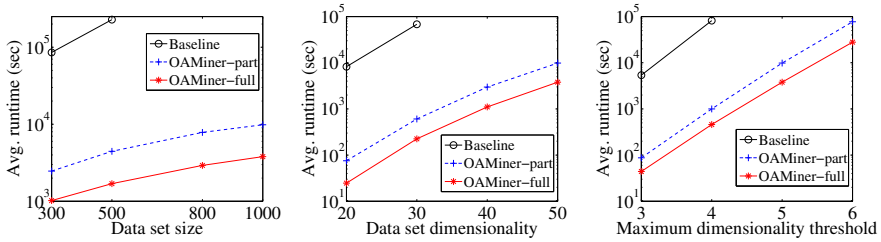
Table 10 The guards having poor ranks in outlying aspects

Outlyingness rank	Name	Outlying aspects
72	Terrence Ross	{11}
70	E'Twaun Moore	{18}
69	C.J. Watson	{8, 12, 13, 14, 18}
61	Jerryd Bayless	{2, 3, 4, 19, 20}
58	Nando De Colo	{1, 2}, {3, 4, 5, 11, 20}
56	Alec Burks	{2, 9, 10, 11}
55	Rodrigue Beaubois	{1, 2, 8, 11, 15}
52	Marco Belinelli	{9, 10, 12}
49	Aaron Brooks	{2, 3, 5, 7, 16}
48	Nick Young	{1, 3, 16, 18, 20}

Table 11 The outlyingness ranks of players ranked top in HiCS or SOD

Position	Name	$rank_{HL}$	$rank_{SOD}$	$rank_S$ (# of outlying aspects)
guard	Quentin Richardson	1	1	1 (54)
	Kobe Bryant	1	9	1 (3)
	Brandon Roy	32	1	1 (4)
forward	Carmelo Anthony	1	5	1 (26)
	Kevin Love	3	1	1 (41)
center	Dwight Howard	1	2	1 (15)
	Andrew Bogut	10	1	1 (9)

5, 10 and 20, and the best ranks were reported. In SOD (Kriegel et al, 2009), the parameter l specifying the size of the reference sets cannot be larger than the number of nearest neighbors. We set it to the number of nearest neighbors. For a given object, we denote by $rank_{HL}$ and $rank_{SOD}$ the ranks computed by HiCS and by SOD, respectively. We denote by $rank_S$ the outlyingness rank computed by OAMiner. Table 11 shows the results. The results clearly show that every player ranked top in either HiCS or SOD has some outlying subspaces where he is ranked number 1. The results of outlying aspect mining are consistent with those of subspace outlier ranking. At the same time, we notice that the rankings of HiCS and SOD are not always consistent with each other, such as for Kobe Bryant, Brandon Roy and Andrew Bogut.



(a) Runtime w.r.t. data set size (b) Runtime w.r.t. data set dimensionality (c) Runtime w.r.t. ℓ

Fig. 6 Efficiency test

6.4 Efficiency

To the best of our knowledge, there is no other method tackling the exact same problem as OAMiner. Therefore, we only evaluate the efficiency of OAMiner and its variations. Specifically, we implemented the baseline method (Algorithm 1 with Pruning Rule 1). Recall that OAMiner uses both upper and lower bounds of quasi-density to speed up the computation of outlyingness ranks. To evaluate the efficiency of our techniques for quasi-density comparison, we implemented a version OAMiner-part that does not use bounds in quasi-density estimation and strategies presented in Section 5.2. Moreover, we implement the full version OAMiner-full that uses all techniques.

Once again, we used a synthetic data set from Keller et al (2012). The dimensionality of the data set is 50, and the data set consists of 1000 data points. We randomly chose 10 data points (non-outliers) from the data set as query objects, and reported the average runtime. Again, we set $\ell = 5$ for all three methods and $\alpha = 1.0$ for OAMiner-full by default.

Figure 6(a) shows the runtime with respect to data set size. The runtime is plotted using the logarithmic scale. The baseline method is time consuming, which is consistent with our analysis. Our pruning techniques can achieve a roughly linear runtime in practice. Both versions of OAMiner are substantially faster than the baseline method. Moreover, OAMiner-full is more efficient than OAMiner-part.

Figure 6(b) shows the runtime with respect to dimensionality. The runtime is also plotted using the logarithmic scale. As dimensionality increases, the runtime increases exponentially. However, our heuristic pruning techniques speed up the search in practice. Again, OAMiner-full is more efficient than OAMiner-part.

Figure 6(c) shows the runtime with respect to maximum dimensionality threshold (ℓ). The runtime is plotted using the logarithmic scale, too. As ℓ increases, more subspaces will be enumerated. Correspondingly, the runtime increases. Once more, both versions of OAMiner are considerably faster than the baseline method, and OAMiner-full is more efficient than OAMiner-part.

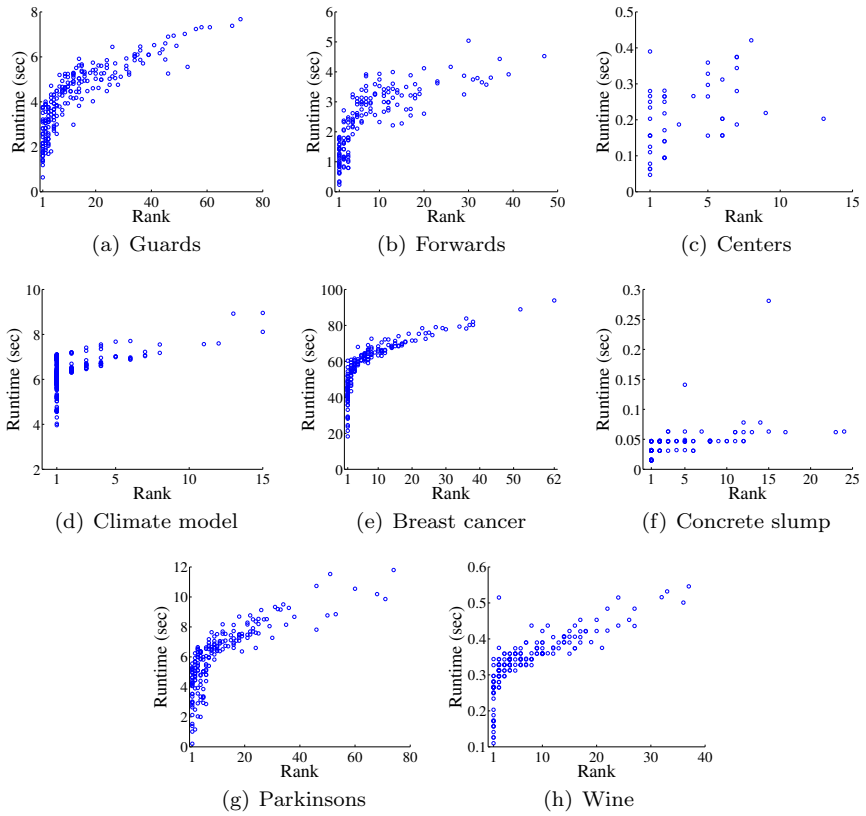


Fig. 7 Runtime w.r.t. outlyingness rank

We also notice that the runtime of OAMiner is related with the outlyingness rank of the query object. Figure 7 shows the runtime with respect to outlyingness rank on each real data set. Not surprisingly, the objects with large outlyingness rank cost more runtime, since OAMiner prunes subspaces based on the rank of the query object by either Pruning Rule 1 or Pruning Rule 2.

Last, we test the sensitivity of the parameter α for bounding quasi-density. We vary the parameter α, which sets the ϵ-neighborhood distance. Table 12 lists the average runtime of OAMiner for a query object on each real data set. The runtime of OAMiner is not sensitive to α in general. Experimentally, the shortest runtime of OAMiner happens when α is in $[0.8, 1.2]$.

7 Discussions and Conclusions

In this paper, we studied the novel and interesting problem of finding outlying aspects of a query object on multidimensional numeric data. We systematically developed a model and a heuristic method. Using both real and synthetic

Table 12 Average runtime of OAMiner w.r.t parameter α

Data set	Average runtime (sec)				
	$\alpha = 0.6$	$\alpha = 0.8$	$\alpha = 1.0$	$\alpha = 1.2$	$\alpha = 1.4$
Guards	4.459	4.234	**4.213**	4.303	4.315
Forwards	2.810	2.519	2.424	2.418	**2.413**
Centers	0.260	0.234	0.216	**0.212**	0.220
Breast cancer	58.476	58.228	57.927	**57.613**	57.982
Climate model	6.334	6.268	6.339	**6.253**	6.410
Concrete slump	0.047	**0.044**	0.044	0.045	0.045
Parkinsons	6.164	6.154	**6.083**	6.218	6.243
Wine	0.351	0.341	**0.339**	0.344	0.350

data sets, we verified that mining outlying aspects is interesting and useful. Moreover, our experiments show that our outlying aspect mining method is effective and efficient.

There are several interesting issues that deserve research effort in the future. First, to further examine the quality of outlying aspects, we plan to compute a statistical confidence interval on the rank via bootstrap sampling, and select the subspace with tighter confidence interval on the rank. Second, since OAMiner ranks the query object among all the objects by their probability densities estimated by a Gaussian kernel , it is interesting to consider using other kernel functions or outlierness degree measures proposed by outlier detection methods, such as SOD (Kriegel et al, 2009) and LOF (Breunig et al, 2000). Third, OAMiner discovers outlying aspects for a given object. Obviously selecting appropriate query points requires domain knowledge. Selecting appropriate background data sets for contrast against the query point also requires background knowledge. It is interesting to explore strategies for incorporating domain knowledge into outlying aspect mining. In practice, it may be the case that a user may want to study a set of objects. In addition, we will explore parallel computation approaches to improve the efficiency of OAMiner, and extend OAMiner for mixed data containing both numerical and non-numerical values. Finally, it is also interesting to investigate concise representation of outlying aspects, such as maximal outlying aspects, explore relations among outlying aspects, and investigate how to measure the interpretability and the interestingness of an outlying aspect.

References

Aggarwal CC (2013) An Introduction to Outlier Analysis. Springer New York

Aggarwal CC, Yu PS (2001) Outlier detection for high dimensional data. In: ACM Sigmod Record, ACM, vol 30, pp 37–46

Agrawal R, Srikant R (1994) Fast algorithms for mining association rules. In: Proc. of the 20th Int'l Conf. on Very Large Data Bases, VLDB '94, pp 487–499

Angiulli F, Fassetti F, Palopoli L (2009) Detecting outlying properties of exceptional objects. ACM Trans Database Syst 34(1):7:1–7:62

Angiulli F, Fassetti F, Palopoli L, Manco G (2013) Outlying property detection with numerical attributes. CoRR abs/1306.3558

Bache K, Lichman M (2013) UCI machine learning repository

Bhaduri K, Matthews BL, Giannella CR (2011) Algorithms for speeding up distance-based outlier detection. In: Proc. of the 17th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining, KDD '11, pp 859–867

Böhm K, Keller F, Müller E, Nguyen HV, Vreeken J (2013) CMI: An information-theoretic contrast measure for enhancing subspace cluster and outlier detection. In: Proc. of the 13th SIAM Int'l Conf. on Data Mining, SDM '13, pp 198–206

Breunig MM, Kriegel HP, Ng RT, Sander J (2000) LOF: identifying density-based local outliers. In: Proc. of the 2000 ACM SIGMOD Int'l Conf. on Management of Data, SIGMOD '00, pp 93–104

Chandola V, Banerjee A, Kumar V (2009) Anomaly detection: A survey. ACM Comput Surv 41(3):15:1–15:58

Han J, Kamber M, Pei J (2011) Data Mining: Concepts and Techniques, 3rd edn. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA

Härdle W (1990) Smoothing Techniques: With Implementations in S. Springer-Verlang, New York

Härdle W, Werwatz A, Müller M, Sperlich S (2004) Nonparametric and Semiparametric Modelss. Springer Series in Statistics, Springer

He Z, Xu X, Huang ZJ, Deng S (2005) FP-outlier: Frequent pattern based outlier detection. Computer Science and Information Systems/ComSIS 2(1):103–118

Keller F, Müller E, Böhm K (2012) HiCS: High contrast subspaces for density-based outlier ranking. In: Proc. of the 28th Int'l Conf. on Data Engineering, ICDE '12, pp 1037–1048

Knorr EM, Ng RT (1999) Finding intensional knowledge of distance-based outliers. In: Proc. of the 25th Int'l Conf. on Very Large Data Bases, VLDB '99, pp 211–222

Kriegel HP, Schubert M, Zimek A (2008) Angle-based outlier detection in high-dimensional data. In: Proc. of the 14th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining, KDD '08, pp 444–452

Kriegel HP, Kröger P, Schubert E, Zimek A (2009) Outlier detection in axis-parallel subspaces of high dimensional data. In: Proc. of the 13th Pacific-Asia Conf. on Advances in Knowledge Discovery and Data Mining, PAKDD '09, pp 831–838

Müller E, Schiffer M, Seidl T (2011) Statistical selection of relevant subspace projections for outlier ranking. In: Proc. of the 27th IEEE Int'l Conf. on Data Engineering, ICDE '11, pp 434–445

Müller E, Assent I, Iglesias P, Mülle Y, Böhm K (2012a) Outlier ranking via subspace analysis in multiple views of the data. In: Proc. of the 12th IEEE Int'l Conf. on Data Mining, ICDM '12, pp 529–538

Müller E, Keller F, Blanc S, Böhm K (2012b) OutRules: A framework for outlier descriptions in multiple context spaces. In: ECML/PKDD (2), pp 828–832

Paravastu R, Kumar H, Pudi V (2008) Uniqueness mining. In: Proc. of the 13th Int'l Conf. on Database Systems for Advanced Applications, DASFAA '08, pp 84–94

Ramaswamy S, Rastogi R, Shim K (2000) Efficient algorithms for mining outliers from large data sets. In: Proc. of the 2000 ACM SIGMOD Int'l Conf. on Management of Data, SIGMOD '00, pp 427–438

Rymon R (1992) Search through systematic set enumeration. In: Proc. of the 3rd Int'l Conf. on Principle of Knowledge Representation and Reasoning, KR '92, pp 539–550

Scott DW (1992) Multivariate Density Estimation: Theory, Practice, and Visualization. Wiley Series in Probability and Statistics, Wiley, New York

Silverman BW (1986) Density Estimation for Statistics and Data Analysis. Chapman and Hall/CRC, London

Tang G, Bailey J, Pei J, Dong G (2013) Mining multidimensional contextual outliers from categorical relational data. In: Proc. of the 25th Int'l Conf. on Scientific and Statistical Database Management, SSDBM '13, pp 43:1–43:4

Zimek A, Schubert E, Kriegel HP (2012) A survey on unsupervised outlier detection in high-dimensional numerical data. Stat Anal Data Min 5(5):363–387

A Proof of Proposition 1

Proof For any dimension $D_i \in S$ $(1 \leq i \leq d)$, the mean value of $\{o.D_i \mid o \in O\}$, denoted by μ_i, is $\frac{1}{|O|}\sum_{o\in O} o.D_i$, the standard deviation of $\{o.D_i \mid o \in O\}$, denoted by σ_i, is $\sqrt{\frac{1}{|O|}\sum_{o\in O}(o.D_i-\mu_i)^2}$, and the bandwidth of D_i (h_i) is $1.06\min\{\sigma_i, \frac{R}{1.34}\}|O|^{-\frac{1}{5}}$, where R is the difference between the first and the third quartiles of O in D_i.

We perform the linear transformation $g(o).D_i = a_i o.D_i + b_i$ for any $o \in O$. Then, the mean value of $\{g(o).D_i \mid o \in O\}$ is $\frac{1}{|O|}\sum_{o\in O}(a_i o.D_i + b_i) = a_i\mu_i + b_i$, and the standard deviation of $\{g(o).D_i \mid o \in O\}$ is $\sqrt{\frac{1}{|O|}\sum_{o\in O}(a_i o.D_i + b_i - a_i\mu_i - b_i)^2} = a_i\sqrt{\frac{1}{|O|}\sum_{o\in O}(o.D_i-\mu_i)^2} = a_i\sigma_i$.

Correspondingly, the bandwidth of D_i is $1.06\min\{a_i\sigma_i, \frac{a_i R}{1.34}\}|O|^{-\frac{1}{5}}$ after the linear transformation. As the distance between two objects in D_i is also enlarged by a_i, the quasi-density calculated by Equation 7 keeps unchanged. Thus, the ranking is invariant under linear transformation. ∎

B Proof of Theorem 1

Proof (i) Given an object $o' \in TN_S^{\epsilon,o}$, for any dimension $D_i \in S$, $\min_{o''\in O}\{|o.D_i - o''.D_i|\} \leq |o.D_i - o'.D_i| \leq \epsilon_{D_i}$. Thus,

$$e^{-\sum_{D_i\in S}\frac{\epsilon_{D_i}^2}{2h_{D_i}^2}} \leq e^{-\sum_{D_i\in S}\frac{|o.D_i-o'.D_i|^2}{2h_{D_i}^2}} \leq e^{-\sum_{D_i\in S}\frac{\min_{o''\in O}\{|o.D_i-o''.D_i|\}^2}{2h_{D_i}^2}}.$$

That is, $dc_S^{\epsilon} \leq dc_S(o,o') \leq dc_S^{max}(o)$.

(ii) Given an object $o' \in LN_S^{\epsilon,o} \setminus TN_S^{\epsilon,o}$, for any dimension $D_i \in S$, $\min\limits_{o''\in O}\{|o.D_i - o''.D_i|\} \leq |o.D_i - o'.D_i| \leq \max\limits_{o''\in O}\{|o.D_i - o''.D_i|\}$. Thus,

$$e^{-\sum\limits_{D_i\in S}\frac{\max\limits_{o''\in O}\{|o.D_i-o''.D_i|\}^2}{2h_{D_i}^2}} \leq e^{-\sum\limits_{D_i\in S}\frac{|o.D_i-o'.D_i|^2}{2h_{D_i}^2}} \leq e^{-\sum\limits_{D_i\in S}\frac{\min\limits_{o''\in O}\{|o.D_i-o''.D_i|\}^2}{2h_{D_i}^2}}.$$

That is, $dc_S^{min}(o) \leq dc_S(o,o') \leq dc_S^{max}(o)$.

(iii) Given an object $o' \in O \setminus LN_S^{\epsilon,o}$, for any dimension $D_i \in S$, $\epsilon_{D_i} < |o.D_i - o'.D_i| \leq \max\limits_{o''\in O}\{|o.D_i - o''.D_i|\}$. Thus,

$$e^{-\sum\limits_{D_i\in S}\frac{\max\limits_{o''\in O}\{|o.D_i-o''.D_i|\}^2}{2h_{D_i}^2}} \leq e^{-\sum\limits_{D_i\in S}\frac{|o.D_i-o'.D_i|^2}{2h_{D_i}^2}} < e^{-\sum\limits_{D_i\in S}\frac{\epsilon_{D_i}^2}{2h_{D_i}^2}}.$$

That is, $dc_S^{min}(o) \leq dc_S(o,o') < dc_S^{\epsilon}$. ∎

C Proof of Corollary 1

Proof We divide O into three disjoint subsets $TN_S^{\epsilon,o}$, $LN_S^{\epsilon,o} \setminus TN_S^{\epsilon,o}$ and $O \setminus LN_S^{\epsilon,o}$. By Theorem 1, for objects belonging to $TN_S^{\epsilon,o}$, we have

$$|TN_S^{\epsilon,o}|\ dc_S^{\epsilon} \leq \sum_{o'\in TN_S^{\epsilon,o}} dc_S(o,o') \leq |TN_S^{\epsilon,o}|\ dc_S^{max}(o)$$

For objects belonging to $LN_S^{\epsilon,o} \setminus TN_S^{\epsilon,o}$, we have

$$(|LN_S^{\epsilon,o}| - |TN_S^{\epsilon,o}|)\ dc_S^{min}(o) \leq \sum_{o'\in LN_S^{\epsilon,o}\setminus TN_S^{\epsilon,o}} dc_S(o,o') \leq (|LN_S^{\epsilon,o}| - |TN_S^{\epsilon,o}|)\ dc_S^{max}(o)$$

For objects belonging to $O \setminus LN_S^{\epsilon,o}$, we have

$$(|O| - |LN_S^{\epsilon,o}|)\ dc_S^{min}(o) \leq \sum_{o'\in O\setminus LN_S^{\epsilon,o}} dc_S(o,o') < (|O| - |LN_S^{\epsilon,o}|)\ dc_S^{\epsilon}$$

As

$$\tilde{f}_S(o) = \sum_{o'\in O} dc_S(o,o') = \sum_{o'\in TN_S^{\epsilon,o}} dc_S(o,o') + \sum_{o'\in LN_S^{\epsilon,o}\setminus TN_S^{\epsilon,o}} dc_S(o,o') + \sum_{o'\in O\setminus LN_S^{\epsilon,o}} dc_S(o,o'),$$

Thus,

$$\begin{aligned}\tilde{f}_S(o) &\geq |TN_S^{\epsilon,o}|\ dc_S^{\epsilon} + (|LN_S^{\epsilon,o}| - |TN_S^{\epsilon,o}|)\ dc_S^{min}(o) + (|O| - |LN_S^{\epsilon,o}|)\ dc_S^{min}(o)\\ &= |TN_S^{\epsilon,o}|\ dc_S^{\epsilon} + (|O| - |TN_S^{\epsilon,o}|)\ dc_S^{min}(o)\end{aligned}$$

$$\begin{aligned}\tilde{f}_S(o) &\leq |TN_S^{\epsilon,o}|\ dc_S^{max}(o) + (|LN_S^{\epsilon,o}| - |TN_S^{\epsilon,o}|)\ dc_S^{max}(o) + (|O| - |LN_S^{\epsilon,o}|)\ dc_S^{\epsilon}\\ &= |LN_S^{\epsilon,o}|\ dc_S^{max}(o) + (|O| - |LN_S^{\epsilon,o}|)\ dc_S^{\epsilon}\end{aligned}$$

Moreover, if $LN_S^{\epsilon,o} \subset O$, i.e. $O \setminus LN_S^{\epsilon,o} \neq \emptyset$, then

$$\tilde{f}_S(o) < |LN_S^{\epsilon,o}|\ dc_S^{max}(o) + (|O| - |LN_S^{\epsilon,o}|)\ dc_S^{\epsilon}$$

∎

D Proof of Corollary 2

Proof Since $O' \subseteq TN_S^{\epsilon,o}$, for objects belonging to $O \setminus O'$, we divide them into $TN_S^{\epsilon,o} \setminus O'$, $LN_S^{\epsilon,o} \setminus TN_S^{\epsilon,o}$ and $O \setminus LN_S^{\epsilon,o}$. Then

$$\tilde{f}_S(o) = \tilde{f}_S^{O'}(o) + \sum_{o' \in TN_S^{\epsilon,o} \setminus O'} dc_S(o,o') + \sum_{o' \in LN_S^{\epsilon,o} \setminus TN_S^{\epsilon,o}} dc_S(o,o') + \sum_{o' \in O \setminus LN_S^{\epsilon,o}} dc_S(o,o'),$$

By Theorem 1, for objects belonging to $TN_S^{\epsilon,o} \setminus O'$, we have

$$(|TN_S^{\epsilon,o}| - |O'|)\ dc_S^{\epsilon} \leq \sum_{o' \in TN_S^{\epsilon,o} \setminus O'} dc_S(o,o') \leq (|TN_S^{\epsilon,o}| - |O'|)\ dc_S^{max}(o)$$

For objects belonging to $LN_S^{\epsilon,o} \setminus TN_S^{\epsilon,o}$, we have

$$(|LN_S^{\epsilon,o}| - |TN_S^{\epsilon,o}|)\ dc_S^{min}(o) \leq \sum_{o' \in LN_S^{\epsilon,o} \setminus TN_S^{\epsilon,o}} dc_S(o,o') \leq (|LN_S^{\epsilon,o}| - |TN_S^{\epsilon,o}|)\ dc_S^{max}(o)$$

For objects belonging to $O \setminus LN_S^{\epsilon,o}$, we have

$$(|O| - |LN_S^{\epsilon,o}|)\ dc_S^{min}(o) \leq \sum_{o' \in O \setminus LN_S^{\epsilon,o}} dc_S(o,o') < (|O| - |LN_S^{\epsilon,o}|)\ dc_S^{\epsilon}$$

Thus,

$$\begin{aligned}\tilde{f}_S(o) &\geq \tilde{f}_S^{O'}(o) + (|TN_S^{\epsilon,o}| - |O'|)\ dc_S^{\epsilon} + (|LN_S^{\epsilon,o}| - |TN_S^{\epsilon,o}|)\ dc_S^{min}(o) + (|O| - |LN_S^{\epsilon,o}|)\ dc_S^{min}(o) \\ &= \tilde{f}_S^{O'}(o) + (|TN_S^{\epsilon,o}| - |O'|)\ dc_S^{\epsilon} + (|O| - |TN_S^{\epsilon,o}|)\ dc_S^{min}(o)\end{aligned}$$

$$\begin{aligned}\tilde{f}_S(o) &\leq \tilde{f}_S^{O'}(o) + (|TN_S^{\epsilon,o}| - |O'|)\ dc_S^{max}(o) + (|LN_S^{\epsilon,o}| - |TN_S^{\epsilon,o}|)\ dc_S^{max}(o) + (|O| - |LN_S^{\epsilon,o}|)\ dc_S^{\epsilon} \\ &= \tilde{f}_S^{O'}(o) + (|LN_S^{\epsilon,o}| - |O'|)\ dc_S^{max}(o) + (|O| - |LN_S^{\epsilon,o}|)\ dc_S^{\epsilon}\end{aligned}$$

Moreover, if $LN_S^{\epsilon,o} \subset O$, i.e. $O \setminus LN_S^{\epsilon,o} \neq \emptyset$, then

$$\tilde{f}_S(o) < \tilde{f}_S^{O'}(o) + (|LN_S^{\epsilon,o}| - |O'|)\ dc_S^{max}(o) + (|O| - |LN_S^{\epsilon,o}|)\ dc_S^{\epsilon}$$

■

E Proof of Corollary 3

Proof Since $TN_S^{\epsilon,o} \subset O' \subseteq LN_S^{\epsilon,o}$, for objects belonging to $O \setminus O'$, we divide them into $LN_S^{\epsilon,o} \setminus O'$ and $O \setminus LN_S^{\epsilon,o}$. Then

$$\tilde{f}_S(o) = \tilde{f}_S^{O'}(o) + \sum_{o' \in LN_S^{\epsilon,o} \setminus O'} dc_S(o,o') + \sum_{o' \in O \setminus LN_S^{\epsilon,o}} dc_S(o,o'),$$

By Theorem 1, for objects belonging to $LN_S^{\epsilon,o} \setminus O'$, we have

$$(|LN_S^{\epsilon,o}| - |O'|)\ dc_S^{min}(o) \leq \sum_{o' \in LN_S^{\epsilon,o} \setminus TN_S^{\epsilon,o}} dc_S(o,o') \leq (|LN_S^{\epsilon,o}| - |O'|)\ dc_S^{max}(o)$$

For objects belonging to $O \setminus LN_S^{\epsilon,o}$, we have

$$(|O| - |LN_S^{\epsilon,o}|)\ dc_S^{min}(o) \leq \sum_{o' \in O \setminus LN_S^{\epsilon,o}} dc_S(o,o') < (|O| - |LN_S^{\epsilon,o}|)\ dc_S^{\epsilon}$$

Thus,

$$\begin{aligned}\tilde{f}_S(o) &\geq \tilde{f}_S^{O'}(o) + (|LN_S^{\epsilon,o}| - |O'|)\ dc_S^{min}(o) + (|O| - |LN_S^{\epsilon,o}|)\ dc_S^{min}(o)\\ &= \tilde{f}_S^{O'}(o) + (|O| - |O'|)\ dc_S^{min}(o)\\ \tilde{f}_S(o) &\leq \tilde{f}_S^{O'}(o) + (|LN_S^{\epsilon,o}| - |O'|)\ dc_S^{max}(o) + (|O| - |LN_S^{\epsilon,o}|)\ dc_S^{\epsilon}\end{aligned}$$

Moreover, if $LN_S^{\epsilon,o} \subset O$, i.e. $O \setminus LN_S^{\epsilon,o} \neq \emptyset$, then

$$\tilde{f}_S(o) < \tilde{f}_S^{O'}(o) + (|LN_S^{\epsilon,o}| - |O'|)\ dc_S^{max}(o) + (|O| - |LN_S^{\epsilon,o}|)\ dc_S^{\epsilon}$$

■

F Proof of Corollary 4

Proof Since $LN_S^{\epsilon,o} \subset O' \subseteq O$, Then

$$\tilde{f}_S(o) = \tilde{f}_S^{O'}(o) + \sum_{o' \in O \setminus O'} dc_S(o, o'),$$

By Theorem 1, for objects belonging to $O \setminus O'$, we have

$$(|LN_S^{\epsilon,o}| - |O'|)\ dc_S^{min}(o) \leq \sum_{o' \in O \setminus O'} dc_S(o, o') \leq (|O| - |O'|)\ dc_S^{\epsilon}$$

Thus,

$$\begin{aligned}\tilde{f}_S(o) &\geq \tilde{f}_S^{O'}(o) + (|O| - |O'|)\ dc_S^{min}(o)\\ \tilde{f}_S(o) &\leq \tilde{f}_S^{O'}(o) + (|O| - |O'|)\ dc_S^{\epsilon}\end{aligned}$$

■

G Proof of Theorem 2

Proof We prove by contradiction.

Given a set of objects O, a subspace S, two neighborhood distances ϵ_1 and ϵ_2. Let $q \in O$ be the query object. For an object $o \in O$, denote by L_{ϵ_1} the lower bound of $\tilde{f}_S(o)$ estimated by ϵ_1, U_{ϵ_2} the upper bound of $\tilde{f}_S(o)$ estimated by ϵ_2.

Assume that $\tilde{f}_S(q) < L_{\epsilon_1}$ and $\tilde{f}_S(q) > U_{\epsilon_2}$.

As L_{ϵ_1} is a lower bound of $\tilde{f}_S(o)$, and U_{ϵ_2} is an upper bound of $\tilde{f}_S(o)$, so that $L_{\epsilon_1} < \tilde{f}_S(o) < U_{\epsilon_2}$. Then, we have $\tilde{f}_S(q) < L_{\epsilon_1} < \tilde{f}_S(o)$ and $\tilde{f}_S(o) < U_{\epsilon_2} < \tilde{f}_S(q)$. Consequently, $\tilde{f}_S(o) < \tilde{f}_S(q) < \tilde{f}_S(o)$. A contradiction.

Thus, $rank_S^{\epsilon_1}(q) = |\{o \in O \mid \tilde{f}_S(o) < \tilde{f}_S(q)\}| + 1 = rank_S^{\epsilon_2}(q)$. ■

H Proof of Theorem 3

Proof We prove by contradiction.

Let Ans be the set of minimal outlying subspaces of q found by OAMiner, r_{best} the best rank. Assume that subspace $S \notin Ans$ satisfying $S \subseteq D$ and $0 < |S| \leq \ell$ is a minimal outlying subspace of q.

Recall that OAMiner searches subspaces by traversing the subspace enumeration tree in a depth-first manner. As $S \notin Ans$, S is pruned by Pruning Rule 1 or Pruning Rule 2.

In the case that S is pruned by Pruning Rule 1, S is not minimal. A contradiction;

In the case that S is pruned by Pruning Rule 2, then there exist a subspace S', such that S' is a parent of S in the subspace enumeration tree and $Comp_{S'}(q) \geq r_{best}$. By the property of competitors, we have $Comp_{S'}(q) \subseteq Comp_S(q)$. Correspondingly, $rank_S(q) \geq |Comp_S(q)| \geq |Comp_{S'}(q)| \geq r_{best}$. A contradiction. ∎