sGrid++: Revising Simple Grid Based Density Estimator for Mining Outlying Aspect

Durgesh Samariya[1(✉)], Jiangang Ma[1], and Sunil Aryal[2]

[1] School of Engineering, Information Technology and Physical Sciences, Federation University, Churchill, VIC, Australia
{d.samariya,j.ma}@federation.edu.au

[2] School of Information Technology, Deakin University, Geelong, VIC, Australia
sunil.aryal@deakin.edu.au

Abstract. In this paper, we address the problem of outlying aspect mining, which aims to identify a set of features (subspace(s) a.k.a aspect(s)) where a given data object stands out from the rest of the data. To detect the most outlying aspect of a given data object, outlying aspect mining algorithms need to compare and rank subspaces with different dimensionality. Thus, they require a fast and dimensionally unbias scoring measure. Existing measures use density or distance to compute the outlyingness of the query in each subspace. Density and distance are dimensionally bias, i.e. density decreases as the dimension of subspace increases. To make them comparable (dimensionally unbias), Z-score normalization is used in the previous works. However, to compute Z-score normalization, we need to compute the outlyingness of each data point in each subspace, which adds significant computational overhead on top of the already expensive density or distance computation.

Recently developed measure called sGrid is a simple and efficient density estimator which allows a fast systemic search. While it is efficient compared to other distance and density-based measures, it is also a dimensionally bias measure and it requires to use Z-score normalization to make it dimensionality unbiased, which makes it computationally expensive. In this paper, we propose a simpler version of sGrid called **sGrid++** that is not only efficient and effective but also dimensionality unbiased. It does not require Z-score normalization. We demonstrate the effectiveness and efficiency of the proposed scoring measure in outlying aspect mining using synthetic and real-world datasets.

Keywords: Outlying aspect mining · Dimensionality-unbiased score · Outlier explanation · Histogram · Density estimation

1 Introduction

Anomaly detection (**AD**) is one of the crucial tasks of data mining, besides clustering and classification, which detects anomalous data points in a data set

R. Chbeir et al. (Eds.): WISE 2022, LNCS 13724, pp. 194–208, 2022.
https://doi.org/10.1007/978-3-031-20891-1_15

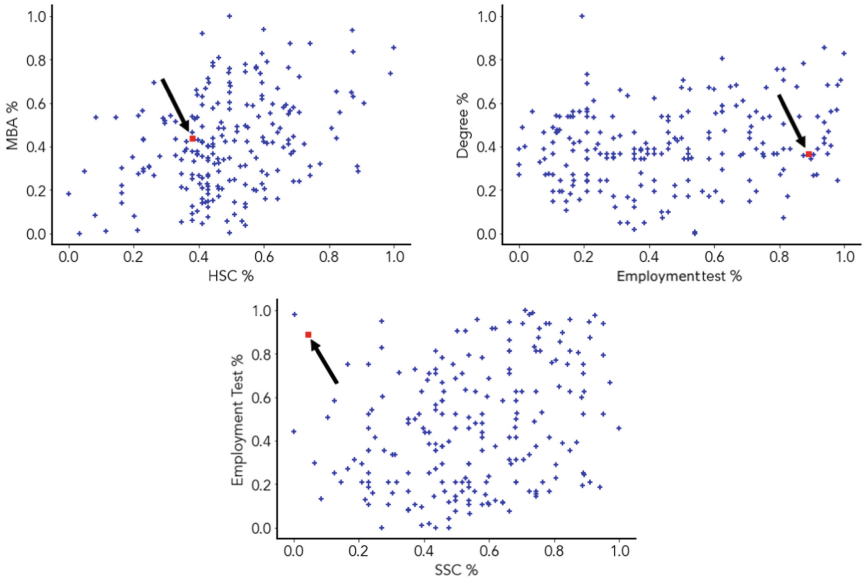


Fig. 1. University students performance on SSC %, HSC %, Degree %, MBA % and Employment test %. The red square point represents candidate A. (Color figure online)

automatically. Anomalies (also refers to as outliers) are data points that are significantly different from the other points in the data set. AD has applications in various domains such as fraud detection, medical or public health, intrusion detection, and machine fault detection [2,13]. While there are wide range of algorithms proposed in the literature to detect anomalies/outliers, they cannot explain why those outliers are flagged.

Lately, as an attempt to provide such explanation, researchers are interested in *Outlying Aspect Mining* (**OAM**) [3,9–12,14,15], where the task is to identify on what aspects (subset of features) a given anomaly/outlier exhibits the most outlying characteristics. In other words, OAM is the task of identifying feature subset(s), where a given query is significantly inconsistent with the rest of the data.

OAM has many real-world applications. For example, when evaluating job applications, recruitment team wants to know strengths/weaknesses of each candidate, i.e., they want to know in what aspect a candidate is outstanding among the applicants. Let's have a look at the example of 215 candidates data[1] with their scores in percentage (%) in their secondary school (SSC), higher secondary (HSC), undergraduate degree, MBA and Employment test. As shown in Fig. 1(c),

[1] Data set is available at https://www.kaggle.com/benroshan/factors-affecting-campus-placement.

Candidate A scored very high percentage in the Employment test while having quite low percentage in SSC.

Another example, assume that you are a football coach or commentator and you want to highlight the strengths and/or weaknesses of a player in the most recent game. Moreover, when the doctor wants to know in which aspect a given patient's condition is not normal [10].

OAM algorithm requires two techniques – (i) scoring measure and (ii) subspace search technique, to detect outlying aspect of a given query. Distance or density-based scoring measures are widely used in the OAM algorithms. The main drawback of these types of measures in OAM is that such measures are computationally expensive and dimensionality biased. Vinh et al. [14] proposed to use Z-score normalization to make density-based scoring measures dimensionality unbiased so that they can be compared to rank subspaces with different dimensionality. It requires computing outlier scores of each data point in each subspace. It adds significant computational overhead making OAM algorithms making them infeasible to run in large and/or high-dimensional datasets.

To summarise, most OAM scoring measures do not work well in practical application due to the following main reasons:

- **High time complexity:** Existing scoring measure uses distance or density, and the known weakness of these measures is, that they are computationally expensive.
- **Dimensionality unbiasedness:** In OAM, subspaces with different dimensionalities are compared to find the best subspace. Thus, we need a measure that is dimensionally unbiased to rank those subspaces. Existing density or distance-based measures are dimensionally biased.

This paper makes the following contributions:

- Propose a new scoring measure for outlying aspect mining algorithm based on sGrid density estimator. We extend the sGrid density estimator and make it dimensionally unbiased measure, thus it does not require any additional normalization. We called the proposed measure sGrid++.
- Compare sGrid++ against three existing OAM scoring measures using synthetic and real-world datasets.
- Through our empirical evaluation, we demonstrate that sGrid++ is a dimensionality unbiased measure. In addition to that, it is faster than existing scoring measures.

The rest of the paper is organized as follows. Section 2 provides a summary of previous work on outlying aspect mining. The proposed scoring measure is presented in Sect. 3. Empirical evaluation results are provided in Sect. 4. Finally, conclusions are provided in Sect. 5.

2 Related Work

Let $\mathcal{O} = \{o_1, o_2, \cdots, o_N\}$ be a collection of N data objects in M-dimensional real domain. Each object $\mathbf{o}$ is represented as M-dimensional vector

$\langle o.1, o.2, \cdots, o.M \rangle$. The feature set $\mathcal{F} = \{F_1, F_2, \cdots, F_M\}$ denotes the full feature space and $\mathcal{S}_\mathcal{F} = \{S_1, S_2, \cdots, S_n\}$ is the set of all possible subspaces (i.e., $|\mathcal{S}_\mathcal{F}| = 2^M$). The problem of OAM is to identify $S_i \in \mathcal{S}_\mathcal{F}$ in which a given query object $o_i \in \mathcal{O}$ is significantly different from the rest of the data.

2.1 Problem Formulation

Definition 1 (Problem definition). *Given a set of N instances $\mathcal{O}$ ($\|\mathcal{O}\| = N$) in M dimensional space, a query $\mathbf{q} \in \mathcal{O}$, a subspace S is called outlying aspect of $\mathbf{q}$ iff,*

- *outlyingness of $\mathbf{q}$ in subspace S is higher than other subspaces; and*
- *there is no other subspace with same or higher outlyingness.*

The main aim of outlying aspect mining is to identify a minimal outlying aspect of a given query.

2.2 Outlying Aspect Mining Techniques

Duan et al. (2015) [3] employs depth-first search [8] with kernel density estimation (KDE) based outlying measure, called OAMiner. For a given query $\mathbf{q}$ and data set $\mathcal{O}$, OAMiner computes outlyingness as follows:

$$f_S(\mathbf{q}) = \frac{1}{N(2\pi)^{\frac{m}{2}} \prod_{i \in S} h.i} \sum_{x \in \mathcal{O}} e^{-\sum_{i \in S} \frac{(\mathbf{q}.i - x.i)^2}{2h.i^2}}$$

where, $f_S(\mathbf{q})$ is a kernel density estimation of $\mathbf{q}$ in subspace S ($|S| = m$), $h.i$ is the kernel bandwidth in dimension i.

OAMiner first ranks a data point based on the density of each data point in each subspace. After ranking the data point in each subspace, it sorts the subspace in ascending order based on the score. Lastly, it returns the top-ranked subspace(s) as an outlying aspect of a given query $\mathbf{q}$.

Vinh et al. (2016) [14] discussed the issue of using density rank as an outlying measure in OAM and provided some examples of where it can be counterproductive. They suggested to use Z-score normalized density to compare subspaces of different dimensionalities. Z-score computes the outlyingness of query $\mathbf{q}$ in subspace S as:

$$\mathrm{Z}(f_S(\mathbf{q})) \triangleq \frac{f_S(\mathbf{q}) - \mu_{f_S}}{\sigma_{f_S}}$$

where μ_{f_S} and σ_{f_S} are the mean and standard deviation of densities of all data instances in subspace S, respectively.

They formulate the concept of dimensionality unbiasedness and proposed to use Z-score normalization to convert any dimensionality-biased measure to unbiased one. Authors have combined the Beam search with the density Z-score measure to identify outlying aspects of a given query.

Wells and Ting (2019) [15] proposed sGrid density estimator, which is a smoothed variant of the traditional grid-based estimator (a.k.a histogram). They also used Z-score normalization to make the score dimensionality unbiased. Because sGrid density can be computed faster than KDE, it allows Beam search OAM to run orders of magnitude faster.

Samariya et al. (2020) [9] proposed a **S**imple **I**solation score using **N**earest **N**eighbor **E**nsemble (SiNNE in short) measure. SiNNE constructs t ensemble of models $(\mathcal{M}_1, \mathcal{M}_2, \cdots, \mathcal{M}_t)$. Each model $\mathcal{M}_i$ is constructed from randomly chosen sub-samples $(\mathcal{D}_i \subset \mathcal{O}, |\mathcal{D}_i| = \psi < N)$. Each model has ψ hyperspheres, where radius of hypersphere is the euclidean distance between a $(a \in \mathcal{D}_i)$ to its nearest neighbor in $\mathcal{D}_i$.

The outlying score of $\mathbf{q}$ in model $\mathcal{M}_i$, $I(\mathbf{q} \| \mathcal{M}_i) = 0$ if $\mathbf{q}$ falls in any of the balls and 1 otherwise. The final outlying score of $\mathbf{q}$ using t models is:

$$\text{SiNNE}(\mathbf{q}) = \frac{1}{t} \sum_{i=1}^{t} I(\mathbf{q} \| \mathcal{M}_i)$$

2.3 Desired Properties of Outlying Scoring Measure

In this section, we provide some desired properties for an ideal outlying aspect mining scoring measure. In Table 1, we summarize the desired properties of existing scoring measures.

Dimensionality Unbiasedness. As we are comparing subspaces with different dimensionality, a scoring measure needs to be unbiased w.r.t. dimensionality. An example of a dimensionally bias scoring measure is the density measure, which decreases as dimension increases. As a result, density is biased towards higher-dimensional subspaces.

Efficiency. To find an outlying aspect of a given query, OAM algorithms are expected to search through a large number of subspaces. Thus, it is essential to have an efficient scoring measure to evaluate subspace efficiently. The efficiency of the scoring measure can be analyzed in terms of time complexity.

Effectiveness. To find the most outlying aspects of a given query, the OAM algorithm ranks each subspace based on its score. Thus, the scoring measure should be effective for getting better outlying aspects. The effectiveness of the scoring measure can be analyzed in terms of the quality of discovered subspaces.

3 sGrid++: The New Proposed Measure

Wells and Ting (2019) [15] introduced a simple and effective alternative of kernel density estimator called sGrid, which allows systematic search method to run faster in outlying aspect mining domain. sGrid is smoothed variant of the grid (a.k.a histogram) based density estimator. sGrid computes the density of multi-dimensional subspace as a multi-dimensional grid. The grid's width is set based on the bin width in one dimension.

Table 1. Summary of desired properties for scoring measures. Time complexity of estimating one query in a subspace is presented. (N = data size; m = dimensionality of subspace; ψ = sub-sample size; t = number of set; w = block size for bit set operation).

Scoring measure	Unbiasedness	Time complexity
Density	✘	✘($O(Nm)$)
Density rank	✔	✘($O(N^2m)$)
Density Z-score	✔	✘($O(N^2m)$)
sGrid	✘	✔($O(Nm/w)$)
sGrid Z-score	✔	✔($O(N^2m/w)$)
SiNNE	✔	✔($O(t\psi^2 + t\psi)$)

sGrid computes the outlying score of a given data point based on a grid in which the point falls and its neighboring grids. sGrid measure is two orders of magnitude faster than kernel density estimation [15]. However, sGrid is dimensionally biased, i.e., sGrid density tends to decreases as dimensions increases. Thus, the authors used Z-score normalization on top of sGrid, which makes sGrid computationally expensive. In addition to that, Samariya et al. [9] shows that Z-normalization is biased towards subspace having high variance. Moreover, Z-score normalization adds additional computation overhead on a measure. Thus, sGrid with Z-score normalization is not effective and due to dimensionality biasedness, it can not be used directly.

Motivated by these limitations, we proposed sGrid++, a simple yet effective variant of sGrid which is dimensionally unbiased in its raw form thus it does not require any additional normalization.

The proposed method consists of two stages. In the first stage (training stage), g number of grids are generated. The second stage is the evaluation stage, which evaluates the outlyingness of a given data point $\mathbf{q}$ in each subspace. Let $\mathcal{O}$ be a M dimensional data set in $\Re^M$, and $S \subset \mathcal{S}_\mathcal{F}$ be a subspace of m, where $m = |S|$ dimensions and $m \leq M$.

For each dimension, the proposed measure first creates equal-width univariate bins. We used the Freedman-Diaconis rule [4] to set the bin width and number of bins in each dimension automatically for a given data set, which means sGrid++ creates b equal width bins overvalue range.

Definition 2 (Histogram). *A histogram is a set of b equal width bins, $H = \{B_1, B_2, \cdots, B_b\}$.*

Once a histogram is created, the proposed measure calculates the mass of each histogram.

Definition 3. *A mass is defined as the number of data points that falls into the region.*

Definition 4. *A mass of a data instance $o \in \Re^M$ with respect to B_i is estimated as follows.*

$$m\bar{a}ss\Big(B_i(o)\Big) = \begin{cases} \textbf{mass}, & \textit{where } \textbf{mass} \textit{ is the mass of bin } B_i \textit{ in which } o \textit{ falls into} \\ 0, & \textit{otherwise} \end{cases}$$

sGrid++ computes density as follows.

$$\text{sGrid++}(\mathbf{q}) = \frac{m\bar{a}ss(\mathcal{G}_S(\mathbf{q}))}{v_(\mathcal{G}_S(\mathbf{q}))} \tag{1}$$

where $\mathcal{G}_S(\mathbf{q})$ is a mass of grid (cell) in which $\mathbf{q}$ falls into and $v(\mathcal{G}_S(\mathbf{q}))$ is a volume of grid $\mathcal{G}_S(\mathbf{q})$.

Assuming that, data is normalized and in the range of $[0, 1]$. sGrid++ creates equal width histogram in each dimension thus it creates b^m grids (cells) of the same volume because they are in same range ([0, 1]). So, volume of each cell in subspace S is

$$v(\mathcal{G}_S(\mathbf{q})) = 1/b^m \tag{2}$$

Plugging Eq. 2 into Eq. 1, sGrid++ density of a query $\mathbf{q}$ is estimated as follows.

$$\text{sGrid++}(\mathbf{q}) = m\bar{a}ss(\mathcal{G}_S(\mathbf{q})) \cdot b^m \tag{3}$$

Let $\mathcal{G}_{(i,j,k)}(\mathbf{q})$ is a grid in which $\mathbf{q}$ falls into and has indices of $(i, j, k) \in \{(1, 1, 1), \cdots, (b_1, b_2, b_3)\}$ in their respective dimensions in 3-dimensional space. To estimate the final outlying score of query $\mathbf{q}$, sGrid++ uses the mass of the grid in which $\mathbf{q}$ falls into. The mass of grid in which $\mathbf{q}$ falls is computed by bit set intersection operation $\mathcal{G}_{(i,j,k)} = b_i \cap b_j \cap b_k$.

$$m\bar{a}ss(\mathcal{G}_S(\mathbf{q})) = \bigcap_{i \in m} B_i(\mathbf{q}) \tag{4}$$

The final outlying score of query $\mathbf{q}$ in subspace S is computed as:

$$grid_S(\mathbf{q}) = m\bar{a}ss(\mathcal{G}_S(\mathbf{q})) \cdot b^m \tag{5}$$

where $\mathcal{G}_S(\mathbf{q})$ is grid in which $\mathbf{q}$ falls into subspace S. $m\bar{a}ss(\mathcal{G}_S(\cdot))$ is the mass of grid $\mathcal{G}_S(\cdot)$, which is computed as shown in Eq. 4.

A working example of the proposed measure in a two-dimensional space is shown in Fig. 2. The $grid_S(x)$ is 3 while $grid_S(y)$ is 1. Thus, point y is considered more outlying.

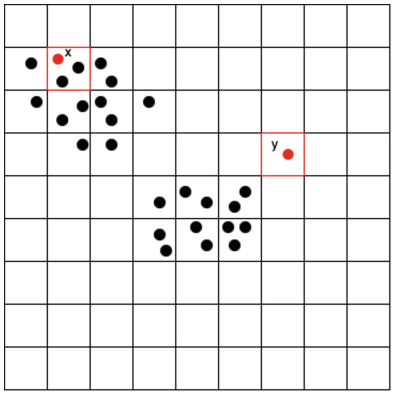


Fig. 2. Example of the new proposed method. The red highlighted regions shows the bins used to estimate outlyingness of two instances x and y. (Color figure online)

Proposition 1. *If point* $\boldsymbol{q}$ *falls within grid* $\mathcal{G}_S$ *which has higher mass then subspace* S *is not an outlying aspect of a query* $\boldsymbol{q}$.

Theorem 1. *The proposed measure* $grid_S(\boldsymbol{q})$ *is dimensionally-unbiased as per dimensionality unbiasedness definition* [14, *Definition 4*].

Proof. Given a data set $\mathcal{O}$ of N data instances drawn from a uniform distribution $\mathcal{U}([0,1]^M)$.

As data is drawn from the uniform distribution, each grid has the same mass,

$$\bar{mass}(\cdot) = \frac{N}{g}$$

where g is total number of grid ($g = b^M$).

If we substitute mass in Eq. 5, for query $\mathbf{q}$, final outlying score is,

$$grid(\mathbf{q}) = \frac{N}{g} \cdot b^M$$

$$= \frac{N}{g} \cdot g = N$$

Thus, an average value of the sGrid++ scoring measure is,

$$E[grid(\mathbf{q})|\mathbf{q} \in \mathcal{O}] = \frac{1}{N} \sum_{\mathbf{q} \in \mathcal{O}} grid(\mathbf{q})$$

$$= \frac{1}{N} \sum_{\mathbf{q} \in \mathcal{O}} N$$

$$= \frac{1}{N} N \cdot N = N, constant\ w.r.t\ |S|$$

The proposed measure is scalable to both huge datasets and high dimensions. We will prove this by our empirical evaluation.

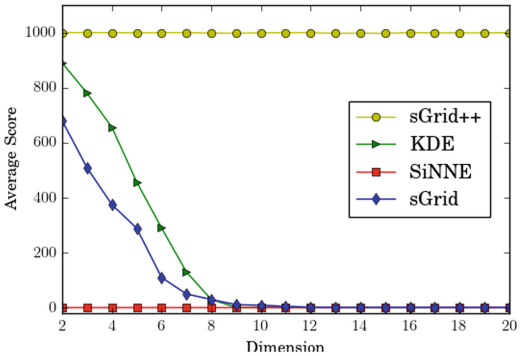


Fig. 3. Dimensionality unbiasedness.

4 Experiments

A series of experiments were performed to answer the following questions:

- **Dimensionality unbiasedness:** Does the proposed measure dimensionally unbiased?
- **Effectiveness:** How accurate is the proposed method?
- **Efficiency:** Does the proposed method scalable compared to its competitor w.r.t. data set size and dimensionality?

We first provided the experimental setup details before detailing our findings.

4.1 Experimental Setup

Algorithm Implementation and Parameters. All measures were implemented in Java using WEKA [5]. We implemented the proposed measure by making the required changes in the Java implementation of sGrid provided by the authors. We used sGrid and SiNNE Java implementations made available by [15] and [9], respectively.

Z(KDE) is performed by using a Gaussian kernel with default bandwidth. sGrid uses the default recommended parameter block size for a bit set operation w as 64. In terms of SiNNE, the sub-sample size $\psi = 8$ and ensemble size $t = 100$. sGrid++ also uses parameter block size for bit set operation $w = 64$.

All experiments were conducted in a macOS machine with a 2.3 GHz 8-core Intel Core i9 processor and 16 GB memory running on macOS Monterey 12.4. We run all jobs for 24 h and killed all uncompleted jobs.

4.2 Desired Properties

Dimensionality Unbiasedness. We generated 19 synthetic datasets, each data set contains 1000 data points from uniform distribution $\mathcal{U}([0,1]^M)$, where

Table 2. Comparison of the proposed measure and its three contenders on five synthetic datasets. **q**-id represents query index, GT represents ground truth. The numbers in a bracket are feature indices (i.e. subspaces).

	q-id	GT	sGrid++	Z(KDE)	Z(sGrid)	SiNNE
Synth_10D	172	{8, 9}	**{8, 9}**	**{8, 9}**	**{8, 9}**	**{8, 9}**
	245	{2, 3, 4, 5}	**{2, 3, 4, 5}**	**{2, 3, 4, 5}**	{3, 4, 5}	**{2, 3, 4, 5}**
	577	{2, 3, 4, 5}	**{2, 3, 4, 5}**	{6, 7}	**{2, 3, 4, 5}**	**{2, 3, 4, 5}**
Synth_20D	43	{0, 1, 2}	**{0, 1, 2}**	**{0, 1, 2}**	**{0, 1, 2}**	**{0, 1, 2}**
	86	{18, 19}	**{18, 19}**	**{18, 19}**	**{18, 19}**	**{18, 19}**
	665	{0, 1, 2}	**{0, 1, 2}**	**{0, 1, 2}**	**{0, 1, 2}**	**{0, 1, 2}**
Synth_50D	121	{21, 22, 23}	**{21, 22, 23}**	**{21, 22, 23}**	**{21, 22, 23}**	**{21, 22, 23}**
	248	{13, 14, 15}	**{13, 14, 15}**	**{13, 14, 15}**	**{13, 14, 15}**	**{13, 14, 15}**
	427	{5, 6, 7, 8}	**{5, 6, 7, 8}**	{8, 9, 48}	{48, 49}	**{5, 6, 7, 8}**
Synth_75D	69	{6, 7, 8}	**{6, 7, 8}**	**{6, 7, 8}**	**{6, 7, 8}**	**{6, 7, 8}**
	145	{0, 1}	**{0, 1}**	**{0, 1}**	**{0, 1}**	**{0, 1}**
	214	{9, 10}	**{9, 10}**	**{9, 10}**	**{9, 10}**	**{9, 10}**
Synth_100D	80	{17, 18}	**{17, 18}**	**{17, 18}**	**{17, 18}**	**{17, 18}**
	105	{10, 11}	**{10, 11}**	**{10, 11}**	**{10, 11}**	**{10, 11}**
	258	{43, 44}	**{43, 44}**	**{43, 44}**	**{43, 44}**	**{43, 44}**

M varied from 2 to 20. We computed the average score of all instances using sGrid++, SiNNE, sGrid, and KDE. The result is presented in Fig. 3. The flat line for the proposed measure and SiNNE shows that both measures are dimensionally unbiased, whereas sGrid and KDE (without Z-score normalization) are not. Note that, in [14], it is shown that using ranks and Z-score normalization, makes any score dimensionally unbiased. Hence, we did not include them in our experiment.

4.3 Mining Outlying Aspects on Synthetic Datasets

We evaluate the performance of sGrid++ and three contending scoring measures on 5 synthetic datasets[2], where number of instances (N) are 1,000 and number of dimensions (M): 10 to 100.

Table 2 summarised the discovered subspaces of three queries[3] by the contending measures on five synthetic datasets. In terms of exact matches, sGrid++ and SiNNE are the best performing measures that detect the ground truth of all 15 queries. Whereas Z(sGrid) and Z(KDE) produced exact matches for 14 queries.

[2] The synthetic datasets are from Keller et al. (2012) [6]. Available at https://www.ipd.kit.edu/~muellere/HiCS/.

[3] We reported three queries only due to page limitation.

Table 3. Comparison of the proposed measure and its three contenders on *student performance* data set. **q**-id represents query index. The numbers in the bracket are feature indices (i.e. subspaces).

q-id	sGrid++	Z(KDE)	Z(sGrid)	SiNNE
5	{1, 3, 4}	{4}	{1}	{1, 3, 4}
52	{0, 3, 4}	{0}	{0}	{0, 2, 4}
68	{0, 1}	{1}	{1}	{0, 1, 2}
156	{0, 3, 4}	{0, 4}	{0, 4}	{0, 3, 4}
197	{2, 3}	{2}	{2}	{1, 2, 3}

4.4 Mining Outlying Aspects on Student Performance Data Set

While evaluating job applications, the recruiting team wants to know in which aspects an applicant is different than other applicants. Considering this example as a case study, we detect outlying aspects of all top k outlier/anomaly students[4]. We used campus placement data set[5] which has 15 features, and we removed non-numerical features and all data points with a missing value.

For each query **q**, we apply the Beam search strategy with four different scoring measures. Table 3 summarizes the outlying aspects found by four different scoring measures on the student data set.

In absence of better quality measures for outlying aspects, we visually present 3 queries in Table 4. Visually, we can say that the sGrid++ and SiNNE detects better subspaces than Z-score based scoring measures – Z(KDE) and Z(sGrid).

4.5 Mining Outlying Aspects on NBA2020 Data Set

Let's assume that, you are an NBA coach, commentator, or agent and you may want to know the strengths or weaknesses of a particular player, using the OAM application one can detect and find that easily. We mine data from Foxsports[6] to prepare technical statistics on *shooting*, *assists* and *defence* stats of NBA 2020.

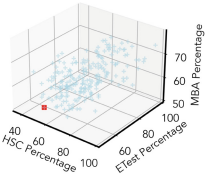
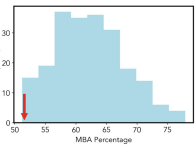
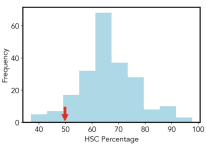
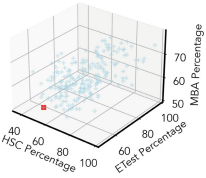
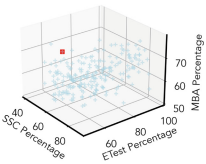
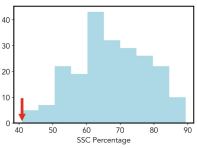
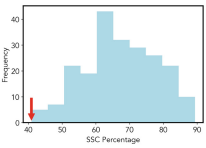
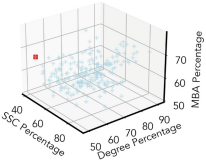
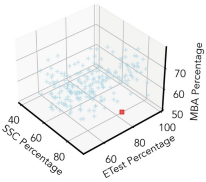
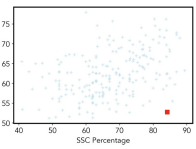
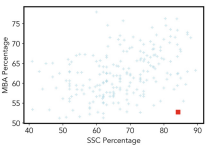
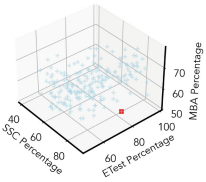
Table 5 summarizes the outlying aspects found by sGrid++ and three contending measures on 3 NBA datasets – *assists*, *defence* and *shooting*. We visually present the results of one query from each data set in Table 6. Visually we can say that out of three sGrid++ detects better subspaces whereas Z-score based measures Z(KDE) and Z(sGrid) are unable to detect best subspaces. SiNNE also detects better subspace as sGrid++. However, SiNNE is slower than sGrid++.

[4] We used a state-of-the-art anomaly detection algorithm called LOF [1] to identify top $k = 5$ anomalies; and used them as queries.

[5] Available at https://www.kaggle.com/benroshan/factors-affecting-campus-placement.

[6] https://www.foxsports.com/nba/stats.

Table 4. Visualization of discovered subspaces by sGrid++, Z(KDE), Z(sGrid) and SiNNE in the *student performance* data set.

q-id	sGrid++	Z(KDE)	Z(sGrid)	SiNNE
5				
52				
156				

4.6 Scale-up Test

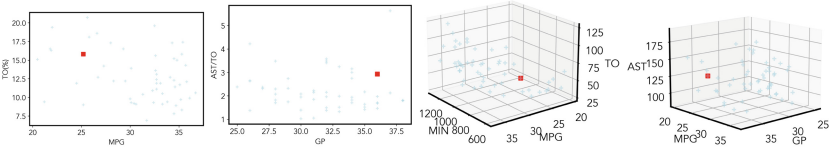
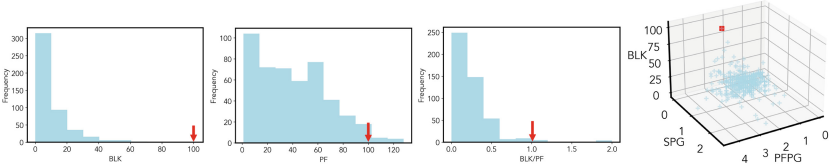
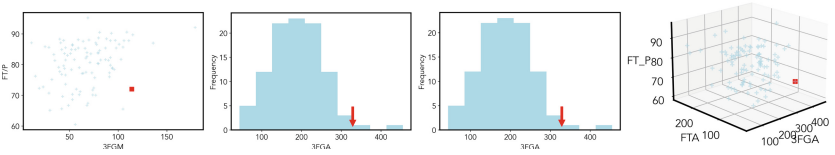
We conducted a scale-up test of these four measures w.r.t. (i) increasing data sizes (N) and (ii) increasing dimensionality (M), using synthetic datasets. We generated three equal-sized Gaussian's with random mean μ = [-10, 10] and variance $\sigma = 1.0$ in each dimension using Python Scikit-Learn [7] library. For each data set, we randomly pick 10 data points as queries and presented the average runtime. Note that, for a fair comparison we set the maximum dimensionality of subspace (ℓ) = 3.

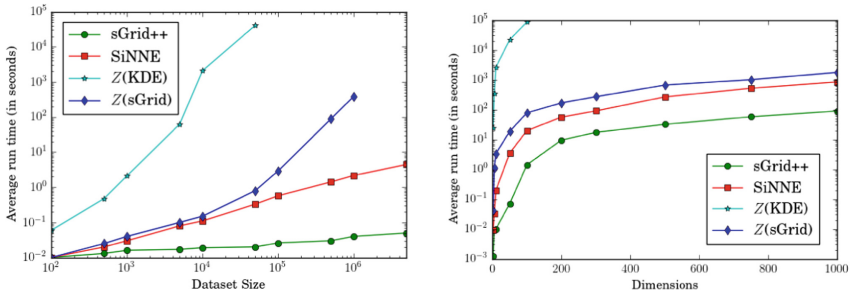
Increasing Data Size. In this scale-up test, we examined the efficiency of the contending scoring measures w.r.t. the number of data sizes (N). A wide range of N values from 100 to 5 million is used and dimension M is fixed to 5. Figure 4a shows the average runtime on 10 queries of the contending measures w.r.t. increasing data set sizes. Note that the runtime and data set size is plotted using a logarithmic scale. sGrid++ and SiNNE are the only measures to finish scale-up test for each data set. However, SiNNE is order of magnitude slower than sGrid++. Z(sGrid) is unable to finish for data set having 5 million data points in 24hrs, whereas Z(Beam) is able to finish upto data set having 50 thousand data points. Overall, sGrid++ is order of magnitude faster than SiNNE, two orders of magnitude faster than Z(sGrid) and four orders of magnitude faster than Z(KDE).

Table 5. Comparison of sGrid++ and its three contenders on *NBA 2020* technical statistics. **q**-id represents query index. The numbers in the bracket are feature indices (i.e. subspaces).

	q-id	sGrid++	Z(KDE)	Z(sGrid)	SiNNE
assists	19	{0,1}	{3, 6}	{0, 7, 8}	{1, 6, 7}
	35	{2, 7}	{0, 8}	{1, 2, 5}	{0, 2, 3}
	52	{4}	{4}	{4}	{0, 3, 9}
defence	51	{2, 3, 4}	{4}	{4}	{4, 5, 12}
	131	{11}	{4}	{13}	{5, 7, 11}
	339	{0, 2, 5}	{0, 1, 5}	{2, 5}	{2, 4, 6}
shooting	4	{11, 21}	{6}	{11, 21}	{4, 6, 11}
	34	{5, 18}	{1}	{1}	{1, 6, 14}
	96	{10, 18}	{12}	{12}	{12, 17, 18}

Table 6. Visualization of discovered subspaces by sGrid++, Z(KDE), Z(sGrid) and SiNNE in the *NBA2020* data set.

q-id	sGrid++	Z(KDE)	Z(sGrid)	SiNNE
assists (**q**-id : 35)				
defence (**q**-id : 131)				
shooting (**q**-id : 96)				



(a) Data size(N). (b) Dimensionality (M).

Fig. 4. Scale-up test.

Increasing Dimensionality. In this scale-up test, we examined the efficiency of the contending measures w.r.t. the increasing number of dimensions. A wide range of M values from 2 to 1000 and a data set size of 10,000 was used. Figure 4b shows the average runtime on 10 queries of the contending measure w.r.t. increasing data set dimensions. Except Z(KDE) all other measures are able to finish scale-up test for each data set with in 24 h. sGrid++ is the fastest measure compare to its contenders followed by SiNNE and Z(sGrid). sGrid++ is order of magnitude faster than SiNNE and Z(sGrid).

5 Conclusion

In this paper, we discussed the issue of existing scoring measures, specifically the existing density estimator sGrid. We proposed a simple yet effective solution for making existing dimensionally biased measure to unbiased. sGrid++ creates univariate histograms in each dimension of subspace. Afterwards, the mass of the grid in which a given data point falls is used to compute the outlyingness of the query in that subspace. Our extensive experiments shows that the proposed scoring measure is dimensionally unbiased and is the fastest measure compared to all its competitors.

Acknowledgments. This work is supported by Federation University Research Priority Area (RPA) scholarship, awarded to Durgesh Samariya. Dr Sunil Aryal is supported by an Air Force Office of Scientific Research (AFOSR) research grant under award number FA2386-20-1-4005.

References

1. Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: LOF: identifying density-based local outliers. In: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, SIGMOD 2000, pp. 93–104. Association for Computing Machinery, New York (2000). https://doi.org/10.1145/342009.335388

2. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: a survey. ACM Comput. Surv. **41**(3), 1–58 (2009). https://doi.org/10.1145/1541880.1541882
3. Duan, L., Tang, G., Pei, J., Bailey, J., Campbell, A., Tang, C.: Mining outlying aspects on numeric data. Data Min. Knowl. Disc. **29**(5), 1116–1151 (2015). https://doi.org/10.1007/s10618-014-0398-2
4. Freedman, D., Diaconis, P.: On the histogram as a density estimator: L 2 theory. Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete **57**(4), 453–476 (1981)
5. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: An update. SIGKDD Explor. Newsl. **11**(1), 10–18 (2009). https://doi.org/10.1145/1656274.1656278
6. Keller, F., Muller, E., Bohm, K.: HiCS: high contrast subspaces for density-based outlier ranking. In: 2012 IEEE 28th International Conference on Data Engineering, pp. 1037–1048 (2012). https://doi.org/10.1109/ICDE.2012.88
7. Pedregosa, F., et al.: Scikit-learn: machine learning in python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
8. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach, 3rd edn. Prentice Hall Press, Upper Saddle River (2009)
9. Samariya, D., Aryal, S., Ting, K.M., Ma, J.: A new effective and efficient measure for outlying aspect mining. In: Huang, Z., Beek, W., Wang, H., Zhou, R., Zhang, Y. (eds.) WISE 2020. LNCS, vol. 12343, pp. 463–474. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-62008-0_32
10. Samariya, D., Ma, J.: Mining outlying aspects on healthcare data. In: Siuly, S., Wang, H., Chen, L., Guo, Y., Xing, C. (eds.) HIS 2021. LNCS, vol. 13079, pp. 160–170. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-90885-0_15
11. Samariya, D., Ma, J.: A new dimensionality-unbiased score for efficient and effective outlying aspect mining. Data Sci. Eng. **7**, 1–16 (2022). https://doi.org/10.1007/s41019-022-00185-5
12. Samariya, D., Ma, J., Aryal, S.: A comprehensive survey on outlying aspect mining methods. arXiv preprint arXiv:2005.02637 (2020)
13. Samariya, D., Thakkar, A.: A comprehensive survey of anomaly detection algorithms. Ann. Data Sci. 1–22 (2021). https://doi.org/10.1007/s40745-021-00362-9
14. Vinh, N.X., et al.: Discovering outlying aspects in large datasets. Data Min. Knowl. Disc. **30**(6), 1520–1555 (2016). https://doi.org/10.1007/s10618-016-0453-2
15. Wells, J.R., Ting, K.M.: A new simple and efficient density estimator that enables fast systematic search. Pattern Recogn. Lett. **122**, 92–98 (2019). https://doi.org/10.1016/j.patrec.2018.12.020, http://www.sciencedirect.com/science/article/pii/S0167865518309371