Unsupervised Class Separation of Multivariate Data through Cumulative Variance-based Ranking

Andrew Foss
Department of Computing Science
University of Alberta
Edmonton, Canada
afoss@cs.ualberta.ca

Osmar R. Zaïane
Department of Computing Science
University of Alberta
Edmonton, Canada
zaiane@cs.ualberta.ca

Sandra Zilles
Department of Computer Science
University of Regina
Regina, Canada
zilles@cs.uregina.ca

***Abstract*—This paper introduces a new extension of outlier detection approaches and a new concept, class separation through variance. We show that accumulating information about the outlierness of points in multiple subspaces leads to a ranking in which classes with differing variance naturally tend to separate. Exploiting this leads to a highly effective and efficient unsupervised class separation approach, especially useful in the difficult case of heavily overlapping distributions. Unlike typical outlier detection algorithms, this method can be applied beyond the 'rare classes' case with great success. Two novel algorithms that implement this approach are provided. Additionally, experiments show that the novel methods typically outperform other state-of-the-art outlier detection methods on high dimensional data such as Feature Bagging, SOE1, LOF, ORCA and Robust Mahalanobis Distance and competes even with the leading supervised classification methods.**

***Keywords*-Outlier Detection; Classification; Subspaces.**

I. Introduction

A common problem in many data mining and machine learning applications is, given a dataset, to identify data points that show significant anomalies compared to the majority of the points in the dataset. These points may be noisy data, which one would like to remove from the dataset, or may contain information that is particularly valuable for the identification of patterns in the data.

The domain of outlier detection [1], [2] deals with the problem of finding such anomalous data, called outliers. Outlier detection can be viewed as a special case of unsupervised binary class separation in the case of 'rare classes'. The dataset is separated into a large class of 'normal cases' and a small class of 'rare cases' or 'outliers'.

Outlier detection is particularly problematic as the dimensionality d of the given dataset increases. Problems are often due to sparsity or due to the fact that distance-based approaches fail because the relative distance between any pair of points tends to become relatively the same, see [3].

One idea to overcome such problems is to *rank* outliers in the high-dimensional space according to how consistently they are outliers in low-dimensional subspaces, in which outlierness is easier to assess. To this end, Lazarevic and Kumar developed Feature Bagging [4] and He et al. SOE1 [5], both taking an ensemble approach combining the outlier results over subspaces. SOE1 is remarkably simple, summing the local densities of each point for each individual attribute. While SOE1 looks only at 1-dimensional subspaces (i.e., at single attributes), Feature Bagging combines the results of the well-known LOF outlier detection method [6] applied to random subspaces with $d/2$ or more attributes (out of a total of d attributes). However, both methods have weaknesses.

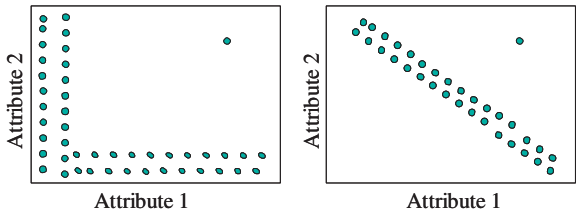

Figure 1. A point that is an outlier in a 2-dimensional space but not in any of the two corresponding 1-dimensional spaces.

SOE1. Looking only at 1-dimensional subspaces is very efficient but not always effective. It is simple to give examples showing when this may lead to missed information; Figure 1 illustrates such a case. The point in the upper right corner is not a clear outlier with respect to either attribute 1 or attribute 2, but is obviously an outlier in the two-dimensional space. In two dimensions this is obvious, but the likelihood of such a scenario arising and being significant clearly declines as the subspace size increases — due to the phenomenon explored in [3].

Feature Bagging. Once the LOF algorithm, applied in the subspaces, becomes less effective (as $d/2$ rises beyond the dimensionality barrier shown by [3], see Section II), bagging will become less effective, too.

Motivated by that, we propose a method that employs a stable outlier detection algorithm for subspaces of a fixed low dimensionality k, $1 < k << d$, and combines the results of that algorithm over all k-dimensional subspaces to provide an outlier ranking in d dimensions.

This results in two major and novel contributions.

Contribution to Outlier Detection: We provide a new framework for outlier detection in an arbitrary number of dimensions, based on rankings obtained by investigating low-dimensional subspaces (as opposed to Feature Bagging) that consist of more than one attribute (as opposed to SOE1). Experiments show that our method is superior not only to SOE1 and Feature Bagging, but also to state-of-the-art outlier detection algorithms designed for multi-variate data. Those algorithms are the two distance-based methods *ORCA* [7] and *Robust Mahalanobis Distance (RMD)* [8], and the density-based *LOF*-method [6].

Contribution to Class Separation: Our outlier ranking can be applied to unsupervised class separation even when neither one of the classes is 'rare' — with marked success. Class separation by means of an outlier score has, incidentally, been used indirectly for the sole purpose of validating outlier detection approaches [4], [5]. For lack of ground truth, unbalanced binary class training data were used with the assumption that the rare class points are outliers vis-à-vis the dominant class [9]. In this work, we not only separate balanced as well as unbalanced classes of high dimensional data but also elucidate the phenomenon that allows this class separation.

The problems we address here are binary class separation problems in which the two classes are assumed to differ in *variance*. We argue that, for two underlying classes A and B of different variance, our outlier ranking basically separates all points in A from all points in B, *even if the two classes overlap completely and are of the same size*. As we will explain below, this is because points in the class of high variance are more likely to be outliers consistently in many low-dimensional subspaces and are thus ranked higher in the resulting outlier ranking.

We test this experimentally with positive results: not only is our method, applied to the case of balanced classes, frequently superior to other outlier detection methods, but it can even compete with supervised classification methods, which are fed with labelled training data.

II. Related Work

Only few existing methods can cope with the problem of outlier detection in high-dimensional data, due to sparsity. The effectiveness of most common methods declines because they rely on distances between points, something that becomes less meaningful in high dimensionality, because the distance between any two points tends to become relatively the same, cf. [3]. Real-world datasets frequently have large numbers of attributes so this poses a significant problem especially because approximation schemes in general and tree indices in particular tend to break down with more than 10-15 dimensions [3]. Beyond this dimensionality barrier, algorithms that work in the full dimensional space face considerable challenges in both efficiency and effectiveness.

This is the primary motivation for using information about outliers in *lower-dimensional subspaces* of the full high-dimensional space in order to determine which points are outliers in the full space.

While the literature on anomaly detection is vast, very few methods aim at investigating subspaces in high-dimensional data. Aggarwal and Yu [10] developed an evolutionary search algorithm to find low density subspaces, though the predominance of such spaces poses challenges, cf. [11]. Knorr and Ng [12] computed a dendogram to show the intensional knowledge in the hierarchy of subspaces in which a point is an outlier. Zhang and Wang [11] proposed HighDOD, a method that uses a sample based learning process based on the sum of the distances to the k nearest neighbours (k-NN) to identify the subspaces in which a given point is an outlier.

SOE1 [5] and Feature Bagging [4] represent the current state-of-the art as far as ranking points according to their outlierness using subspaces is concerned.[1]

Validating outlier detection methods was largely lacking until the idea of using rare classes as outliers in unbalanced supervised classification training data was introduced in [9]. Since then, others have used the separation of a rare class from a dominant class as a means to validate an outlier detection approach. However, the stated objective was never class separation per-se and the data used was typically heavily unbalanced for the exact purpose of validating outliers. Exploiting outliers for genuine class separation was never intended or explained.

III. The T* Framework

We assume a *dataset* $D \subset \mathbb{R}^d$ in d dimensions. Each of the dimensions represents a different *attribute*. For every point $x \in D$ and every $i \in \{1, \ldots, d\}$ we denote by x_i the value in the i^{th} attribute of x, i.e., $x = (x_1, \ldots, x_d)$. If $k \leq d$ and $S = \{i_1, \ldots, i_k\} \subseteq \{1, \ldots, d\}$, then D_S denotes the set $\{(x_{i_1}, \ldots, x_{i_k}) \mid x \in D\}$, i.e., the projection of D on the attributes indexed by S.

For every finite set Z, let $|Z|$ denote the cardinality of Z.

A. Algorithmic Idea

In this section we assume an efficient algorithm *Outlier*, which, given a subspace D_S for a 'small' set S of attributes (in our experiments, $|S| = 2$ is sufficient) and a point $x \in D_S$, determines a degree of 'outlierness' of x in D_S. Let

$$\mathrm{out}(x, D_S)$$

denote this degree.

We will describe two such algorithms in detail in Appendix A; however, the specific *Outlier* algorithm used for this purpose is not critical for our outlier detection approach

[1]The work of Knorr and Ng [12] could be developed further to rank outliers, but they have not pursued this direction so far.

in high dimensionality — other heuristics could be applied in the same way.

Note that we have not defined what an 'outlier' in D_S is, since there is no unique commonly accepted definition of that term. Different versions of the algorithm *Outlier* correspond to different interpretations of the term 'outlier in D_S'.

Our algorithmic approach to outlier detection in D can be sketched as follows. Intuitively, our algorithm accumulates an outlier score for every data point, by counting how often and to which degree it is an 'outlier' in a set of all subspaces of a low dimension k. The top-ranked points are considered outliers.

1) Compute an outlier score for every point in D.
   a) Set the parameter $k << d$ (size of subspaces).
   b) Let $S_1 \subseteq \{1, \ldots, d\}$, ..., $S_z \subseteq \{1, \ldots, d\}$ be all subsets of $\{1, \ldots, d\}$ of size k, i.e., $|S_i| = k$ for all $i \in \{1, \ldots, z\}$.
   c) For every point $x \in D$, compute an outlier score
   $$\mathrm{score}(x) = \sum_{1 \leq i \leq z} \mathrm{out}(x, D_{S_i})$$
   Note here that we assume the values of out to be calibrated over the different subspaces, such that a high out value in one subspace cannot become predominant over those obtained in other subspaces.
2) Rank the points in D with respect to their outlier scores.
   a) Compute a ranking $x^1, \ldots x^{|D|}$ over the points in D such that the values of $\mathrm{score}(x^i)$ are non-increasing when i is increasing.
   b) If queried for the top N outliers in D, return $x^1, \ldots, x^N$.

The crucial question in this approach is over which subspaces we should accumulate the outlier scores, i.e., how to choose the dimensionality k. On the one hand, taking all subspaces of a relatively high dimension k would be intractable. On the other hand, as argued in Section I, it is not advisable to accumulate only over subspaces of dimension 1 and when d is large, so remains $d/2$.

As the experiments described in Section IV demonstrate, a value of $k = 2$ is sufficient to outperform standard outlier detection methods on typical datasets, while at the same time resulting in a very efficient and effective method.

B. A Note on the Dimensionality d of the Original Dataset

It is important to note here that a high enough dimensionality d of the original dataset is *essential* for our method to work as discussed in Section III-D.

Note that none of the closely related work on outlier detection attempts genomic or proteomic data. Outlier detection (classically) seeks to find individual anomalies or rare classes in the presence of major classes or distributions. Genomic and proteomic data typically have a very small number of interesting features and no large coherent class(es). Thus such data has not been the focus of research on outlier detection so far and therefore we have also not studied this kind of data. The datasets we use for empirical evaluation below have between 20 and 166 attributes. Datasets over 10-15 dimensions can be considered 'high' due to the onset of high-dimensional effects [3].

C. Application to Unsupervised Class Separation

T* can be used for unsupervised (binary) class separation even beyond the 'rare classes' case immediately, if the given dataset fulfills the following conditions.

- The data are of high enough dimensionality for T* to produce a discriminative ranking of the data points.
- The given data separate in two classes, A and B.[2]
- Class B is of higher variance than class A.

Under the given assumptions, we propose the following straightforward unsupervised class separation method.

1) Run T* on the given dataset D to obtain a ranking $x^1, \ldots, x^{|D|}$ of the points in D.
2) For every $t \in \{1, \ldots, |D|\}$ assign a likelihood of the point x^t being in B in a way such that this likelihood decreases as t increases. Alternatively, if the class ratio $|A|/|B|$ is known, label the points $x^1, \ldots, x^{|B|}$ with B and the remaining points with A.

The reason this method is expected to work well is based on the same intuition that our outlier detection method is based on:

> *If a point is contained in a class of high variance, then this point is likely to be an 'outlier' in many low-dimensional subspaces, and vice versa.*

Let us explain this intuition in more detail, assuming a density-based *Outlier* method applied in 2-dimensional subspaces (the argument easily carries over to other cases).

First, assume a point x is ranked high in the outlier ranking returned by T*. Then, for 'many' 2-dimensional subspaces $D_{\{i,j\}}$ of D, x is assigned a high outlier degree in $D_{\{i,j\}}$ by the corresponding *Outlier* method.

This means that, for 'many' 2-dimensional subspaces $D_{\{i,j\}}$ of D, x is isolated (i.e., an outlier). Since the class A has low variance, points in A are expected to be not isolated in 'most' 2-dimensional subspaces. Hence x is more likely to belong to B than to belong to A. Consequently, if a point has a score value above a certain threshold, this point is most likely to belong to B.

Second, assume a point x is ranked low in the outlier ranking returned by T*. Then, for 'most' 2-dimensional subspaces $D_{\{i,j\}}$ of D, x is *not* considered an outlier in $D_{\{i,j\}}$ by the corresponding *Outlier* method. This means

[2]We assume underlying true populations $\mathcal{A}$ and $\mathcal{B}$ and denote by A and B the respective fixed sets of datapoints contained in the given dataset D.

that, for 'most' 2-dimensional subspaces $D_{\{i,j\}}$ of D, x is in a dense region. Since the class B has high variance, points in B are expected to be isolated in 'many' 2-dimensional subspaces. Hence x is more likely to belong to A than to belong to B. Consequently, if a point has a score value below a certain threshold, this point is more likely to belong to A.

In particular, if the difference between the variance of A and the variance of B is high enough, and if d is high enough to provide a sufficient number of 2-dimensional subspaces, we expect thus to separate basically *all points in* B from *all points in* A — just because basically all points in B will have higher outlier scores than any point in A.

Since this observation is totally independent of the mean μ around which the points in a class are distributed, our algorithm works well even if two classes overlap completely — as long as the two classes differ significantly in variance.

Figure 2 illustrates the general phenomenon. In Example (i), there are two equal variance classes and this results in their members being well mixed in the outlier score ranking. However, in Example (ii), the variances of the two classes are different and there is a tendency for the higher variance class c_v to populate the higher outlier score rankings. If the classes are heavily overlapping, in any given subspace, only a few members of c_v will *visually* appear as outliers. However, as their underlying variance is higher, accumulating over multiple subspaces, eventually almost all can differentiate themselves from the lower variance class.

In fact, for many real-world datasets it is the case that they contain two classes that differ significantly in variance. For instance, when considering medical data, it is often the case that the class representing healthy cases has lower variance than the class representing unhealthy cases. To date, outlier detection has been understood as beneficial in separating two such classes in the case of an extreme class imbalance, i.e., if the 'healthy' class is predominant in the sense that it contains many more data points than the 'unhealthy' class. However, our outlier detection method allows for unsupervised class separation even in the case of perfectly balanced classes, as long as the classes differ in variance.

D. Concentration of Measure

The basis of this intuition lies in the law of large numbers and its extension by V. Milman [13] which he termed the concentration of measure. Milman showed that this is applicable to a large class of functions (Lipschitz) that are smooth and have a finite mean. A measure over an ensemble size m of independent variables strongly tends to the mean as $m \to \infty$. A simple example is coin tosses. If a coin of unknown bias p is thrown m times giving the results $\{X_0, X_1, ...\}$, then

$$\forall \epsilon > 0, P\left(\left|\frac{\sum_i^m Xi}{m} - p\right| \geq \epsilon\right) \leq 2e^{-2\epsilon^2 m} \qquad (1)$$

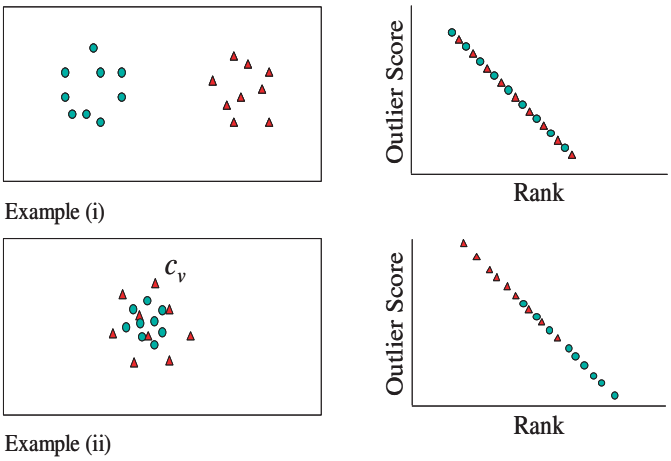

Figure 2. In Example (i) two classes with equal variance show no separation in the outlier ranking. In Example (ii), the difference in variance leads to a degree of separation.

For example, if a balanced coin is thrown once there is complete uncertainty regarding the outcome. If the coin is thrown 1000 times, the number of heads will, in all probability, be rather close to 500. The larger the number of tosses, the more predictable the outcome. Similarly, high-dimensional vectors radiating from a distribution mean will tend to be found concentrating at a mean length which is a function of the variance of the distribution. This is the basis of separating concentric classes that are randomly generated with different variances. This phenomenon is also the cause of the phenomenon of converging maximum and minimum interpoint distances with increasing dimensionality elucidated by Beyer et al. [3].

Thus, concentration of measure, while making points increasingly similar and thus undermining the concept of outlierness, also improves our ability to estimate certain properties of the underlying distributions, as we demonstrate for variance. Equation 1 shows the bound tightens exponentially with m and Beyer et al. showed empirically that the effect is important for greater than 10-15 dimensions. An ensemble of completely enumerated subspaces size k over d independent dimensions will contain at least $d - k + 1$ independent sets of subspaces. Thus datasets with more than 15 dimensions may exhibit this phenomenon when an ensemble approach such as that described in this paper ($k = 2$) is applied.

IV. Experimental Results

In the framework of T*, many different heuristics for determining outliers in low dimensionality could be plugged in. We propose two novel and effective methods. One based on entropy, T*ENT, and the other based on the Resolution-based Outlier Factor (ROF) [14], called T*ROF. Both these original algorithms are presented in Appendix A. Since Feature Bagging uses LOF as a basis for determining outlier

scores in subspaces, we also experimented with using LOF in the T* framework (T*LOF). While the Feature Bagging method of [4] uses a relatively small sample of high dimensional spaces, T*LOF enumerates all the 2D spaces using LOF. These three are compared with Feature Bagging, SOE1, Robust Mahalanobis Distance (RMD), ORCA, and full-dimensional LOF.

As it is difficult to find data with any ground truth ranking of outliers, we consider two types of datasets for evaluation.

Type 1. The data are ranked and we expect a relationship between this ranking and outlierness. In this case, the accuracy of T*is measured by a correlation with the provided rank. As type 1 datasets we used three National Hockey League (NHL) datasets [15], which are popular validation sets in outlier detection research. In professional sports data, the bulk of the players are often hard to differentiate from each other but the top players tend to stand out on many attributes. Players could potentially be outliers for many reasons, not just because they are top players in the league. However, due to the fact that many attributes (e.g., number of goals scored or number of assists) have many players with 'bad' values, we expect low-ranked players to be in a fairly dense area and thus unlikely to be outliers. For illustration, consider Figure 3 (Appendix A). More isolated points can be found in the upper right corner, where both attributes have their higher and 'better' values. It is reasonable, therefore, to expect a stronger correlation of an outlier ranking for the leading players than for the less successful players.

Type 2. The data are labelled in two classes A and B and we expect B to contain outliers more often than A. In this case, the accuracy of the outlier algorithms is measured by testing how many of the topmost ranked outliers are in B, using the actual number of points in B as a cutoff. We also measure area under the ROC curve (AUC). Type 2 data have been exploited this way for validating outlier detection methods before [4], [10] using 'rare' classes. In our case a range of UCI datasets [16] are used; they all have higher dimensionality and have data of two classes that are likely to differ in variance. For example, medical data are often of type 2.

A. Evaluation on Type 1 Data.

The NHL datasets for seasons 2003/04, 2005/06, 2006/07 (there was no 04/05 season) each provide an official rank and values in 15 or 16 attributes for about 1000 hockey players. The attributes vary from having fairly continuous values to having no more than four possible values.

The algorithms are evaluated under the assumption that top players are frequently outliers. Table I shows the results for the three most successful algorithms. Every 'Cor.' value is the Pearson Correlation between the outlier score and the NHL ranking. For 2006/07 the top seven outliers determined by T*ENT method contain the players with NHL ranks 3,4,5,6,8 (2,3,4,5,6,7,9,10 for 2005/06 and 1,2,4,6 for 2003/04) and none with rank lower than 60 out of about 1000 players. T*ROF also put high ranking players at the top of the outlier scores. SOE1 had less obvious success with the leading players. (None of the other algorithms yielded a statistically significant correlation so the ranking appears quite random.) To further illustrate the results, it is also shown for each algorithm, down to which depth in the resulting ranking one has to look in order to find any 5 out of the top 10 NHL-ranked players.

Table I
CORRELATION COEFFICIENTS (ABBREVIATED BY COR.; ALL VALUES SIGNIFICANT AT $p < 0.0005$) AND TOP 10 OUTLIER PLAYERS (IN ORDER) BY NHL RANKING NUMBER. PLAYERS THAT ACTUALLY HAVE ONE OF THE TOP 10 NHL RANKING NUMBERS ARE HIGHLIGHTED IN BOLDFACE. '5/10' DENOTES THE RANK AT WHICH 5 OUT OF THE TOP 10 (BY NHL RANK) PLAYERS ARE COVERED (LOWER NUMBERS ARE BETTER).

	2003-2004		
Method	Cor.	Top Outliers	5/10
T*ENT	0.852	**1**,**2**,41,57,**6**,21,19,37,**4**,16	14
T*ROF	0.686	16,**2**,41,**6**,18,**5**,12,33,**3,4**	**10**
SOE1	0.843	37,91,34,90,41,131,7,28,155,**1**	47
	2005-2006		
Method	Cor.	Top Outliers	5/10
T*ENT	0.851	**2**,**3**,**4**,11,16,**6**,**10,7,5,9**	**7**
T*ROF	0.690	15,**9**,**6**,**7**,83,**5**,20,**3,10,4**	8
SOE1	0.858	11,15,16,59,27,72,165,35,**9**,51	33
	2006-2007		
Method	Cor.	Top Outliers	5/10
T*ENT	0.858	60,**4**,**8**,**6**,**5**,**3**,14,11,17,24	**6**
T*ROF	0.706	**6**,**4**,**5**,34,**9**,47,48,**3,7**,27	8
SOE1	0.864	**5**,60,90,77,105,53,**3**,17,55,24	44

This type of data drew a clear line between two groups of algorithms as the other comparison algorithms performed poorly on or could not handle this data. RMD could not complete the NHL data, ORCA did not achieve a significant correlation (e.g. 2003/04: $r = -0.071$), and LOF, as published, and thus Feature Bagging, are not suitable for certain attributes of the NHL dataset due to many identical data points and gave non-significant results when the task was attempted.

B. Evaluation on Type 2 Data.

The algorithms were tested on all UCI Repository [16] binary problem datasets, with 20 or more attributes, which they would all reasonably be expected to complete. This meant requiring non-categorical data without missing values or a very large number of attributes (≥ 500). While these limitations could be reduced by modifying certain of the algorithms, they serve as a useful set of conditions to define a group of datasets that is both broad in scope and free of

any question of experimenter bias. Out of the 12 datasets that meet these criteria, two were left out each being one of a pair of datasets based on the same data. For example, the SPECTF data with the highest number of attributes was chosen. The 'Hill-Valley' dataset was omitted since the task should not be amenable to this approach — whether a sequence has a peak or a trough does not imply any likely difference in variance between the classes. On the other datasets, it is reasonably possible that the two classes may have different variances due to some meaningful tendency in the data. The statistics for the datasets are given in Table II.

Table II
CLASS SIZES, DATASET SIZE AND NUMBER OF ATTRIBUTES.

	$\|D\|$	d	Target class	Other class
Ionosphere	351	32	126	225
SPECTF	267	44	212	55
Sonar	208	60	111	97
WDBC	569	30	212	357
Parkinsons	195	22	148	48
Spambase	4601	57	1813	2788
Creditcard	1000	24	700	300
Musk	476	166	207	269
Insurance	5822	85	348	5474

Table II gives an overview of these datasets showing that they represent various balances including a preponderance of the class B expected to be more often contributing outliers. Hardly any would meet the usual 'rare' class criterion. For example, the WDBC set consists of 569 samples labelled in two classes, 212 malignant and 357 benign with 30 attributes for each sample. In this dataset as in many medical scenarios, normal biopsies tend to have quite strongly clustered results while abnormal ones exhibit a higher variance.

The results on the UCI datasets are summarised in Tables III and IV. Our results show that overall T*ROF and T*ENT were the most effective though RMD and SOE1 performed well. However, RMD is hampered by high run time and SOE1 by parameter sensitivity. Being parameter free, T*ROF has a clear edge as an effective and user friendly approach. Since T*ENT gives a binary score for each subspace it does best if $\binom{d}{2} \geq |D|$ (the highest possible score is not less than the number of points).

Table V gives an indication of how the run-time of the various algorithms scales with dataset size and dimensionality. Results for T*LOF are given but the implementation used, while effective, was inefficient for low dimensional spaces and thus these run times could be significantly improved. In general, the scaling is in line with the theoretical run times suggested by the authors. Purely for the purpose of this scaling experiment, the WDBC dataset was used as a seed to create datasets 10, 50 and 100 times the size of the actual WDBC set. This was done by creating multiple copies of each point and adding a random amount of jitter to each, not exceeding 1%, thus preserving the core characteristics of the dataset. Similarly, to test scaling with respect to the number of attributes, the WDBC data was used as a seed with up to 10% jitter to minimize correlation effects.

Table V
RUN TIME (IN SECONDS): SCALING BY DATASET SIZE FOR VARIOUS ALGORITHMS (30 ATTRIBUTES) AND SCALING BY NUMBER OF ATTRIBUTES (569 DATA POINTS). †EXCEEDED TIME LIMIT.

Algorithm	Dataset Size				# Attributes			
	569	5,690	28450	56,900	30	60	90	120
T*ENT	43.4	317.4	821.0	1782.2	43.4	174.8	405.0	704.5
T*ROF	43.6	321.0	828.4	1798.3	43.6	178.3	411.5	717.0
SOE1	0.0	0.1	0.7	1.5	0.0	0.0	0.0	0.1
FBAG	20.4	1620.5	†	†	20.4	30.2	42.9	53.8
LOF	0.5	44.0	1309.7	4331.9	0.5	0.8	1.2	1.5
ORCA	0.6	0.8	1.0	1.2	0.6	0.8	1.2	1.2
RMD	188.5	2225.1	†	†	188.5	729.9	1619.1	2635.2
T*LOF	77.7	5907.9	†	†	77.7	393.7	1178.4	1990.8

An extensive, but not necessarily exhaustive, literature search for the best results on these datasets using supervised classifiers yielded the following:

Ionosphere: Wang [17] using a range of k-NN classifiers and 10-fold cross-validation (MCV) achieved an accuracy of 87.8%. A Support Vector Machine (SVM) ensemble tested with 10-fold MCV achieved 95.20% [18].

SPECTF: Previously, the CLIP3 algorithm was used to generate classification rules from these patterns that were 81.34% accurate (as compared with cardiologists' diagnoses) [19]. Ali et al. [20] report that the best Linear Dimensionality (LD) reduction Quadratic classifier had an error of 4.44% while the LD Linear classifier studied had an error rate of 17.64%.

Sonar: The Sonar dataset is known to be difficult to separate. Harmeling et al. [21] tested a wide variety of supervised methods including Gaussian mixtures, Support Vector Data Description (SVDD), Parzen, a k-means based approach, and several k-NN methods. Their concept was using various local density based measures to rank the points for their degree of being typical as an outlier method does. Results varied from AUC 0.596 (SVDD) to 0.870 (new γkNN method).

WDBC: Jiang and Zhou used a neural network to edit the training data for kNN classifiers [22]. With a minimum of 250 points used as training data, 100% separation of the classes was been achieved. With 200 training points they achieved 96% accuracy. Ali et al. [20] report that the best Linear Dimensionality (LD) reduction Quadratic classifier had an error of 4.02% while the LD Linear classifier studied had an error rate of 2.99%.

Parkinsons: No comparable supervised classification results were located in our literature search.

Spambase: Neural Expert networks have been reported with an accuracy of 85% [23] using 10 fold MCV.

Table III
AREA UNDER THE CURVE (AUC) AND PERCENT ACCURACY ON THE TARGET CLASS FOR VARIOUS UCI DATASETS. †COULD NOT COMPUTE COVARIANCE MATRIX.

	Ionosphere		**SPECTF**		**Sonar**		**WDBC**		**Parkinsons**	
	AUC	%	AUC	%	AUC	%	AUC	%	AUC	%
T*ROF	0.8915	74.60	0.8849	90.57	**0.9255**	**83.51**	**0.9574**	**87.74**	0.7405	**83.67**
T*ENT	0.8429	84.89	**0.8959**	**92.45**	0.6235	61.26	0.8474	85.85	0.5319	80.27
SOE1	0.7691	80.00	0.7642	84.43	0.601	56.76	0.9203	80.66	**0.7456**	80.95
FBAG	0.6048	69.78	0.436	78.30	0.5018	54.05	0.5161	39.63	0.4239	72.11
LOF	0.5836	66.22	0.4154	77.83	0.4954	54.96	0.4699	35.85	0.4386	72.79
ORCA	0.4575	60.89	0.5569	80.19	0.5056	49.48	0.5091	39.62	0.5169	76.19
RMD	**0.9479**	**87.40**	0.7527	84.36	0.5857	62.73	0.9131	77.25	†	
T*LOF	0.4297	63.56	0.3753	75.94	0.4836	52.25	0.4834	38.21	0.4249	72.79

Table IV
AREA UNDER THE CURVE (AUC) AND PERCENT ACCURACY ON THE TARGET CLASS FOR VARIOUS UCI DATASETS. †COULD NOT COMPUTE COVARIANCE MATRIX. *EXCEEDED TIME LIMIT.

	Spambase		**Credit Card**		**Musk**		**Insurance**	
	AUC	%	AUC	%	AUC	%	AUC	%
T*ROF	**0.9077**	**79.15**	0.3237	63.14	**0.7055**	**58.45**	0.5213	4.31
T*ENT	0.7278	78.26	**0.7689**	**89.14**	0.3297	28.99	0.5386	10.06
SOE1	0.7078	58.36	0.4422	69.14	0.3359	29.95	**0.5489**	**10.63**
FBAG	0.4742	37.51	0.5201	68.57	0.4967	43.48	0.4999	8.33
LOF	0.4668	36.62	0.494	69.14	0.4906	43.00	0.4994	6.61
ORCA	0.4909	38.44	0.5551	71.71	0.5084	46.38	0.5359	8.62
RMD	†		†		†		†	
T*LOF	0.4825	39.33	0.5168	71.57	0.4958	42.51	*	

Credit Card: This is the Statlog (German Credit Card) dataset. Eggermont et al. used Genetic Programming and compared with C4.5 with Bagging and Boosting and other algorithms. The best result was a 27.1% misclassification rate [24].

Musk: This is a highly studied dataset. For example, Zafra and Venture report accuracy with SVMs of 87% and 93% with a genetic algorithm [25]. This dataset is only nominally two-class as both musks and non-musks are made up of multiple classes and thus is possibly not suitable for the outlier approach.

Insurance: This was part of the COIL 2002 competition and proved difficult for all algorithms. The best results were only a few percentage points better than the best in Table IV (See e.g. [26]).

The very best supervised classifiers outperform the outlier methods on most datasets tested but the margins are generally small. Remarkably, one of our unsupervised approaches achieved the best result on two datasets — Sonar (T*ROF) and Credit Card (T*ENT). T*ENT and T*ROF are remarkably competitive considering that they simply exploit the difference in variance between the classes and operate unsupervised.

C. Parameters

T*ROF has no user adjustable parameters. T*ENT has a parameter θ which determines the minimum cluster size for a cluster to be considered 'major' and thus not contribute to the outlier result. In all the Type 1 experiments and most of the Type 2 experiments, this was set by the heuristic advised in [27] for TURN*, that is $\theta = \min\{100, \frac{|D|}{100}\}$. On two datasets, WDBC and Parkinsons, somewhat better results were obtained when this value was reduced, showing some sensitivity to this setting. SOE1 builds a histogram of the data and thus requires either digitised data or a digitisation parameter. For each experiment, a series of different settings were tried and the best results reported. SOE1 proved quite sensitive to this parameter. A mean could have been reported but then this risked a bias due to the range of values tested. LOF and Feature Bagging have a parameter $minpts$ but we found that a value of 30 generally gave the best results as previously reported. RMD and ORCA required no parameter settings.

V. CONCLUSION

This paper introduces an entirely novel concept of separating classes based on variance rather than spatial location. This distinguishes it from statistical and other approaches, which use variance to validate separation generated by exploiting spatial differences. We have shown that this can be done very successfully in two-class problems even when the classes are balanced showing that outlier detection has application beyond the traditional problem of finding small

numbers of extreme data or rare classes. Two novel outlier methods are presented that outperform the state-of-the-art in outlier detection algorithms. These demonstrate that, while inspecting 2D spaces is more expensive than simply analysing 1D spaces, a clear effectiveness benefit is obtained and the cost is still well below that of standard statistical methods that involve computing a covariance matrix.

These methods are provably efficient and empirically shown to be very effective and robust with respect to parameter settings. Due to its framework nature, single components (the outlier detection method used in the subspaces) can be replaced.

The benefit of our outlier scoring technique shows in its application to unsupervised class separation beyond the 'rare classes' case.

If good methods for outlier detection in $3, 4, 5, \ldots$ dimensions can be developed, it would be interesting to analyze our framework for different values of k, i.e., for different subspace sizes. Of course it might be too inefficient to detect outliers in *all* 3-dimensional spaces (there are in general too many of these), but instead of looking at all of them, a random sample might be sufficient.

APPENDIX

Two *Outlier* Algorithms Employable by T*

As stated in Section III, one could use different variants of *Outlier* algorithms. We describe two variants here, both building on TURN* [27], a state-of-the-art clustering algorithm which has shown to outperform various others (DBSCAN, CHAMELEON, CURE, ROCK, Wavecluster, and k-Means) and has the additional advantage of being parameter-free. Since TURN* has proven to work best in 2 dimensions so far, but scales exponentially in d, we restrict our use of the T* framework to $k = 2$ in this paper. However, our experimental results will show that this already yields a method outperforming the state-of-the-art methods on real-world datasets.

To summarize the TURN* method, we first need to introduce the concepts of nearest neighbours and of clusters, given a real-valued parameter r called *resolution*.

Neighbours and Nearest Neighbours. Let $x, x' \in D$, $S = \{i, j\} \subseteq \{1, \ldots, d\}$, $i \neq j$. x is a *neighbour* of x' in D with respect to S if $|x_i - x'_i| \leq r$ and $|x_j - x'_j| \leq r$. x is a *nearest neighbour* of x' with respect to S (an S-NN for short) if x is a neighbour of x' with respect to S and there is no $y \in D$ with

$$\begin{aligned} y_j \neq x'_j \quad &\text{and } (x_i < y_i \leq x'_i \text{ or } x_i > y_i \geq x'_i) \text{ or} \\ y_i \neq x'_i \quad &\text{and } (x_j < y_j \leq x'_j \text{ or } x_j > y_j \geq x'_j)\,. \end{aligned}$$

Internal Points and Clusters. Let $S = \{i, j\} \subseteq \{1, \ldots, d\}$, $i \neq j$, and $x, x' \in D$. x is *internal* in D_S if x has at least 4 S-NNs. x and x' are called *reachable* in D_S if there are internal points $n_0, \ldots, n_z$ in D_S such that x is an S-NN of n_0, n_m is an S-NN of n_{m+1} for all $m < z$, and n_z is an S-NN of x'. A *cluster* in D_S is a maximal set of points that are pairwise reachable in D_S.

TURN* consists of two modules. The first component is a clustering algorithm, which, given a dataset D and a resolution r, assigns all points in D to clusters. Clusters are built starting with a not yet touched internal point and recursively adding nearest neighbours for every internal point reached. The second component varies the resolution fed to the first component in order to find the 'best' clustering, repeatedly calling the first component. The reader is referred to [27] for details. It is important to note that the resolution parameter is adjusted by TURN* and need not be dealt with by the user.

In what follows we describe two *Outlier* methods; the corresponding variants of T* are called T*ENT and T*ROF.

A. T*ENT — Finding the Best Resolution by Entropy

The *Outlier* method in T*ENT varies the resolution selection criterion in the second TURN* component. It calls TURN* and collects a series of mean cluster density values (see below) of the clusterings obtained while varying over different resolution values r. The entropy of the clusterings are computed for all the resolution values at which the mean density changes its trend, i.e., at which the series shows a 'knee' suggesting an area of stability in the clustering results. Finally the clustering with the lowest entropy is selected as this likely has the least number of outliers. Once this clustering is found, all points in clusters that are smaller than a certain threshold θ are defined outliers. The outlier degree out for a given point is then simply 1 if the point is an outlier (in a cluster of size $< \theta$) and 0 if the point is not an outlier (in a cluster of size $\geq \theta$). In particular, the outlier degree is just a binary value expressing whether or not we consider a point an outlier rather than a real value expressing how much we consider a point an outlier.

In more detail, the behaviour of the *Outlier* method in T*ENT, applied to a 2-dimensional dataset $D_{\{i,j\}}$, can be described as follows.

1) Run TURN* on $D_{\{i,j\}}$.
2) Let $r_1, r_2, \ldots, r_T$ be the sequence of resolutions TURN* goes through.
3) For every resolution r_t, $1 \leq t \leq T$, compute a mean density with respect to the corresponding TURN* clustering as follows. For every $x \in D$ compute a local density

$$\left(\sqrt{L_i(x)^2 + R_i(x)^2} + \sqrt{L_j(x)^2 + R_j(x)^2}\right)^{-1}.$$

Here $L_i(x)$ is the closest nearest neighbour (for resolution r) whose value in attribute i is not higher than x_i (and accordingly with j instead of i); $R_i(x)$ is the closest nearest neighbour (for resolution r) whose value in attribute i is not smaller than x_i (and accordingly with j instead of i). The mean density is

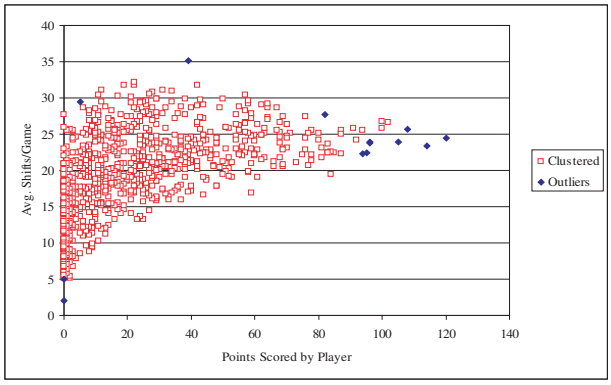

Figure 3. Outliers in a 2-dimensional subspace, automatically detected by the *Outlier* method in T*ENT (sample attribute pair, NHL data, $\theta = \min\{100, \frac{|D|}{100}\}$). Filled points are flagged as outliers in this 2-dimensional subspace, the others are not considered outliers in this space.

then the mean over all local densities of non-outlier points $x \in D$.

4) Detect all the resolutions r_t^* for which there is a change in the second differential of the series of mean density values.[3]
5) Of all those resolutions r_t^*, pick the one for which the corresponding clustering C has the lowest entropy value given by

$$H = \sum_{c \in C} p_c ln(p_c)$$

where p_c is the probability of a datapoint falling in cluster c.

6) For every $x \in D$, let

$$\text{out}(x, D_{\{i,j\}}) = \begin{cases} 1\,, & \text{if } x \text{ is in a cluster of size} < \theta\,, \\ 0\,, & \text{otherwise}\,. \end{cases}$$

Note that the *Outlier* method in T*ENT requires setting the parameter θ. We address this point in Section IV.

For illustration, consider Figure 3 for an NHL dataset [15] in which each point is a hockey player described by 16 attributes. The figure shows outliers flagged by the method used in T*ENT in the 2-dimensional space spanned by the attributes showing (i) the number of points scored by a player over the season, and (ii) the average number of shifts a player had per game.

B. T*ROF — Accumulating Outlierness over Different Resolutions

The *Outlier* method in T*ROF applies TURN* without the stopping criterion for optimal resolutions. It simply computes the out value of a point x as the resolution-based outlier factor (ROF) over all different resolutions that TURN* goes through over the resolution range. The ROF is the sum of the ratios of cluster sizes of the cluster the point x is contained in, as resolution changes. ROF was previously applied successfully to a 3-dimensional engineering dataset, cf. [14], but not developed for higher dimensionality. In more detail, the behaviour of the *Outlier* method in T*ROF, applied to a 2-dimensional dataset $D_{\{i,j\}}$, can be described as follows.

1) Run TURN* on $D_{\{i,j\}}$.
2) Let $r_1, r_2, \ldots, r_T$ be the sequence of resolutions TURN* goes through.
3) For every $x \in D$, let

$$\text{out}(x, D_{\{i,j\}}) = \sum_{1 \le t \le T-1} \frac{|C(x, r_t)| - 1}{|C(x, r_{t+1})|}\,,$$

where, for every t, $C(x, r_t)$ denotes the cluster to which the first component of TURN* assigns the point x, when this component is run with resolution r_t.

Note that T*ROF has no user-definable parameters.

C. A Remark on Complexity

T*ENT and T*ROF have a run time cost in $O(d^2|D|log|D|)$. For all of the $O(d^2)$ attribute pairs, all of the points in D have to be sorted according to their attribute values along both dimensions; this is what dominates the run time. The space complexity is dominated by holding a $|D| \times d$ matrix in memory. However, the algorithm only requires two attributes to be processed at any one time so the minimum size is $O(2|D|)$.

ACKNOWLEDGMENTS

We gratefully acknowledge support by the Alberta Ingenuity Fund and NSERC. We thank Robert Holte for his helpful comments.

REFERENCES

[1] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys*, vol. 41, no. 3, 2009, article 15.

[2] M. Petrovskiy, "Outlier detection algorithms in data mining systems," *Program. Comput. Softw.*, vol. 29, no. 4, pp. 228–237, 2003.

[3] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, "When is "nearest neighbor" meaningful," in *Int. Conf. on Database Theory*, 1999, pp. 217–235.

[4] A. Lazarevic and V. Kumar, "Feature bagging for outlier detection," in *Proc. ACM SIGKDD*, 2005, pp. 157–166.

[5] Z. He, X. Xu, and S. Deng, "A unified subspace outlier ensemble framework for outlier detection in high dimensional spaces," in *Proc. of the 6th International Conference, WAIM 2005*, 2005, pp. 632–637.

[6] M. Breunig, H.-P. Kriegel, R. Ng, and J. Sander, "LOF: Identifying density-based local outliers," in *Proc. SIGMOD Conf.*, 2000, pp. 93–104.

[3] This technique is routinely used in time series analysis to render a series stationary [28].

[7] S. Bay and M. Schwabacher, "Mining distance-based outliers in near linear time randomization and a simple pruning rule," in *Proc. SIGKDD*, 2003.

[8] P. Rousseeuw and K. V. Driessen, "A fast algorithm for the minimum covariance determinant estimator," *Technometrics*, vol. 41, no. 3, pp. 212–223, 1999.

[9] C. Aggarwal and P. Yu, "Outlier detection for high dimensional data," in *Proc. ACM SIGMOD Intl. Conf. on Management of Data*, 2001, pp. 37–46.

[10] ——, "An efficient and effective algorithm for high-dimensional outlier detection," *VLDB Journal*, vol. 14, no. 2, pp. 211–221, 2005.

[11] J. Zhang and H. Wang, "Detecting outlying subspaces for high-dimensional data: the new task, algorithms, and performance," *Knowl. Inf. Syst.*, vol. 10, no. 3, pp. 333–355, 2006.

[12] E. Knorr and R. Ng, "Finding intensional knowledge of distance-based outliers," in *Proc. VLDB Conf.*, 1999, pp. 211–222.

[13] V. Milman, "The heritage of P. Levy in geometrical functional-analysis," *Asterisque*, vol. 157, pp. 273–301, 1988.

[14] H. Fan, O. R. Zaiane, A. Foss, and J. Wu, "A nonparametric outlier detection for effectively discovering top-n outliers from engineering data," in *Proc. of PAKDD'06*, 2006, pp. 557–566.

[15] NHL, "Official web site: www.nhl.com," 2008. [Online]. Available: www.nhl.com

[16] C. Blake and C. Merz, "UCI repository of machine learning databases," http://archive.ics.uci.edu/ml/, 1998.

[17] H. Wang, "Nearest neighbors by neighborhood counting," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 28, no. 6, pp. 942 – 953, 2006.

[18] H. Kim and S. H. Park, "Data reduction in support vector machines by a kernelized ionic interaction model," in *Proc. of SDM'04*, 2004.

[19] L. A. Kurgan, K. J. Cios, R. Tadeusiewicz, M. R. Ogiela, and L. S. Goodenday, "Knowledge discovery approach to automated cardiac spect diagnosis," *Artificial Intelligence in Medicine*, vol. 23, p. 149, 2001.

[20] M. L. Ali, L. Rueda, and M. Herrera, "On the performance of chernoff-distance-based linear dimensionality reduction techniques," *Advances in Artificial Intelligence*, vol. 4013, pp. 467–478, 2006.

[21] S. Harmeling, G. Dornhege, D. Tax, F. Meinecke, and K.-R. Müller, "From outliers to prototypes: Ordering data," *Neurocomputing*, vol. 69, pp. 1608–1618, 2006.

[22] Y. Jiang and Z.-H. Zhou, "Editing training data for kNN classifiers with neural network ensemble," *Lecture Notes in Computer Science 3173*, pp. 356–361, 2004.

[23] V. A. Petrushin and L. K. (Eds), *Multimedia data mining and knowledge discovery*. Springer, 2007.

[24] J. Eggermont, J. N. Kok, and W. A. Kosters, "Genetic programming for data classification: Partitioning the search space," in *Proc. of the 2004 Symposium on applied computing (ACM SAC'04)*, 2004, pp. 1001–1005.

[25] A. Zafra and S. Ventura, "Multi-objective genetic programming for multiple instance learning," in *Lecture Notes in Computer Science, Machine Learning: ECML '07*, 2007.

[26] B. Zadrozny and C. Elkan, "Transforming classifier scores into accurate multiclass probability estimates," in *Proc. of KDD '02*, 2002.

[27] A. Foss and O. R. Zaïane, "A parameterless method for efficiently discovering clusters of arbitrary shape in large datasets," in *Proc. of the IEEE International Conference on Data Mining (ICDM'02)*, 2002, pp. 179–186.

[28] T. Masters, *Neural, Novel and Hybrid Algorithms for Time Series Prediction*. John Wiley & Sons, 1995.