Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 329 (2016) 61–77

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

An Unsupervised Approach for Combining Scores of Outlier Detection Techniques, Based on Similarity Measures

José Ramón Pasillas-Díaz [1,2] Sylvie Ratté[3]

Department of Software and IT Engineeringg
École de Technologie Supérieure
Montreal, QC, Canada

Abstract

Outlier detection, the discovery of observations that deviates from normal behavior, has become crucial in many application domains. Numerous and diverse algorithms have been proposed to detect them. These algorithms identify outliers using precise definitions of the concept of outliers, thus their performance depends largely on the context of application. The construction of ensembles has been proposed as a solution to increase the individual capacity of each algorithm. However, the unsupervised scenario (absence of class labels) in the domains where outlier detection operates restricts the use of approaches relying on the existence of labels. In this paper, two novel unsupervised approaches using ensembles of heterogeneous types of detectors are proposed. Both approaches construct the ensemble using solely the results produced by each algorithm, identifying and giving more weight to the most suitable techniques depending on the particular dataset under examination. Through experimental evaluation in real world datasets, we demonstrate that our proposed algorithm provides a significant improvement over the base algorithms and even over existing approaches for ensemble outlier detection.

Keywords: outlier detection, ensembles

1 Introduction

Our capacity to collect and store data increases in an exponential manner but our capacity to analyze it has not followed the same trend. Despite the explosion of available data, the discovery of truly interesting patterns is a rare event. Outlier detection the discovery of observations that deviates from normal behavior has been widely studied in recent years [26,15,7], resulting in a set algorithms designed to detect these rare but potentially crucial events. In some specific contexts an outlier is a data point that can be considered either as an abnormality or noise,

[1] This work was supported by CONACYT Mexico, Scholarship 214609.

[2] Email: jose-ramon.pasillas-diaz.1@ens.etsmtl.ca

[3] Email: sylvie.ratte@etsmtl.ca

http://dx.doi.org/10.1016/j.entcs.2016.12.005

whereas anomaly refers to a special kind of outlier which is of interest to the analyst. However, the terms outlier and anomaly, in general, have been used interchangeably in the literature [7].

One of the core definitions of outliers was made in 1980 by Grubbs [12]: An outlying observation, or outlier, is one that appears to deviate markedly from other members of the sample in which it occurs. However, this definition lacks one important characteristic, this is, the case where the outlying points conglomerates to form their own group of outliers; Barnett and Lewis [2] improved the definition of outliers by considering as outlier not only a single and isolated point, but also a group of points deviating from the normal behavior.

The effect of undetected outliers in different application domains (i.e. medical, intrusion detection, fraud detection, geographical) could have deep and disastrous consequences. An example is the detection of breast cancer where an undetected positive case implies an untreated patient; another example is a failed attempt to detect strange behavior in the use of a stolen credit card resulting in a financial impact for the credit card holder. In both of these examples, the minority of the cases represents the class of interest.

The process of outlier detection represents a very specific classification scenario: first, the quantity of outliers is very small in proportion to the quantity of normal instances; and second, the use of labels (supervised approach) in outlier detection is limited due to the fact that, by definition, the outliers that we are trying to detect represent a new or unseen behavior. Despite the fact that some algorithms (techniques) can operate using only labels for the normal class [23] (semi supervised approach) and use this information to increase the detection rate, unsupervised approaches have the undeniable advantage of operating over unlabeled data. Furthermore, unlabeled data are usually easier to obtain and represents the more common scenario in outlier detection [10].

The use of an unsupervised outlier detection approach also has the benefit of avoiding the bias introduced by training an algorithm with anomalous observations, labeled wrongly as normal data, causing the misclassification of future similar observations.

Due to the large spectrum of domains where outlier detection can operate, there are a wide variety of outlier detection algorithms mainly based on: classification, clustering, nearest neighborhood and statistical approaches [7]. However, their use is application dependent; no single outlier detection algorithm is best suited for all the different data scenarios that we could encounter in real world datasets [19]. Some algorithms work better when the data tend to form clusters, whereas others are most suitable to use in the presence of neighborhoods in the data.

Despite the fact that by working on an unsupervised scenario it is not possible to know which algorithm is better for a specific dataset in advance, the performance of these algorithms can be improved.

Similar to ensemble classifier learning, where heterogeneous assumptions are used to produce a unified output [25,24], in ensemble outlier detection, diverse (heterogeneous) assumptions are also needed to produce a meaningful result, potentially

complementing each other. There is no gain if the techniques that form the ensemble produce exactly the same output.

The more common scenario is to construct a diversified ensemble with techniques whose results are uncorrelated, using class labels (supervised approach) and algorithm outputs to determine the similarity between techniques. However, when the class labels necessary to compare the agreements between techniques are absent (as is the case in an unsupervised setup), a different way to establish diversity must be found. In this regard, some approaches ensure diversity in the ensemble by providing different samples of features, but apart from the fact that multiple iterations are required to analyze each sample, some datasets will require the use of the complete set of features to identify the outlier observations.

The approach we are proposing reaches diversity not by comparing the output of the algorithms and the class labels, but by creating the ensemble with a varied set of algorithms.

Combining outputs of different classifiers is not a novel task; however, outlier detection has to face two additional problems [20]. First, an ensemble of classifiers works with discrete labels whereas outlier detection is mainly concerned with scores. Second, an ensemble of classifiers generally relies on the existence of training data (supervised approach), whereas outlier detection generally does not have access to labeled data (unsupervised approach).

We propose two novel approaches based on a weighted combination of outlier detection algorithms, both of which give more weight to algorithms whose outputs offer an expected better performance for a specific data representation, and improve the differentiation between outlier and inlier by increasing the relative distances between the scores of outliers and those of inliers.

The rest of the paper is organized as follows: section II provides a background about techniques for outlier detection, ensemble methods, and evaluation procedures; section III introduces our approach in detail; section IV illustrates some experiments with real life datasets and section V concludes our research and discusses the scope for future work.

2 Background and related work

Outlier detection is a very active research area where new approaches are proposed each year. Nevertheless, the detection of outliers was first contemplated in the statistical community in 1887 [9]. Since then, different techniques based on various approaches such as classification, clustering, density based and statistical inference have been proposed.

An important characteristic of an outlier algorithm is its output, which can be either a score or binary label [1]. The former type of output assigns a score to each observation and in general can be used to rank the observations depending on its level of outlierness. The latter assigns binary labels, commonly using 1 for outliers and 0 to designate normal observations (inliers).

A score has the advantage of retaining more detail by providing a degree of

outlierness, whereas a binary output offers a more simplistic classification of an observation as either inlier or outlier. Despite the convenience of a binary output, the information retained in the scores could offer more insights about the outlierness of an observation.

The construction of an ensemble of outlier algorithms seems like a viable solution when the objective is to increase the detection rate of outliers (e.g. breast cancer detection) while diminishing the variance introduced by each outlier detection algorithm. However, no gain will be obtained by using algorithms whose results are identical. Therefore, two important factors must be taken into account when constructing an ensemble: accuracy and diversity. Accuracy measures the output quality of each algorithm, while diversity endeavors to build an ensemble whose results are distinct and, in theory, complementary. Accuracy depends on the right association of technique and dataset; diversity can be established using variations of the search space (data and feature sampling) or by the use of different types of algorithms [28]. Combining different types of algorithms could yield better performance than simply using parametric variations of the same algorithm [27]. However, a balance between accuracy and diversity is needed in order to obtain an improved ensemble detection rate [32]; highly diverse, but inaccurate algorithms, results in an ensemble whose components are truly diverse, but without the accuracy component is unable to converge near the true classification output, resulting in an ensemble whose detection rate is below that of its individual members.

The process of building an ensemble involves three main considerations: the choice of the algorithms, the organization (modular or ensemble) and the combination method [6]. A multiclassifier can be categorized as modular or ensemble. A multiclassifier is modular when each member is responsible for a specific part of the process and the algorithms are used in a series of steps, using the results of the previous algorithm. It is an ensemble when each single member works on the same search space and a combinatorial process joins the results to produce a unified output. In this paper we are focusing on the latter type. The most important component is the combinatorial approach chosen so that each single member (classifier) contributes to improve the overall performance.

One critical factor in the construction of an ensemble is to mix members (algorithms) whose errors are not identical; doing so assures us that these members complement each other, therefore producing potential improved results. However, the majority of such approaches assume that a measure of accuracy for each member is available, using class labels for each observation. Still, considering that outlier detection is mainly an unsupervised field, it is not practical to measure accuracy using output labels. In our proposed approach, we do not assume highly accurate classifiers trained with the use of labeled data; instead we estimate accuracy by considering only the output scores of each algorithm and attempting to achieve diversity using different types of outlier detection techniques.

In our empirical studies, four detectors are used: a density based approach (Local Outlier Factor or LOF), two distance based approaches (k-means & hierarchical clustering) and a statistical based approach (modified boxplot). The density based

approach LOF is considered one of the most performing outlier detection algorithms [19]. This technique computes a degree of isolation that depends on two factors: first, the distance between a point and its neighbors, and second, the density of the neighborhood. The detection of outliers using boxplots [30,18] is one of the most simple model based techniques; this statistical approach makes no specific assumptions about the data distribution determining as outliers those points beyond a specific threshold. The first distance based approach relies on the k-means algorithm [13]; the data is divided into different groups depending on the closest centroid; the outlierness of a point is equal to the distance to its closest centroid. An outlier algorithm using hierarchical clustering [29] divide the data into binary clusters recursively until the data cannot be divided any further; in this case outliers consist of those observations that present more resistance to being merged into a cluster.

While increasing the detection rate of the ensemble using a combination of only highly accurate classifiers seems like a good idea, the unsupervised nature of the datasets where outlier detection operates is a limiting factor. When considering an unsupervised scenario, it is crucial to use selfsufficient measurements of diversity that are based only on the results of the members of the ensemble and not assume the existence of labels for the normal instances (semi-supervised approach) or labels for both normal and outlier instances (supervised approach).

Therefore, our focus is on self-sufficient measurement of diversity. Previous studies such as feature bagging (FB) [20] use variations of the search space to induce diversity in the ensemble; a similar study [22] uses both variation in the search space and different outlier detection techniques.

The feature bagging approach starts by randomly choosing without replacements different subsamples of features; then in a series of rounds, each outlier technique analyzes these subsamples producing a set of output scores. Finally, the process of joining the scores can be performed with any of the two methods provided by the authors of feature bagging: *Breadth First* and *Cumulative Sum*.

The *Breadth First* method first sorts the outlier scores from all the iterations of feature bagging, next takes the index of the record with the highest score and then inserts its index in a vector, and so on. If an index is already in the vector, it is omitted. The final output is a vector of indices pointing to its corresponding scores.

The second variant of feature bagging is *Cumulative Sum*. This method simply adds up the scores of each iteration of feature bagging, and the outliers are those observations with a resulting high score.

The *Breadth First* approach is exposed to a critical observation: it is highly sensitive to the order in which the outlier detection algorithms were applied. This means that the first technique in the ensemble has priority to decide about the outlierness of a given data record. Also, the methodology of this approach does not indicate how to establish the order of the algorithms.

Cumulative Sum reports better performance overall when compared with the *Breadth First* method [20]. This way of combining the outputs overcomes the order problem of the the members in *Breadth First*. However, neither of the two variants

of feature bagging takes the use of different types of algorithms into account.

The authors of feature bagging used only one algorithm (LOF) for their experiments and there is no mention on how to join scores in different scales. To achieve better performance, their experiments assume the existence of labels for the normal instances (inliers).

The authors of feature bagging [20] report improvements on performance over a single outlier detection technique; their results provide solid foundation upon which to compare new approaches. However, we hypothesize that better performance can be achieved by joining the outputs of different types of algorithms and setting specific weights, without assuming any knowledge of the output labels.

Receiver operating characteristics (ROC) curves are very useful when measuring the performance of outlier detectors. These curves consist in plotting the true positive rate (TPR=ratio of true positives to actual positives) versus the false positive rate (FPR=ratio of false positives to actual negatives) using a variation of a discriminant threshold. For that matter, the area under the curve (AUC) is often used as the benchmark in outlier analysis [19,20,27,22,17,11]. AUC is the probability that a randomly selected positive instance will be ranked higher than a randomly selected negative one. AUC is a convenient metric to evaluate the performance of outliers algorithms when it is not possible to predetermine a threshold and instead of a ROC curve a single measure is required [3]. The higher the AUC, the better the expected performance of the technique; an AUC=1 indicates a perfect performance, whereas an AUC=0.5 indicates performance similar to a simple random choice.

Besides ROC curves and AUC, other commonly used evaluation measures are accuracy and precion@n [8]. The former, is commonly used in the classification scenario to evaluate the results of classification algorithms; however, in outlier detection the highly imbalanced datasets can bias this measure; e.g. a simplistic classifier assigning all the observations to the inlier class will produce a high and misleading accuracy value, when truly it is erroneously classifying all the outlier observations, which are in outlier detection the observation that the final user is, indeed, trying to find. The latter, is another measure that can be used to evaluate outlier detection algorithms; however, this measure is highly sensitive to the selection of n [5]; e.g. in a toy scenario with only 2 outliers and 100 inliers, an outlier detection algorithm ranks the true outliers in the third and fourth position (almost perfectly considering an unsupervised outlier detection scenario), a selection of n=4 would result in a precision@n=0.5; however, setting n=2 would give a precision@n=0, despite that the classifier has indeed highly classified the outliers. Precision@n requires the user to have at least some knowledge about the expected number of outliers in the data; in outlier detection, being in general an unsupervised setting, it is neither possible to know in advance the ground truth class labels nor the number of outliers present in the data.

ROC curves are widely used in the literature to evaluate unsupervised outlier detection algorithms, then their use facilitates the comparability with previous research works[9].

3 The approaches

We propose two novel approaches for combining the outputs of heterogeneous outlier detection algorithms in an unsupervised scenario: ensemble of detectors with correlated votes (*EDCV*) and ensemble of detectors with variability votes (*EDVV*).

With prior knowledge of which detector will work better for each dataset, it is possible to predetermine a specific weight for each algorithm. However, working in an unsupervised approach requires measuring the ability of each algorithm independently of the existence of labels. The main difference between *EDCV* and *EDVV* is the measure used to estimate the coefficients or weights when the outputs of the algorithms are compared. *EDCV* uses correlation coefficients as a similarity measure, whereas *EDVV* uses the mean of the absolute deviations between outputs (MAD) as a dissimilarity measure in the form of 1-MAD. The two also use a modified boxplot method to determine the number of outlierness votes that each observation receives from the algorithms. In this way, both approaches assign weights but in two different ways: first, by measuring the performance of each algorithm over the specific dataset (similarity/dissimilarity measures), and second, by giving a number of votes to each individual score produced by each algorithm.

At this point, two different measures (correlation for *EDCV* and MAD for *EDVV*) are used to determine the individual performance of the algorithms over a specific dataset. The similarity/dissimilarity measures assign specific weights to each one of the algorithms of the ensemble, giving more influence to those algorithms whose outputs are similar.

The approaches use two different similarity/dissimilarity measures for numerical values: correlation and MAD; we use them to measure the similitude between the outputs of different classifiers. The former can be used to evaluate the statistical correlation between different outputs; also it is indifferent to the scale of the input values and will produce a result of 1 for perfectly correlated values, 0 for uncorrelated values and -1 for negatively correlated values. The latter is used to measure the absolute deviation between different outputs. MAD produces results relative to the scale of its components. Whereas MAD tends to assign low values to similar scores, correlation coefficient assigns high values to correlated scores.

3.1 General approach

The two approaches we are proposing are based on the same procedure described in Algorithm 1, however, they differ critically in the way they assign the weights to each algorithm. In this subsection we present the first phase of both approaches leaving the weight assignation for the following subsections 1) (*EDCV*) and 2) (*EDVV*).

As shown in Algorithm 1, a given dataset (*DS*) of size m is first examined by applying each of the algorithms in a series of T rounds, where T represents the number of algorithms available in the ensemble. For testing purposes we are using T=4 . Nonetheless, T can take different values, meaning that our approach is not constrained either to the use of specific outlier algorithms or by the number of them. We expect that our approach can be applied using the majority of outlier detection

Algorithm 1 General Approach for combining outlier detection scores

Input: Given a dataset DS=((x_1),(x_2)(x_m)) of size m, where x_i represent a specific observation. T equals the set of algorithms in the ensemble; T_i refers to a specific algorithm in T.

Output: Ensemble outlier scores $Ffinal$

procedure GENERAL APPROACH
  for each i in $t \in T$ **do**
    Select randomly, without replacements, a set of features $F(t)$ from D of random size between $d/2$ and d-1
    Apply outlier algorithm T_i to DS
    The output of T_i is output score F_i
    Standardize F_i
  end for
  Determine votes (V)
  Determine weights (W)
  Combine the output scores F and produce a final ensemble output $Ffinal$
end procedure

Output: $Ofinal$

algorithms that are capable of producing results in the form of scores.

The different algorithms for outlier detection produce scores on different scales; for example while LOF tend to produce values close to 1, hierarchical clustering produces results with a much larger range. We have determined that the best way to normalize these results is to use a standardization procedure. Standardization is frequently used as a normalization method in ensemble outlier detection [14,20], bringing the different outputs to comparable scale and maintaining the relative larger scores of the outliers compared with those of inliers, avoiding in this way that algorithms with the largest range of results dominate the final result. The standardization method we are using consists in transforming the output scores (F) into Z scores with the conventional procedure Z= *(Xi-mean)/SD* (where *SD* is the standard deviation). This standardization step allows for an observation with a large score in one technique to maintain a large value after joining the ensemble.

Using these standardized outputs (F) from each algorithm, we then apply a modified boxplot technique to detect those outputs whose deviations are greater than the rest. In this way we produce a vector of votes (V) of size m * T (number of observations multiplied by the number of algorithms) that contains the number of votes of each algorithm for each observation. An observation receives a vote if its score is greater that 1.5*IQR (where IQR is the inter quartile range). We determine the IQR in the conventional way [31] IQR=Q3-Q1, where Q3 and Q1 stand for third quartile and first quartile respectively. Accordingly, the output matrix V in this step has the same dimensions as the matrix containing the standardized scores F. Each score in F will have a corresponding number of votes in V; for example V_{ij} corresponds to the number of votes assigned to F_{ij}.

The following two subsections (1) *EDCV* Approach & 2) *EDVV* Approach)

Algorithm 2 The *EDCV* approach for joining outlier scores

1: **procedure** *EDCV*
2: Compute matrix (C) of correlation coefficients between the standardized output scores F.
3: For each technique, produce w_n as the average of its corresponding column of correlations C_m:
4: **for** each n in T **do**

$$Ofinal = \frac{(\sum_{m=1}^{T} Cmn) - 1}{T - 1}$$

5: **end for**
6: **end procedure**

Output: Return matrix of weights $W = \{w_1, w_2, w_n\}$

describe the calculation of the matrix of weights (W). Although both approaches used the same general procedure, they differ in how they calculate the matrix W.

The matrix W measures the individual capacity of each algorithm over the specific dataset under examination, increasing the weight received for outliers while maintaining those of inliers. While it is obvious that each outlier algorithm has already assigned an intrinsic weight with the scores assigned to each observation, we attempt increasing the weight of outliers, while maintaining those of inliers, to have a better differentiation between outlier and non-outlier.

The main difference between the votes V and the weights W is that the votes are intended to increase the difference between outliers and non-outliers and are produced individually for each observation whereas the weights will not be specific to a particular observation but instead reflect the apparent capacity of the algorithm over the dataset under examination.

The subsection A.1)B explains how the F scores are combined using the votes (V) and weights (W) to produce the final score, *Ffinal*.

3.1.1 EDCV approach

The process of obtaining the weights (W) for each algorithm (T) using the *EDCV* approach is displayed in Algorithm 2. First, we obtain a matrix of correlations C (1) with dimensions m=size of T by n=size of T by calculating the correlation between the standardized scores F. For example, as represented in (1), C_{mn} stands for the correlation coefficient between scores F_m and F_n. Next, we divided the average of the correlations corresponding to each F_n by the size of T to obtain the matrix W; given that the correlation of an algorithm with itself is meaningless as it corresponds invariably to a perfect correlation with value 1, then we subtracted 1 from both the numerator and denominator. The resulting matrix of weights

W=$\{w_1,w_2,w_n\}$ represents the specific weights for each algorithm.

$$C = \begin{array}{c} \\ F_1 \\ F_2 \\ \vdots \\ F_n \end{array} \begin{array}{c} \begin{array}{cccc} F_1 & F_2 & \dots & F_n \end{array} \\ \begin{pmatrix} C_{11} & C_{12} & \dots & C_{1n} \\ C_{21} & C_{22} & \dots & C_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ C_{m1} & C_{m2} & \dots & C_{mn} \end{pmatrix} \end{array} \tag{1}$$

3.1.2 EDVV approach

The second variant of our approach, *EDVV*, obtains W with the process displayed in Algorithm 3. First, a matrix (D) (2) with dimensions m=size of T by n=size of T is produced by calculating the MAD between the standardized scores F.

$$D = \begin{array}{c} \\ F_1 \\ F_2 \\ \vdots \\ D_n \end{array} \begin{array}{c} \begin{array}{cccc} F_1 & F_2 & \dots & F_n \end{array} \\ \begin{pmatrix} D_{11} & D_{12} & \dots & D_{1n} \\ D_{21} & D_{22} & \dots & D_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ D_{m1} & D_{m2} & \dots & D_{mn} \end{pmatrix} \end{array} \tag{2}$$

Note that the matrix D is similar in size and structure to that produced by the other variant of our approach *EDCV*; however, in the present case the values of the matrix D (2) represent deviations and not correlations. MAD assigns lower values to similar output scores and our general framework expects that the highest weights of W represent the most suitable algorithms, so when feeding the matrix D with MAD values we transform them to a compatible form with our general approach by using the complement 1-MAD.

After this step, the average of the each F_n in matrix D is divided by the size of T-1 to produce the matrix W. This is different from the *EDCV* approach where we subtracted 1 from both the numerator and denominator; in the *EDVV* we only subtract 1 from the numerator, owing to the fact that a MAD between the same algorithm equals 0.

The resulting matrix W=$\{w_1,w_2,w_n\}$ is formed with the specific weights for each algorithm.

3.2 Putting it all together

The last phase of our general approach uses the weights W produced by either of our proposed variants: *EDCV* or *EDVV*.

The final process is displayed in Algorithm 3. First, we calculate the product of each of the standardized scores F and their corresponding votes in matrix V, then the resulting values are updated by applying the weights W obtained by either *EDCV* or *EDVV*. Finally, the updated scores from each algorithm are simply added together and divided by the size of T.

The output of this last phase is a vector of size m (number of observations) with the weighted and voted scores of all the algorithms of the ensemble. These

Algorithm 3 The *EDVV* approach for joining outlier scores

1: **procedure** *EDVV*
2: Compute a matrix (D) of mean absolute deviations (MAD) between the standardized output scores F.
3: For each technique, produce w_n as the average of its corresponding column of deviations D_m:
4: **for** each n in T **do**

$$Ofinal = \frac{\sum_{m=1}^{T} Dmn}{T-1}$$

5: **end for**
6: **end procedure**

Output: Return matrix of weights $W = \{w_1, w_2, w_n\}$

final scores have two main advantages over a simple averaging approach: first, they increase the relative distance between potential outlier and inliers, and second, they promote the outputs of the algorithms exhibiting the better expected performance.

In the following section, we present the experiments using real world datasets comparing the proposed approaches with 3 similar approaches: simple averaging, feature bagging *Cumulative Sum* and feature bagging *Breadth First*).

4 Experiments and evaluatoion

4.1 Methods and parameters

For our experiments, we compare the results of our approach with those of simple averaging, feature bagging *Cumulative Sum* and feature bagging *Breadth First*. We set the number of iterations for feature bagging to 50, while for simple averaging, *EDCV* and *EDVV* we used 4 iterations (one for each algorithm).

Feature bagging in its two variants (*Cumulative Sum* and *Breadth First*) uses only a single algorithm applied n times. The authors report their results using LOF as the single algorithm of their ensemble, thus when comparing our results with those of feature bagging, we also use LOF.

We set the number of algorithms in both approaches (*EDCV* and *EDVV*) equal to 4. The algorithms used in our ensemble are: LOF, k-means clustering, hierarchi-

Algorithm 4 Final averaged output after applying the corresponding votes and weights

1: **procedure** FINAL OUTPUT
2: **for** each i in m **do**

$$Ffinal = \frac{\sum_{j=1}^{T}(F(i,j) * V(i,j) * W(j)}{T}$$

3: **end for**
4: **end procedure**

Output: Return $Ffinal$

cal clustering and a modified boxplot method.

We use LOF as the technique with the expected best performance in our ensemble and the rest is formed with techniques whose performances are not expected to be better or significantly better than those provided by LOF.

The choice of the algorithms composing the ensemble was made in order to obtain a diversified set; by diversified we refer not only to the type of technique (distance or density based), but also to the quality of the results. In this way, the resulting set consists of different types of algorithms with different performances. The idea is to simulate a real world scenario where it is not possible to know in advance which technique is the more suitable for the dataset under study.

Where possible we use the default values of each algorithm, and in the case of clustering and LOF that need some adjustment in their parameters, we do not try to tune the configuration values to the specific domain or dataset. Instead, we use the same parameters with all the datasets; obviously tuning these values would result in a better overall performance, but we are simulating a scenario where there is no additional information about a particular dataset.

The goal in our experiments is to mimic a real estimation of the performance of the ensemble methods and not the performance of perfectly tuned outlier detection algorithms. Differently from the experiments performed by the authors of feature bagging who used the labels for the inliers (normal instances), we do not suppose the existence of labels, given that our experiments are based on a completely unsupervised approach. Despite this, we acknowledge that the inclusion of labels for the inliers will increase the performance of the algorithms and thus that of the ensemble.

Our results are also compared to a simple average of the scores of each algorithm, which surprisingly gives interesting results (see subsection IV.C).

To choose the configuration values for LOF and k-means, we follow the suggestions from [13,4]. For LOF, the parameter indicating the number of neighbors was set to 20; this decision was made by averaging the authors suggestion to use a value between 10 and 30 in the absence of more knowledge about the dataset under study. For the k-means clustering algorithm, we set the number of centers to eleven (k=11). The remaining two algorithms, hierarchical clustering and modified box-plot, were used with their default values.

4.2 Datasets

The datasets were selected based on: (a) real world problems, (b) different proportions of classes, (c) different number of variables and (d) used by previous and similar research on outlier detection. Table 1 gives the characteristics of the selected datasets located on the UCI machine learning repository (see table 1) [21].

For the breast cancer and ionosphere datasets, we did not perform any modification; we simply took the smallest class as the outlier class, and the rest as the normal (inlier) class. With the former dataset, the smallest class represents a classification of malignant cell nuclei, whereas the bigger class represents the benign case. The later dataset consists of measures from high-frequency antennas

detecting free electrons in the ionosphere; the majority class is composed of those measures representing some structure in the ionosphere, and the minority class by those cases where there is no evidence of structure formation in the ionosphere. For the satimage dataset, we use the smallest class as the outlier and merged the rest to be considered as the normal class. In this dataset, the classes represent multispectral values of pixels in a satellite image. When performing experiments on lymphography, we selected classes one and four (less than 5%) to be the outlier class and used classes two and three as the normal class.

To increase the number of available datasets, we used a procedure commonly used in similar studies [20,16], which consists in the adaptation of datasets not directly related with the problem of outlier detection. The procedure consists of transforming a multivariate problem into a two class problem in two steps: first, we identify the smallest class or a subset of the smallest classes, and consider them as the outlier class, then, the majority - or the rest of the classes - are merged and used as the normal class. Following this method, we formed 7 additional datasets based on ann_thyroid and shuttle datasets. Accordingly, for the ann_thyroid dataset, which contains three classes, the smallest two are related with hyperfunction and subnormal function (less than 10% of the dataset), and a third not hypothyroid class (normal condition); in this case, we produced 2 datasets by using each one of the minority classes in turns as the outlier class versus the normal condition.

Finally, for the shuttle dataset containing 6 classes, we selected class 1 (80% of the data) as the normal class and each of the remaining 5 classes in turns as the outlier class, obtaining 5 additional datasets.

Table 1
Datasets characteristics (Cl=Classes, At=Attributes, O=Outliers, I=Inliners)

Dataset	Cl	At	O	I	O (%)	Modifications
Breast cancer	2	32	212	357	37.26	Class 2 v/s. 1
Ionosphere	2	34	126	225	35.90	Class 2 v/s. 1
Lymphography	4	18	6	142	4.05	Merged class 1 & 4 v/s. rest
Satimage	7	36	626	5809	9.73	Small class v/s rest
Ann_thyroid (average)	3	21	73-177	3178	2.24-5.28	Each class v/s. 3
Shuttle (average)	6	9	2-809	11478	0.02-6.58	Classes 2,3,5,6 & 7 vs. class 1

4.3 Results

The results of our experiments on the resulting 11 datasets were presented in table 2. The ROC curves for simple average, feature bagging (*Cumulative Sum* and *Breadth First*), *EDCV* and *EDVV* were displayed in Fig 1. In the case of the ann_thyroid and shuttle datasets that were adapted to a binary class problem, the results were presented using the average of the AUC over the artificially produced datasets; their ROC curves were not presented for space reasons. For breast cancer, ionosphere, lymphography and satimage datasets, we presented both the AUC and the computed ROC curve.

Table 2 showed that both *EDCV* and *EDVV* outperformed simple average, FB *Cumulative Sum*, and FB *Breadth First* in almost all the datasets, the exception being the ann_thyroid dataset, where FB *Breadth First* showed better results; the main reason for this behavior is the dependence of *Breadth First* on the order in which the outputs of the algorithms are presented. Nevertheless, the authors of the *Breadth First* approach do not contemplate a procedure to sort these outputs and consequently, this approach relies on a random order, in the case of ann_thyroid the resulting random order was favorable to *Breadth First*. Despite that, both *EDCV* and *EDVV* showed better performance than FB *Cumulative Sum* and simple averaging.

As expected the worst performance for all algorithms was with the datasets adapted to a binary class problem. This is understandable since the union of different classes produced a single class with different distributions that are very difficult to detect by the individual algorithms of the ensembles. However, even on the artificially generated datasets, *EDCV* and *EDVV* offered an improved performance compared with the rest of the approaches. The advantage of *EDCV* and *EDVV* is that they do not assume an exceptional and constant good performance of the algorithms over all the different types of datasets, but instead, assign weights to the algorithms based on their performance on each dataset in particular.

Surprisingly, a simple average of the scores produced by the outlier detection algorithms gave a constant good performance.

Table 2
AUC (area under the curve) for simple averaging, feature bagging (FB) cumulative sum, feature bagging (FB) breadth first and our proposed approaches EDCV and EDVV.

Dataset	**Simple Average**	**FB cum.sum**	**FB Breadth first**	**EDCV**	**EDVV**
Breast cancer	0.8439	0.6475	0.6695	0.8489	**0.8609**
Ionosphere	0.8711	0.8654	0.8824	0.8916	**0.8980**
Lymphography	0.9871	0.9871	0.9765	**0.9894**	**0.9894**
Satimage	0.6439	0.5149	0.5079	**0.6517**	0.6326
Ann_thyroid (average)	0.7331	0.7081	**0.8360**	0.7501	0.7485
Shuttle (average)	0.9955	0.9133	0.9096	**0.9972**	0.9970

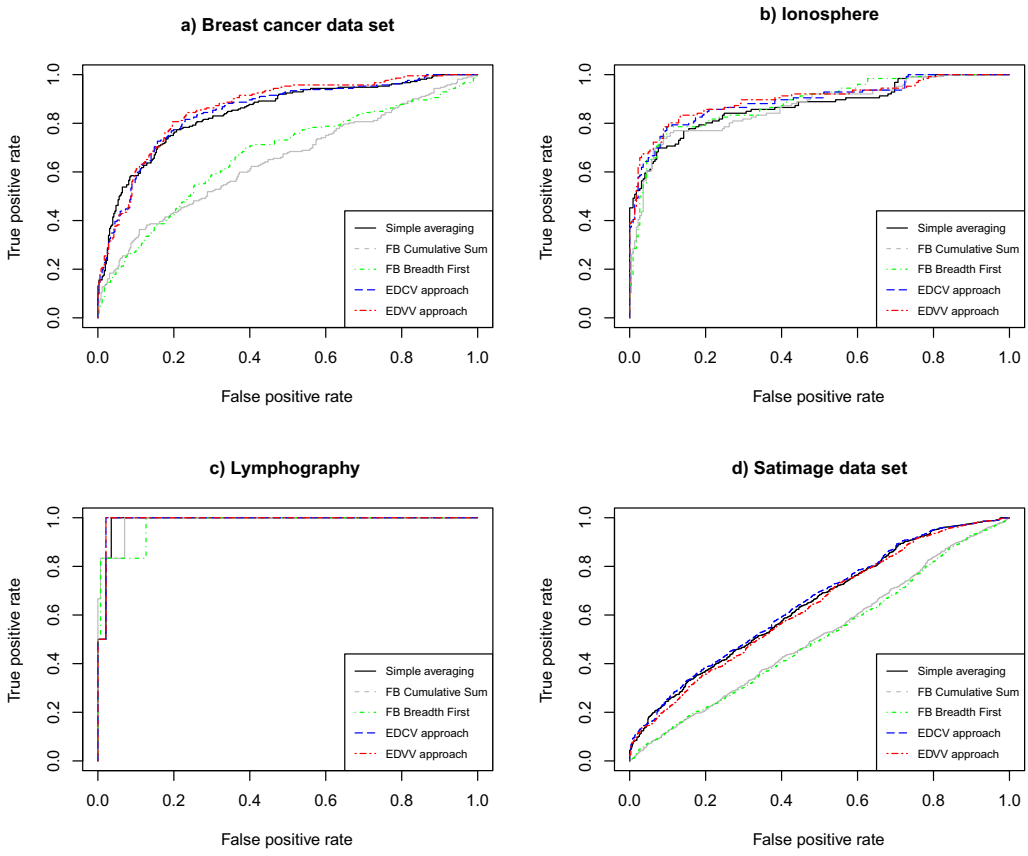

Fig. 1. ROC curves for LOF, Feature bagging and FBSO in Segmentation, Satimage, Waveform and Gisette datasets.

More constant improvements in *EDCV* and *EDVV* were found in the datasets originally designed for a binary classification (Fig 1). Table 2 showed that the AUC for both approaches (*EDCV* and *EDVV*) was better in the datasets of breast cancer, ionosphere, lymphography and Shuttle. Besides ann_thyroid, satimage was an exception where only *EDCV* had higher AUC than the rest of the ensembles.

5 Conclusions

In this paper, two novel and completely unsupervised ensemble approaches for combining the output scores of different outlier detection algorithms were presented: ensemble of detectors with correlated votes (*EDCV*) and ensemble of detectors with variability votes (*EDVV*). Experiments on several popular real life datasets suggested that both approaches can achieve better performance than similar methods. Also, it is worth considering that our results were obtained using only 4 iterations of the ensemble, while for feature bagging we set the number of iterations to 50.

These improvements were related to the fact that *EDCV* and *EDVV* do not make presumptions about the performance of the algorithms until they are capable

of comparing their outputs; thus the advantage is that both approaches are not expecting an exceptional and constant performance from all the algorithms on different types of datasets. Moreover, not expecting a constant performance of the algorithms allows for the inclusion of different types of outlier detection algorithms. While similar approaches like feature bagging *Cumulative Sum* and feature bagging *Breadth First* introduce diversity through variation on the search space, *EDCV* and *EDVV* attempt to ensure diversity by using different types of algorithms, which results in a more widely applicable approach.

Despite this, we consider that our results can be improved by using feature bagging variation of the search space as a way to deal with noisy attributes. In future work, we will attempt to address this possibility.

References

[1] Aggarwal, C. C., "Outlier Analysis," Springer Science & Business Media, 2013.

[2] Barnett, V. and T. Lewis, **3**, Wiley New York, 1994.

[3] Bradley, A. P., *The use of the area under the roc curve in the evaluation of machine learning algorithms*, Pattern recognition **30** (1997), pp. 1145–1159.

[4] Breunig, M. M., H.-P. Kriegel, R. T. Ng and r. Sander, *Lof: identifying density-based local outliers*, SIGMOD Rec. **29** (2000), pp. 93–104.

[5] Campos, G. O., A. Zimek, J. Sander, R. J. Campello, B. Micenková, E. Schubert, I. Assent and M. E. Houle, *On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study*, Data Mining and Knowledge Discovery (2015), pp. 1–37.

[6] Canuto, A. M., M. C. Abreu, L. de Melo Oliveira, J. C. Xavier and A. d. M. Santos, *Investigating the influence of the choice of the ensemble members in accuracy and diversity of selection-based and fusion-based methods for ensembles*, Pattern recognition letters **28** (2007), pp. 472–486.

[7] Chandola, V., A. Banerjee and V. Kumar, *Anomaly detection: A survey*, ACM Comput. Surv. **41** (2009), pp. 1–58.

[8] Craswell, N., *R-precision*, in: *Encyclopedia of Database Systems*, Springer, 2009 pp. 2453–2453.

[9] Edgeworth, F., *Xli. on discordant observations*, The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science **23** (1887), pp. 364–375.

[10] Eskin, E., A. Arnold, M. Prerau, L. Portnoy and S. Stolfo, *A geometric framework for unsupervised anomaly detection*, in: *Applications of data mining in computer security*, Springer, 2002 pp. 77–101.

[11] Fawcett, T., *Roc graphs: Notes and practical considerations for researchers*, Machine learning **31** (2004), pp. 1–38.

[12] Grubbs, F. E., *Procedures for detecting outlying observations in samples*, Technometrics **11** (1969), pp. 1–21.

[13] Hartigan, J. A. and M. A. Wong, *Algorithm as 136: A k-means clustering algorithm*, Journal of the Royal Statistical Society. Series C (Applied Statistics) **28** (1979), pp. 100–108.

[14] Hawkins, D. M., **11**, Springer, 1980.

[15] Hodge, V. J. and J. Austin, *A survey of outlier detection methodologies*, Artificial intelligence review **22** (2004), pp. 85–126.

[16] Joshi, M. V. and V. Kumar, *Credos: Classification using ripple down structure (a case for rare classes).*, in: *SDM*, SIAM, 2004, pp. 321–332.

[17] Kriegel, H.-P., P. Kröger, E. Schubert and A. Zimek, "Interpreting and Unifying Outlier Scores." SDM, 2011 pp. 13–24.

[18] Laurikkala, J., M. Juhola, E. Kentala, N. Lavrac, S. Miksch and B. Kavsek, *Informal identification of outliers in medical data*, in: *Fifth International Workshop on Intelligent Data Analysis in Medicine and Pharmacology*, 2000, pp. 20–24.

[19] Lazarevic, A., L. Ertoz, V. Kumar, A. Ozgur and J. Srivastava, *A comparative study of anomaly detection schemes in network intrusion detection*, , **3** (2003), pp. 25–36.

[20] Lazarevic, A. and V. Kumar, *Feature bagging for outlier detection*, , **21**, 2005, pp. 157–166.

[21] Lichman, M., *UCI machine learning repository* (2013).
URL http://archive.ics.uci.edu/ml

[22] Nguyen, H., H. Ang and V. Gopalkrishnan, "Mining Outliers with Ensemble of Heterogeneous Detectors on Random Subspaces," Lecture Notes in Computer Science **5981**, Springer Berlin / Heidelberg, 2010 pp. 368–383.

[23] Noto, K., C. Brodley and D. Slonim, *Anomaly detection using an ensemble of feature models*, in: *2010 IEEE International Conference on Data Mining*, IEEE, 2010, pp. 953–958.

[24] Opitz, D. and R. Maclin, *Popular ensemble methods: An empirical study*, Journal of Artificial Intelligence Research **11** (1999), pp. 169–198.

[25] Oza, N. C. and K. Tumer, *Classifier ensembles: Select real-world applications*, Information Fusion **9** (2008), pp. 4–20.

[26] Patcha, A. and J.-M. Park, *An overview of anomaly detection techniques: Existing solutions and latest technological trends*, Computer networks **51** (2007), pp. 3448–3470.

[27] Schubert, E., R. Wojdanowski, A. Zimek and H.-P. Kriegel, *On evaluation of outlier rankings and outlier scores*, in: *Proceedings of the 12th SIAM International Conference on Data Mining (SDM), Anaheim, CA, 2012*, 2012, pp. 1047–1058.

[28] Tan, K. M. and R. A. Maxion, *The effects of algorithmic diversity on anomaly detector performance*, in: *2005 International Conference on Dependable Systems and Networks (DSN'05)*, IEEE, 2005, pp. 216–225.

[29] Torgo, L., *Resource-bounded fraud detection*, in: *Portuguese Conference on Artificial Intelligence*, Springer, 2007, pp. 449–460.

[30] Torgo, L., "Data mining with R: learning with case studies," Chapman & Hall/CRC, 2010.

[31] Tukey, J. W., *Exploratory data analysis*, Addison-Wesley Series in Behavioral Science: Quantitative Methods, Reading, Mass.: Addison-Wesley, 1977 **1** (1977).

[32] Zimek, A., R. J. Campello, and r. Sander, *Ensembles for unsupervised outlier detection: challenges and research questions a position paper*, SIGKDD Explor. Newsl. **15** (2014), pp. 11–22.