International Journal of Pattern Recognition and Artificial Intelligence

Bagged One-Class Classifiers in the Presence of Outliers

SANTI SEGUÍ, LAURA IGUAL and JORDI VITRIÀ
Computer Vision Center
Campus UAB, Edifici O, 08193, Bellaterra, Spain

Dept. Matemàtica Aplicada i Anàlisi
Universitat de Barcelona, Gran Via 585, 08007, Barcelona, Spain

The problem of training classifiers only with target data arises in many applications where non-target data are too costly, difficult to obtain, or not available at all. Several one-class classification methods have been presented to solve this problem, but most of the methods are highly sensitive to the presence of outliers in the target class. Ensemble methods have therefore been proposed as a powerful way to improve the classification performance of binary/multi-class learning algorithms by introducing diversity into classifiers. However, their application to one-class classification has been rather limited. In this paper, we present a new ensemble method based on a non-parametric weighted bagging strategy for one-class classification, to improve accuracy in the presence of outliers. While the standard bagging strategy assumes a uniform data distribution, the method we propose here estimates a probability density based on a forest structure of the data. This assumption allows the estimation of data distribution from the computation of simple univariate and bivariate kernel densities. Experiments using original and noisy versions of 20 different datasets show that bagging ensemble methods applied to different one-class classifiers outperform base one-class classification methods. Moreover, we show that, in noisy versions of the datasets, the non-parametric weighted bagging strategy we propose outperforms the classical bagging strategy in a statistically significant way.

Keywords: One-class Classifier and Ensemble Methods and Bagging and Outliers

1. Introduction

One-class classification is a special case of machine learning problems [1]. In contrast to conventional classification problems, one-class classification tries to distinguish one class of data, called the *target class*, from all other possible data, called the *outlier class*, without any information about the outlier class. The most common strategy for one-class classification is to build a description of the target class from a training dataset, in order to detect any sample which does not resemble the set of training examples.

One of the main applications of one-class classification methods is outlier detection [2,3,4,5,6,7]. Outliers are usually samples that have exceptionally large or small feature values in comparison to the other target-class samples.

Another situation in which one-class classification can be applied is when one of the classes is well defined, but the other class is undersampled and/or extremely

heterogeneous. This is a classic scenario in some medical problems [8], where samples of the healthy class have little variability and are easily obtained, while patient samples are more difficult to obtain and their intraclass variability can be high.

Finally, one-class classification can also be useful when comparing two datasets [9]. For example, consider the case of a target class whose statistical properties change over time in unforeseen ways. This can cause problems, since the classifier becomes less accurate as time passes. One-class classification can be used in such a scenario to check data stability.

There are three main approaches for solving the one-class classification problem: density, boundary and reconstruction methods. In general terms, density methods are used when a large number of data are available, while boundary and reconstruction methods are used when the aim is for classifiers to be learned despite there being no large set of data. It has been shown that the methods can be applied to heterogeneous datasets with well-defined labels and a small number of outliers in the target class.

However, the problem of dealing with one-class classifiers on under-sampled and *contaminated* datasets, i.e data with the presence of a large number of outliers or ill-defined labels, is still a challenge. Nowadays, many real problems deal with under-sampled and contaminated datasets. An instance of such a problem is the classifying of healthy subjects and patients with intestinal dysfunction from the INTES dataset of endoscopy videos, as presented by Malagelada et al. [10]. In this particular case, patients with intestinal dysfunctions are difficult to find and costly (undersampled dataset) and furthermore, the set of patients presents a high intraclass variability (heterogeneous class). That is why it is convenient to pose this problem as a one-class classification problem. This real problem presents another difficulty: although the target class (healthy subjects) is an homogeneous class, it may be contaminated for several different reasons: 1) diagnostic test errors; 2) healthy subjects with abnormal behavior; or 3) patients suffering from other kinds of dysfunction without a positive diagnosis.

Ensemble methods [11] have been proposed as an effective way to improve the performance of classifiers in two-class and multi-class settings. Such methods are characterized by the production of a set of various classifiers that are used to classify new samples by a voting combination rule. However, their application to one-class classifiers has been rather limited [12,13,14].

The main strategy within ensemble methodology is to combine the output of several classifiers by following a specific combination rule. Initially, the combination rule for the ensemble was based solely on averaging the output of each classifier; but more sophisticated algorithms have been proposed over recent years. Two key issues have been identified when designing a classifier combination method [15,16]: accuracy and diversity.

Constructing a diverse ensemble in which each classifier is as different as possible, while still maintaining consistency with the training data, is known to be

a theoretically important property of these methods [17]. However, the creation of diverse ensembles is an issue that has not been solved. Kuncheva et al. [11] noted that ensemble methods which focus on creating diversity in heuristic ways seem to yield very good results; however, methods that measure diversity and use that measurement explicitly in the process of creating the ensemble, apparently do not benefit from the same improvement.

There is neither a strict definition nor an explicit measure for diversity, but in the literature we can find three main ways of creating diversity: the first consists of considering a different pool of samples for each classifier in the ensemble; the second consists of considering different subsets of features for each classifier; and the last one consists of using different classifier methods to build each member of the ensemble. The most popular approaches are the first and the second.

Bagging [18] is a method that is commonly adopted to generate different pools of samples for each classifier. It trains each classifier in the ensemble with a re-sampled version of the training set. The method is useful for unstable classifiers in which small changes in the training set cause large changes in the estimated boundary [19]. Weighted bagging [13] is an extension of the original bagging method which establishes a different weight for each example to be included in the bootstrapped samples. In order to compute the weight of each sample, the methods mentioned above estimate a density function using an iterative method and assuming a normal data distribution. Recently, the term *weighted bagging* has also been used to describe a new bagging strategy for classification and regression in two-class problems [20].

Boosting is another way to construct diverse classifier ensembles by varying the inputs. Boosting has been proposed and refined in a series of works by Freund and Schapire [21], leading to its most successful implementation to date called AdaBoost (Adaptive Boosting). While bagging relies on random and independent changes in the training data, boosting changes the training data to direct further classifiers toward more difficult cases. In this way, desirable diversity is induced in the classifier ensemble.

In this paper, we propose a new hybrid weighted bagging ensemble method based on a non-parametric density estimation method that combines the benefits of boundary methods and density estimation. The non-parametric density estimation method is specially designed so as to be robust when estimating high-dimensional data distributions by assuming that the density function can be well represented by a forest graphical model. Once this model has been estimated, it can be used to generate several weighted bootstrap samples of the data. Then, we can build a set of classifiers and use their votes to classify new samples.

We analyze the general benefits of using bagging ensemble methods for different boundary one-class classification methods in the presence of outliers. In particular, we compare bagging and the new weighted bagging ensemble methods on the INTES dataset and 19 different datasets artificially contaminated with outliers. We show that using bagging and weighted bagging ensemble methods for one-class

classification can dramatically improve the classification results, especially when datasets are contaminated with outliers.

The rest of the paper is organized as follows: Section 2 reviews some of the most useful one-class classifier methods; Section 3 introduces our new method for combining multiple one-class classifiers; Section 4 presents the experimental results; and Section 5 rounds off the paper with our conclusions and prospects for future work.

2. Background: One-Class Classifiers

There are three main approaches for solving the one-class classification problem: density, boundary and reconstruction methods.

Density methods aim to estimate the probability density of the training set and fix a threshold value on that density function. For instance, density can be estimated by using a Gaussian model [22] or a mixture of Gaussians [23]. A Parzen estimator method [24,25] can be used if a non-parametric method is needed. All these methods yield excellent results when the probability model fits the data and the sample size is sufficient. However, the methods do not provide good results in high-dimensional spaces due to the lack of enough samples and the difficulty in estimating a reliable density function [1].

Boundary methods aim to estimate directly the boundary that encloses the target class samples, which, in some cases, can be seen as a simpler problem than estimating the probability density function. The k-center method [26] can be used to estimate the boundary by using a set of multi-dimensional spheres with minimal radius. Schölkopf et al. [27] introduced the One-Class Support Vector Machine (OCSVM), which uses a hyperplane to separate target samples from the origin with the maximal margin. Later, the Support Vector Data Description (SVDD) method was introduced by Tax et al. [28]. That method can be seen as an evolution of OCSVM and it consists of determining the smallest hypersphere that contains the training data. Recently, a graph-based one-class classifier method, the Minimum Spanning Tree Class Descriptor (MST_CD), was proposed by Juszczak et al. [29]. In that method, a graph-based description of the target class is calculated by using the minimum spanning tree (MST) [30], and the classification rule is based on the distance to the closest edge of the MST. Finally, Nearest Neighbor Data Description (NNDD) [23] is a boundary method which extends the nearest neighbor density estimator. In general, the main drawback of boundary methods is that they depend strongly on the metric used, so they tend to be very sensitive to the scaling of features and to the presence of outliers in the training set.

Reconstruction methods make assumptions about the clustering characteristics of the data or about their distribution in subspaces. Then, a set of prototypes or subspaces are defined and a reconstruction error is minimized. For example, K-mean clustering [22] and Learning Vector Quantization (LVQ) [31] are representative methods of this class that assume that the data are clustered and can be represented

by a few prototypes.

Of the different one-class classifier strategies, here we focus on boundary methods, since they have been shown to be appropriate for both high- and low-dimensional spaces. In the following subsections, we review three of the most popular boundary-based one-class classifiers: NNDD, SVDD and MST_CD.

From now on, we consider $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_n]^T$ as an $n \times d$ matrix that corresponds to the training dataset (see notation[a]); n is the number of examples and $\mathbf{x}_i = [x_1, ..., x_d]^T$ is a data example described by d features.

2.1. *Nearest Neighbor Data Description (NNDD)*

The NNDD method is an extension of the nearest neighbor density estimator [23]. It avoids estimation of the density function of the data and uses only the distances to the first nearest neighbor. The function that describes the distance of a sample $\mathbf{x}$ to the boundary is given by:

$$f_{NN}(\mathbf{x}) = \frac{V(||\mathbf{x} - NN^{tr}(\mathbf{x})||)}{V(||NN^{tr}(\mathbf{x}) - NN^{tr}(NN^{tr}(\mathbf{x}))||)} \tag{1}$$

where $NN^{tr}(\mathbf{x})$ represents the nearest neighbor to sample $\mathbf{x}$ in the training dataset and $V(r)$ is the volume of the hypersphere of radius r.

2.2. *Support Vector Data Descriptor (SVDD)*

The SVDD method consists of building a shaped boundary around the training data $\mathbf{X}$. In particular, it defines a hypersphere of radius r and center $\mathbf{a}$ which encloses the maximum number of samples possible while having the minimum volume. This requirement can be stated as a minimization problem:

$$\min r^2 + C \sum_{i=1}^{n} \xi_i \quad \text{with} \quad ||\mathbf{x}'_i - \mathbf{a}|| \leq r^2 + \xi_i \tag{2}$$

where ξ_i are the slack variables introduced to allow for the presence of outliers in the training data and C is the trade-off parameter that controls how much the slack variables are penalized.

Equation (2) is solved using the Lagrange multipliers approach, which transforms the problem to one of the maximization of the following function F with

[a]Bold capital letters denote matrices, bold lower-case letters denote column vectors and non-bold letters denote scalar variables. $\mathbf{x}_j$ and $\mathbf{x}^T$ represent the transpose of vector $\mathbf{x}$; $||\mathbf{x}||$ designates the Euclidean norm of vector $\mathbf{x}$; and $\mathbf{x}_i \cdot \mathbf{x}_j$ denotes the inner product of vectors $\mathbf{x}_i$.

respect to the Lagrange multipliers $\alpha = (\alpha_1, \ldots, \alpha_n)$:

$$F = \sum_{i=1}^{n} \alpha_i (\mathbf{x}_i \cdot \mathbf{x}_i) - \sum_{i,j=1}^{n} \alpha_i \alpha_j (\mathbf{x}_i \cdot \mathbf{x}_j)$$

$$\text{with} \quad 0 \leq \alpha_i \leq C, \forall i = 1, \ldots, n \text{ and } \sum_{i=1}^{n} \alpha_i = 1.$$

Finally, the function describing the distance between a given sample $\mathbf{x}$ and the estimated boundary is given by:

$$f(\mathbf{x}, \alpha) = (\mathbf{x} \cdot \mathbf{x}) - 2 \sum_{i=1}^{n} \alpha_i (\mathbf{x} \cdot \mathbf{x}_i) + \sum_{i,j=1}^{n} \alpha_i \alpha_j (\mathbf{x}_i \cdot \mathbf{x}_j). \tag{3}$$

When a hypersphere is not a good fit for the boundary estimation of the data in the original representation space, the inner product can be generalized by a kernel function where a mapping of the data to a new feature space is implicitly applied.

2.3. *Minimum Spanning Tree Class Descriptor (MST_CD)*

This classifier method is based on a graph that represents the training data. The graph is computed to capture the structure of the data. In particular, the MST is used.

The process of training the classifier is reduced to solving the standard MST problem for a dataset. Several algorithms have been proposed for finding the MST in polynomial time; with Prim's [32] and Kruskal's [33] the most popular. The MST defines a graph without loops which connects all the vertices, such that the total length of the edges is minimal. The length of an edge that connects two target samples $\mathbf{x}_i$ and $\mathbf{x}_j$ is usually measured by the Euclidean distance between the nodes.

This classifier method does not consider only vertices but also graph edges for classifying, thereby providing a much richer representation of the data. The classification of a new object, $\mathbf{x}$, is based on the distance to the nearest vertex or edge.

The projection of $\mathbf{x}$ onto an edge defined by the vertices $\{\mathbf{x}_i, \mathbf{x}_j\}$ is:

$$p_{e_{ij}}(\mathbf{x}) = \mathbf{x}_i + \frac{(\mathbf{x}_j - \mathbf{x}_i) \cdot (\mathbf{x} - \mathbf{x}_i)}{(\|\mathbf{x}_j - \mathbf{x}_i\|)} (\mathbf{x}_j - \mathbf{x}_i).$$

If the projection $p_{e_{ij}}$ lies between $\mathbf{x}_i$ and $\mathbf{x}_j$, then the distance $d(\mathbf{x} \| e_{ij})$ between $\mathbf{x}$ and the edge e_{ij} is computed as the Euclidean distance. Otherwise, $d(\mathbf{x} \| e_{ij})$ is derived as the shortest Euclidean distance to one of the vertices $\{\mathbf{x}_i, \mathbf{x}_j\}$

The distance of the new object, $\mathbf{x}$, to the target class is defined as the minimum distance to the set of $(n-1)$ edges of the MST:

$$D_{\text{MST\_CD}}(\mathbf{x}, \mathbf{X}) = \min_{e_{ij} \in MST} d(\mathbf{x} \| e_{ij}). \tag{4}$$

The decision as to whether $\mathbf{x}$ belongs to the target or non-target class is based on a threshold, θ, set by the distances D_{MST_CD}. This threshold cannot be derived as an error in the training set, since the distance of all target objects is equal to zero according to the definition of the distance. Therefore, θ is determined by a quantile function of the distribution of the edge weights in the given MST. The quantile function is defined as:

$$\theta \equiv \vartheta_\sigma(\tilde{\mathbf{e}}) = \|e_{([\sigma n])}\| \tag{5}$$

where $\tilde{\mathbf{e}} = (e_{(1)}, e_{(2)}, ..., e_{(n)})$ is the sorted sequence of scalar weights of the edges in the MST, such that: $\|e_{(1)}\| \leq \|e_{(2)}\| \leq, ..., \leq \|e_{(n)}\|$; $[a]$ returns the nearest integer of a; and $\sigma \in [0, 1]$. Thus, $\vartheta_0(\tilde{\mathbf{e}})$ returns the minimum distance between two edges of the MST, $\vartheta_1(\tilde{\mathbf{e}})$ is maximum, and $\vartheta_{0.5}(\tilde{\mathbf{e}})$ is the median weight of the edges.

3. Bagging Strategy for One-Class Classifier Methods

In this work, we focus on one-class classification problems with outlying data. Since many real problems are undersampled problems, as is the case with the INTES dataset, we focus on boundary-based one-class classifiers. In this context, boundary one-class classifiers are unstable. This means that small changes in data result in large changes in the estimated boundary. In order to improve the performance of these one-class classifiers when dealing with a contaminated dataset, we propose the use of bagging-based ensemble methods. The bagging strategy has been shown to yield good results in binary/multi-class classifications, when it is applied on unstable classifiers [34].

It must be pointed out that not all ensemble strategies that have been proposed for two-class and multi-class problems can be used directly in a one-class scenario. The acceptance/rejection decision for all one-class classifiers depends on a threshold that is fixed by the estimated probability or distance. This threshold is usually fixed by setting the percentage of target examples that should be accepted. Since we have no information regarding the non-target class, methodologies such as boosting, which are based on measuring the empirical error in all classes, cannot be applied to one-class classification.

3.1. *Bagging*

Bagging generates L new training datasets, $\mathbf{X}'_\ell$, of size n' ($n' \leq n$) by sampling from $\mathbf{X}$ uniformly and with replacement. Some instances of $\mathbf{X}$ may not appear in $\mathbf{X}'_\ell$, and some may appear duplicated. After the construction of the L different (and varied) classifiers, every new sample is classified by computing the majority vote from the ensemble.

As can be seen, bagging is a simple strategy that can be used in any learning scenario: two-class, multi-class or even one-class classification. In classical bagging, uniform sampling is applied, and for this reason all the samples have the same probability of being present in the training set of each classifier of the ensemble. If

all the samples have the same probability, then outliers are likely to be included in most of the bootstrap samples. This characteristic, which has been shown to be beneficial for clean datasets, can be a problem for datasets with outliers, as we will show in Section 4. For this reason, we propose the use of a weighted bagging strategy, which overcomes the presence of outliers by defining a sampling policy that minimizes the probability of an outlier being present in the bootstrap samples.

3.2. *Non-Parametric Weighted Bagging for One-Class Classification*

In order to minimize the probability of an outlier being present in the bootstrap samples, the density function of the data can be used in the sampling procedure. This idea was proposed by Shieh and Kamm [13] to overcome the problem of training OCSVMs in the presence of outliers. The method weights points based on how close they are to the target class, using a kernel density estimator. In this way, it assigns lower probability weights on outliers (points far from the target class). Later, Seguí et al. [35] proposed the use of weighted bagging to combine MST_CD one-class classifiers. Their preliminary results showed that weighted bagging can also reduce the influence of outliers in this boundary method.

The success of weighted bagging is directly related to how well the data density is estimated. The method presented by Shieh and Kamm [13] is suitable for problems where a kernel density estimator can be directly applied to the data samples; but it is severely limited by the capacity of kernel density estimators to represent high-dimensional data.

In order to avoid this problem, we propose the application of a non-parametric density estimator that was proposed recently [36]. This method, called the Forest Density Estimation method, is a non-parametric method specially designed to compute the optimal density for under-sampled data in high-dimensional spaces.

The most common way of dealing with the problem of under-sampled data is to impose assumptions on the data distribution. The Forest Density Estimation method uses an alternative approach to deal with this problem: it restricts feature dependencies to simplify the estimation of the unknown data distribution in an optimal way. Feature dependencies are restricted to those that can be represented using a graph structure of features where edges between conditionally independent features are removed and not considered for the density estimation. In this way, instead of computing a single multivariate density function of the data, the estimation problem is transformed into a series of simpler problems that compute several univariate and bivariate marginal densities.

Let $\mathbf{x}_i = [x_1, ..., x_d]^T$ be a data sample described by d features. Let $\mathbf{X}$ be a dataset composed of n data samples. Let $\mathcal{G} = (V, E)$ be an acyclic graph with d vertices that represent the data feature x_i and the $e \ll d^2$ edges connecting the pairs of features (x_i, x_j) that are estimated to be important in order to properly approximate $p(\mathbf{x})$. The density estimation $p(\mathbf{x})$ is then obtained using the following

expression:

$$p(\mathbf{x}) = \prod_{(i,j)\in E} \frac{p(x_i, x_j)}{p(x_i)p(x_j)} \prod_{k=1}^{d} p(x_k), \tag{6}$$

where $p(x_i)$ is the univariate marginal density of the variable x_i and $p(x_i, x_j)$ is the bivariate marginal density of features x_i and x_j.

In order to apply this model, we must solve the following problems: (1) How to decide which pairs of features (x_i, x_j) are important in order to properly approximate $p(\mathbf{x})$; (2) how to estimate $p(x_i)$ and $p(x_i, x_j)$; and (3) how to prevent overfitting when working with a small sample. Following Gupta et al. [36], we propose these solutions:

(1) Finding the best forest structure for $p(\mathbf{x})$ can be recast as the problem of finding the maximum weight spanning forest for a weighted graph, where the weight of the edge connecting x_i and x_j is the mutual information between those variables. This approach was originally proposed by Chow and Liu [37] back in 1968. The method proceeds by iteratively adding an edge connecting the pair of variables with maximum mutual information from all all pairs not yet visited by the algorithm. The method can be stopped at any iteration, k, to get a k-edged weighted forest.

(2) The univariate marginal density of x_i can be computed using a kernel density estimation method. Given an evaluation point x_i, its univariate kernel density estimate based on the observations $x_i^{(s)}$ is:

$$p(x_i) = \frac{1}{n} \sum_{s=1}^{n} \frac{1}{h_1} K\left(\frac{x_i^{(s)} - x_i}{h_1}\right) \tag{7}$$

where h_1 is a bandwidth parameter tuned for optimal estimation [38]. The bivariate marginal density of two features x_i and x_j can also be computed using a two-dimensional kernel density estimation method:

$$p(x_i, x_j) = \frac{1}{n} \sum_{s=1}^{n} \frac{1}{h_{2i} h_{2j}} K\left(\frac{x_i^{(s)} - x_i}{h_{2i}}\right) K\left(\frac{x_j^{(s)} - x_j}{h_{2j}}\right) \tag{8}$$

where h_{2i} and h_{2j} are bandwidth parameters tuned for optimal estimation [38].

(3) If the estimated graph $\mathcal{G}$ is a full connected tree, it may lead to overfitting. In order to reduce this problem, the graph $\mathcal{G}$ is pruned to $k \leq d-1$ edges, using the following procedure:

   (a) Randomly divide the training set into two sets, D_1 and D_2, of sizes n_1 and n_2, where $n_1 = n_2$ and $n_1 + n_2 = n$.
   (b) Use D_1 to estimate the univariate, $p_{n_1}(x_i)$, and bivariate, $p_{n_1}(x_i, x_j)$, density functions.
   (c) Compute the mutual information matrix $\hat{I}(x_i, x_j)$.
   (d) Use $\hat{I}(x_i, x_j)$ to compute the maximum weight spanning tree $\mathcal{G}_{n_1}^{d-1}$.

(e) Use D_2 to prune the graph $\mathcal{G}_{n_1}^{d-1}$ and find the forest $\mathcal{G}_{n_1}^{k}$ with k edges by maximizing the following equation:

$$\arg\max_{k\in\{0,\ldots,d-1\}} \frac{1}{n_2} \sum_{s\in D_2} log\Big(\prod_{(i,j)\in E^{(k)}} \frac{p_{n_1}(x_i^{(s)}, x_j^{(s)})}{p_{n_1}(x_i^{(s)}) p_{n_1}(x_j^{(s)})} \Big) \quad (9)$$

where $E^{(k)}$ corresponds to the set of edges of the forest $\mathcal{G}_{n_1}^{k}$.

Once the forest is pruned to k edges, the forest density function, which determines the weight of each sample to be selected in a bootstrap sample of the ensemble, is defined as:

$$p(\mathbf{x}) = \prod_{(i,j)\in E^{(k)}} \frac{p_{n_1}(x_i, x_j)}{p_{n_1}(x_i) p_{n_1}(x_j)} \prod_k p_{n_1}(x_k). \quad (10)$$

The proposed method can be used to build a density function for high-dimensional data while computing only univariate and bivariate kernel density estimates of the features, and it can readily be used to bias the sampling process during bagging, as shown in Algorithm 1. In that algorithm, the *base model* is not specified, but there is no special restrictions on it. Hence, non-parametric weighted bagging can be seen as a wrapper method that can be used with any one-class classifier. In our experiments, we have considered NNDD, SVDD and MST_CD classifiers as alternative *base models*.

Figure 1 represents the application of the Forest Density Estimation method to one of the databases considered.

Algorithm 1 Non-parametric weighted bagging.

Require: A dataset $\mathbf{X}$ with n samples $\mathbf{x_i^{(s)}} = [x_1^{(s)}, ..., x_d^{(s)}]^T$ described by d features.

1: Compute the optimal k_1-edged weighted forest $\mathcal{G}^{k_1} = (V, E^{(k_1)})$, $k_1 = d-1$, for representing $p(\mathbf{x})$ using the Chow Liu algorithm [37].
2: Based on the data samples, compute $p(x_i)$ for every x_i in V and $p(x_i, x_j)$ for every pair (x_i, x_j) in $E^{(k_1)}$.
3: Build $\mathcal{G}^{k_2} = (V, E^{(k_2)})$ by pruning $\mathcal{G}^{k_1}$ to $k_2 < d-1$ edges.
4: Define $p(\mathbf{x}) = \prod_{(i,j)\in E^{(k_2)}} \frac{p_{n_1}(x_i,x_j)}{p_{n_1}(x_i)p_{n_1}(x_j)} \prod_k p_{n_1}(x_k)$.
5: Compute a weight $w_i = p(\mathbf{x_i^{(s)}})$ for every sample.
6: **for** $l = 1$ to L **do**
7: Select N samples from the training set by performing weighted sampling with replacement.
8: Train a *base model* M_l on the samples.
9: **end for**
10: **return** $M_1, \ldots, M_L$.

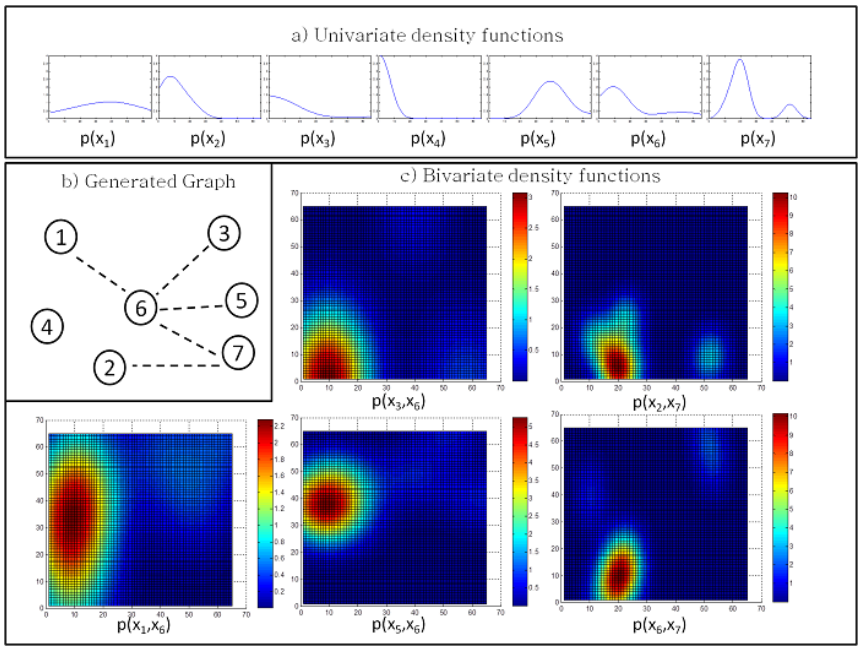


Fig. 1. Forest density estimation for E. coli dataset: a) estimated univariate density functions $p(x_i)$; b) estimated forest structure; c) estimated bivariate density functions $p(x_i, x_j)$ corresponding to the connected features of the forest structure.

4. Results

In this section, we show the results of bagging and weighted bagging ensemble methods using one-class classifiers in original and noisy versions of 20 datasets.

4.1. *Datasets*

The experiments were performed using datasets obtained from 20 real databases. All of them were obtained, as detailed below, from the UCI repository [39], with the exception of the INTES dataset:

- INTES dataset. This dataset, which was the initial motivation for our work, is a standard dataset which contains a set of 118 samples of 19 features extracted from wireless capsule endoscopy videos of healthy subjects and patients with intestinal motility disorders [10]. It has the following particularities:

  (1) Both the symptoms and the extracted features from the wireless capsule endoscopy that the patients present are highly heterogeneous.
  (2) We refer to the set of volunteers as healthy subjects and we call "patients" those subjects who returned a negative motility test result from manometric devices [40].

Table 1. Datasets.

Dataset (target class)	Dimensions	Target samples	Outlier samples
Liver (present)	6	145	200
Liver (absent)	6	200	145
Iris (versicolor)	4	50	100
Iris (setosa)	4	50	100
Biomed (healthy)	5	127	67
Biomed (patient)	5	67	127
Heartstatlog (absent)	13	150	120
Heartstatlog (present)	13	120	150
E. coli (periplasm)	7	52	284
Ionosphere (good)	34	225	126
Hepatitis (normal)	19	123	32
Housing (<35)	13	48	458
Imports (low risk)	25	88	71
Vehicle (Opel)	18	212	634
Vehicle (Saab)	18	217	629
Sonar (rocks)	60	97	111
Cancer (wpbc ret)	33	47	151
Arrhythmia (normal)	278	237	183
Breast(malignant)	9	458	241
INTES (healthy)	19	105	19

(3) It is expected that a small percentage (2% - 5%) of healthy subjects would present abnormal behavior, i.e., an outlier sample.
(4) The motility test has a considerable percentage of error [41].

These particularities taken together suggest to us that this is a case of a one-class classification problem, where the target (healthy) class is contaminated with outliers.

- UCI datasets. We randomly selected 19 datasets from the UCI repository [39] in order to extensively test the different methods we aim to compare.

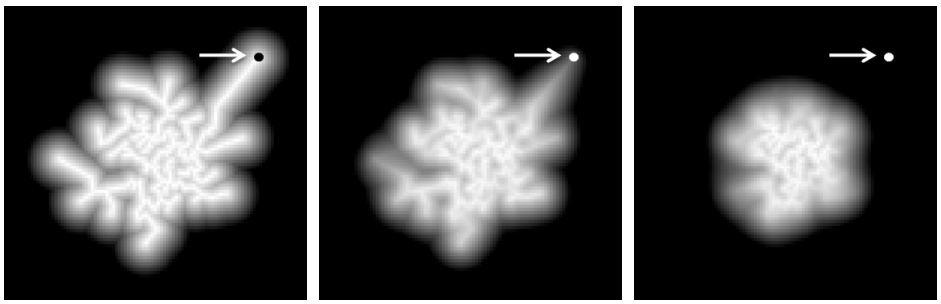
Table 1 contains a list of the datasets considered, with the corresponding number of features and number of examples of the *target* and *outlier* class.

4.2. *Performance Evaluation*

All the experiments were performed using MATLAB on a standard personal computer. The MATLAB version of the MST_CD, SVDD and NNDD methods, publicly available in the *DD_TOOL* toolbox [42] [b] of the `prtools` library, were used. For all the methods, the default parameters were used: (1) MST_CD is computed over the full dataset; (2) $SVDD$: $sigma = 5$; (3) $NNDD$: $k = sqrt(d)$ where d is the data dimensionality.

In order to evaluate the performance of the ensemble one-class classifier methods, the area under the receiver operator characteristics (ROC) curve (AUC) was computed [43]. The AUC measure is the total performance of a classifier integrated over all possible thresholds. By "'thresholds"' we refer to the density/distance to

[b] http://ict.ewi.tudelft.nl/~davidt/dd_tools.html



(a) MST_CD (b) B_MST_CD (c) WB_MST_CD

Fig. 2. Results of the: MST_CD (a); bagging MST_CD (b); and non-parametric weighted bagging MST_CD (c), methods for a synthetic Gaussian dataset plus one outlier (indicated by the arrow). It can be seen that only the weighted bagging method was able to clearly rule out the outlier.

the estimated class. A large AUC value therefore means better performance of the one-class classifier; a value lower than 50% means that the classifier performs worse than random guessing.

All the experiments were performed using 50% of the target class as the training set, and the other 50% of the target class together with the outlier class (samples from other dataset classes) as test set. The experiments were repeated 20 times and the mean value of the trials is presented.

To measure the significance of the results, two different statistical tests were performed. On the one hand, the paired Student t-test (as suggested in [44]) was used to evaluate the statistical significance between pairs of classifiers over a particular dataset. On the other hand, the non-parametric Friedman test [45] was used in order to evaluate the proposed ensemble methods over several datasets. In this latter test, the algorithms are ranked for each dataset according to their performance (with 1 the highest rank and k, the number of methods evaluated, the lowest rank). The average rank over all the datasets is used to evaluate the statistical test. For both tests, the statistical significance level $\alpha = 0.05$ was used.

4.3. *Synthetic Toy Experiment*

A toy experiment using synthetic data was performed in order to illustrate the problem. We trained the MST_CD classifier using a 2D cloud of Gaussian distributed data with one outlier. The estimated boundaries using a base classifier and both ensemble bagging and weighted bagging of MST_CD are reproduced in Figure 2. It can clearly be seen that the outlier sample (indicated by the arrow) present in the training set was enclosed in the estimated boundary for the original and ensemble bagging MST_CD; however, ensemble weighted bagging MST_CD rejected this sample.

4.4. *Ensemble of One-Class Classifiers*

In this second experiment, we evaluated bagging and the non-parametric weighted bagging for one-class classification. In particular, we used MST_CD, SVDD and NNDD on the 20 original datasets. The number of classifiers in the ensembles was set to $L = 100$.

Table 2 presents the results for the three one-class classification methods described in Section 2. Bold numbers in the table correspond to the best result or a non-significantly worse than best result, according to the Student t-test performed for each dataset and one-class method. In addition, the last row of the table presents the number of cases where the methods obtain the best or a non-significantly worse than best result (according to the paired Student t-test). As can be appreciated from the table, the non-parametric weighted bagging method always delivered either the best result or a non-significant worse than best result using MST_CD and NNND, and only in 3 cases out of 20 did it get a statistically worse result than the best result using SVDD.

Finally, Table 3 presents the rankings from the Friedman test. The best and not statistically worse than best results appear in bold. As can be observed, in all cases the ensemble bagging and non-parametric weighted bagging obtain better rankings than base one-class classifier methods, showing a statistically significant improvement for MST_CD and NNDD.

Table 2. Classification results using original versions of the datasets. The mean AUC measure from 20-hold out repetitions, is presented for the score of each method. The best and not significantly worse than best results appear in bold. The last row shows the number of times that each method is considered the best, or not worse than the best.

	MST_CD			SVDD			NNDD		
DB (target class)	original	bagging	w. bagging	original	bagging	w. bagging	original	bagging	w. bagging
Liver (present)	**58.67**	**59.41**	**59.36**	**54.46**	**54.31**	**54.53**	**57.71**	**58.85**	**58.92**
Liver (absent)	**51.13**	**51.09**	**51.17**	**52.86**	**51.97**	50.20	**47.90**	**49.17**	**49.30**
Iris (versicolor)	97.60	**98.06**	**98.37**	**98.30**	**98.26**	**98.27**	**98.14**	**97.99**	**98.19**
Iris (setosa)	**99.50**	**99.50**	**99.50**	**99.50**	**99.50**	**99.50**	**99.50**	**99.50**	**99.50**
Biomed (healthy)	**90.32**	**90.68**	**90.66**	86.53	87.43	**88.33**	**90.85**	**90.96**	**90.92**
Biomed (patient)	**44.51**	**46.30**	**45.94**	**75.79**	**73.91**	69.76	**47.43**	**45.05**	**44.74**
Heartstatlog (absent)	79.47	81.44	**82.72**	73.03	75.73	**78.68**	81.03	80.52	**81.11**
Heartstatlog (present)	66.50	**69.09**	**70.25**	**70.68**	**71.06**	**71.55**	68.40	**67.45**	**68.29**
E. coli (periplasm)	88.91	**92.89**	**93.47**	88.56	**92.44**	**93.48**	92.66	**91.61**	**92.20**
Ionosphere (good)	**96.67**	**97.01**	**96.91**	**97.21**	**97.20**	**97.09**	**96.69**	**96.67**	**96.75**
Hepatitis (normal)	79.15	**81.23**	**82.27**	76.85	78.56	**80.41**	80.74	**80.72**	**80.96**
Housing (<35)	**84.87**	**85.55**	**85.21**	**82.29**	**83.25**	**83.76**	**84.86**	**85.25**	**85.45**
Imports (low risk)	**79.57**	**78.81**	**77.53**	**74.12**	**74.94**	**74.27**	**79.31**	**80.05**	**79.92**
Vehicle (Opel)	75.61	**76.93**	**77.01**	67.66	**68.10**	**68.84**	76.08	**76.12**	**76.27**
Vehicle (Saab)	76.38	**77.83**	**77.31**	**77.02**	**76.20**	74.75	76.49	**76.43**	**76.46**
Sonar (rocks)	**71.17**	**71.01**	**70.33**	**68.77**	**69.07**	**68.90**	**71.03**	**71.61**	**71.49**
Cancer (wpbc ret)	**59.39**	**61.12**	**61.13**	**57.57**	**58.74**	**58.72**	**58.96**	**60.59**	**60.57**
Arrhythmia (normal)	**79.09**	**79.64**	**79.63**	**79.25**	**79.42**	**79.44**	**78.33**	**77.86**	**77.82**
Breast(malignant)	**98.68**	**98.74**	**98.81**	**98.78**	**98.76**	**98.84**	**98.71**	**98.72**	**98.72**
INTES (healthy)	90.29	**91.48**	**91.84**	**90.63**	**91.34**	**91.62**	90.59	**90.54**	**90.72**
#Statistical wins	12	19	20	15	17	17	13	19	20

Table 3. Average rank from the Friedman test using original versions of the datasets. Best and not significantly worse than best results appear in bold.

	Original	Bagging	Weighted Bagging
MST_CD	2.70	**1.70**	**1.60**
SVDD	**2.35**	**2.00**	**1.75**
NNDD	2.60	**1.65**	**1.65**

4.5. *Ensembles of One-Class Classifiers in Noisy Data*

In this third experiment, we evaluated the behavior of original one-class classifiers (MST_CD, SVDD, NNDD), bagging and the non-parametric weighted bagging in a noisy version of the 19 datasets from the UCI repository and also the INTES dataset. The noisy version of datasets was obtained by swapping a percentage of the samples from the target class for samples from the outlier class. The experiment was repeated twice: first, swapping 10% of the target class samples with outliers; and second, swapping 25%.

Tables 4 and 6 show the results of this experiment using 10% and 25% of outliers in the target class, respectively. First of all, note that if we compare the results using the original data (Table 2) and the noisy versions of the data (Table 4 and 6), the accuracy of the classifiers is clearly reduced as more noise is added to target class. However, this reduction is not the same for all the datasets. For example, it can be observed that in some datasets (iris and biomed) the AUC is reduced by more than 10%, while in other datasets (cancer or heart) the reduction is only 2% or 3%. This is due to the different data distributions and/or dimensionality of the databases.

Additionally, we can observe in Tables 4 and 6, that the proposed non-parametric weighted bagging outperforms the results of the base classifier and the classical bagging strategy in most of the datasets. Regarding the result of the Student t-test, we observe that, in most of the cases, the non-parametric weighted bagging strategy obtains the best result or a non-significantly worse than best result (bold numbers in the table).

Finally, the results from the Friedman statistical tests are displayed in Tables 5 and 7. They show that forest weighted bagging is statistically better than all base classifiers, and also outperforms the results of ensemble bagging in MST_CD and NNDD.

Table 4. Classification results using noisy versions of the datasets (10% of outliers in the target class). The mean AUC measure, from 20-hold out repetitions, is presented for the score of each method. The best and not significantly worse than best result appear in bold. The last row shows the number of times that each method is considered the best or not worse than the best.

	MST_CD			SVDD			NNDD		
DB (target class)	original	bagging	w. bagging	original	bagging	w. bagging	original	bagging	w. bagging
Liver (present)	**55.94**	**57.25**	**57.66**	**51.58**	**52.56**	**53.02**	**56.79**	**56.92**	**57.24**
Liver (absent)	**46.54**	**46.54**	**46.99**	**52.33**	**51.34**	49.80	**46.73**	**46.00**	**46.17**
Iris (versicolor)	81.39	93.16	**95.62**	89.58	**94.60**	**96.18**	93.47	**89.90**	**91.91**
Iris (setosa)	90.04	96.70	**98.36**	84.18	92.00	**96.88**	97.09	**94.89**	**96.14**
Biomed (healthy)	78.89	86.13	**88.33**	66.63	74.67	**82.53**	89.02	**84.20**	**86.53**
Biomed (patient)	**36.90**	**36.96**	**37.52**	**66.91**	**66.65**	**63.48**	**36.76**	**35.86**	**36.39**
Heartstatlog (absent)	68.65	70.73	**74.13**	64.00	67.72	**72.35**	71.20	69.94	**71.10**
Heartstatlog (present)	53.41	**55.84**	**58.36**	**66.51**	**65.32**	**65.02**	55.43	**54.64**	**56.03**
E. coli (periplasm)	82.39	**87.33**	**89.22**	85.62	**88.03**	**89.76**	89.59	**86.06**	**87.31**
Ionosphere (good)	87.59	94.00	**95.00**	93.30	**95.06**	**95.73**	89.65	**88.05**	**88.60**
Hepatitis (normal)	58.87	63.67	**68.67**	65.44	67.82	**71.86**	62.56	**61.09**	**61.77**
Housing (<35)	**80.18**	**80.80**	**79.95**	**72.58**	**73.22**	**73.33**	**80.88**	**81.34**	**81.62**
Imports (low risk)	**67.73**	**66.46**	**66.67**	**65.12**	**65.23**	**65.98**	**68.10**	**68.99**	**69.05**
Vehicle (Opel)	67.22	70.25	**72.24**	59.60	60.45	**62.50**	67.91	**67.61**	**68.18**
Vehicle (Saab)	67.63	**70.57**	**71.35**	**67.60**	**67.80**	**67.46**	68.38	**68.03**	**68.25**
Sonar (rocks)	**63.08**	**63.62**	**63.73**	**61.93**	**63.08**	**63.50**	**63.51**	**64.12**	**64.04**
Cancer (wpbc ret)	**60.25**	**61.25**	**61.35**	**57.88**	**59.16**	**59.12**	**60.04**	**61.56**	**58.36**
Arrhythmia (normal)	72.43	**73.59**	**74.84**	**72.60**	**72.76**	**72.97**	72.11	**71.88**	**71.79**
Breast(malignant)	91.22	97.22	**98.22**	**98.91**	**98.88**	**99.00**	97.35	95.23	**95.78**
INTES (healthy)	63.16	72.69	**81.19**	78.49	**84.11**	**86.09**	66.15	66.10	**66.47**
#Statistical wins	7	11	20	11	15	19	7	17	20

Table 5. Average rank from the Friedman test using noisy versions of the datasets (10% of outliers in the target class). Best and not significantly worse than best results appear in bold.

	Original	Bagging	Weighted Bagging
MST_CD	2.80	2.05	**1.15**
SVDD	2.60	**1.95**	**1.45**
NNDD	2.75	2.05	**1.20**

Table 6. Classification results using noisy versions of the datasets (25% of outliers in the target class). The mean AUC measure, from 20-hold out repetitions, is presented for the score of each method. The best and not significantly worse than best results appear in bold. The last row shows the number of times that each method is considered the best or not worse than the best.

	MST_CD			SVDD			NNDD		
DB (target class)	original	bagging	w. bagging	original	bagging	w. bagging	original	bagging	w. bagging
Liver (present)	52.06	53.36	**54.39**	51.20	**52.27**	**53.11**	55.69	**53.92**	**54.47**
Liver (absent)	37.80	38.80	**40.19**	**49.89**	**49.49**	48.57	43.67	40.29	**40.80**
Iris (versicolor)	68.49	77.92	**86.72**	88.52	90.01	**92.65**	82.51	76.17	**81.79**
Iris (setosa)	54.84	65.92	**74.85**	58.58	59.30	**62.90**	75.99	67.23	**72.89**
Biomed (healthy)	64.64	74.75	**81.18**	56.02	62.39	**71.41**	86.32	76.58	**80.13**
Biomed (patient)	**28.88**	26.89	27.10	**59.02**	**58.69**	56.26	**25.66**	25.02	24.86
Heartstatlog (absent)	55.72	57.41	**61.03**	58.46	60.47	**64.93**	58.36	56.76	**57.75**
Heartstatlog (present)	43.43	44.25	**45.93**	**60.51**	**59.08**	57.60	44.07	43.20	**43.98**
E. coli (periplasm)	73.61	78.48	**80.67**	76.60	**79.46**	**80.73**	82.97	77.56	**79.08**
Ionosphere (good)	72.26	87.50	**89.73**	84.74	87.86	**90.04**	80.07	**73.73**	**75.18**
Hepatitis (normal)	35.55	42.04	**49.72**	57.95	59.94	**63.43**	40.90	37.10	**38.40**
Housing	73.13	**75.01**	**74.82**	57.67	**61.09**	**63.63**	**75.52**	**74.44**	**74.71**
Imports (low risk)	57.00	**60.39**	**61.41**	**65.07**	**65.77**	**65.77**	58.15	**57.77**	**58.08**
Vehicle (Opel)	54.80	57.43	**59.42**	**55.27**	**55.53**	**55.73**	56.04	55.48	**55.97**
Vehicle (Saab)	60.55	**62.79**	**63.65**	**63.75**	**63.29**	62.43	61.14	**60.46**	**60.66**
Sonar (rocks)	54.61	55.04	**56.15**	54.65	56.01	**56.98**	54.96	**54.94**	**54.95**
Cancer (wpbc ret)	**55.80**	**55.21**	**55.25**	**53.25**	**53.35**	**53.39**	**55.99**	**56.04**	**55.99**
Arrhythmia (normal)	62.96	65.04	**67.96**	**63.10**	**63.09**	**63.77**	63.10	63.09	**63.77**
Breast(malignant)	62.96	65.04	**67.96**	**63.10**	**63.09**	**63.77**	95.93	93.59	**93.78**
INTES (healthy)	43.44	54.96	**70.18**	76.41	82.40	**84.91**	76.41	82.40	**84.91**
#Statistical wins	2	4	19	9	12	16	3	7	19

Table 7. Average rank from the Friedman test using noisy versions of the datasets (25% of outliers in the target class). Best and not significantly worse than best results appear in bold.

	Original	Bagging	Weighted Bagging
MST_CD	2.80	2.05	**1.15**
SVDD	2.50	**2.10**	**1.40**
NNDD	2.80	2.05	**1.15**

5. Conclusions

In this paper, we deal with the problem of one-class classification in the presence of outliers. We study the application of ensemble methods for one-class classification, which has been rather limited until now. In particular, we evaluate the classical bagging strategy and propose a new non-parametric weighted bagging strategy in order to improve robustness when dealing with outliers in high-dimensional spaces.

The method we propose estimates the data density by assuming a forest structure of the data and constructs bootstrap samples according to the data density obtained: the higher the estimated density of a sample, the greater its likelihood of being selected for a bootstrap of the ensemble.

An extensive experimental study using original and noisy versions of 20 datasets was performed. The experiments show that the ensemble strategies improve the classification accuracy for different one-class classifier methods and provide increased robustness when dealing with noise. Furthermore, we can infer from the results that the weighted bagging ensemble strategy achieves better results when dealing with original and especially noisy datasets than base one-class classifiers. It also offers a statistically significant improvement in comparison with base one-class classifier and bagging ensemble methods.

In future work we will study new ensemble strategies, other than bagging, to build ensemble methods for one-class classification. Moreover, in the density function computation, new distance metrics can be considered that take advantage of the data distribution.

Acknowledgment

This work was partially supported by the MINECO grants: TIN2009-14404-C02 and TIN2012-38187-C03-01.

Electronic version of an article published as [International Journal of Pattern Recognition and Artificial Intelligence, Vol. 27, Issue 5, 213] [10.1142/S0218001413500146] [copyright World Scientific Publishing Company] [http://www.worldscinet.com/ijprai]

References

1. D. M. J. Tax, One-class classification, Ph.D. thesis, Delft. University of Technology (2001).

2. J. H. M. Janssens, I. Flesch, E. O. Postma, Outlier detection with one-class classifiers from ml and kdd, in: ICMLA '09: Proceedings of the 2009 International Conference on Machine Learning and Applications, 2009, pp. 147–153.
3. V. Hodge, J. Austin, A survey of outlier detection methodologies, Artificial Intelligence Review 22 (2) (2004) 85–126.
4. M. Markou, S. Singh, Novelty detection: a review–part 1: statistical approaches, Signal Processing 83 (12) (2003) 2481 – 2497.
5. S. Harmeling, G. Dornhege, D. M. J. Tax, F. Meinecke, K.-R. M From outliers to prototypes: Ordering data, Neurocomputing 69 (13-15) (2006) 1608 – 1618, blind Source Separation and Independent Component Analysis - Selected papers from the ICA 2004 meeting, Granada, Spain, Blind Source Separation and Independent Component Analysis.
6. F. Angiulli, Prototype-based domain description for one-class classification, IEEE Trans. Pattern Anal. Mach. Intell. 34 (6) (2012) 1131–1144.
7. X. Xu, W. Wang, An incremental gray relational analysis algorithm for multi-class classification and outlier detection, IJPRAI 26 (6).
8. Q. Guo, M. Kelly, C. Graham, Predicting distribution of a new forest disease using one-class svms, in: ICDM '03: Proceedings of the Third IEEE International Conference on Data Mining, IEEE Computer Society, 2003, p. 719.
9. K. Hempstalk, E. Frank, I. H. Witten, One-class classification by combining density and class probability estimation, in: Machine Learning and Knowledge Discovery in Databases, Springer, 2008, pp. 505–519.
10. C. Malagelada, F. D. Iorio, F. Azpiroz, A. Accarino, S. Seguí, P. Radeva, J.-R. Malagelada, New insight into intestinal motor function via noninvasive endoluminal image analysis, Gastroenterology 135 (4) (2008) 1155 – 1162.
11. L. I. Kuncheva, Combining Pattern Classifiers: Methods and Algorithms, 2004.
12. R. Perdisci, G. Gu, W. Lee, Using an ensemble of one-class svm classifiers to harden payload-based anomaly detection systems, in: ICDM '06: Proceedings of the Sixth International Conference on Data Mining, 2006, pp. 488–498.
13. A. D. Shieh, D. F. Kamm, Ensembles of one class support vector machines, in: MCS '09: Proceedings of the 8th International Workshop on Multiple Classifier Systems, 2009, pp. 181–190.
14. V. Cheplygina, D. M. J. Tax, Pruned random subspace method for one-class classifiers, in: MCS, 2011, pp. 96–105.
15. T. G. Dietterich, Ensemble methods in machine learning, in: MCS '00: Proceedings of the First International Workshop on Multiple Classifier Systems, 2000, pp. 1–15.
16. L. K. Hansen, P. Salamon, Neural network ensembles, IEEE Transactions on Pattern Analysis and Machine Intelligence 12 (10) (1990) 993–1001.
17. L. I. Kuncheva, C. J. Whitaker, Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy (2003).
18. L. Breiman, Bagging predictors, Machine Learning 24 (2) (1996) 123–140.
19. E. Bauer, R. Kohavi, An empirical comparison of voting classification algorithms: Bagging, boosting, and variants, Machine Learning 36 (1-2) (1999) 105–139.
20. Q. Yu, Weighted bagging: a modification of adaboost from the perspective of importance sampling, Journal of Applied Statistics 38 (3) (2011) 451–463.
21. Y. Freund, R. E. Schapire, A short introduction to boosting, in: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, 1999, pp. 1401–1406.
22. C. M. Bishop, Neural Networks for Pattern Recognition, 1st Edition, Oxford University Press, USA, 1996.

23. R. O. Duda, P. E. Hart, Pattern Classification and Scene Analysis, John Wiley & Sons Inc, 1973.
24. E. Parzen, On estimation of a probability density function and mode, The Annals of Mathematical Statistics 33 (3) (1962) 1065–1076.
25. D.-Y. Yeung, C. Chow, Parzen-window network intrusion detectors, in: Proceedings of the Sixteenth International Conference on Pattern Recognition, 2002, pp. 385–388.
26. A. Ypma, E. Ypma, R. P. Duin, Support objects for domain approximation, in: International Conference on Artificial Neural Networks, 1998, pp. 719–724.
27. B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, R. C. Williamson, Estimating the support of a high-dimensional distribution, Neural Computation 13 (1999) 2001.
28. D. M. J. Tax, R. P. W. Duin, Support vector data description, Machine Learning 54 (1) (2004) 45–66.
29. P. Juszczak, D. M. J. Tax, E. Pekalska, R. P. W. Duin, Minimum spanning tree based one-class classifier, Neurocomputing 72 (7-9) (2009) 1859–1869.
30. R. L. Graham, P. Hell, On the history of the minimum spanning tree problem, IEEE Annals of the History of Computing 7 (1) (1985) 43–57.
31. H. J. Lee, S. Cho, Application of lvq to novelty detection using outlier training data, Pattern Recognition Letters 27 (13) (2006) 1572–1579.
32. R. C. Prim, Shortest connection networks and some generalizations, Bell System Technology Journal 36 (1957) 1389–1401.
33. J. B. Kruskal, On the shortest spanning subtree of a graph and the traveling salesman problem, Proceedings of the American Mathematical Society 7 (1) (1956) 48–50.
34. L. Breiman, Bagging predictors, Machine learning 24 (2) (1996) 123–140.
35. S. Seguí, L. Igual, J. Vitrià, Weighted bagging for graph based one-class classifiers, in: MCS, 2010, pp. 1–10.
36. A. Gupta, J. D. Lafferty, H. Liu, L. A. Wasserman, M. Xu, Forest density estimation, in: COLT, 2010, pp. 394–406.
37. C. Chow, C. Liu, Approximating discrete probability distributions with dependence trees, Information Theory, IEEE Transactions on 14 (3) (1968) 462–467.
38. A. B. Cybakov, Introduction to nonparametric estimation, Springer, 2009.
39. D. N. A. Asuncion, UCI machine learning repository, http://www.ics.uci.edu/∼mlearn/MLRepository.html (2007).
40. E. M. Quigley, Gastric and small intestinal motility in health and disease, Gastroenterology Clinics of North America 25 (1996) 113–145.
41. D. Wingate, M. Hongo, J. Kellow, G. Lindberg, A. Smout, Disorders of gastrointestinal motility: Towards a new classification1, Journal of Gastroenterology and Hepatology 17 (2002) S1–S14.
42. D. M. J. Tax, Ddtools, the data description toolbox for matlab, version 1.7.3 (Dec 2009).
43. A. P. Bradley, The use of the area under the ron curve in the evaluation of machine learning algorithms, Pattern Recognition 30 (7) (1997) 1145–1159.
44. T. G. Dietterich, Approximate statistical tests for comparing supervised classification learning algorithms, Neural Computation 10 (1998) 1895–1923.
45. J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.