Ensembles of One Class Support Vector Machines

Albert D. Shieh[1] and David F. Kamm[2]

[1] Department of Statistics, Harvard University,
Cambridge, MA 02138, USA
shieh@fas.harvard.edu
[2] Department of Computer Science, Stanford University,
Stanford, CA 94305, USA
dkamm@cs.stanford.edu

Abstract. The one class support vector machine (OCSVM) is a widely used approach to one class classification, the problem of distinguising one class of data from the rest of the feature space. However, even with optimal parameter selection, the OCSVM can be sensitive to overfitting in the presence of noise. Bagging is an ensemble method that can reduce the influence of noise and prevent overfitting. In this paper, we propose a bagging OCSVM using kernel density estimation to decrease the weight given to noise. We demonstrate the improved performance of the bagging OCSVM on both simulated and real world data sets.

1 Introduction

In binary classification, it is typically assumed that training data is available for both classes. However, in some real world applications, there is little or no data available for one of the classes. For example, in the diagnosis of rare diseases, there are many healthy patients but few sick patients to collect data from. Therefore, one class is represented well (positive class), but the other class is not represented well or at all (negative class). One class classification is the partially unsupervised learning problem of training on positive data only and distinguishing the positive class from the rest of the feature space, which comprises all possible negative classes. One class classifiers decide to either accept or reject a given point into the positive class [13].

There are two main types of one class classifiers. Density methods estimate the probability distribution of the positive class and accept points that have a high probability of belonging to the positive class. Boundary methods estimate a boundary that encloses the positive class and accept points inside the boundary. Although density methods provide information about the entire structure of the positive class, boundary methods often perform better since they solve a fundamentally easier problem [13]. In particular, the one class support vector machine (OCSVM), a boundary method adapting the support vector machine (SVM) to one class classification [12,11], has become one of the most widely used one class classifiers.

J.A. Benediktsson, J. Kittler, and F. Roli (Eds.): MCS 2009, LNCS 5519, pp. 181–190, 2009.

Although the OCSVM has been applied widely, the estimated boundary can be sensitive in practice [10]. When the training data is noisy and contains many outliers near or in the negative class, the OCSVM will estimate a large boundary that encloses areas of the feature space where the positive class has low density, resulting in many false positives [6]. This can be highly problematic for many applications where the number of false positives must be kept to a minimum. For example, the accidental diagnosis of a sick patient as healthy may result in death. Optimal parameter selection can tighten the estimated boundary to a certain extent [15], but ideally the OCSVM should be modified in order to decrease the influence of outliers on the estimated boundary.

One way of viewing the sensitivity of the OCSVM to outliers is that the OCSVM is unstable. Although the estimated boundary robustly encloses the positive class, the introduction of outliers can arbitrarily expand the estimated boundary. Bagging is an ensemble method that combines multiple unstable classifiers trained on resampled data to produce an improved classifier [3]. Bagging has been applied to other unstable one class classifiers such as one class decision trees [9] with success. Other types of ensemble methods, such as combining different one class classifiers [14] and training on different sets of features [8], have been applied to the OCSVM with success. However, since the OCSVM has been traditionally viewed as stable, bagging OCSVM has not been explored to our knowledge.

In this paper, we propose a bagging OCSVM using weights determined by kernel density estimation. Our bagging OCSVM combines the advantages of density methods, boundary methods, and bagging. The OCSVM is still used to estimate a boundary, but kernel density estimation is used to find outliers and bagging is used to decrease the influence of outliers on the estimated boundary. Our bagging OCSVM is inspired by the recent success of combining density and boundary methods [5]. We demonstrate that the bagging OCSVM tightens the estimated boundary and reduces false positives on both simulated and real world data sets. The rest of the paper is organized as follows. In Section 2, we describe the OCSVM. In Section 3, we describe the bagging OCSVM. In Section 4, we experimentally compare the normal and bagging OCSVM. In Section 5, we give our final remarks.

2 One Class Support Vector Machines

The OCSVM, as formulated in [11], estimates a set that encloses most of a given sample of m points $\{\mathbf{x}_i\}_{i=1}^m, \mathbf{x}_i \in \mathbb{R}^d$. Each point $\mathbf{x}_i$ is transformed by a map $\phi : \mathbb{R}^d \to \mathcal{K}$ from the feature space $\mathbb{R}^d$ into a high dimensional kernel space $\mathcal{K}$ generated by the kernel $k(\mathbf{x}, \mathbf{y})$. The kernel corresponds to an inner product in the kernel space through $k(\mathbf{x}, \mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle$. The OCSVM finds a hyperplane in the kernel space that separates the data from the origin with maximum margin. If no such hyperplane exists, slack variables ξ_i allow for some points to be within the margin (outliers) and a free parameter $\nu \in [0, 1]$ controls the cost of such violations. In general, ν provides an upper bound on the fraction

of outliers [11]. The hyperplane in the kernel space induces a nonlinear surface in the feature space.

More precisely, the OCSVM solves the quadratic program

$$\begin{aligned} \min_{\mathbf{w},\boldsymbol{\xi},\rho} \ & \frac{1}{2}\|\mathbf{w}\|^2 + \frac{1}{\nu m}\sum_{i=1}^{m}\xi_i - \rho \\ \text{s.t.} \ & \langle \mathbf{w}, \phi(\mathbf{x}_i)\rangle \geq \rho - \xi_i, \ i = 1, \ldots, m \\ & \xi_i \geq 0, \ i = 1, \ldots, m \end{aligned} \tag{1}$$

where $\mathbf{w}$ is the normal vector to the hyperplane and ρ is the margin. The quadratic program can be solved efficiently by sequential minimal optimization (SMO) of its dual form [11]. The decision function

$$g(\mathbf{x}) = \text{sign}(\langle \mathbf{w}, \phi(\mathbf{x})\rangle - \rho) \tag{2}$$

determines whether a given point $\mathbf{x}$ is in (positive) or out (negative) of the estimated set.

In practice, a Gaussian kernel

$$k(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{y}\|^2}{2\sigma^2}\right) \tag{3}$$

with a width parameter σ is used and is the only kernel that has been successfully applied to the OCSVM [11]. The Gaussian kernel maps the data into the same orthant in the kernel space, justifying the principle of separating the data from the origin. The OCSVM with a Gaussian kernel is equivalent to the support vector domain description (SVDD) [12], which finds a hypersphere in the kernel space that encloses the data with minimum volume.

Optimal selection of σ is critical to the performance of the OCSVM, since it controls the shape of the estimated boundary. When σ is not selected properly and distances are inhomogeneous, the estimated boundary is often elongated and encloses large, empty areas of the feature space. Selecting σ using methods such as kernel whitening [15] can tighten the estimated boundary to a certain extent. However, when there are many outliers in the data, tuning σ alone is not sufficient to create a compact estimated boundary [6]. Therefore, the OCSVM should be modified in order to tighten the estimated boundary and exclude outliers.

3 Weighted Bagging

Bagging is a popular ensemble method that trains each classifier in the ensemble on a resampled version of the training data [3]. Given training data of m points $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^m$, a bootstrap sample $\mathcal{Y}$ is generated by drawing m points randomly with replacement from $\mathcal{X}$ with probability weight $w(i)$ for point $\mathbf{x}_i$. In classical bagging, all points are given the same probability weight $w(i) = 1/m$. Each bootstrap sample is the same size as the training data, but some points from the training data can either be left out or repeated multiple times. An ensemble of n

classifiers is formed by training a classifier on n bootstrap samples. New points are typically classified by a simple majority vote of the ensemble.

More precisely, the bagging algorithm is:

1. Generate n bootstrap samples of m points $\{\mathcal{Y}_j\}_{j=1}^n$ from $\mathcal{X}$ with probability weights $w(i)$.
2. For $j = 1, \ldots, n$, train a classifier g_j on the bootstrap sample $\mathcal{Y}_j$.
3. Classify new points using the majority vote of the ensemble $\{g_j\}_{j=1}^n$.

Bagging is only useful when the classifier is unstable and small changes to the training data in the bootstrap samples can create large changes in the classifier [1]. The OCSVM is unstable in the sense that the estimated boundary expands greatly in the presence of outliers. However, the OCSVM is also stable in the sense that the estimated boundary always encloses the positive class.

Directly applying bagging to the OCSVM is not useful since the OCSVM is stable for the majority of the training data. Bagging only tightens the estimated boundary when outliers are excluded from the bootstrap samples. However, since all points are given the same probability weight, outliers are likely to be included in many of the bootstrap samples. Bagging will still exclude some outliers from the bootstrap samples by chance, but the estimated boundary will not be very robust. Since we want to exclude outliers from the bootstrap samples, we should give outliers lower probability weight. We propose to give probability weights to points based on how close they are to the positive class.

Kernel density estimation is a popular nonparametric method to estimate the probability distribution of the training data [7]. The kernel density estimator is a sum of Gaussian kernels at each point in the training data

$$f(\mathbf{x}) = \sum_{j=1}^{m} \frac{1}{(2\pi)^{d/2}\sigma^d m} k(\mathbf{x}, \mathbf{x}_j). \tag{4}$$

We could use the kernel density estimate as the probability weight $w(i) = f(\mathbf{x}_i)$ for point $\mathbf{x}_i$. However, kernel density estimation in multiple dimensions can be unreliable. Therefore, a more suitable probability weight would be a measure of how well the kernel density estimator fits the training data.

We use an iterative method from [4] based on cross validation of a weighted kernel density estimator to determine the probability weights. Instead of boosting up probability weights for outliers, we boost them down. Initially, all points are given the same probability weight. At iteration k, given the probability weights $w_k(i)$, the weighted kernel density estimator is

$$f_k(\mathbf{x}_i) = \sum_{j=1}^{m} \frac{w_k(i)}{(2\pi)^{d/2}\sigma^d} k(\mathbf{x}_i, \mathbf{x}_j) \tag{5}$$

and the leave one out weighted kernel density estimator is

$$f'_k(\mathbf{x}_i) = \sum_{j=1}^{m} \frac{w_k(i)}{(2\pi)^{d/2}\sigma^d} k(\mathbf{x}_i, \mathbf{x}_j) I(j \neq i). \tag{6}$$

The probability weights are updated by adding the log odds ratio of the weighted kernel density estimate to the leave one out weighted kernel density estimate.

More precisely, the probability weight algorithm is:

1. Initialize with uniform probability weights by $w_0(i) = 1/m$.
2. For $k = 1, \ldots, n$, update the probability weights by

$$w_k(i) = w_{k-1}(i) + \log\left(\frac{f_{k-1}(\mathbf{x}_i)}{f'_{k-1}(\mathbf{x}_i)}\right).$$

3. Invert and normalize the final probability weights by

$$w(i) = \left(\frac{1}{w_n(i)}\right) / \left(\sum_{j=1}^{m} \frac{1}{w_n(j)}\right).$$

The final probability weights will be low for points in the negative class and high for points in the positive class.

4 Experiments

We compared the normal and bagging OCSVM on both simulated and real world data sets. All experiments were performed in MATLAB on a standard personal computer. We used the SMO algorithm from [11] as implemented in the LIBSVM library to solve the OCSVM. We selected σ for each data set using a simple grid search as in [6] to maximize the number of negative points outside the estimated boundary plus the number of positive points inside the estimated boundary. We used $n = 10$ samples in bagging and $k = 5$ iterations in determining the probability weights for all data sets. We used the same ν and σ values for each individual OCSVM in the ensemble as the normal OCSVM. The bagging OCSVM was significantly more computationally intensive than the normal OCSVM due to the kernel density estimation step, but was still fast enough to be used practically. Typical runtimes for the bagging OCSVM were less than a minute.

First, we evaluated the normal and bagging OCSVM on three simulated data sets similar to those used in [6]:

- **Square noise** contains 450 points and 2 features. First, 400 points were drawn randomly from the square $\{(x, y) : x \in [0.4, 2.6], y \in [0.4, 0.6] \cup [2.4, 2.6]\} \cup \{(x, y) : x \in [0.4, 0.6] \cup [2.4, 2.6], y \in [0.4, 2.6]\}$. Next, 50 points of noise were drawn randomly from the area $\{(x, y) : x \in [0, 3], y \in [0, 3]\}$. We set $\nu = 1/9$ and $\sigma = 0.35$.
- **Line noise** contains 450 points and 2 features. First, 400 points were drawn randomly from the line $\{(x, y) : x = y, x \in [0, 3], y \in [0, 3]\}$. Next, 50 points of noise were drawn randomly from the area $\{(x, y) : x \in [0, 3], y \in [0, 3]\}$. We set $\nu = 1/9$ and $\sigma = 0.35$.

– **Sphere** contains 450 points and 2 features. All 450 points were drawn from a bivariate Gaussian distribution with mean $(1.5, 1.5)$ and variance 0.1. We set $\nu = 1/10$ and $\sigma = 2$.

The estimated boundaries of the normal and bagging OCSVM are shown in Figure 1. On the Square noise and Line noise data sets, the bagging OCSVM

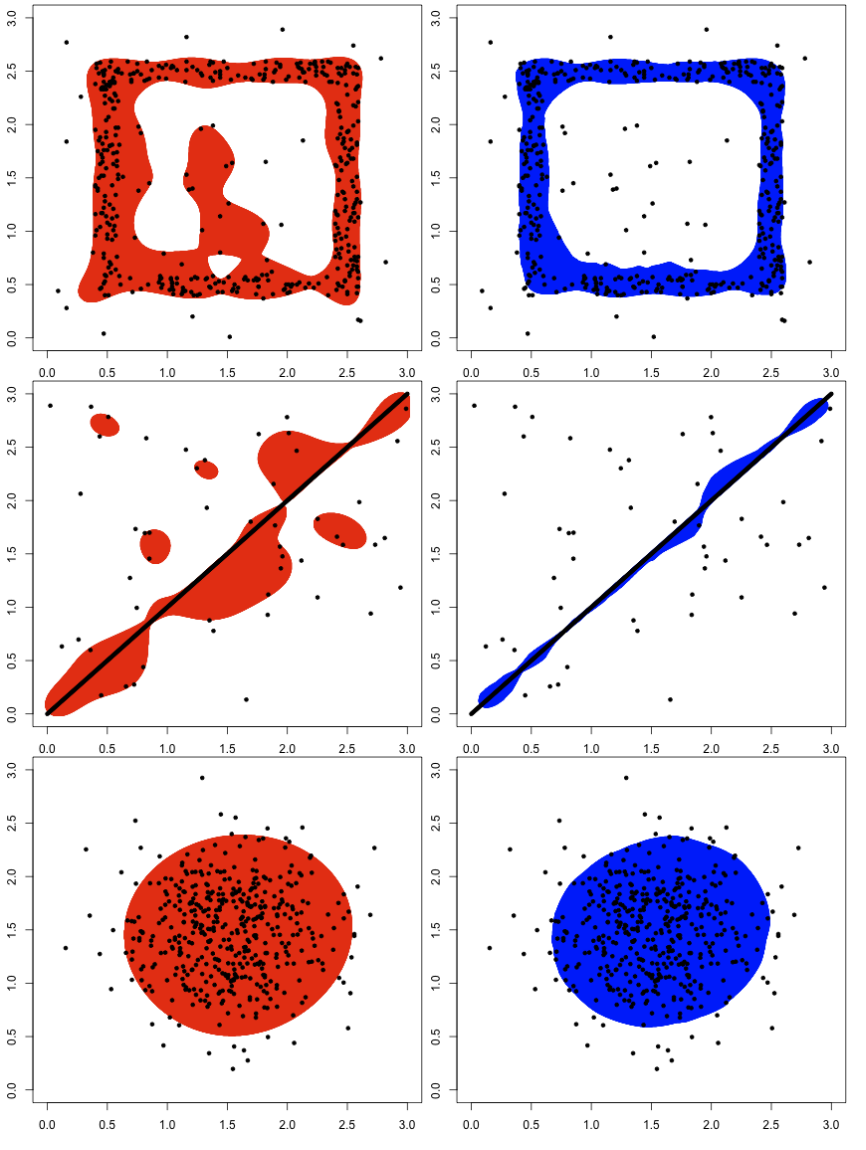

Fig. 1. Estimated boundaries of the normal (left) and bagging (right) OCSVM on the Square noise (top), Line noise (middle), and Sphere (bottom) data sets

performs much better than the normal OCSVM. The estimated boundary of the normal OCSVM is influenced by outliers and encloses large areas of the feature space with low density, including disconnected areas of the feature space. On the other hand, the estimated boundary of the bagging OCSVM tightly encloses the shapes of the positive class in the feature space. Therefore, the bagging OCSVM appears to be less sensitive than the normal OCSVM to noise. On the Sphere data set, the bagging OCSVM performs similarly to the normal OCSVM. Therefore, the bagging OCSVM does not appear to arbitrarily tighten the estimated boundary. Although we chose the ν values optimally according to the proportion of noise, we found that adjusting the ν values only shrunk or expanded the estimated boundaries uniformly for both the normal and bagging OCSVM.

In order to determine whether the probability weights alone are sufficient to eliminate outliers without bagging, we removed 50 points with the lowest probability weights from the Square noise and Line noise data sets and trained a normal OCSVM. The amount of outliers removed corresponds to the amount of added noise. The estimated boundaries of the normal OCSVM are shown in Figure 2. Although the estimated boundaries of the normal OCSVM improved significantly after outlier removal, the probability weights are not necessarily reliable for outlier detection. In regions of the feature space where the data is sparse, the probability weights are low, so thresholding can lead to discontinuities in the estimated boundary. In particular, the estimated boundary for the Square noise data set contains a gap in the top side. Therefore, sampling from a weighted distribution in bagging appears to be important for averaging out sparse regions of the data.

Next, we evaluated the performance of the normal and bagging OCSVM on three real world data sets from the UCI Machine Learning Repository:

– **USPS** contains 256 features and 2651 points (821 positive class, 1830 negative class). The data set was randomly partitioned into a training data set

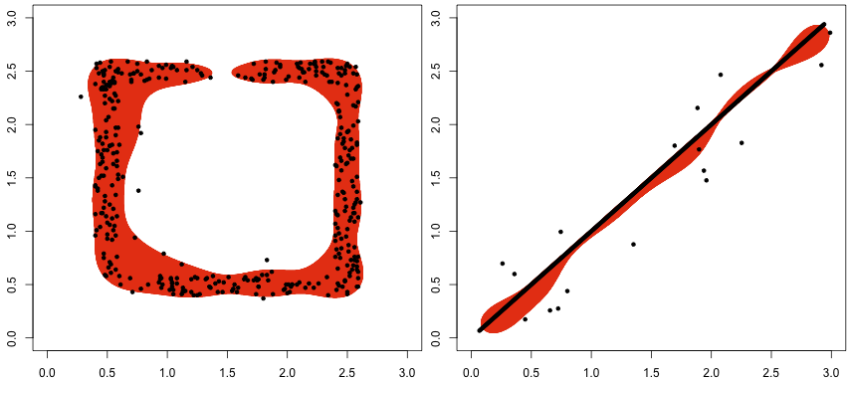

Fig. 2. Estimated boundaries of the normal OCSVM on the Square noise (left) and Line noise (right) data sets after outlier removal

Table 1. Performance of the normal and bagging OCSVM on the USPS, Breast cancer, and Ionosphere data sets

Data set	FPR	TPR		
		Normal	Bagging	Difference
USPS	0%	45.2%	49.7%	+4.5%
	1%	61.6%	65.0%	+3.4%
	5%	75.7%	78.5%	+2.8%
	10%	89.8%	92.7%	+2.9%
Breast cancer	0%	67.2%	88.9%	+21.7%
	1%	82.0%	89.8%	+7.8%
	5%	86.1%	90.2%	+4.1%
	10%	93.0%	92.2%	−0.8%
Ionosphere	0%	8.8%	18.4%	+9.6%
	1%	28.0%	37.6%	+9.6%
	5%	58.4%	60.0%	+1.6%
	10%	87.2%	90.4%	+3.2%

of 644 points and a test data set of 2007 points (177 positive class, 1830 negative class). We set $\sigma = 1$.

- **Breast cancer** contains 10 features and 683 points (444 positive class, 239 negative class). The data set was randomly partitioned into a training data set of 200 points and a test data set of 483 points (244 positive class, 239 negative class). We set $\sigma = 1$.
- **Ionosphere** contains 34 features and 351 points (225 positive class, 126 negative class). The data set was randomly partitioned into a training data set of 100 points and a test data set of 251 points (125 positive class, 126 negative class). We set $\sigma = 16$.

We used the true positive rate (TPR) and false positive rate (FPR) as our performance metrics. Since ν is an upper bound on the FPR, we varied ν in order to control the FPR. However, varying ν was not sufficient to compute a full receiver operating characteristic (ROC) curve since the FPR was always below 25%. In real world applications of one class classification, the target FPR is typically very low since there are large consequences for false positives. Therefore, we compared the TPR of the normal and bagging OCSVM at four typical target FPRs of 0%, 1%, 5%, and 10% as in [8]. The TPRs of the normal and bagging OCSVM are shown in Tables 1 and 2.

The bagging OCSVM achieves higher TPRs than the normal OCSVM for almost all target FPRs on all data sets, suggesting that real world data sets are noisy enough that tightening the estimated boundary is important. The performance improvement of the bagging OCSVM is highest for low target FPRs. For a target FPR of 0%, the difference in TPR for the bagging OCSVM ranges from +4.5% to +21.7%. As the target FPR increases, the performance of the bagging OCSVM approaches that of the normal OCSVM, probably because tightening the estimated boundary becomes less important than enclosing all of the positive class. For a target FPR of 10%, the difference in TPR for the

Table 2. Performance of the normal and bagging OCSVM on the noisy versions of the USPS, Breast cancer, and Ionosphere data sets

Data set	FPR	TPR		
		Normal	Bagging	Difference
USPS	0%	32.2%	38.4%	+6.2%
	1%	48.6%	53.1%	+4.5%
	5%	68.4%	73.4%	+5.0%
	10%	81.9%	84.7%	+2.8%
Breast cancer	0%	56.1%	83.6%	+27.5%
	1%	59.8%	86.9%	+27.1%
	5%	66.8%	87.3%	+20.5%
	10%	78.3%	87.8%	+9.5%
Ionosphere	0%	0.0%	11.2%	+11.2%
	1%	25.6%	36.0%	+10.4%
	5%	56.8%	58.4%	+1.6%
	10%	59.2%	60.0%	+0.8%

bagging OCSVM ranges from −0.8% to +3.2%. Therefore, the bagging OCSVM appears to be well suited for applications of one class classification that require a low target FPR.

In order to further evaluate the robustness, noisy versions of the data sets were generated by randomly swapping 25% of the points in the training data set with points in the test data set. The performance improvement of the bagging OCSVM was even larger on the noisy versions of the data sets, suggesting that tightening the estimated boundary is especially important in the presence of noise. The noisy versions of the training data sets contained points from both the positive class and the negative class, which probably expanded the estimated boundaries of the normal OCSVM. The performance of the bagging OCSVM is not as sensitive to noise since the bagging OCSVM decreases the weight given to points far from the positive class. Therefore, the bagging OCSVM appears to be well suited for unlabeled training data.

5 Conclusion

In this paper, we proposed a bagging OCSVM using weights determined by kernel density estimation to tighten the estimated boundary of the normal OCSVM, which can be sensitive to noise. We demonstrated that the estimated boundary of the bagging OCSVM fits the shape of the positive class well on three simulated data sets and that the bagging OCSVM achieves significantly higher TPRs than the normal OCSVM on three real data sets at common target FPRs. The bagging OCSVM is especially useful for applications of one class classification that require low target FPRs, such as the diagnosis of rare diseases.

Acknowledgments. The authors would like to thank the anonymous reviewers, whose comments helped improve the manuscript.

References

1. Bauer, E., Kohavi, R.: An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. Machine Learning 36, 105–139 (1999)
2. Bicego, M., Figueiredo, M.A.T.: Soft clustering using weighted one-class support vector machines. Pattern Recognition 42, 27–32 (2009)
3. Breiman, L.: Bagging predictors. Machine Learning 24, 123–140 (1996)
4. Di Marzio, M., Taylor, C.C.: Boosting kernel density estimates: a bias reduction technique? Biometrika 91, 226–233 (2004)
5. Hempstalk, K., Frank, E., Witten, I.H.: One-class classification by combining density and class probability estimation. In: Daelemans, W., Goethals, B., Morik, K. (eds.) ECML PKDD 2008, Part I. LNCS (LNAI), vol. 5211, pp. 505–519. Springer, Heidelberg (2008)
6. Hoffmann, H.: Kernel PCA for novelty detection. Pattern Recognition 40, 863–874 (2007)
7. Parzen, E.: On estimation of a probability density function and mode. Annals of Mathematical Statistics 33, 1065–1076 (1962)
8. Perdisci, R., Gu, G., Lee, W.: Using an ensemble of one-class SVM classifiers to harden payload-based anomaly detection systems. In: 6th IEEE International Conference on Data Mining, pp. 488–498. IEEE Press, New York (2006)
9. Li, C., Zhang, Y.: Bagging one-class decision trees. In: 5th International Conference on Fuzzy Systems and Knowledge Discovery, pp. 420–423. IEEE Press, New York (2008)
10. Roth, V.: Kernel fisher discriminants for outlier detection. Neural Computation 18, 942–960 (2006)
11. Scholköpf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J., Williamson, R.C.: Estimating the support of a high-dimensional distribution. Neural Computation 13, 1443–1471 (2001)
12. Tax, D.M.J., Duin, R.P.W.: Support vector domain description. Pattern Recognition Letters 20, 1191–1999 (1999)
13. Tax, D.M.J.: One-class classification. Ph.D thesis, Delft University of Technology (2001)
14. Tax, D.M.J., Duin, R.P.W.: Combining one-class classifiers. In: Kittler, J., Roli, F. (eds.) MCS 2001. LNCS, vol. 2096, pp. 299–308. Springer, Heidelberg (2001)
15. Tax, D.M.J., Juszczak, P.: Kernel whitening for one-class classification. In: Lee, S.-W., Verri, A. (eds.) SVM 2002. LNCS, vol. 2388, pp. 40–52. Springer, Heidelberg (2002)