LEARNING MINIMUM VOLUME SETS WITH SUPPORT VECTOR MACHINES

Mark A. Davenport, Richard G. Baraniuk *

Rice University
Department of Electrical and Computer Engineering

Clayton D. Scott †

Rice University
Department of Statistics

ABSTRACT

Given a probability law P on d-dimensional Euclidean space, the minimum volume set (MV-set) with mass β, $0 < \beta < 1$, is the set with smallest volume enclosing a probability mass of at least β. We examine the use of support vector machines (SVMs) for estimating an MV-set from a collection of data points drawn from P, a problem with applications in clustering and anomaly detection. We investigate both one-class and two-class methods. The two-class approach reduces the problem to Neyman-Pearson (NP) classification, where we artificially generate a second class of data points according to a uniform distribution. The simple approach to generating the uniform data suffers from the curse of dimensionality. In this paper we (1) describe the reduction of MV-set estimation to NP classification, (2) devise improved methods for generating artificial uniform data for the two-class approach, (3) advocate a new performance measure for systematic comparison of MV-set algorithms, and (4) establish a set of benchmark experiments to serve as a point of reference for future MV-set algorithms. We find that, in general, the two-class method performs more reliably.

1. INTRODUCTION

In anomaly detection the goal is to identify measurements $\mathbf{x} \in \mathbb{R}^d$ as being either normal/typical or abnormal/anomalous. We seek a subset of $\mathbb{R}^d$ such that points inside the set correspond to typical data while points outside are anomalies. In practice, it is often the case that such a set must be "learned" from a collection $\{\mathbf{x}_i\}_{i=1}^M$ of training samples gathered under normal conditions. In other words, we must be able to detect anomalies without knowing what they look like. For example, in machine fault detection, we would like to predict when a machine is about to fail, but cannot gather data from a failed machine because it would entail breaking the machine.

1.1. Minimum volume sets

In this situation one possibility is to estimate a *minimum volume set* (MV-set) for the probability measure P governing the typical data. Specifically, given a known reference measure μ, the MV-set with mass at least β, $\beta \in (0, 1)$, is

$$G_\beta^* = \arg\min\{\mu(G) : P(G) \geq \beta, G \text{ measurable}\}.$$

In this paper we focus on the common case where μ is the Lebesgue measure, although our techniques extend easily to other measures. The parameter β is chosen by the user and reflects a desired false alarm rate of $1 - \beta$. MV-sets summarize regions where the mass of P is most concentrated. For example, if P is a multivariate Gaussian distribution and μ is the Lebesgue measure, then the MV-sets are ellipsoids. See [1–4] and references therein for additional discussion.

*Supported by NSF grants CCF-0431150, CNS-0435425, and CNS-0520280; ONR grant N00014-02-1-0353; AFOSR grant FA9550-04-0148; DARPA grant N66001-06-1-2011; and the Texas Instruments Leadership University Program.

†Supported by an NSF VIGRE postdoctoral training grant.

Email: {md,richb,cscott}@rice.edu, Web: dsp.rice.edu

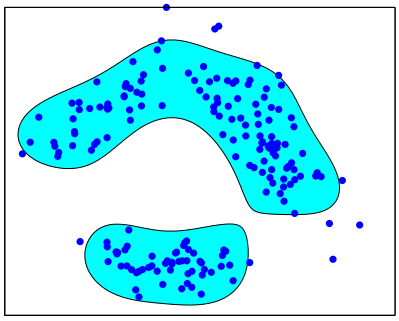


Fig. 1. *Estimated minimum volume set for "banana" data ($\beta = 0.9$).*

Our task is the following: Given a reference measure μ, $\beta \in (0, 1)$, and M realizations of a distribution P, construct a set $\widehat{G}_\beta$ that approximates the true MV-set G_β^*. An example is shown in Figure 1. In this work we focus on one of the most successful and widely applied family of learning algorithms, support vector machines (SVMs).

1.2. One-class methods

Since MV-sets are density level sets [4], with β defining a density level and vice-versa, an obvious approach to estimating an MV-set is to estimate the density of P and then compute an appropriate level set. However, density estimation is a notoriously difficult problem in the case where we do not have a parametric model for our data, and in fact these methods typically perform very poorly on real problems. A better strategy is to avoid estimating the entire density and only estimate a single level-set. The so-called one-class SVM (OC-SVM) [3,5] is one of the more powerful methods for doing this. Our application of the OC-SVM entails carefully setting the free parameters to achieve the desired mass/volume tradeoff.

1.3. Two-class methods: Reducing to NP classification

In contrast to the one-class approach described above, a second approach is to reduce MV-set estimation to Neyman-Pearson (NP) classification [4, 6]. The advantage of this approach is that almost any standard classification algorithm can be modified to perform NP classification. The disadvantage is that this conversion often requires the introduction of an additional free parameter (to affect the tradeoff between false alarms and misses). We focus on a strategy for NP classification using SVMs developed in [7].

In classification our training data consist of samples $\{\mathbf{x}_i\}_{i=1}^N$ together with labels $y_i \in \{-1, +1\}$ for each sample. We assume that when $y_i = +1$, $\mathbf{x}_i$ is drawn from Q_+ and when $y_i = -1$, $\mathbf{x}_i$ is drawn from Q_-, where Q_+ and Q_- are unknown probability measures. In what follows, let $f : \mathbb{R}^d \to \{+1, -1\}$ be a classifier, and let

$$P_F(f) = Q_-(\{\mathbf{x} : f(\mathbf{x}) = +1\}) \quad \text{and}$$
$$P_M(f) = Q_+(\{\mathbf{x} : f(\mathbf{x}) = -1\})$$

denote the *false alarm* and *miss* rates of f. In NP classification, the goal is to design a classifier $\hat{f}_\alpha$ that minimizes the miss rate while constraining the false alarm rate to not exceed some user-specified significance level α. For a more detailed motivation for the Neyman-Pearson paradigm, see [6].

To estimate the MV-set using NP classification, we can think of setting $Q_- = 1 - P$ and $Q_+ = \mu$. In this case the MV-set and NP classification solutions coincide. Specifically, if $\alpha = 1 - \beta$ and f^*_α is the optimal NP classifier, then $G^*_\beta = \{x : f^*_\alpha = -1\}$. To implement this idea we assign the observed data, $\{\mathbf{x}_i\}_{i=1}^M$, labels of -1. We then simulate[1] a number of points from the reference measure μ, and assign these points labels of $+1$. Constraining $P_F(f) \leq \alpha = 1 - \beta$, ensures that the probability mass of the set where $f(\mathbf{x}) = -1$ is at least β, and since we draw the positively labeled class from the reference measure μ, minimizing $P_M(f)$ is equivalent to minimizing μ. From this perspective, the reference measure μ is a prior on the distribution for anomalies. Taking μ to be the Lebesgue/uniform measure can be interpreted as assuming a noninformative prior on anomalies. In summary, by taking our MV-set to be $\widehat{G}_\beta = \{\mathbf{x} : \widehat{f}_\alpha(\mathbf{x}) = -1\}$ we can estimate the MV-set of our data using NP classification algorithms.

The idea of reducing a supervised problem to an unsupervised problem by sampling from a reference measure has apparently been known for some time [8]. Although they do not speak in terms of "Neyman-Pearson classification," the reduction outlined above is essentially described in [9]. The two-class idea was also applied in [10] to reduce density level set estimation to cost-sensitive classification. In a kind of hybrid between one- and two-class methods, [11] employs artificial uniform data to select the parameters of the OC-SVM.

1.4. The challenge of generating uniform data

The two-class approach entails generating realizations from the reference measure μ. In the case where μ is the Lebesgue measure, it suffices to draw points uniformly from some hypercube containing the data, with some extra care necessary when dealing with discrete-valued data. However, as the number of points drawn grows, so does the computational complexity of the training process. With a limited number of simulated points, independent generation of the uniform data may suffer because P may be concentrated in a very small volume of space. Furthermore, in high dimensions, the average inter-point distance increases, and more and more of the simulated points will be so far from the data as to be useless in estimating the volume. This can be viewed as one aspect of the "curse of dimensionality".

We consider two alternative methods for overcoming these challenges. The first involves drawing many more points than are ultimately desired and then adaptively removing points, or *thinning*, to get the desired number of points. This approximately results in a "packing set" with a large minimum distance between neighboring points. While thinning does offer a significant gain with respect to independent uniform sampling, it does not account for the "vastness of space" in high dimensions. A second approach, called *manifold sampling*, does address this concern and also adapts to potential manifold structure in the data.

1.5. Performance analysis and benchmark experiments

Most papers proposing algorithms for MV/level set estimation or NP/cost-sensitive classification do not adopt a systematic methodology for comparing different methods. The typical paper introduces a new algorithm and provides an ROC curve that conveys the ability of the algorithm to trade off false alarms for misses. However, in most practical settings, we desire a specific false alarm rate. Therefore, in evaluating a specific algorithm, we need to ask not only "Can we trade off false alarms for misses?" but also "How precisely can we guarantee a desired false alarm rate?"

Toward this end, we advocate a recently introduced performance measure for comparing algorithms for MV-set estimation [12]. We conduct several numerical experiments and use this performance measure to compare the one-class and two-class methods. Since our benchmark datasets are obtained by taking one class from a data set for binary classification, we have two obvious options for measuring performance on anomalous data. We can either estimate the volume of the set, which assumes that anomalies are actually uniformly distributed, or we can treat the second class as anomalies and compare the MV-set with a classifier that has access to both classes. The latter gives a way of assessing the robustness of the implicit assumption that the anomalies are uniformly distributed.

Since satisfying the targeted mass constraint is so crucial in MV-set estimation, our emphasis on a systematic comparison using a scalar performance measure highlights the importance of *error estimation*. Not only should an algorithm be flexible in the sense of having a large area under its ROC, but it should also be possible to accurately estimate the enclosed mass and volume so that the free parameters of the algorithm can be set appropriately.

1.6. Summary of contributions

We contribute two methods for generating the artificial uniform data that outperform independent sampling. Employing a practical performance measure, we report an experimental comparison of a one-class method (the OC-SVM) and a two-class method (based on the algorithm in [7]) on five benchmark datasets. We find that the two-class method consistently outperforms the one-class method, and conjecture that this is in part due to the improved error estimation capabilities of this method. We also find that the MV-set performs nearly as well in some cases as a classifier that was trained having access to examples of actual anomalies. This indicates that the implicit uniform prior on anomalies is often quite robust. Finally, we have made our code, which is based on the LIBSVM package [13], available at www.dsp.rice.edu/software.

2. SUPPORT VECTOR MACHINES

Support vector machines (SVMs) are among the most effective methods for learning classifiers from training data [14]. Conceptually, we construct the support vector classifier in a two step process. In the first step we transform the $\mathbf{x}_i \in \mathbb{R}^d$ via a mapping $\Phi : \mathbb{R}^d \to \mathcal{H}$ where $\mathcal{H}$ is a high (possibly infinite) dimensional Hilbert space. The intuition is that we should be able to separate these classes more easily in $\mathcal{H}$ than in $\mathbb{R}^d$. For algorithmic reasons, we choose Φ so that we can compute inner products in $\mathcal{H}$ through the *kernel* operator $k(\mathbf{x}, \mathbf{x}') = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle_{\mathcal{H}}$.

In the second step, we determine a hyperplane in the induced feature space according to the max-margin principle, which states that, in the case where we can separate the two classes by a hyperplane, we should pick the hyperplane that maximizes the *margin* — the distance between the decision boundary and the closest point to the boundary. This hyperplane is then our decision boundary. Thus, if $\mathbf{w} \in \mathcal{H}$ and $b \in \mathbb{R}$ are the normal vector and affine shift (or *bias*) defining the max-margin hyperplane, then the support vector classifier is given by $f_{\mathbf{w},b}(\mathbf{x}) = \text{sgn}(\langle \mathbf{w}, \Phi(\mathbf{x}) \rangle_{\mathcal{H}} + b)$.

2.1. Neyman-Pearson support vector machines

There are several different formulations of the SVM. In this paper we will focus on the cost-sensitive ν-SVM, or the 2ν-SVM,

[1] Note that the MV-set does not change if we truncate μ outside of the support of P. Thus, if μ is Lebesgue measure then it suffices to simulate points uniformly on some region containing the support of P.

first proposed in [15]. Let $I_+ = \{i : y_i = +1\}$ and $I_- = \{i : y_i = -1\}$, and set $n_+ = |I_+|$ and $n_- = |I_-|$. The 2ν-SVM has the primal formulation:

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi},\rho} \ & \frac{1}{2}\|\mathbf{w}\|^2 - 2\nu_+\nu_-\rho + \frac{\nu_-}{n_+}\sum_{i\in I_+}\xi_i + \frac{\nu_+}{n_-}\sum_{i\in I_-}\xi_i \\ \text{s.t.} \quad & y_i(k(\mathbf{w},\mathbf{x}_i)+b) \geq \rho - \xi_i \qquad \text{for } i = 1,2,\ldots,n \\ & \xi_i \geq 0 \qquad \text{for } i = 1,2,\ldots,n \\ & \rho \geq 0. \end{aligned}$$

In [7] it was shown that we can use the 2ν-SVM to achieve the desired false alarm rate by adjusting ν_+ and ν_- appropriately. Specifically, we conduct a grid search over the SVM parameters and estimate P_F and P_M using some error estimation technique such as cross-validation, denoting these estimates $\widehat{P}_F$ and $\widehat{P}_M$. Finally, we select the parameter combination minimizing $\widehat{P}_M$ such that $\widehat{P}_F \leq \alpha$, and train an SVM on the full training set using these parameters. The dual formulation of ($P_{2\nu}$) is feasible if and only if $\nu_+ \leq 1$ and $\nu_- \leq 1$, with a trivial solution if $\nu_+ \leq 0$ or $\nu_- \leq 0$ [16]. Therefore, to search over the parameters of the 2ν-SVM it suffices to conduct a search over a uniform grid of (ν_+, ν_-) in $[0,1] \times [0,1]$.

Furthermore, we can significantly improve the performance of the basic grid search method. First, the additional parameter in the 2ν-SVM can render a full grid search somewhat time consuming. Fortunately, a simple speed-up is possible. A simple coordinate descent search approach can perform almost as well as a full grid search, but is much faster [7]. In addition, for the full grid search over (ν_+, ν_-), after estimating the error at each point on the grid, we can low-pass filter both $\widehat{P}_F$ and $\widehat{P}_M$ with a Gaussian window. For coordinate descent we window along lines in the grid. This effectively reduces the variance of the error estimates. It is especially effective for high variance estimates such as cross-validation, and can significantly improve the performance. Without windowing, some grid points will look much better than they actually are, due to chance variation [7].

2.2. One-Class support vector machines

The one-class SVM (OC-SVM) was proposed in [3] for the problems of estimating the support of a high-dimensional distribution and novelty detection. The OC-SVM can be formulated as

$$\begin{aligned} \min_{\mathbf{w},\boldsymbol{\xi},\rho} \ & \frac{1}{2}\|\mathbf{w}\|^2 - \nu\rho + \frac{1}{n}\sum_{i=1}^{n}\xi_i \\ \text{s.t.} \quad & k(\mathbf{w},\mathbf{x}_i) \geq \rho - \xi_i \qquad \text{for } i = 1,2,\ldots,n \\ & \xi_i \geq 0 \qquad \text{for } i = 1,2,\ldots,n. \end{aligned}$$

The resulting decision function

$$f(\mathbf{x}) = \text{sgn}(k(\mathbf{w},\mathbf{x}) - \rho)$$

will be positive on a set containing most $\mathbf{x}_i$. Thus, in this algorithm the MV-set is chosen to be $\widehat{G}_\beta = \{\mathbf{x} : f(\mathbf{x}) > 0\}$.

However, the user must set any kernel parameters and the parameter ν. It is not immediately clear how to choose these parameters so that $\widehat{G}_\beta$ reasonably approximates G^*_β. The challenge lies in the fact that while we can estimate $P(G)$ from the data, thus ensuring that $P(G) > \beta$, there will in general be many possible parameter settings that result in sets G that satisfy this requirement, and we must select only one. Specifically, we would like to choose the one with *minimum volume*. Thus we must estimate the volume of the set induced by each parameter setting. We do so by drawing a large number of points from μ and then calculating the fraction of these points that lie in G for each parameter setting, as in [11]. We will now proceed to describe several different methods for generating these points.

3. GENERATING UNIFORM DATA

The two-class method entails generating realizations from the reference measure μ. In the case where μ is the Lebesgue measure and the features are real-valued, it suffices to draw points uniformly from some hypercube containing the data. In some cases we will have training data where some (or all) of the features assume a finite number of discrete values. For example, one feature might be gender, in which case the data points will assume only one of two possible values. In this case it makes little sense to draw training points uniformly from a hypercube containing the data, thus we instead draw points uniformly from the discrete set of values the feature can assume.

In both cases, drawing these points is a straightforward procedure. However, as the number of points drawn grows, so does the computational complexity of the training process. Thus, in practice, we must only draw a small number of points. Unfortunately, with a limited number of simulated points, independent generation of the uniform data may suffer because P may be concentrated in a very small volume of space. Furthermore, in high dimensions, the average interpoint distance increases, and more and more of the simulated points will be so far from the data as to be useless in estimating the volume. This can be viewed as one aspect of the "curse of dimensionality"

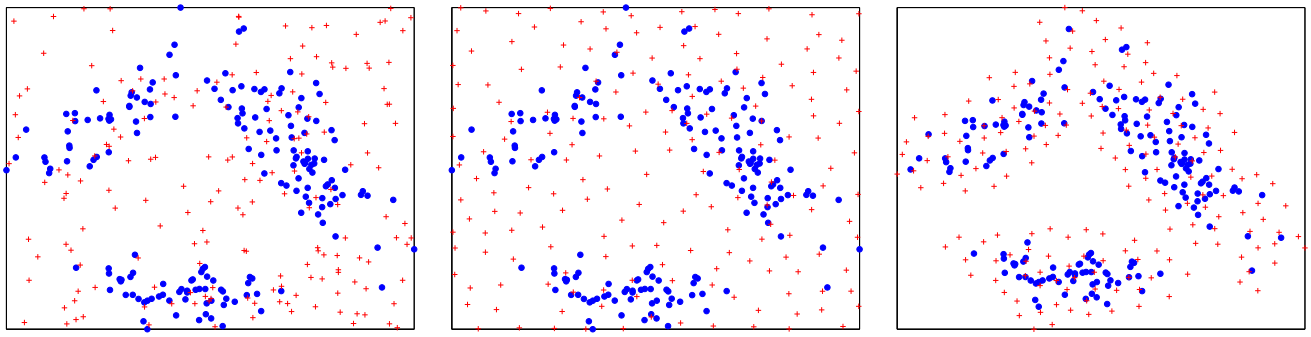
3.1. Thinning

The *thinning* approach is to draw many more points than are ultimately desired, and then adaptively remove points to get the desired number of points. Specifically, say that we draw m points and ultimately want n points, where $m \gg n$. We then compute the Euclidean distance between all possible pairs of points. We can iteratively remove points by considering the remaining points and selecting the pair of points that are closest to each other. We throw away one of these points by removing the one that is closest to any of the remaining points. When iteratively applied, this results in a data set where the points are ensured to be separated by a relatively large minimum distance [17].

3.2. Manifold sampling

The thinning approach described above helps to evenly distribute the points throughout space. This is potentially problematic in high dimensions. When dealing with high-dimensional data, it is common for the data to occupy a very small fraction of the total volume of a hypercube containing the data. For example, our data might lie on a low-dimensional manifold embedded in a high-dimensional space. In this case, a small number of points drawn uniformly on the hypercube may be of little use in estimating the MV-set — it is extremely unlikely that any points will lie within the MV-set, and hence it is essentially impossible to estimate the volume of the set.

We propose a second approach that models the observed data as lying on a low-dimensional manifold. First, compute the average distance between a point and its k^{th} nearest neighbor, where k is chosen by the user (in our experiments we take $k = 10$). Then, generate a large number m of points by selecting an $\mathbf{x}_i$ at random, and then randomly drawing a point from the sphere centered at $\mathbf{x}_i$ whose radius is the number computed in the first step. We can think of the union of these spheres as a *thickened* manifold within which the data lie. Again, we can apply the thinning technique to get a reduced set of points that are separated by a large minimum distance. These points also lie within a set that contains the data but potentially



(a) independent uniform sampling (b) thinned uniform sampling (c) manifold sampling

Fig. 2. *Methods for generating uniform data. (a) n points sampled independently from a uniform distribution. (b) 10 n points sampled from a uniform distribution thinned to n points. (c) 10 n points sampled from a thickened manifold thinned to n points.*

has a much smaller volume than that of the bounding hypercube. This technique and those described above are illustrated in Figure 2.

4. MEASURING PERFORMANCE

Any experimental comparison of different methods for MV-set estimation requires a measure of performance, that is, a scalar criterion that can be used to compare two estimated MV-sets head-to-head. For example, suppose we have a pair of sets G, G'; how can we determine which set is "better" in a meaningful way? The tendency in previous work has been to adjust the parameters of the different algorithms *a posteriori* until $P(G) \approx P(G')$, and then say that G is "better" if $\mu(G) < \mu(G')$. However, this simply avoids the issue of actually making a direct comparison of G and G' when $P(G) < P(G')$ and $\mu(G) < \mu(G')$, as will often happen in practice.

One possible scalar measure of performance is to assign a "score" of $\mu(G)$ to the set if $P(G) \geq \beta$, and ∞ otherwise. This is problematic, however, because we must estimate $P(G)$, and this estimate is susceptible to error. Moreover, in practical settings, it is often acceptable (and sometimes unavoidable) to have $P(G)$ be a small amount less than β. Thus, it seems preferable to have some tolerance for estimates whose mass is slightly less than β.

As an alternative, we evaluate an MV-set using

$$\mathcal{E}_\mu(G) = \frac{1}{1-\beta} \max\{\beta - P(G), 0\} + \mu(G)$$

As discussed in [12], this measure has the following desirable properties: (*i*) It is minimized by the set G^*_β. (*ii*) It can be accurately estimated from a test sample using the simple plug-in estimate. (*iii*) It has the appealing property that as β draws closer to 1, a stiffer penalty is exacted on classifiers that violate the constraint. In other words, it penalizes the *relative* error $(\beta - P(G))/(1-\beta)$.

In our experiments we employ benchmark data sets for binary classification and perform MV-set estimation using only the negatively labeled class. We use the test set to estimate $P(G)$, and we estimate $\mu(G)$ by generating a large test set of uniform data. In addition, since we only use one class for training, we have a second performance measure:

$$\mathcal{E}_+(G) = \frac{1}{1-\beta} \max\{\beta - P(G), 0\} + Q_+(G),$$

where $Q_+(\{\mathbf{x} : f(\mathbf{x}) = -1\})$ is the probability of error on the class not used during training. In some sense this is a more appropriate metric because μ is effectively a prior for the anomaly distribution, while Q_+ is the actual anomaly distribution. We will consider this measure as well since we would like our algorithm to perform well regardless of the structure of the anomalous data.

5. EXPERIMENTS

In our experiments with the OC-SVM we used the LIBSVM package [13]. For the NP-SVM we adapted the LIBSVM package to implement the 2ν-SVM, which is available online at www.dsp.rice.edu/software.

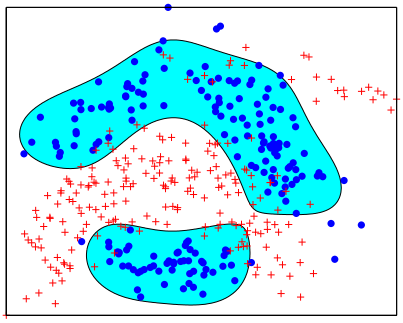
We ran our algorithms on the benchmark datasets "banana", "breast-cancer", "heart", "thyroid", and "ringnorm". The datasets are available online with documentation.[2] The first and last data set are synthetic, while the other three are based on real data collected from various repositories on the web. The "breast-cancer" and "heart" datasets have a mixture of discrete and continuous features, while the other datasets have exclusively continuous features. The dimensions of the datasets are 2, 9, 13, 5, and 20 respectively. For our experiments we used the negatively labeled training vectors as the realizations of the typical distribution, and estimated average performance over 30 different permutations of each data set into training and test data. The data sets contained 400, 200, 170, 140, and 400 training vectors respectively, although the number of negatively labeled training vectors varies over the different permutations. The targeted β is 0.9.

In all of our experiments we used a radial basis function (Gaussian) kernel and searched for the bandwidth parameter σ over a logarithmically spaced grid of 50 points from 10^{-4} to 10^4. For the 2ν-SVM method we considered a 50×50 regular grid of $(\nu_+, \nu_-) \in [0,1] \times [0,1]$. For the OC-SVM we considered a 50 point logarithmically spaced grid of ν from 10^{-4} to 1.

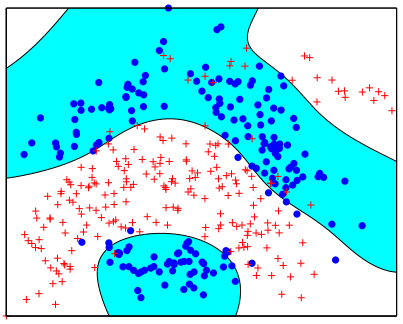
For each permutation of each data set we used the negatively labeled training and test vectors as our normal data set. We then ran our algorithms on the training data and estimated P, μ, and Q_+ using the test vectors and a large set of vectors drawn independently from a hypercube containing the data (or uniformly on the discrete set of feature values as appropriate). Table 1 reports the mean values for P, μ, and Q_+ over 30 permutations, along with standard errors. For each permutation we also computed the performance measures $\mathcal{E}_\mu$ and $\mathcal{E}_+$, and we show the mean and median values in the table. The methods compared are the 2ν-SVM with a windowed grid search over (ν_+, ν_-), and the OC-SVM. For both of these methods, we considered estimating the volume using independent sampling from a hypercube, thinned sampling, and manifold sampling. For all methods, the parameters were selected using 5-fold cross-validation.

Furthermore, in Table 2 we show the results of applying an estimated MV-set to the problem of NP classification. Specifically, we compare the performance of the technique outlined above with

[2] http://ida.first.fhg.de/projects/bench



(a) Classifier (MV-set estimate) *without* knowledge of "+" class.



(b) Classifier *with* knowledge of "+" class.

Fig. 3. *Performance of MV-sets applied to NP classification.*

the performance of a 2ν-SVM which is trained with access to the anomalous data set. An example of this is illustrated in Figure 3.

6. CONCLUSIONS

Our experimental results provide some interesting conclusions as well as some questions for further research. While we do not state the results in Table 1, we compared the methods shown with a thresholded kernel density estimate. While this method seemed to perform relatively well on the "banana" dataset, it was not competitive at all on the other, higher-dimensional datasets. Next, we can see in Table 1 that the two-class method consistently outperforms the one-class when measured with respect to both the mean and median (with respect to the 30 permutations) values of $\mathcal{E}_\mu$ and $\mathcal{E}_+$. In particular, as measured by $\mathcal{E}_\mu$, the manifold sampling two-class method always outperforms all of the one-class methods. When measuring performance using $\mathcal{E}_+$, the manifold sampling two-class method fails to beat the one-class methods on the "ringnorm" dataset, but is still competitive.

Regarding the various sampling methods: For the one-class, the results are generally within the standard errors, and so no conclusions can be drawn except for the case of "ringnorm". This is a high-dimensional dataset, and as expected, the thinning and manifold sampling strategies result in a clear improvement over independent sampling. For the two-class, the three methods are again indistinguishable (up to standard errors) on "banana" and "thyroid", and as before, the manifold sampling strategies result in a marked improvement on "ringnorm". However, there is a clear drop in performance on "heart" and "breast-cancer" when we use the thinning technique. Recall that these two datasets are the only two with any discrete features. The loss in performance in this case is likely due to the fact that our thinning strategy is based on trying to maximize the minimum Euclidean distance between the points. In this setting it might be more appropriate to only apply the thinning technique to the features that are continuous. Note also that as measured by $\mathcal{E}_\mu$, the manifold sampling technique seems to perform extremely well on these two datasets, while this performance does not carry over to $\mathcal{E}_+$. We regard $\mathcal{E}_+$ as the more meaningful quantity in this case, and since this only occurs for the datasets with discrete features, we suspect that for similar reasons as above, the manifold sampling method only *appears* to perform significantly better than independent sampling on these data sets. Both of these issues warrant further investigation.

Finally, Table 2 shows that in four of the five cases, the NP classifier trained on two real classes outperforms the MV-set classifier, which trains on only one class. This is not surprising, although somewhat unexpected is that the miss rates (Q_+) for the two methods are comparable on three of the five datasets. This would seem to indicate that the uniform prior is often a reasonable assumption.

7. REFERENCES

[1] W. Polonik, "Minimum volume sets and generalized quantile processes," *Stochastic Processes and their Applications*, vol. 65, pp. 1–24, 1997.

[2] G. Walther, "Granulometric smoothing," *Annals of Statistics*, vol. 25, pp. 2273–2299, 1997.

[3] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Computation*, vol. 13, pp. 1443–1471, 2001.

[4] C. D. Scott and R. D. Nowak, "Learning minimum volume sets," *Journal of Machine Learning Research*, 2006, Accepted.

[5] D. M. J. Tax and R. P. W. Duin, "Support vector domain description," *Pattern Recognition Letters*, , no. 26, pp. 1191–1199, 1999.

[6] C. D. Scott and R. D. Nowak, "A Neyman-Pearson approach to statistical learning," *IEEE Trans. on Information Theory*, vol. 51, no. 11, pp. 3806–3819, 2005.

[7] M. A. Davenport, R. G. Baraniuk, and C. D. Scott, "Controlling false alarms with support vector machines," in *Proc. Int. Conf. on Acoustics, Speech, and Signal Proc. (ICASSP)*, 2006.

[8] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer, New York, 2001.

[9] J. Theiler and D. M. Cai, "Resampling approach for anomaly detection in multispectral images," in *Proc. SPIE*, 2003, vol. 5093, pp. 230–240.

[10] I. Steinwart, D. Hush, and C. Scovel, "A classification framework for anomaly detection," *Journal of Machine Learning Research*, vol. 6, pp. 211–232, 2005.

[11] D. M. J. Tax and R. P. W. Duin, "Uniform object generation for optimizing one-class classifiers," *Journal of Machine Learning Research*, vol. 2, pp. 155–713, 2001.

[12] C. D. Scott, "Performance measures for Neyman-Pearson classification," Tech. Rep., Rice University, Dept. of Statistics, 2005, See http://www.stat.rice.edu/~cscott.

[13] C. C. Chang and C. J. Lin, *LIBSVM: a library for support vector machines*, 2001, See http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[14] B. Schölkopf and A. J. Smola, *Learning with Kernels*, MIT Press, Cambridge, MA, 2002.

[15] H. G. Chew, R. E. Bogner, and C. C. Lim, "Dual-ν support vector machine with error rate and training size biasing," in *Proc. Int. Conf. on Acoustics, Speech, and Signal Proc. (ICASSP)*, 2001, pp. 1269–1272.

[16] M. A. Davenport, "The 2ν-SVM: A cost-sensitive extension of the ν-SVM," Tech. Rep. TREE 0504, Rice University, Dept. of Elec. and Comp. Engineering, October, 2005, See http://www.ece.rice.edu/~md.

[17] R. S. Wagner, D. B. Johnson R. G. Baraniuk, S. Du, and A. Cohen, "An architecture for distributed wavelet analysis and processing in sensor networks," in *Info. Proc. in Sensor Networks (IPSN)*, 2006.

Table 1. *Mean values and standard errors of P, μ, and Q_+ (over 30 permutations of each data set) and the associated mean and median error scores ($\mathcal{E}_\mu$ and $\mathcal{E}_+$) for the six methods. The tested methods are the OC-SVM and the NP-SVM approaches, where for both methods we tried estimating the volume using independent sampling, thinned sampling, and manifold sampling (denoted Ind, Thin, and Man in the table). Here $\beta = 0.9$, and μ is estimated by generating 10,000 points in a hypercube containing the data.*

		mass(P)	volume (μ)	mean $\mathcal{E}_\mu$	median $\mathcal{E}_\mu$	Q_+	mean $\mathcal{E}_+$	median $\mathcal{E}_+$
banana	OC-Ind	.811 ± .25	.384 ± .12	1.358 ± 2.35	0.500	.326 ± .17	1.301 ± 2.40	0.398
	OC-Thin	.810 ± .24	.383 ± .12	1.360 ± 2.35	0.503	.329 ± .17	1.305 ± 2.40	0.396
	OC-Man	.804 ± .25	.384 ± .12	1.430 ± 2.36	0.500	.343 ± .18	1.389 ± 2.41	0.398
	NP-Ind	.895 ± .03	.367 ± .05	0.532 ± 0.25	0.423	.198 ± .04	0.363 ± 0.23	0.256
	NP-Thin	.895 ± .02	.343 ± .05	0.470 ± 0.16	0.420	.184 ± .03	0.310 ± 0.14	0.246
	NP-Man	.899 ± .02	.350 ± .05	0.444 ± 0.15	0.403	.193 ± .04	0.287 ± 0.13	0.231
breast	OC-Ind	.884 ± .05	.297 ± .08	0.550 ± 0.49	0.369	.800 ± .09	1.053 ± 0.45	0.926
	OC-Thin	.889 ± .04	.298 ± .08	0.494 ± 0.32	0.354	.801 ± .10	0.997 ± 0.31	0.895
	OC-Man	.919 ± .04	.586 ± .15	0.693 ± 0.26	0.702	.882 ± .07	0.988 ± 0.26	0.917
	NP-Ind	.897 ± .04	.121 ± .02	0.293 ± 0.29	0.141	.625 ± .09	0.797 ± 0.29	0.688
	NP-Thin	.826 ± .11	.905 ± .12	1.747 ± 0.86	1.388	.724 ± .17	1.567 ± 0.81	1.221
	NP-Man	.927 ± .04	.007 ± .00	0.064 ± 0.14	0.008	.773 ± .09	0.830 ± 0.16	0.811
heart	OC-Ind	.883 ± .06	.283 ± .08	0.628 ± 0.39	0.472	.613 ± .09	0.958 ± 0.38	0.823
	OC-Thin	.883 ± .07	.292 ± .08	0.647 ± 0.45	0.480	.619 ± .10	0.973 ± 0.45	0.820
	OC-Man	.916 ± .04	.458 ± .10	0.577 ± 0.24	0.508	.729 ± .10	0.848 ± 0.22	0.779
	NP-Ind	.883 ± .05	.136 ± .04	0.432 ± 0.36	0.268	.411 ± .09	0.707 ± 0.38	0.542
	NP-Thin	.834 ± .09	.496 ± .25	1.257 ± 0.64	1.061	.282 ± .16	1.044 ± 0.66	0.829
	NP-Man	.918 ± .06	.002 ± .00	0.155 ± 0.31	0.002	.607 ± .14	0.761 ± 0.24	0.695
thyroid	OC-Ind	.877 ± .07	.371 ± .11	0.767 ± 0.57	0.579	.145 ± .10	0.541 ± 0.56	0.314
	OC-Thin	.867 ± .07	.382 ± .11	0.837 ± 0.63	0.502	.117 ± .09	0.572 ± 0.61	0.332
	OC-Man	.897 ± .07	.528 ± .16	0.866 ± 0.48	0.741	.161 ± .11	0.499 ± 0.43	0.302
	NP-Ind	.880 ± .05	.319 ± .10	0.630 ± 0.46	0.511	.240 ± .15	0.551 ± 0.45	0.385
	NP-Thin	.866 ± .07	.325 ± .10	0.788 ± 0.56	0.621	.097 ± .05	0.561 ± 0.51	0.384
	NP-Man	.870 ± .07	.294 ± .09	0.701 ± 0.67	0.519	.034 ± .04	0.441 ± 0.65	0.240
ringnorm	OC-Ind	.904 ± .03	.022 ± .01	0.106 ± 0.16	0.035	.005 ± .002	0.089 ± 0.16	0.009
	OC-Thin	.925 ± .02	.041 ± .01	0.053 ± 0.03	0.042	.008 ± .003	0.020 ± 0.03	0.008
	OC-Man	.926 ± .02	.039 ± .01	0.053 ± 0.03	0.042	.007 ± .003	0.020 ± 0.03	0.009
	NP-Ind	.892 ± .03	.020 ± .01	0.174 ± 0.20	0.079	.005 ± .001	0.158 ± 0.20	0.057
	NP-Thin	.960 ± .03	.104 ± .04	0.105 ± 0.04	0.094	.018 ± .009	0.020 ± 0.01	0.019
	NP-Man	.941 ± .02	.052 ± .03	0.058 ± 0.03	0.049	.010 ± .005	0.015 ± 0.03	0.009

Table 2. *Mean values and standard errors of Q_- and Q_+ (over 30 permutations of each data set) and the associated mean and median error scores. The two methods we compare are denoted "Without" and "With". The "Without" method uses the NP-SVM technique with an artificially generated positive class (generated using manifold sampling). The "Without" method only has access to the negative class during training. The "With" method also uses the NP-SVM technique but has access to both classes during training. Here $\beta = 0.9$, or equivalently, $\alpha = 0.1$.*

		Q_-	Q_+	mean $\mathcal{E}_+$	median $\mathcal{E}_+$
banana	Without	.102 ± .02	.193 ± .04	.287 ± .13	.231
	With	.104 ± .02	.124 ± .02	.243 ± .14	.160
breast	Without	.073 ± .04	.773 ± .09	.830 ± .16	.811
	With	.112 ± .06	.689 ± .10	.985 ± .41	.821
heart	Without	.082 ± .06	.607 ± .14	.761 ± .24	.542
	With	.113 ± .05	.231 ± .07	.497 ± .37	.326
thyroid	Without	.130 ± .07	.034 ± .04	.441 ± .65	.240
	With	.087 ± .06	.032 ± .05	.222 ± .37	.051
ringnorm	Without	.059 ± .02	.010 ± .01	.015 ± .03	.009
	With	.074 ± .02	.008 ± .01	.021 ± .04	.008