RICE UNIVERSITY

Error Control for Support Vector Machines

by

Mark A. Davenport

A THESIS SUBMITTED
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE

Master of Science

Approved, Thesis Committee:

Richard Baraniuk, Chair,
Victor E. Cameron Professor,
Electrical and Computer Engineering

Don H. Johnson,
J. S. Abercrombie Professor,
Electrical and Computer Engineering,
Statistics

Rudolf H. Riedi,
Associate Professor,
Electrical and Computer Engineering,
Statistics

Clayton D. Scott,
Assistant Professor,
Electrical Engineering and Computer Science,
University of Michigan

Houston, Texas

April 2007

Abstract

Error Control for Support Vector Machines
by
Mark A. Davenport

In binary classification there are two types of errors, and in many applications these may have very different costs. We consider two learning frameworks that address this issue: minimax classification, where we seek to minimize the maximum of the false alarm and miss rates, and Neyman-Pearson (NP) classification, where we seek to minimize the miss rate while ensuring the false alarm rate is less than a specified level α. We show that our approach, based on cost-sensitive support vector machines, significantly outperforms methods typically used in practice. Our results also illustrate the importance of heuristics for improving the accuracy of error rate estimation in this setting. We then reduce anomaly detection to NP classification by considering a second class of points, allowing us to estimate minimum volume sets using algorithms for NP classification. Comparing this approach with traditional one-class methods, we find that our approach has several advantages.

Acknowledgements

The work presented in this thesis was accomplished with a great deal of help from Clay Scott. I greatly appreciate his assistance and insight, and especially his occasional prodding. Thanks also to my advisor, Richard Baraniuk, for his enthusiastic encouragement, and to the additional members of my committee, Don Johnson and Rudolf Riedi, for their helpful feedback.

I am also deeply indebted to Ryan King and Brandon Skeen who helped provide the computational resources without which this work would not have been possible.

Finally, I would like to thank all of my other friends and family for their support. Thanks especially to my parents for (perhaps unwittingly) pointing me down this road, and to Kim for her constant reassurance, optimism, and encouragement.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The goal of any learning algorithm is to formulate a rule, or *classifier*, that generalizes from the given data to new situations in a "reasonable" way. Specifically, given a sample of *training vectors* $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$ along with corresponding *labels* $y_1, \ldots, y_n \in \{-1, +1\}$, we aim to find a function $f : \mathbb{R}^d \to \{+1, -1\}$ that "accurately" predicts the label when presented with a new feature vector, where accuracy is usually equated with having a small probability of error.

However, there are two distinct types of errors, and in many important applications these may have very different costs. In tumor classification, for example, the impact of mistakenly classifying a benign tumor as malignant is much less than that of the opposite mistake. Furthermore, even if the two types of errors have comparable costs, we would often prefer to avoid classifiers that perform much better on one class than the other. This is particularly a problem when we have many more training samples from one class than from the other, since algorithms that attempt to minimize the probability of error will tend to place less emphasis on the smaller class.

1.1 Cost-Sensitive Learning

We shall explore two related frameworks for addressing these challenges. In the *minimax* framework we seek to minimize the maximum of the false alarm and miss rates. This criterion is a natural goal in the absence of any knowledge about the prior probabilities for the two classes, as it will avoid classifiers that perform much better on one class than the other. Alternatively, in the *Neyman-Pearson* (NP) framework the user sets a target false alarm rate α, and the goal of the algorithm is to minimize the miss rate subject to the condition that the false alarm rate is no greater than α. This classical approach is also quite natural in many settings, especially when one class is more or less important than the other. The two frameworks are related in that they both attempt to design classifiers that operate at a specific point of the *receiver operating characteristic* (ROC) *curve*, but differ in how the specific point is chosen. In NP classification, the user selects the point through α, while in minimax classification α is automatically chosen to obtain equal false alarm and miss rates.

In contrast to the *two-class* problems described above, *anomaly detection* is inherently a *one-class* problem in that we are only able to observe data from a single class. Given such training data, we aim to estimate a classifier that discriminates between a typical feature vector (corresponding to the observed class) and an anomalous feature vector. A natural approach here is to estimate the *minimum volume set* (MV-set), which is roughly the set with the smallest volume among all those containing a fraction of at least β of the observed data, where β is a parameter set by the user. We show that it is possible to use an NP classification algorithm to estimate an MV-set by artificially generating a second class of points that serves as an estimator for the volume of candidate MV-sets.

Traditional wisdom holds that we should be able to solve all of these problems by simply applying the "cost-sensitive" extensions that exist for many common classification algorithms. These algorithms seek to minimize a more general "misclassification cost" rather than the probability of error. For example, support vector machines (SVMs) – a powerful, kernel-based method for classification – can be modified to do this by introducing an additional parameter that that penalizes errors from one class more than the other. The challenge is that in practical settings like those described above, we have specific error rates we would like to achieve, i.e., we wish to operate at a specific point of the ROC. Obtaining this performance will require that we set the additional parameter appropriately. This work highlights the importance of accurate error estimation in solving these problems – it is precisely the ability of an algorithm to leverage accurate error estimation techniques to adjust the free parameters appropriately that will determine whether an algorithm will be able to give us the desired performance. Furthermore, in comparing different algorithms we emphasize that great care should be taken to ensure that we use a performance measure that is truly meaningful for the problem of interest.

1.2 Minimax Learning

First, we consider the traditional binary classification setting. Hence, 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. Let $f : \mathbb{R}^d \to \{+1, -1\}$ be a classifier, and let

$$P_F(f) = Q_-(\{\mathbf{x} : f(\mathbf{x}) = +1\}) = \Pr(f(\mathbf{x}) = +1|y = -1) \quad \text{and} \tag{1.1}$$

$$P_M(f) = Q_+(\{\mathbf{x} : f(\mathbf{x}) = -1\}) = \Pr(f(\mathbf{x}) = -1|y = +1) \tag{1.2}$$

denote the *false alarm* and *miss* rates of f, respectively.

As an example, suppose that our two classes represent patients with two different types of cancer. In the absence of some prior information about these classes, such as the relative frequencies or severity of the two types, we have no reason to favor one class over the other. However, unless we have exactly the same number of training vectors from each class, minimizing the probability of error will implicitly do exactly that – assigning more weight to the class with more training vectors. For this reason (among others), many researchers prefer to use classifiers operating at the the *break even point* (BEP) or *equal error rate* (EER) (where $P_F(f) = P_M(f)$). See, for example, [1, 27]. Of course, if an algorithm is sufficiently flexible, it is possible that it may yield many classifiers satisfying this constraint. We seek the best one possible, which we shall denote the *minimax* classifier. Specifically, the minimax classifier is defined as

$$f^*_{mm} = \arg \min_f \max \{P_F(f), P_M(f)\}. \tag{1.3}$$

In order to adapt SVMs to estimate the minimax classifier, we will obtain estimates of $P_F(f)$ and $P_M(f)$ for different parameter settings, and then select the "optimal" parameters according to (1.3). An alternative kernel-based approach to minimax classification is that of the so-called minimax probability machine (MPM) [15]. We compare these methods as well as additional SVM-based approaches in detail in Chapter 3.

The minimax framework has two major advantages with respect to conventional classification criteria that seek to minimize an expected misclassification cost. First, assigning the costs appropriately is often difficult (see [2,20]). Second, unlike efforts to minimize a misclassification cost, the minimax framework does not implicitly assume knowledge of the prior probabilities of the two classes. This is extremely important in applications where the class frequencies in the training data do not accurately reflect class probabilities in the larger population. In fact, it could be argued that most classification problems of interest fit this description.

1.3 Neyman-Pearson Learning

In the minimax setting we are concerned with the challenges posed by unknown prior probabilities and potentially disparate costs for the two types of errors. An alternative approach to addressing these two challenges is through the NP paradigm.

NP classification is typically motivated by describing situations where errors from one class are much more costly than errors from another class. As an example, suppose now that the positive class represents patients with cancer and the negative class represents patients without cancer. Here a miss corresponds to failing to detect cancer that is present in a patient, while a false alarm corresponds to detecting cancer when it is not actually present. Clearly, the cost associated with a miss is much different than that associated with a false alarm. In this case we might only be able to tolerate a certain level of false alarms, in which case we would want to have the lowest miss rate possible provided the false alarm rate satisfies some constraint. This approach corresponds to the goal of the Neyman-Pearson (NP) classification framework. Specifically, given a user-specified significance level α, the NP-optimal classifier is defined as

$$f_\alpha^* = \underset{f:P_F(f)\leq\alpha}{\arg\min}\ P_M(f). \tag{1.4}$$

Figure 1.1 illustrates an example of an estimate of such a classifier.

The NP framework shares the same advantages over conventional cost-sensitive classification approaches as the minimax framework. Specifically, specifying a desired false alarm rate is typically more natural than assigning the costs, and the NP framework does not implicitly assume knowledge of the prior probabilities of the two classes. See the theoretical analysis provided in [25] for a more in-depth discussion of NP classification. Building on the initial results published in [11], we illustrate that SVMs can be adapted to this setting through the use of the same techniques used to

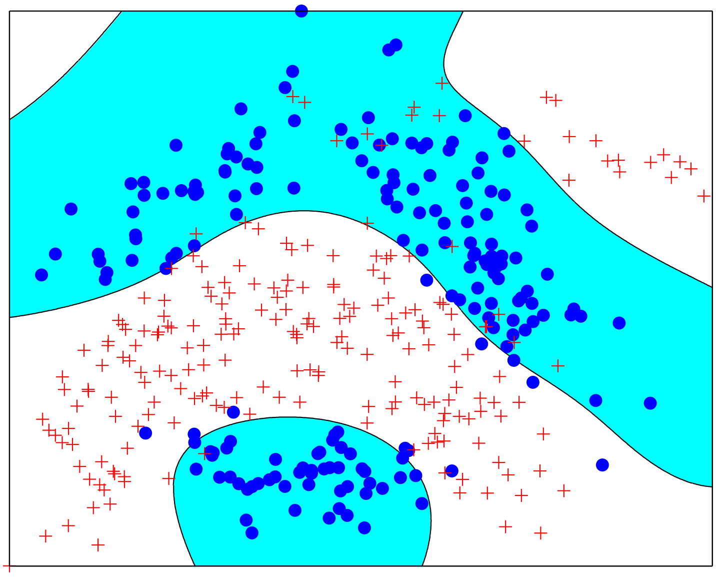


Figure 1.1: *Estimated Neyman-Pearson classifier for "banana" data set* ($\alpha = 0.1$).

adapt SVMs to the minimax setting, providing an efficient and practical algorithm for NP classification. While similar to our approach to minimax classification, there are important differences in both the methods for parameter selection and performance evaluation since in the NP setting we wish to attain a specified false alarm rate.

1.4 Anomaly Detection with Minimum-Volume Sets

We now consider the problem of *anomaly detection*, where we are only given training vectors from only one of the two classes – the *normal*, or *typical* class. Our goal is to use this data to find a classifier that categorizes new feature vectors as being either normal/typical or abnormal/anomalous without ever observing any abnormal/anomalous data. 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. In addition, even in cases where we do observe anomalous data, it is often difficult or impossible to identify the anomalous training vectors *a priori*. In essence, we would like to be able to detect anomalies without knowing what they look like.

We can think of our classifier as defining a subset of $\mathbb{R}^d$ such that the points inside the set correspond to typical samples, while points outside the subset are anomalies. We want this set to contain most of the typical samples, but be as small as possible so as to maximize our chances of detecting an anomaly. Specifically, let $\beta \in (0, 1)$ be given and suppose P is the probability measure governing the typical data and μ is a known reference measure. We would like to estimate the *minimum volume set* (MV-set)

$$G^*_\beta = \arg\min\{\mu(G) : P(G) \geq \beta, G \text{ measurable}\}, \tag{1.5}$$

or in other words, we want to find the set of minimum volume (as measured by μ) among all those containing a fraction of at least β of the probability mass of P. The parameter β is chosen by the user and reflects a desired false alarm rate of $1 - \beta$.

We can interpret MV-sets in a number of ways. First, 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. Furthermore, MV-sets are density level sets, with β defining a density level and vice-versa. See [12, 19, 22, 26, 38] and references therein for additional discussion. We can

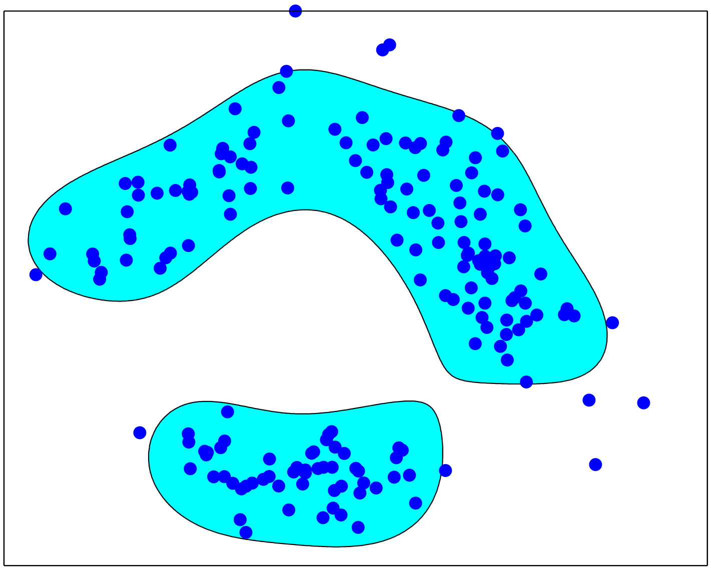


Figure 1.2: *Estimated minimum volume set for "banana" data set ($\beta = 0.9$).*

also view the reference measure μ in multiple ways: either as our definition of volume on $\mathbb{R}^d$ or as a prior on the distribution for anomalies. In this light, taking μ to be the Lebesgue/uniform measure can be interpreted as assuming a noninformative prior on anomalies. Thus, in this thesis we focus on the common case where μ is the Lebesgue measure, although most of our techniques extend easily to other measures.

Hence, our task is the following: given n realizations of a distribution P, a reference measure μ, and $\beta \in (0,1)$, construct a set $\widehat{G}_\beta$ that approximates the true MV-set G^*_β. An example of such a set is shown in Figure 1.2. Since MV-sets are density level sets, one possible approach to MV-set estimation is to use the so-called one-class SVM (OC-SVM) [22, 31] to estimate the appropriate level set. Our application of the OC-SVM entails carefully setting the free parameters to achieve the desired mass/volume tradeoff. Alternatively, we may reduce MV-set estimation to Neyman-Pearson (NP) classification as described in [26]. Specifically, 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. (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 training vectors, $\{\mathbf{x}_i\}_{i=1}^n$, labels of -1. We then draw a set of new training vectors according to 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 μ, minimizing $P_M(f)$ is equivalent to minimizing an estimate of $\mu(\{\mathbf{x} : f(\mathbf{x}) = -1\})$. Hence, 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. This idea was initially explored in [12].

The idea of reducing a supervised problem to an unsupervised problem by sampling from a reference measure has apparently been known for some time [14]. Although they do not speak in terms of "Neyman-Pearson classification," the reduction outlined above is essentially described in [33]. The two-class idea was also applied in [30] to reduce density level set estimation to cost-sensitive classification. In a kind of hybrid between one- and two-class methods, [32] employs artificial uniform data to select the parameters of the OC-SVM.

The OC-SVM has recently been proven to be a universally consistent estimator of density level sets [36], and from the consistency results of [29] for the two-class SVM, we can see that the NP approach is consistent as well. However, it is not immediately apparent which method will be superior with finite sample sizes. A key challenge in both approaches is that they require the generation of 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. In the OC-SVM approach, we use this data to adjust free-parameters appropriately. In the NP approach we train our SVM using the data, and thus in this case we must limit the number of realizations from μ to control training time. Thus in the NP setting, 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 effect can be viewed as one aspect of the "curse of dimensionality".

We consider two methods for overcoming these challenges. The first involves drawing many more points than are ultimately desired and then adaptively removing points, or *thinning*, to obtain 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, which we call *manifold sampling*, does address this concern and also adapts to potential manifold structure in the data.

1.5 Cost-Sensitive SVMs

SVMs are a powerful kernel-based classification method which can be extended to the cost-sensitive setting by introducing an additional parameter that penalizes the errors asymmetrically. This approach has been taken by several authors to develop cost-sensitive versions of the C-SVM, and a closely related algorithm, the ν-SVM (for example, see [5, 6, 17, 18, 35]). We shall refer to these extensions as the $2C$-SVM and the 2ν-SVM respectively. It was shown in [17] that the C-SVM and ν-SVM are equivalent in the sense that they explore the same set of potential classifiers as their free parameters (C and ν) are varied. We extend this result to show that the same is true of the $2C$-SVM and the 2ν-SVM.

The challenge in adapting either of these extensions to minimax or NP classification is that we cannot know *a priori* how to set the relative costs in order to achieve the desired false alarm and miss rates. The way around this problem is to estimate the error rates for a grid of possible values of the cost parameters using an error estimation method such as cross-validation, selecting the parameters which yield the desired performance. Obviously this error estimation process is critical to achieving the desired false alarm and miss rates.

With this in mind, we argue that the 2ν-SVM has several advantages over the $2C$-SVM. In particular, the 2ν-SVM can be formulated so that its two free parameters,

ν_+ and ν_-, each lie in the interval $[0, 1]$, which is not possible for the $2C$-SVM. This makes the starting points, ending points, and spacing for any grid search much less arbitrary. Furthermore, a number of heuristics for improving the parameter selection process are more natural in this setting. Therefore we will restrict our attention to the 2ν-SVM, although many of our conclusions would likely hold for the $2C$-SVM as well.

1.6 Performance Evaluation

Suppose we have two classifiers (e.g., two SVMs with different parameter settings) and compute empirical estimates of the false alarm and miss rates for each. Which classifier should we choose? In some cases there may be an expert who can decide. But what if no expert is available? Or, what if we want to compare two learning algorithms across several datasets? Even if an expert is available, experts usually cost money or have limited time, and their decisions may be subjective or prone to human error.

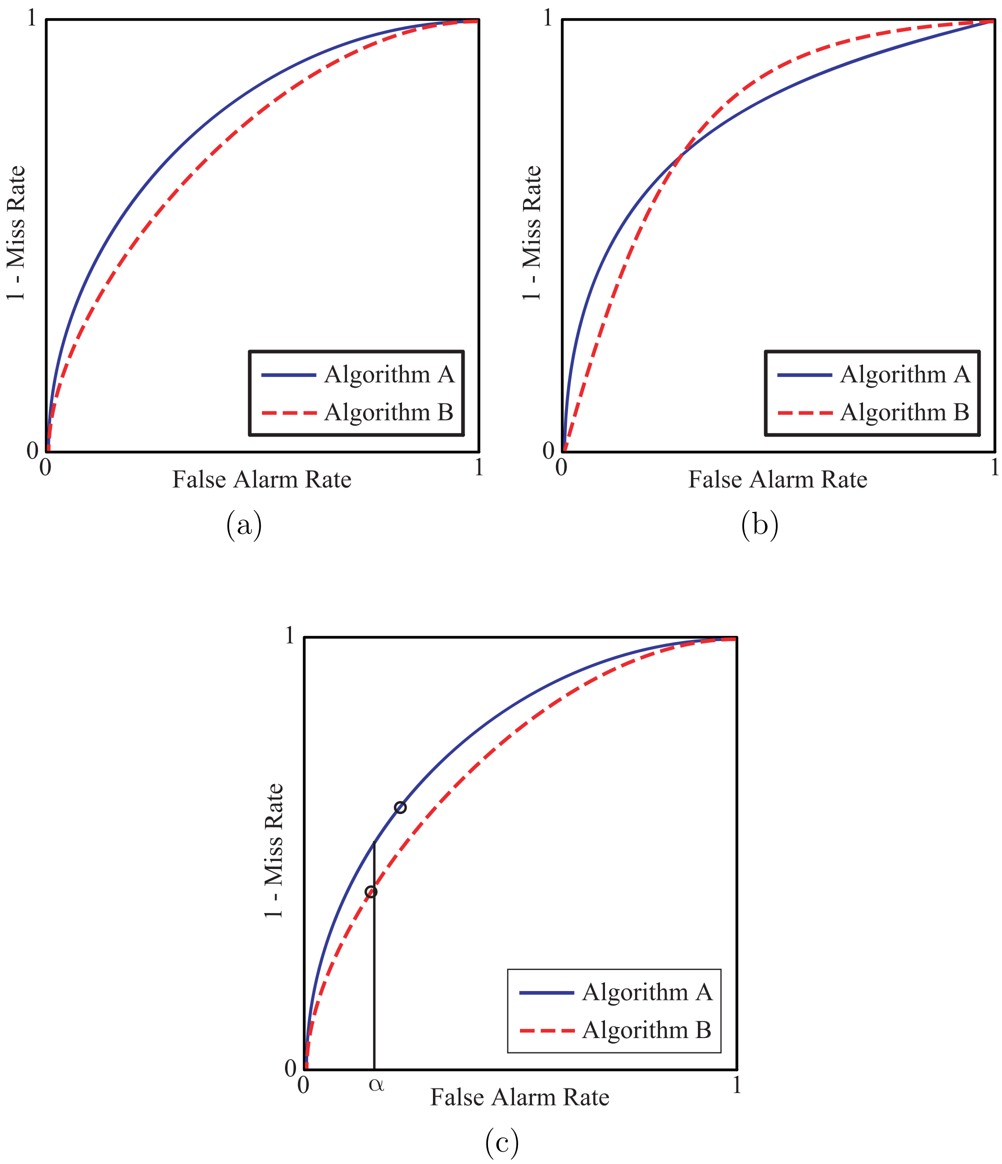
While it seems desirable to have an objective measure for evaluating and comparing the trained classifiers, most papers proposing algorithms for cost-sensitive classification or anomaly detection do not adopt a systematic methodology for comparing different algorithms. The typical paper introduces a new algorithm and provides an ROC curve that conveys the ability of the algorithm to trade off false alarms for misses. An algorithm is said to *dominate* another algorithm if its ROC curve is always equal to or above the other. For example, in Figure 1.3(a) algorithm A clearly dominates algorithm B. One clear problem with using ROC curves to measure performance is that in practice no single algorithm will dominate the others, as is illustrated in Figure 1.3(b). To compare two or more algorithms we really need a scalar performance criterion. The most common such criterion is the area under the ROC curve (AUC). Numerous studies indicate the advantages of AUC maximization over misclassification rate minimization, even when the final objective is to minimize misclassification rate [2, 8, 20, 21].

However, in most practical settings – including many of the settings commonly used to motivate the use of the AUC – we really want to operate at a *specific point* on the ROC curve, and unfortunately there is no oracle to tell us how to set the free parameter(s) in the algorithm to achieve this point. In other words, the AUC evaluates a family of classifiers (that trace out an ROC), but it does not recommend a particular member of that family. For example, in NP classification or MV-set estimation, we have a target false alarm rate (α) at which we hope to operate. In this case, it does not matter if an algorithm has a slightly higher ROC curve or AUC if the algorithm gives us little control over where on the ROC curve we actually operate. As an example, in Figure 1.3(c) algorithm A dominates algorithm B, but we are able to tightly control the false alarm rate with algorithm B, while with algorithm A we can greatly exceed the desired level. In this case it seems clear that even though the AUC of algorithm A exceeds that of algorithm B, algorithm B is actually the better choice for our purposes. While this may seem unlikely, it has been observed in practical settings, with extremely negative consequences for the case of minimax classification in particular, as described in [1]. In summary, 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 (or miss) rate?"

Thus, in the case where we desire a specific false alarm rate, as in NP classification or MV-set estimation, we advocate a new performance measure for comparing alternative algorithms [24]. In the minimax classification problem we simply argue that one should do the obvious, use the maximum of the false alarm and miss rates, rather than the often used error rate or AUC. Our emphasis on a systematic comparison highlights the importance of *error estimation* to cost-sensitive classification or anomaly detection problems. Not only should an algorithm be flexible in the sense of having a large AUC, but it should also be possible to accurately estimate the error rates for various parameter settings so that the free parameters of the algorithm can be set appropriately.

1.7 Outline of Thesis

In Chapter 2 we briefly review SVMs, introduce the cost-sensitive extensions of the C-SVM and ν-SVM – the $2C$-SVM and 2ν-SVM, and list some of their properties, including a detailed description of the relationship between these two formulations which generalizes results of [4,7]. In Chapters 3, 4, and 5 we describe how the 2ν-SVM



(a)

(b)

(c)

Figure 1.3: *Examples of ROC curves. (a) A pair ROC curves where algorithm A dominates algorithm B. (b) A more typical example of a pair of ROC curves where neither algorithm A nor algorithm B dominates. (c) A practical example where we want to operate at a specific point on the ROC curve – even though algorithm A dominates algorithm B, we might prefer algorithm B if it offers better control over where on the ROC curve we actually operate.*

can be used for minimax classification, NP classification, and MV-set estimation, providing detailed experimental results for each algorithm on a collection of benchmark data sets. Finally, we conclude in Chapter 6 with discussion and directions for future research.

Chapter 2

Support Vector Machines

2.1 Review of SVMs

SVMs are among the most effective methods for classification. Conceptually, we construct a support vector classifier in a two step process. In the first step, we transform the $\mathbf{x}_i$ via a mapping $\Phi : \mathbb{R}^d \to \mathcal{H}$ where $\mathcal{H}$ is a high (possibly infinite) dimensional Hilbert space. Our intuition is that we should be able to more easily separate the two classes in $\mathcal{H}$ than in $\mathbb{R}^d$. For algorithmic reasons, Φ must be chosen so that the *kernel* operator $k(\mathbf{x}, \mathbf{x}') = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle_{\mathcal{H}}$ is positive definite. This criterion allows us to compute inner products in $\mathcal{H}$ without explicitly evaluating Φ.

In the second step, we determine a hyperplane in the induced feature space according to the max-margin principle. In the case where we can separate the two classes by a hyperplane, the SVM chooses the hyperplane that maximizes the *margin* – the distance between the decision boundary and the closest point to the boundary. When we cannot separate the classes by a hyperplane, we relax the constraints through the introduction of slack variables ξ_i. If $\xi_i > 0$, this means that the corresponding $\mathbf{x}_i$ lies inside the margin and is called a *margin error.* If $\mathbf{w} \in \mathcal{H}$ and $b \in \mathbb{R}$ are the normal vector and affine shift defining the max-margin hyperplane, then the support vector classifier is given by $f_{\mathbf{w},b}(\mathbf{x}) = \mathrm{sgn}(\langle \mathbf{w}, \Phi(\mathbf{x}) \rangle_{\mathcal{H}} + b)$. The offset parameter b is often called the *bias.*

There are two different formulations of the SVM. The original SVM [9] – the C-SVM – can be formulated as the following quadratic program:

$$(P_C) \qquad \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \xi_i$$

$$\begin{aligned} \text{subject to} \qquad y_i(k(\mathbf{w}, \mathbf{x}_i) + b) &\geq 1 - \xi_i \qquad && \text{for } i = 1, \dots, n \\ \xi_i &\geq 0 && \text{for } i = 1, \dots, n \end{aligned}$$

where $C \geq 0$ is a parameter that controls overfitting.

For computational reasons, it is often easier to solve (P_C) by solving its dual:

$$(D_C) \qquad \min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) - \sum_{i=1}^{n} \alpha_i$$

$$\begin{aligned} \text{subject to} \qquad 0 \leq \alpha_i &\leq C \qquad && \text{for } i = 1, \dots, n \\ \sum_{i=1}^{n} \alpha_i y_i &= 0. \end{aligned}$$

We derive this formulation by forming the Lagrangian ($\boldsymbol{\alpha}$ is a Lagrange multiplier). The primal and the dual are related through $\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \Phi(\mathbf{x}_i)$. We will often have that $\alpha_i = 0$ for most $\mathbf{x}_i$. We call the $\mathbf{x}_i$ for which $\alpha_i \neq 0$ the *support vectors.*

An alternative (but equivalent) formulation of the C-SVM is the ν-SVM [23], which replaces C with a different parameter $\nu \in [0,1]$ that serves as an upper bound on the fraction of margin errors and a lower bound on the fraction of support vectors. The ν-SVM has the primal formulation

$$
\begin{array}{llll}
(P_\nu) & & \displaystyle\min_{\mathbf{w},b,\boldsymbol{\xi},\rho} \quad \frac{1}{2}\|\mathbf{w}\|^2 - \nu\rho + \frac{1}{n}\sum_{i=1}^{n}\xi_i & \\
 & \text{subject to} & y_i(k(\mathbf{w},\mathbf{x}_i)+b) \geq \rho - \xi_i & \text{for } i=1,\ldots,n \\
 & & \xi_i \geq 0 & \text{for } i=1,\ldots,n \\
 & & \rho \geq 0 &
\end{array}
$$

and dual formulation

$$
\begin{array}{llll}
(D_\nu) & & \displaystyle\min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j k(\mathbf{x}_i,\mathbf{x}_j) & \\
 & \text{subject to} & \displaystyle 0 \leq \alpha_i \leq \frac{1}{n} & \text{for } i=1,\ldots,n \\
 & & \displaystyle\sum_{i=1}^{n}\alpha_i y_i = 0, \quad \sum_{i=1}^{n}\alpha_i \geq \nu. &
\end{array}
$$

2.2 Cost-Sensitive SVMs

The above formulations implicitly penalize errors in both classes equally. However, as described in the introduction, in many applications different are costs associated with the two different kinds of errors. To address this issue, cost-sensitive extensions of both the C-SVM and the ν-SVM have been proposed – the $2C$-SVM and the 2ν-SVM.

First we will consider the $2C$-SVM proposed in [18]. Let $I_+ = \{i : y_i = +1\}$ and $I_- = \{i : y_i = -1\}$. The $2C$-SVM has primal

$$
(P_{2C}) \qquad \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\gamma \sum_{i \in I_+} \xi_i + C(1-\gamma) \sum_{i \in I_-} \xi_i
$$

$$
\begin{array}{lll}
\text{subject to} & y_i(k(\mathbf{w}, \mathbf{x}_i) + b) \geq 1 - \xi_i & \text{for } i = 1, \ldots, n \\
 & \xi_i \geq 0 & \text{for } i = 1, \ldots, n
\end{array}
$$

and dual

$$
(D_{2C}) \qquad \min_{\boldsymbol{\alpha}} \quad \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) - \sum_{i=1}^{n} \alpha_i
$$

$$
\begin{array}{lll}
\text{subject to} & 0 \leq \alpha_i \leq C\gamma & \text{for } i \in I_+ \\
 & 0 \leq \alpha_i \leq C(1-\gamma) & \text{for } i \in I_- \\
 & \sum_{i=1}^{n} \alpha_i y_i = 0 &
\end{array}
$$

where $C > 0$ is again a parameter that controls overfitting and $\gamma \in [0, 1]$ is a parameter for trading off the two types of errors. Similarly, [6] proposed the 2ν-SVM as a cost-sensitive extension of the ν-SVM. The 2ν-SVM has primal

$$
(P_{2\nu}) \qquad \min_{\mathbf{w},b,\boldsymbol{\xi},\rho} \frac{1}{2}\|\mathbf{w}\|^2 - \nu\rho + \frac{\gamma}{n} \sum_{i \in I_+} \xi_i + \frac{1-\gamma}{n} \sum_{i \in I_-} \xi_i
$$

$$
\begin{array}{lll}
\text{subject to} & y_i(k(\mathbf{w}, \mathbf{x}_i) + b) \geq \rho - \xi_i & \text{for } i = 1, \ldots, n \\
 & \xi_i \geq 0 & \text{for } i = 1, \ldots, n \\
 & \rho \geq 0 &
\end{array}
$$

and dual

$$(D_{2\nu}) \qquad \min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j)$$

$$\begin{aligned} \text{subject to} \qquad & 0 \le \alpha_i \le \frac{\gamma}{n} && \text{for } i \in I_+ \\ & 0 \le \alpha_i \le \frac{1-\gamma}{n} && \text{for } i \in I_- \\ & \sum_{i=1}^{n} \alpha_i y_i = 0, \quad \sum_{i=1}^{n} \alpha_i \ge \nu \end{aligned}$$

where $\nu \in [0,1]$ again controls overfitting and $\gamma \in [0,1]$ allows us to trade off between the two types of errors.

2.3 Properties of the 2ν-SVM

Before illustrating the relationship between the $2C$-SVM and the 2ν-SVM, we establish some of the basic properties of the 2ν-SVM.

Proposition 1. *Fix $\gamma \in [0,1]$ and let $n_+ = |I_+|$, $n_- = |I_-|$. Then $(D_{2\nu})$ is feasible if and only if $\nu \le \nu_{\max} \le 1$, where*

$$\nu_{\max} = \frac{2\min(\gamma n_+, (1-\gamma)n_-)}{n}.$$

Proof. First, assume that $\nu \le \nu_{\max}$. Then we can construct an $\boldsymbol{\alpha}$ that satisfies the constraints of $(D_{2\nu})$. Specifically, let

$$\alpha_i = \frac{\nu_{\max}}{2n_+} = \frac{\min(\gamma, (1-\gamma)n_-/n_+)}{n} \leq \frac{\gamma}{n} \; for \; i \; \in I_+$$

and

$$\alpha_i = \frac{\nu_{\max}}{2n_-} = \frac{\min(\gamma n_+/n_-, 1-\gamma)}{n} \leq \frac{1-\gamma}{n} \; for \; i \; \in I_-.$$

Then $\sum_{i \in I_+} \alpha_i + \sum_{i \in I_+} \alpha_i = \nu_{\max} \geq \nu$ and $\sum_{i=1}^n \alpha_i y_i = 0$. Thus we have a feasible solution, and so $(D_{2\nu})$ is feasible.

Now assume that $(D_{2\nu})$ is feasible. Then there exists an $\boldsymbol{\alpha}$ such that $\sum_{i=1}^n \alpha_i \geq \nu$ and $\sum_{i \in I_+} \alpha_i = \sum_{i \in I_-} \alpha_i$. Combining this we obtain $\nu \leq 2 \sum_{i \in I_+} \alpha_i$. Since we also have $0 \leq \alpha_i \leq \gamma/n$ for $i \in I_+$, we see that $\nu \leq 2 \sum_{i \in I_+} \alpha_i \leq 2\gamma n_+/n$, and therefore, $\nu \leq 2\gamma n_+/n$. Similarly, $\nu \leq 2(1-\gamma)n_-/n$. Thus $\nu \leq \nu_{\max}$.

Finally, we can see that

$$\nu_{\max} = \frac{2\min(\gamma n_+, (1-\gamma)n_-)}{n} \leq \frac{2\min(n_+, n_-)}{n} \leq 1,$$

as desired. □

Proposition 2. *Fix $\gamma \in [0,1]$ and $\nu \in [0, \nu_{\max}]$. There is at least one optimal solution of $(D_{2\nu})$ that satisfies $\sum_{i=1}^n \alpha_i = \nu$. In addition, if the optimal objective value of $(D_{2\nu})$ is not zero, all optimal solutions of $(D_{2\nu})$ satisfy $\sum_{i=1}^n \alpha_i = \nu$.*

Proof. This proposition was proved in [4] for (D_ν). The proof relies only on the form of the objective function of (D_ν), which is identical to that of $(D_{2\nu})$. Thus, we omit it for the sake of brevity and refer the reader to [4]. □

We will later make use of the fact that the 2ν-SVM as proposed in [6] is parameterized in a different manner than $(D_{2\nu})$. Specifically, instead of parameters ν and γ, we can formulate $(D_{2\nu})$ using ν_+ and ν_-, where

$$\nu = \frac{2\nu_+\nu_- n_+ n_-}{(\nu_+ n_+ + \nu_- n_-)n}, \quad \gamma = \frac{\nu_- n_-}{\nu_+ n_+ + \nu_- n_-} = \frac{\nu n}{2\nu_+ n_+}.$$

or equivalently

$$\nu_+ = \frac{\nu n}{2\gamma n_+}, \quad \nu_- = \frac{\nu n}{2(1-\gamma)n_-}.$$

This parametrization has the benefit that ν_+ and ν_- have a more intuitive meaning illustrated by the following result.

Proposition 3. *Suppose that the optimal objective value of* $(D_{2\nu})$ *is not zero. Then for the optimal solution of* $(D_{2\nu})$*:*

1. ν_+ *is an upper bound on the fraction of margin errors from class* $+1$.

2. ν_- *is an upper bound on the fraction of margin errors from class* -1.

3. ν_+ *is a lower bound on the fraction of support vectors from class* $+1$.

4. ν_- *is a lower bound on the fraction of support vectors from class* -1.

Proof. See [6] for the proof. □

Proposition 4. *($D_{2\nu}$) is feasible if and only if $\nu_+ \leq 1$ and $\nu_- \leq 1$.*

Proof. From Proposition 1 we have that ($D_{2\nu}$) is feasible if and only if

$$\nu \leq \frac{2\min(\gamma n_+, (1-\gamma)n_-)}{n}.$$

Thus, ($D_{2\nu}$) is feasible if and only if

$$\frac{2\nu_+\nu_- n_+ n_-}{(\nu_+ n_+ + \nu_- n_-)n} \leq \frac{2\min\left(\frac{\nu_- n_+ n_-}{\nu_+ n_+ + \nu_- n_-}, \frac{\nu_+ n_+ n_-}{\nu_+ n_+ + \nu_- n_-}\right)}{n},$$

which is equivalent to $\nu_+\nu_- \leq \min(\nu_-, \nu_+)$, or $\nu_+ \leq 1$ and $\nu_- \leq 1$. □

2.4 Relationship Between the 2ν-SVM and $2C$-SVM

The following theorems illustrate the relationship between (D_{2C}) and ($D_{2\nu}$). The first shows how solutions of (D_{2C}) are related to solutions of ($D_{2\nu}$), and the second shows how solutions of ($D_{2\nu}$) are related to solutions of (D_{2C}). The third theorem, our main result, shows that increasing (decreasing) ν is similar to decreasing (increasing) C. The proofs of these theorems were previously given in [10] and can be found in the Appendix.

Theorem 1. *Fix $\gamma \in [0,1]$. For any $C > 0$, let $\boldsymbol{\alpha}^C$ be any optimal solution of (D_{2C}) and set $\nu = \sum_{i=1}^n \alpha_i^C/(Cn)$. Then any $\boldsymbol{\alpha}$ is an optimal solution of (D_{2C}) if and only if $\boldsymbol{\alpha}/(Cn)$ is an optimal solution of ($D_{2\nu}$).*

Theorem 2. *Fix $\gamma \in [0,1]$. For any $\nu \in (0, \nu_{\max}]$, assume ($D_{2\nu}$) has a nonzero optimal objective value, in which case $\rho > 0$, and set $C = 1/(\rho n)$. Then any $\boldsymbol{\alpha}$ is an optimal solution of (D_{2C}) if and only if $\boldsymbol{\alpha}/(Cn)$ is an optimal solution of ($D_{2\nu}$).*

Theorem 3. *Fix $\gamma \in [0,1]$ and let $\boldsymbol{\alpha}^C$ be any optimal solution of (D_{2C}). Define*

$$\nu_* = \lim_{C \to \infty} \frac{\sum_{i=1}^n \alpha_i^C}{Cn}$$

and

$$\nu^* = \lim_{C \to 0} \frac{\sum_{i=1}^n \alpha_i^C}{Cn}.$$

Then $0 \leq \nu_* \leq \nu^* = \nu_{\max} \leq 1$. For any $\nu > \nu^*$, ($D_{2\nu}$) is infeasible. For any $\nu \in (\nu_*, \nu^*]$ the optimal objective value of ($D_{2\nu}$) is strictly positive, and there exists at least one $C > 0$ such that any $\boldsymbol{\alpha}$ is an optimal solution of (D_{2C}) if and only if $\boldsymbol{\alpha}/(Cn)$ is an optimal solution of ($D_{2\nu}$). For any $\nu \in [0, \nu_*]$, ($D_{2\nu}$) is feasible with an optimal objective value of zero(and a trivial solution).

Remark 1. *Consider the case where the data can be perfectly separated by a hyperplane. In this case, as $C \to \infty$ margin errors are penalized more heavily, and thus for some sufficiently large C, the solution of (D_{2C}) will correspond to the separating hyperplane. Thus there exists some C^* such that $\boldsymbol{\alpha}^{C^*}$ (corresponding to the separating hyperplane) is an optimal solution of (D_{2C}) for all $C \geq C^*$. In this case, as $C \to \infty$, $\sum_{i=1}^n \alpha_i^C / Cn \to 0$, and thus $\nu_* = 0$.*

Using the definitions of ν_+ and ν_- in Section 2.3, it is easy to see that Theorem

3 implies that if γ is fixed and we let $C \to \infty$, (D_{2C}) is equivalent to $(D_{2\nu})$ (in the sense described above) if we let

$$\nu_+ \to \frac{\nu_* n}{2\gamma n_+} \geq 0, \quad \nu_- \to \frac{\nu_* n}{2(1-\gamma)n_-} \geq 0.$$

Similarly, if γ is fixed and we let $C \to 0$, (D_{2C}) is equivalent to $(D_{2\nu})$ if we let

$$\nu_+ \to \frac{\nu_{\max} n}{2\gamma n_+} = \min\left(1, \frac{(1-\gamma)n_-}{\gamma n_+}\right), \quad \nu_- \to \frac{\nu_{\max} n}{2(1-\gamma)n_-} = \min\left(1, \frac{\gamma n_+}{(1-\gamma)n_-}\right).$$

Chapter 3

Minimax Learning

We now return to the problem of minimax classification. We consider several strategies for estimating minimax classifiers using SVMs. For example, one approach is to train a C-SVM or ν-SVM and then shift b (the bias) to minimize $\max(P_F(f), P_M(f))$. A more powerful approach is to use the $2C$-SVM or 2ν-SVM to attempt to minimize $\max(P_F(f), P_M(f))$ by adjusting γ (and also possibly b) appropriately. In Chapter 2 we saw that (P_{2C}) and $(P_{2\nu})$ are closely related and equivalent in the sense that they explore the same set of possible solutions. Therefore, we lose nothing in restricting our attention to the 2ν-SVM. In fact, in light of Theorem 3, we can now see that there are some good reasons to prefer the 2ν-SVM to the $2C$-SVM. In particular, as noted in the previous section, for any fixed value of γ, if the data is separable – which is *always* the case if we use a radial basis function as our kernel [4] – then there exists an unknown value C^* such that for all $C \geq C^*$ we will obtain the same solution. Therefore, if we want to implement any kind of grid search procedure for parameter selection with the $2C$-SVM, we are forced to make some rather arbitrary choices regarding the maximal (and minimal) value of C to use in our grid search. If we set the maximum value of C too high we will repeatedly train SVMs for values of C that all result in the same classifier, and hence waste a great deal of training time, while if we set it too low we will not adequately explore the parameter space. This issue, combined with the numerical problems faced by most implementations when C becomes large (inspiring ad-hoc restrictions such as "never pick $C > 1000$"), have led us to believe that the 2ν-SVM is a more appropriate formulation when implementing the kinds of procedures we now describe.

In what follows $P_F(f)$ and $P_M(f)$ denote the false alarm and miss rates of a classifier f as in equations (1.1) and (1.2), and $\widehat{P}_F(f)$ and $\widehat{P}_M(f)$ denote empirical estimates of these quantities.

3.1 2ν-SVM Approach to Minimax Classification

As described in Section 2.3, the 2ν-SVM as proposed in [6] is parameterized in a different manner than $(P_{2\nu})$. Our parametrization has the benefit that 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 $\nu_- \leq 0$ (from Theorem 3). Therefore, to search over the parameters of the 2ν-SVM it suffices to conduct a search over a uniform grid of (ν_+, ν_-) in $[0,1]^2$. Hence, the full algorithm for minimax classification with the 2ν-SVM is to search over ν_+, ν_-, and any kernel parameters, obtain estimates of $P_F(f)$ and $P_M(f)$ using an error estimation technique such as cross-validation, and select the parameter combination minimizing $\max\{\widehat{P}_F(f), \widehat{P}_M(f)\}$.

3.1.1 Smoothing the Error Estimates

We have observed across a wide range of datasets that $\widehat{P}_F(f)$ and $\widehat{P}_M(f)$ tend to vary smoothly when plotted as functions of (ν_+, ν_-). However, these estimates also appear somewhat "noisy". Thus, a heuristic offering potential improvement for the full grid search over (ν_+, ν_-) is to smooth the estimates with a low-pass filter both $\widehat{P}_F(f)$ and $\widehat{P}_M(f)$ after estimating the error at each point on the grid. In our experiments we consider two smoothing strategies: we can either apply a two-dimensional smoothing filter (we use a simple Gaussian window) to the error estimates for $(\nu_+, \nu_-) \in [0,1]^2$ separately for each value of the kernel parameter, or we can apply a three-dimensional smoothing filter to the error estimates, smoothing across different kernel parameter values. Both strategies effectively reduce the variance of the error estimates. This approach is especially effective for high variance estimates like cross-validation. Without smoothing, some grid points will look much better than they actually are, due to chance variation. This technique illustrates another advantage of the (ν_+, ν_-) parametrization since the ability to discretize the parameter space of the 2ν-SVM with a uniform grid plays a key role in justifying this heuristic.

3.1.2 Coordinate Descent: Speeding Up the 2ν-SVM

The additional parameter in the 2ν-SVM renders a full grid search somewhat time consuming, especially for large data sets. Fortunately, a simple speed-up is possible. Again inspired by the smoothness of $\widehat{P}_F(f)$ and $\widehat{P}_M(f)$ as functions of

(ν_+, ν_-), instead of conducting a full grid search over (ν_+, ν_-) we propose a coordinate descent search. Several variants are possible, but the ones we employ run as follows: For a fixed value of the kernel parameter, find the best parameters on grids placed along the lines $\nu_+ = 1/2$ and $\nu_- = 1/2$. From then on, conduct a line search in the direction orthogonal to the previous line search, at each step selecting the parameters minimizing $\max\{\widehat{P}_F(f), \widehat{P}_M(f)\}$, repeating this procedure for each kernel parameter value. Just as before, a simple three-dimensional extension of this algorithm is also considered, along with various approaches to smoothing the data. Note that this strategy would again be more difficult to justify with the $2C$-SVM because the choice of endpoints and grid spacing would ultimately be arbitrary and data-dependent.

3.2 Alternative Approaches to Minimax Classification

3.2.1 Bias-Shifting

A potential advantage of the *bias-shifting* strategy is the ability to separate the training into two stages. First, we search over the parameters of the SVM (ν and any kernel parameters). Using an error estimation method such as cross-validation (CV), we then select the parameters that minimize $\max\{\widehat{P}_F(f), \widehat{P}_M(f)\}$ or the misclassification rate. Second, once ν has been selected we shift the bias of the corresponding classifier and, again using some form of error estimation, select the bias that further minimizes $\max\{\widehat{P}_F(f), \widehat{P}_M(f)\}$. In our experiments, we use the resubstitution estimate to select the bias. Resubstitution is generally a poor estimate when the set of classifiers is complex; however, once we fix a normal vector $\mathbf{w}$, the set of possible shifted hyperplanes is a class with low complexity, and so resubstitution is in fact a reasonable error estimate. Note that we can apply the same technique to an SVM trained using the 2ν-SVM approach described above in the hope that it will improve the performance of that method as well.

3.2.2 Balanced ν-SVM

A common motivation for minimax classification is that some datasets are unbalanced in the sense that they have many more samples from one class than from the other. In light of Proposition 3, another possible algorithm is to apply the strategy described above for the ν-SVM, but instead to use a 2ν-SVM with $\nu_+ = \nu_-$. We refer to this method as the *balanced ν-SVM"*. Since ν_+ and ν_- are upper bounds on the fractions of margin errors from their respective classes, we might expect that this method will be superior to the traditional ν-SVM. Note that this method has the same computational complexity as the traditional ν-SVM.

3.2.3 Minimax Probability Machine

The *minimax probability machine* (MPM) is a kernel-based alternative to SVMs that is specifically designed for minimax classification [15]. The general idea is to use the training data to estimate the mean and covariance matrices for each of the two classes, and then select the (hyperplane) classifier that minimizes $\max\{\widehat{P}_F(f), \widehat{P}_M(f)\}$ for the worst-case over all possible choices of class-conditional densities whose (class-conditional) means and covariance matrices match those estimated from the training data. Since the means and covariance matrices estimated from the training data will be subject to some error, the user must set up to four parameters that reflect the uncertainty in these estimates. The MPM can be kernelized in a similar manner to the SVM, and the two algorithms have similar computational complexity, although the MPM has a greater number of free parameters to tune.

3.3 Experiments

We ran our algorithms on a collection of benchmark datasets that are available online with documentation.[1] The datasets comprise a mixture of synthetic datasets and datasets based on real data collected from various repositories on the web. The datasets are summarized in Table 3.1. For each of the first 9 data sets, we have 100 permutations of the training and test data, and for the last two ("image" and "splice") we have 20 permutations.

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 ν-SVM method we searched over a uniform grid of 50 points of the parameter ν, and for the balanced ν-SVM we searched over a uniform grid of 50 points of the parameter $\nu_+ = \nu_-$. For the 2ν-SVM methods we considered a 50×50 regular grid of $(\nu_+, \nu_-) \in [0,1]^2$. For each parameter combination, we

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

Table 3.1: *Description of benchmark datasets used in our experiments: d denotes the dimension of the feature vectors, n (Training/Testing) the number of feature vectors in the training and test sets, and $\bar{n}_+$ ($\bar{n}_-$) the average number of feature vectors in the training set from the positive (negative) class. We also list whether the features are real-valued, discrete-valued, or comprise a combination of real and discrete-valued features.*

Dataset	d	n (Training)	$\bar{n}_+$	$\bar{n}_-$	n (Testing)	Features
banana	2	400	182	218	4900	Real
breast-cancer	9	200	59	141	77	Discrete
diabetes	8	468	164	304	300	Mixed
flare-solar	9	666	368	298	400	Discrete
heart	13	170	76	94	100	Mixed
ringnorm	20	400	199	201	7000	Real
thyroid	5	140	43	97	75	Real
twonorm	20	400	202	198	7000	Real
waveform	21	400	132	268	4600	Real
image	18	1300	746	554	1010	Mixed
splice	60	1000	483	517	2175	Discrete

estimated $P_F(f)$ and $P_M(f)$ using 5-fold cross-validation. In adjusting the bias for any of these methods we selected the optimal bias according to the resubstitution estimate. We applied a 3×3 Gaussian window to the error estimates to implement 2-dimensional smoothing, and we applied a $3 \times 3 \times 3$ Gaussian window to the error estimates to implement 3-dimensional smoothing. The standard deviation of the Gaussian window was set to the length of one grid interval. Different window sizes and widths were tried, but without much change in performance. For the coordinate descent methods we used the same grid structure described above. In our experiments with the ν-SVM we used the LIBSVM package [3]. For the 2ν-SVM we implemented our own version that is available online at www.dsp.rice.edu/software.

For each permutation of each dataset we ran our algorithms on the training data and estimated the false alarm and miss rates using the test data. On any given permutation, our performance metric is $\max\{\widehat{P}_F(f), \widehat{P}_M(f)\}$, where $\widehat{P}_F(f)$ and $\widehat{P}_M(f)$ denote the false positive and miss rates estimated using the test data. To generate a more reliable performance estimate, we repeat this for each permutation and then average the minimax scores over all permutations – a procedure known as Monte Carlo cross-validation [28]. To evaluate performance on unbalanced datasets, we repeated these experiments retaining only 10% of the negatively labeled training data.

We use two main statistical tests to compare the algorithms described above, as advocated in [13]. In cases where we want to make a direct comparison between only two algorithms, we use the Wilcoxon signed-ranks test, which ranks the differences in performances of the two classifiers over the 11 datasets, and then compares the ranks for the positive and negative differences to test if the observed differences between the two algorithms is statistically significant. When reporting results from the Wilcoxon signed-ranks test, we will give the p-value, or probability of obtaining the observed differences by chance.

When we wish to compare more than two algorithms on multiple datasets, we use a two-step procedure. First we use the Friedman test, which is a statistical test similar to the Wilcoxon signed-ranks test in that it allows us to determine the probability of obtaining the observed performances by chance. Next, once we have rejected the null-hypothesis (that the differences have occurred by chance) we apply the Nemenyi test which involves computing a ranking of the algorithms for each dataset, and then an average ranking for each algorithm. Along with these rankings, we provide the so-called critical difference for a significance level of 0.05. (If the average ranking of two algorithms differs by more than this value, the performance of the two algorithms is significantly different with a p-value of 0.05.) See [13] for a more thorough discussion of and motivation for these techniques.

3.4 Results

We begin by evaluating the performance of the 2ν-SVM. Perhaps somewhat surprisingly, bias-shifting actually results in uniformly worse performance for every 2ν-SVM-based method, with p-values below 0.05 in almost every case (see Table 3.2). As we will see again in our discussion of the balanced ν-SVM and the ν-SVM, bias-shifting only leads to improved performance when the SVM parameters have been selected to minimize the error rate. When the SVM parameters are selected to minimize $\max\{\widehat{P}_F(f), \widehat{P}_M(f)\}$, bias-shifting has a negative impact on overall performance.

On the other hand, smoothing and coordinate descent are extremely effective. The results of smoothing are shown in Table 3.3, and they clearly indicate that 2-D and 3-D smoothing offer a statistically significant gain in performance, with 3-D smoothing offering a slight edge. Similarly, the results in Table 3.4 show that 3-D smoothing combined with either 2-D or 3-D coordinate descent offer gains in performance as well, which is particularly helpful since these methods speed up the parameter selection process considerably.

Table 3.2: *The effect of bias-shifting on the 2ν-SVM methods for minimax classification. For every case, bias-shifting leads to worse performance compared to a 2ν-SVM without bias-shifting (on both balanced and unbalanced data sets). The table lists the p-values calculated using the Wilcoxon signed-ranks test indicating the significance of this difference in performance are listed for each 2ν-SVM method.*

Smoothing	Coordinate Descent	Balanced	Unbalanced
None	None	.148	.042
2-D	None	.001	.001
3-D	None	.005	.001
None	2-D	.107	.007
None	3-D	.032	.007
2-D	2-D	.001	.001
3-D	2-D	.002	.001
3-D	3-D	.032	.001

Table 3.3: *Comparison of smoothing methods for the 2ν-SVM for minimax classification. The table lists the average ranking for each approach. (Friedman test yields p-values of .001 for both balanced and unbalanced experiments. The critical difference for the Nemenyi test at 0.05 is 1.10.)*

Smoothing	Balanced	Unbalanced
None	2.91	2.91
2-D	1.73	1.64
3-D	1.36	1.45

Table 3.4: *Comparison of coordinate descent methods for the 2ν-SVM for minimax classification. The table lists the average ranking for each approach. (Friedman test yields p-values of .002 for balanced and .001 for unbalanced experiments. The critical difference for the Nemenyi test at 0.05 is 1.92.)*

Smoothing	Coordinate Descent	Balanced	Unbalanced
None	2-D	4.18	4.18
None	3-D	3.91	4.00
2-D	2-D	2.73	2.82
3-D	2-D	2.00	2.00
3-D	3-D	2.18	2.00

Table 3.5: *Comparison of different methods using the ν-SVM for minimax classification. The table lists the average ranking for each approach. (Friedman test yields p-values of .200 for balanced and .002 for unbalanced experiments. The critical difference for the Nemenyi test at 0.05 is 1.50.)*

Method	Smoothing	Balanced	Unbalanced
MM-NoBS	No	3.00	3.64
MM-NoBS	Yes	2.18	2.73
PE-BS	No	2.82	1.73
PE-BS	Yes	2.00	1.91

Before we directly compare the smoothing and coordinate descent methods, let us briefly evaluate the balanced ν-SVM and the traditional ν-SVM. We consider two main strategies: (1) adjust ν (or $\nu_+ = \nu_-$ for the balanced ν-SVM) to minimize $\max\{\widehat{P}_F(f), \widehat{P}_M(f)\}$ (with 2-D smoothing between ν and the kernel parameter optional), and do not adjust the bias, or (2) adjust ν (or $\nu_+ = \nu_-$) to minimize the misclassification rate (with 2-D smoothing between ν and the kernel parameter again optional), and adjust the bias to minimize $\max\{\widehat{P}_F(f), \widehat{P}_M(f)\}$. In Tables 3.5 and 3.6 we denote strategy (1) by MM-NoBS and strategy (2) by PE-BS. For the traditional ν-SVM, we find that strategy (2) appears most effective, as can be seen in Table 3.5. Alternatively, for the balanced ν-SVM the results indicate that the best method is to use smoothing and follow strategy (1), as can be seen in Table 3.6. This result seems to indicate that the parametrization of the balanced ν-SVM does indeed lend itself more naturally to minimax classification than the traditional ν-SVM, which needs to adjust the bias after training the SVM to get the desired performance.

We are finally in a position to compare the 2ν-SVM strategies to the balanced

Table 3.6: *Comparison of different methods using the balanced ν-SVM for minimax classification. The table lists the average ranking for each approach. (Friedman test yields p-values of .183 for balanced and .0001 for unbalanced experiments. The critical difference for the Nemenyi test at 0.05 is 1.50.)*

Method	Smoothing	Balanced	Unbalanced
MM-NoBS	No	2.91	2.09
MM-NoBS	Yes	1.82	1.27
PE-BS	No	2.82	3.09
PE-BS	Yes	2.45	3.55

ν-SVM and traditional ν-SVM. In Tables 3.7 and 3.8 we give the minimax error rates for the 3-D smoothing approach (labeled 3D-GS), the 2-D and 3-D coordinate descent methods (labeled 2D-CD and 3D-CD – both use 3-D smoothing), the balanced ν-SVM without bias-shifting (labeled Bal ν-SVM), and the traditional ν-SVM with bias-shifting (labeled ν-SVM). Table 3.7 shows the performance of each algorithm on each dataset averaged over the permutations, and Table 3.8 shows the same for the unbalanced datasets. Table 3.9 gives the results of the Nemenyi test for these algorithms. In the balanced experiments, the 2ν-SVM methods appear to exhibit stronger performance, but this is not statistically significant. However, for the *un*balanced case, there is a clear and significant difference, with the 2ν-SVM methods being clearly superior. The 3D-GS method appears to be the best performing overall, but the coordinate descent methods exhibit very similar performance.

Finally, we can compare the 2ν-SVM with the MPM. We compare the 2ν-SVM and the MPM with a linear kernel, and we set the parameters for the MPM to be the ideal parameters based on the test set, which should give the MPM an unfair advantage.

Table 3.7: *Minimax rates on the* balanced *datasets for the best 2ν-SVM methods, the balanced ν-SVM (without bias-shifting), and the ν-SVM (with bias-shifting). Scores reported are* $\max\{\widehat{P}_F(f), \widehat{P}_M(f)\}$*, averaged over all 100 (or 20) permutations.*

Dataset	3D-GS	2D-CD	3D-CD	Bal ν-SVM	ν-SVM
banana	.129	.129	.129	.133	.132
breast-cancer	.414	.419	.431	.490	.425
diabetes	.302	.301	.303	.304	.289
flare-solar	.355	.352	.433	.415	.365
heart	.226	.219	.224	.231	.221
ringnorm	.024	.023	.021	.022	.027
thyroid	.075	.078	.070	.076	.081
twonorm	.032	.031	.030	.029	.034
waveform	.119	.117	.116	.123	.135
image	.043	.050	.065	.039	.040
splice	.114	.118	.118	.113	.157

Table 3.8: *Minimax rates on the* unbalanced *datasets for the best 2ν-SVM methods, the balanced ν-SVM (without bias-shifting), and the ν-SVM (with bias-shifting). Scores reported are* $\max\{\widehat{P}_F(f), \widehat{P}_M(f)\}$*, averaged over all 100 (or 20) permutations.*

Dataset	3D-GS	2D-CD	3D-CD	Bal ν-SVM	ν-SVM
banana	.193	.194	.189	.226	.218
breast-cancer	.451	.460	.477	.564	.737
diabetes	.340	.340	.338	.455	.449
flare-solar	.410	.412	.425	.595	.548
heart	.271	.286	.275	.413	.490
ringnorm	.048	.049	.040	.055	.088
thyroid	.133	.139	.126	.126	.135
twonorm	.060	.060	.058	.079	.099
waveform	.168	.171	.168	.210	.181
image	.134	.133	.157	.151	.097
splice	.195	.196	.200	.379	.335

Table 3.9: *Comparison of best 2ν-SVM methods for minimax classification, the balanced ν-SVM (without bias-shifting), and the ν-SVM (with bias-shifting). The table lists the average ranking for each approach. (Friedman test yields p-values of .502 for balanced and .0003 for unbalanced experiments. The critical difference for the Nemenyi test at 0.05 is 1.92.)*

Method	Balanced	Unbalanced
3D-GS	2.73	2.00
2D-CD	2.64	2.64
3D-CD	2.73	2.00
ν-SVM	3.64	4.09
Bal ν-SVM	3.27	4.27

Recall that the parameters for the MPM represent the uncertainty in our knowledge of the class-dependent means and covariance matrices. We can calculate this uncertainty *exactly* by calculating the differences between the means and covariances based on the training set andthose based on the test set, which allows us to realize the best performance possible with the MPM. To make a fair comparison, we only set two free parameters – we assume that the uncertainty in the means and covariances is the same for both classes, and thus the MPM and 2ν-SVM would require roughly the same computational complexity to set the parameters in a practical setting. We then compare this to the performance of the 2ν-SVM where the parameters for the 2ν-SVM are chosen via cross-validation. The results are given in Table 3.10. In the unbalanced case we do not see a significant difference, with each algorithm doing better on roughly half the datasets, although in this case we do see that when the 2ν-SVM outperforms the MPM it tends to do so by a large amount compared to cases where the MPM outperforms the 2ν-SVM. However, in the balanced case we get a

Table 3.10: *Minimax rates for the 2ν-SVM with linear kernel (where ν_+ and ν_- are selected through cross validation, with smoothing of the error estimates) and the linear MPM where the parameters are chosen to be optimal for the test data set. Scores reported are* $\max\{\widehat{P}_F(f), \widehat{P}_M(f)\}$*, averaged over all 100 (or 20) permutations.*

Dataset	Balanced		Unbalanced	
	SVM	MPM	SVM	MPM
banana	.558	.482	.619	.517
breast-cancer	.396	.401	.453	.421
diabetes	.291	.311	.332	.319
flare-solar	.350	.360	.392	.399
heart	.218	.205	.285	.238
ringnorm	.287	.308	.335	.302
thyroid	.197	.387	.253	.392
twonorm	.031	.027	.064	.038
waveform	.141	.180	.175	.218
image	.194	.341	.230	.362
splice	.175	.184	.239	.296

clear difference in performance. Furthermore, in a more practical setting, in which the parameters for the MPM are set imperfectly, the 2ν-SVM would likely out-perform the MPM by an even greater margin.

Chapter 4

Neyman-Pearson Learning

We now turn to the NP classification setting. NP classification has much in common with minimax classification, and we will consider slight modifications of many of the same algorithms described above. However, there are important differences in how we set the SVM parameters and in how we evaluate the performance of our algorithms.

4.1 Approaches to Neyman-Pearson Classification

4.1.1 2ν-SVM Approach

Our approach to NP classification using the 2ν-SVM is for the most part identical to our approach to minimax classification using the 2ν-SVM, with the main difference being the way in which we select our free parameters. After conducting a grid search over ν_+, ν_-, and any kernel parameters, we use estimates of $P_F(f)$ and $P_M(f)$ to select the parameter combination minimizing $P_M(f)$ subject to the constraint that $P_F(f) < \alpha$. Just as before, smoothing the error estimates will prove to be a useful heuristic. Our coordinate descent procedure can also be used in this setting provided that we make a minor modification. At the beginning of the coordinate descent search, it is possible that we may not find any parameter settings that yield a classifier satisfying the false alarm constraint. In this event, we simply pick the classifier minimizing $P_F(f)$, and proceed with our search until we find a parameter setting that does satisfy the false alarm constraint.

4.1.2 Bias-Shifting Approaches

As an alternative to the 2ν-SVM approach, we again consider the traditional ν-SVM and the balanced ν-SVM, along with the strategy of bias-shifting. Based on the results for minimax classification, we do not expect these methods to perform as well as the 2ν-SVM methods, but they will serve as a meaningful reference to compare to, and are representative of the techniques most frequently used in practice.

4.2 Measuring Performance

As described in the Introduction, we would like to have a scalar performance measure, but find the AUC to be inadequate in our setting. An obvious option is to estimate $P_F(f)$ and $P_M(f)$ and assign a "score" of $\widehat{P}_M(f)$ to the classifier if $\widehat{P}_F(f) \leq \alpha$, and ∞ otherwise. Unfortunately, there are a number of problems with this approach.

First, our classifiers have been estimated from random training samples, and hence the false alarm rate of a classifier is itself a random quantity. The performance of an algorithm on one dataset may not give a fair indication of its performance on others. This contrasts with conventional classification, where the probability of error for an unfavorable training sample will at least be somewhat close to the typical probability of error for that distribution. In terms of expectation, the expected performance of a learning rule will be infinite as long as there is some collection of training samples (that occurs with nonzero probability) for which $P_F(f) > \alpha$. It seems preferable for a performance measure to show more leniency to violations of the false alarm constraint so that "rare" training samples do not mislead us about an algorithm's typical performance.

Second, it is not possible to estimate the performance from data precisely. Estimates of $P_F(f)$ and $P_M(f)$ are based on random data and thus will have some chance error. It is generally impossible to be certain whether a given classifier does or does not satisfy the false alarm constraint. In fact, if we evaluate the performance of f with $\widehat{P}_M(f)$ when $\widehat{P}_F(f) \leq \alpha$ and with ∞ otherwise, then the bias of this estimate is *infinite* whenever $0 < P_F(f) \leq \alpha$.

Third, many practitioners would be content to have the false alarm rate slightly exceed the constraint if the decrease in the miss rate is substantial. In other words, it might be nice to explore a small region along the receiver operating characteristic in the vicinity of α.

In summary, the indicator function-based performance measures just discussed are sufficient for theoretical purposes, but their empirical counterparts are not. A good performance measure for NP classification should not reject outright a classifier that appears to violate the false alarm constraint. Such a violation may be (1) the result of a rare training sample on which an otherwise good learning rule performs poorly, (2) the result of chance error in estimating the false alarm rate, and (3) acceptable to a practitioner.

To alleviate these problems we could instead measure performance with $c\mathbb{I}(P_F(f) > \alpha) + P_M(f)$, where $c > 1$ and $\mathbb{I}(\cdot)$ is an indicator. Yet the same three arguments used above for the case $c = \infty$ still apply. For example, suppose we estimate this quantity with a plug-in estimate based on an independent test sample. This estimate has a very large bias when $P_F(f)$ is close to α, even for large test sample sizes. In fact, if $P_F(f) = \alpha$, the bias does not tend to zero as the test sample size increases.

Another possibility, which does show some leniency to classifiers violating the false alarm constraint, is to measure performance with $|P_F(f) - \alpha| + |P_M(f) - \beta_\alpha|$ or some other "distance" in the $(P_F(f), P_M(f))$ plane. Yet this kind of measure too has drawbacks. It requires knowledge of β_α which is unknown, and hence the distance cannot be estimated. In addition, it penalizes classifiers for having $P_F(f) < \alpha$ or $P_M(f) < \beta_\alpha$, which seems unreasonable.

To remedy this situation, we consider the Neyman-Pearson score,

$$\mathcal{E}(f) = \frac{1}{\alpha} \max \{P_F(f) - \alpha, 0\} + P_M(f). \tag{4.1}$$

as a practical performance measure for evaluation and comparison of classifiers in NP classification. In [24] it is shown that the global minimizer of $\mathcal{E}$ is indeed f_α^*, consistent with the task of NP classification. Furthermore, the NP score has additional properties, desirable from a statistical point of view. For example, it can be reliably estimated from data. It also has the appealing property that it tolerates small violations of the false alarm constraint. That is, it will prefer a classifier with $P_F(f) > \alpha$ to one with $P_F(f) \leq \alpha$ if the payoff in $P_M(f)$ is large enough. At the same time, as α draws closer to 0, a stiffer penalty is exacted on classifiers that violate the constraint. This makes sense because exceeding α by 0.01, for example, is much more significant when $\alpha = 0.01$ than when $\alpha = 0.1$. Said another way, the NP score in (4.1) penalizes the *relative* error $(P_F(f) - \alpha)/\alpha$.

4.3 Experiments and Results

We again begin by evaluating the performance of the 2ν-SVM. Our experimental procedure was identical to that described in Section 3.3, with the only difference being the criterion used to select the various parameters. As was the case in minimax classification, bias-shifting results in uniformly worse performance for every 2ν-SVM-based method, with p-values below 0.05 in almost every case (see Table 4.1). This result is consistent with our previous observation that bias-shifting is only beneficial when the false alarm (or miss) rate of the base classifier is far from the desired false alarm (or miss) rate.

The results for smoothing and coordinate descent also mirror those for minimax

Table 4.1: *The effect of bias-shifting on the 2ν-SVM methods for NP classification. For every case, bias-shifting leads to worse performance compared to a 2ν-SVM without bias-shifting (on both balanced and unbalanced data sets). The table lists the p-values calculated using the Wilcoxon signed-ranks test indicating the significance of this difference in performance are listed for each 2ν-SVM method.*

Smoothing	CD	Balanced (α)				Unbalanced (α)			
		.01	.05	.1	.2	.01	.05	.1	.2
None	None	.002	.010	.032	.175	.024	.024	.067	.067
2-D	None	.001	.001	.001	.001	.001	.001	.001	.001
3-D	None	.001	.001	.001	.001	.001	.001	.001	.001
None	2-D	.001	.003	.001	.019	.002	.002	.003	.007
None	3-D	.001	.001	.001	.003	.003	.002	.001	.001
2-D	2-D	.001	.001	.001	.001	.001	.001	.001	.001
3-D	2-D	.001	.001	.001	.001	.001	.001	.001	.001
3-D	3-D	.001	.001	.001	.001	.001	.001	.001	.001

Table 4.2: *Comparison of smoothing methods for the 2ν-SVM for NP classification. The table lists the average ranking for each approach. (Friedman test yields p-values of less than .01 for all cases. The critical difference for the Nemenyi test at 0.05 is 1.10.)*

Smoothing	Balanced (α)				Unbalanced (α)			
	.01	.05	.1	.2	.01	.05	.1	.2
None	3.00	3.00	2.73	3.00	2.73	2.82	2.64	2.73
2-D	1.91	1.91	2.09	1.82	1.82	1.82	2.00	1.91
3-D	1.09	1.09	1.18	1.18	1.45	1.36	1.36	1.36

classification. The results of smoothing are shown in Table 4.2, again indicating that 2-D and 3-D smoothing offer a statistically significant gain in performance, with 3-D smoothing performing slightly better. Similarly, the results in Table 4.3 show that 3-D smoothing combined with either 2-D or 3-D coordinate descent offer gains in performance as well.

Next we again briefly evaluate the balanced ν-SVM and the traditional ν-SVM.

Table 4.3: *Comparison of coordinate descent methods for the 2ν-SVM for NP classification. The table lists the average ranking for each approach. (Friedman test yields p-values of less than .05 for all cases. The critical difference for the Nemenyi test at 0.05 is 1.92.)*

Smoothing	CD	Balanced (α)				Unbalanced (α)			
		.01	.05	.1	.2	.01	.05	.1	.2
None	2-D	4.09	3.82	3.82	4.55	4.45	4.36	4.36	4.18
None	3-D	3.91	4.18	3.55	4.00	4.18	3.91	3.64	3.82
2-D	2-D	2.73	2.36	2.64	2.91	2.36	2.82	3.36	3.18
3-D	2-D	2.09	1.91	1.91	1.55	2.00	2.00	1.91	2.09
3-D	3-D	2.18	2.73	3.09	2.00	2.00	1.91	1.73	1.73

We now consider three main strategies: (1) adjust ν (or $\nu_+ = \nu_-$ for the balanced ν-SVM) to minimize $\widehat{P}_M(f)$ subject to $\widehat{P}_F(f) < \alpha$ (with 2-D smoothing between ν and the kernel parameter optional), and do not adjust the bias, (2) repeat (1) but shift the bias to further minimize $\widehat{P}_M(f)$, or (3) adjust ν (or $\nu_+ = \nu_-$) to minimize the misclassification rate (with 2-D smoothing between ν and the kernel parameter again optional), and adjust the bias to minimize $\widehat{P}_M(f)$ subject to $\widehat{P}_F(f) < \alpha$. In Tables 4.4 and 4.5 we denote strategy (1) by NP-NoBS, (2) by NP-BS, and (3) by PE-BS. For the traditional ν-SVM, we find that strategy (1) appears most effective on the balanced datasets, and strategy (2) seems to be most effective on the unbalanced datasets, as can be seen in Table 4.4. Alternatively, for the balanced ν-SVM strategy (1) seems to be the best (or competitive with the best) method regardless of whether the data is balanced or unbalanced, as can be seen in Table 4.5. Notice that in both cases, strategy (3), which is the method most commonly used in practice, is clearly inferior. For comparison with the 2ν-SVM, we will consider strategy (2) for the traditional

Table 4.4: *Comparison of different methods using the ν-SVM for NP classification. The table lists the average ranking for each approach. (Friedman test yields p-values of less than .05 for all cases. The critical difference for the Nemenyi test at 0.05 is 2.35.)*

Method	Smoothing	Balanced (α)				Unbalanced (α)			
		.01	.05	.1	.2	.01	.05	.1	.2
NP-NoBS	No	2.68	2.00	2.09	2.36	5.64	4.73	3.82	2.59
NP-NoBs	Yes	2.23	2.27	1.64	1.73	5.18	4.27	3.73	3.05
NP-BS	No	4.86	4.45	4.27	4.36	2.00	2.27	2.36	2.77
NP-BS	Yes	5.23	5.73	5.45	5.09	1.91	2.00	2.55	3.50
PE-BS	No	3.09	3.55	3.91	4.00	2.64	3.27	3.55	4.00
PE-BS	Yes	2.91	3.00	3.64	3.45	3.64	4.45	5.00	5.09

Table 4.5: *Comparison of different methods using the balanced ν-SVM for NP classification. The table lists the average ranking for each approach. (Friedman test yields p-values of less than .1 for all cases. The critical difference for the Nemenyi test at 0.05 is 2.35.)*

Method	Smoothing	Balanced (α)				Unbalanced (α)			
		.01	.05	.1	.2	.01	.05	.1	.2
NP-NoBS	No	3.00	2.27	2.18	2.45	3.36	3.09	2.95	2.41
NP-NoBs	Yes	2.27	2.45	1.82	1.27	3.09	3.00	2.50	2.23
NP-BS	No	4.55	4.64	5.09	4.73	2.18	2.55	2.86	3.23
NP-BS	Yes	4.27	5.45	5.09	5.18	2.27	2.18	2.68	3.86
PE-BS	No	3.36	3.36	3.64	3.91	4.73	4.73	4.36	4.18
PE-BS	Yes	3.55	2.82	3.18	3.45	5.36	5.45	5.64	5.09

ν-SVM and strategy (1) for the balanced ν-SVM.

We are now in a position to compare the 2ν-SVM strategies to the balanced ν-SVM and traditional ν-SVM. In Table 4.6 we give the results of the Nemenyi test for the 3-D smoothing approach (labeled 3D-GS), the 2-D and 3-D coordinate descent methods (labeled 2D-CD and 3D-CD – both use 3-D smoothing), the balanced ν-SVM without bias-shifting (labeled Bal ν-SVM), and the traditional ν-SVM with bias-shifting (labeled ν-SVM). In this case we see that the 2ν-SVM methods clearly

Table 4.6: *Comparison of 2ν-SVM methods for NP classification with the balanced ν-SVM and the ν-SVM with bias-shifting. The table lists the average ranking for each approach. (Friedman test yields p-values of less than .001 for all cases. The critical difference for the Nemenyi test at 0.05 is 1.92.)*

Method	Balanced (α)				Unbalanced (α)			
	.01	.05	.1	.2	.01	.05	.1	.2
3D-GS	2.64	2.18	2.36	2.45	3.36	3.27	3.18	2.45
2D-CD	2.45	2.27	2.18	2.27	1.91	1.91	2.09	2.55
3D-CD	2.00	2.73	2.73	2.55	1.73	1.64	1.64	1.91
ν-SVM	5.00	5.00	4.91	4.91	3.45	3.73	4.18	4.55
Bal ν-SVM	2.91	2.82	2.82	2.82	4.55	4.45	3.91	3.55

outperform the traditional ν-SVM methods, and also outperform the balanced ν-SVM, but by a smaller margin on the balanced datasets. Perhaps the most surprising result is that the 3-D coordinate descent method is not only competitive with the full grid search, but is even better than the grid search on the unbalanced datasets.

Chapter 5

Learning Minimum-Volume Sets

We now examine how the techniques for NP classification described above can be used for anomaly detection through the introduction of an artificially generated class of points, allowing us to estimate MV-sets as described in the Introduction. While the NP classification algorithms used in this setting are identical to those described above, we must take care in how we generate the artificial data, especially in high dimensions.

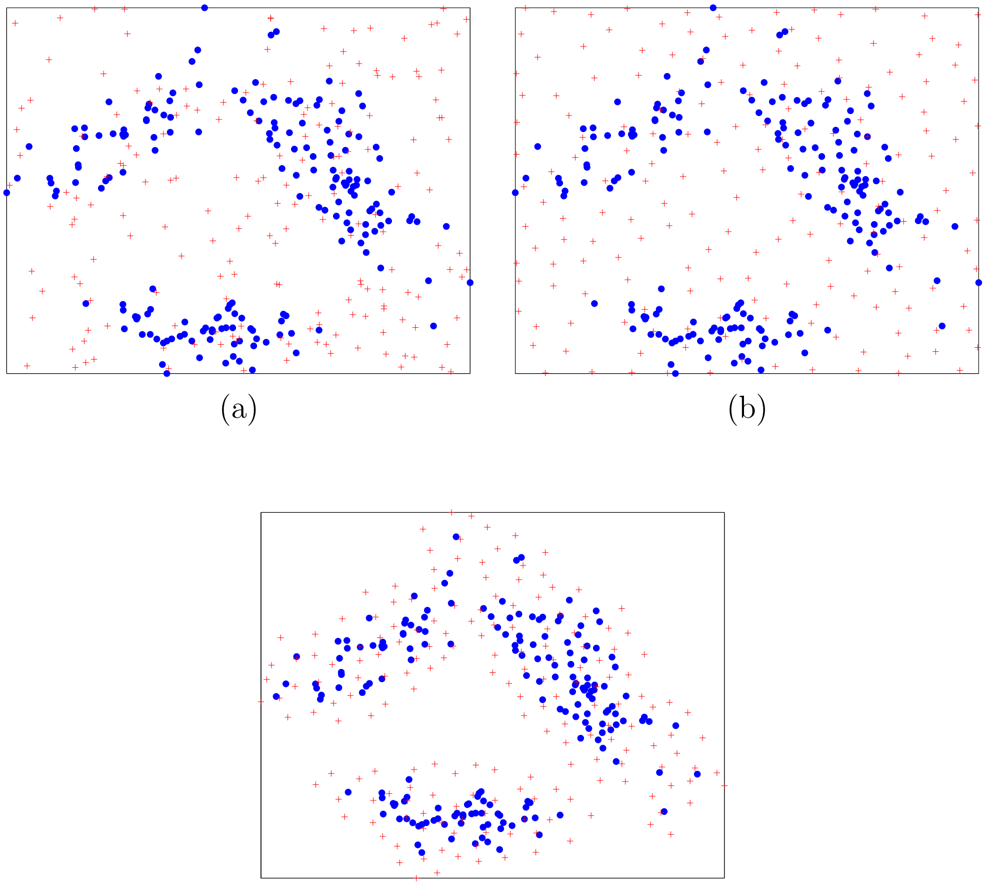
5.1 2ν-SVM Approach

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 effect can be viewed as one aspect of the "curse of dimensionality".

5.1.1 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 [37].



(a)

(b)

(c)

Figure 5.1: *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.*

5.1.2 Manifold Sampling

The thinning approach described above helps to distribute the points evenly throughout space. This approach 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 of points by randomly drawing points from the spheres centered at the $\mathbf{x}_i$'s, whose radii are 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 has a much smaller volume than that of the bounding hypercube. This technique and those described above are illustrated in Figure 5.1.

5.2 One-Class SVMs

The OC-SVM was proposed in [22] for the problems of estimating the support of a high-dimensional distribution and novelty detection. The OC-SVM can be formulated as

$$\min_{\mathbf{w},\boldsymbol{\xi},\rho} \quad \frac{1}{2}\|\mathbf{w}\|^2 - \nu\rho + \frac{1}{n}\sum_{i=1}^{n}\xi_i$$

$$\begin{aligned} \text{s.t.} \qquad & k(\mathbf{w}, \mathbf{x}_i) \geq \rho - \xi_i && \text{for } i = 1, 2, \ldots, n \\ & \xi_i \geq 0 && \text{for } i = 1, 2, \ldots, n. \end{aligned}$$

The resulting decision function

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

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 [32].

The MPM can also be adapted to perform MV-set estimation. We defer the reader to [16] for a thorough description of the OC-MPM. Given the superior performance of the 2ν-SVM as compared to the traditional MPM, we do not compare with the OC-MPM, but would expect it to exhibit inferior performance.

5.3 Measuring Performance

Just as we reduced the problem of estimating MV-sets to that of NP classification, we can use the same performance measures as before. Specifically, we can evaluate an MV-set using

$$\mathcal{E}_{\mu}(G) = \frac{1}{1-\beta} \max\left\{\beta - P(G), 0\right\} + \mu(G), \tag{5.1}$$

where $\mu(G)$ is an estimate of the volume obtained by independently generating a large test set according to μ, and $P(G)$ is estimated from a separate test set that is excluded during training.

However, in our experiments we employ benchmark data sets for binary classification and perform MV-set estimation using only the negatively labeled class. Thus, since we only use one class for training, we have a second performance measure:

$$\mathcal{E}_{+}(G) = \frac{1}{1-\beta} \max\left\{\beta - P(G), 0\right\} + Q_{+}(G), \tag{5.2}$$

where $Q_{+}(\{\mathbf{x} : f(\mathbf{x}) = -1\})$ is the probability of error on the class not used during training. In some sense this metric is more appropriate 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. Both of these measures have the same advantages as the measure defined in (4.1).

5.4 Experiments and Results

In our experiments with the OC-SVM we again used the LIBSVM package [3], and for the two-class methods, we used the 2ν-SVM described in Section 4. We used the same data sets as before, but only the negatively labeled training vectors. In all of our experiments we again 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 again 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 all methods we smoothed the error estimates using a 3-D Gaussian window.

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).

Tables 5.1—5.4 report the performance of the OC-SVM and the 2ν-SVM methods as measured by $\mathcal{E}_\mu$ and $\mathcal{E}_+$. Unlike the results presented thus far, there is no method that is superior on all datasets, although some general observations can be made. First, the manifold sampling approach does tend to work better on high-dimensional datasets as expected, and the thinning approach tends to work best on low-dimensional datasets. However, both methods seem to perform very poorly on a small number of (different) datasets. This seems to be rooted in the fact that these methods have more difficulty satisfying the constraint on the probability mass compared to the OC-SVM. While the reasons for this are not clear, further investigation that considers various possible hybrids between our approach and the OC-SVM should help shed some light on this issue.

On the other hand, we can compare the performance of the estimated MV-sets to the performance of our NP classification algorithms. Specifically, we compare the performance of the technique outlined above with the performance of a 2ν-SVM which is trained with access to the anomalous data set. While the approach which has access to the anomalies will obviously do better, we find that on several of the datasets we do surprisingly well, and the two-class techniques do a much better job at the achieving a low error rate on the unobserved class, as shown in Table 5.5. An example is illustrated in Figure 5.2.

Table 5.1: *Performance of OC-SVM and 2ν-SVM methods for MV-set estimation. Scores reported are $\mathcal{E}_\mu$ and $\mathcal{E}_+$, averaged over all 100 (or 20) permutations, for $\beta = 0.8$.*

Dataset	$\mathcal{E}_\mu$			$\mathcal{E}_+$		
	OC	Thin	Man	OC	Thin	Man
banana	0.69	1.10	0.31	0.17	0.89	0.16
breast-cancer	0.90	0.71	0.63	0.72	1.10	1.04
diabetes	0.99	1.20	0.58	0.65	1.44	0.93
flare-solar	1.25	0.09	0.13	0.95	0.62	0.85
heart	0.97	0.75	0.87	0.59	0.93	1.04
ringnorm	0.99	1.98	0.19	0.00	1.46	0.18
thyroid	1.09	0.50	1.51	0.17	0.48	1.53
twonorm	1.01	1.84	0.03	0.24	1.40	0.25
waveform	1.00	0.26	0.13	0.73	0.98	0.65
image	1.00	0.04	0.15	0.63	0.74	0.35
splice	0.03	0.64	1.00	0.89	0.77	1.00

Table 5.2: *Performance of OC-SVM and 2ν-SVM methods for MV-set estimation. Scores reported are $\mathcal{E}_\mu$ and $\mathcal{E}_+$, averaged over all 100 (or 20) permutations, for $\beta = 0.9$.*

Dataset	$\mathcal{E}_\mu$			$\mathcal{E}_+$		
	OC	Thin	Man	OC	Thin	Man
banana	1.01	1.38	0.43	0.98	1.10	0.30
breast-cancer	0.64	0.90	1.02	0.94	1.46	1.52
diabetes	0.96	1.34	0.17	0.83	1.81	0.84
flare-solar	0.88	0.17	0.32	0.91	0.73	1.06
heart	0.84	0.88	2.02	0.84	1.18	2.24
ringnorm	1.00	4.41	0.54	0.03	3.89	0.53
thyroid	0.88	0.84	2.84	0.36	0.74	2.87
twonorm	1.09	4.11	0.20	0.40	3.69	0.44
waveform	1.01	0.52	0.35	0.89	1.22	0.99
image	1.00	0.46	0.39	0.66	1.12	0.63
splice	0.01	2.10	1.00	0.96	2.12	1.00

Table 5.3: *Performance of OC-SVM and 2ν-SVM methods for MV-set estimation. Scores reported are $\mathcal{E}_\mu$ and $\mathcal{E}_+$, averaged over all 100 (or 20) permutations, for $\beta = 0.95$.*

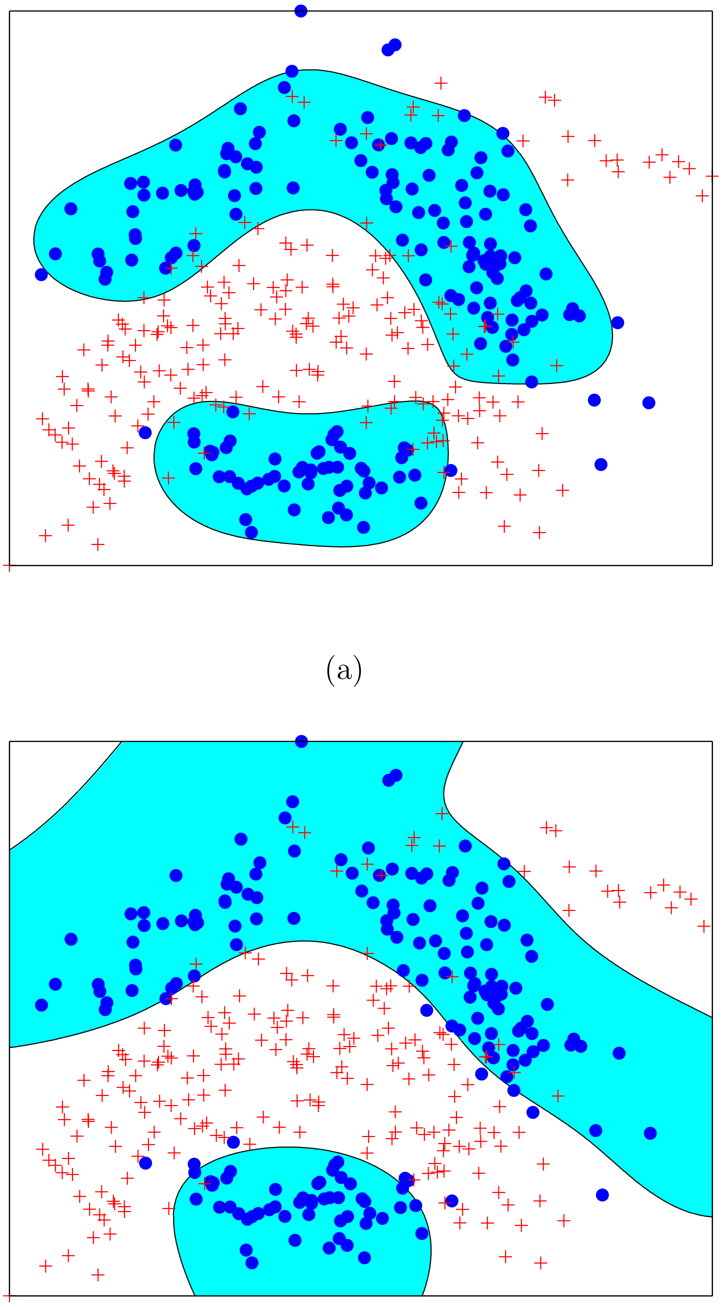
Dataset	$\mathcal{E}_\mu$			$\mathcal{E}_+$		
	OC	Thin	Man	OC	Thin	Man
banana	1.50	3.15	0.62	1.93	2.89	0.61
breast-cancer	0.60	0.98	1.29	1.25	1.63	1.88
diabetes	0.94	0.95	0.29	0.98	1.62	1.04
flare-solar	0.82	0.51	0.46	1.03	1.19	1.25
heart	1.25	1.30	2.60	1.50	1.58	2.95
ringnorm	1.20	9.19	1.45	0.26	8.66	1.43
thyroid	0.91	0.92	4.42	0.81	0.80	4.45
twonorm	1.37	8.67	1.37	0.76	8.24	1.61
waveform	0.92	0.92	1.30	0.93	1.62	1.96
image	1.00	1.45	0.67	0.70	2.09	1.13
splice	0.13	3.04	1.00	1.11	3.07	1.00

Table 5.4: *Performance of OC-SVM and 2ν-SVM methods for MV-set estimation. Scores reported are $\mathcal{E}_\mu$ and $\mathcal{E}_+$, averaged over all 100 (or 20) permutations, for $\beta = 0.99$.*

Dataset	$\mathcal{E}_\mu$			$\mathcal{E}_+$		
	OC	Thin	Man	OC	Thin	Man
banana	0.92	9.59	1.73	1.70	9.30	1.76
breast-cancer	4.22	1.57	1.29	4.88	1.89	1.57
diabetes	1.33	2.90	24.09	1.90	3.59	24.13
flare-solar	1.91	2.97	5.98	2.29	3.89	6.76
heart	7.85	2.21	18.43	8.10	2.36	18.62
ringnorm	5.89	42.16	3.78	4.94	41.58	3.74
thyroid	3.68	13.78	11.75	3.97	13.66	11.80
twonorm	7.01	44.20	7.35	6.40	43.79	7.65
waveform	2.43	4.17	2.93	2.64	4.86	3.80
image	1.37	5.57	1.48	1.84	6.17	2.15
splice	4.22	3.05	1.00	5.20	3.08	1.00

Table 5.5: *Performance of MV-set estimation techniques for NP classification. Scores reported are $P_F(f)$ and $P_M(f)$, averaged over all 100 (or 20) permutations, for $\beta = 0.9$ ($\alpha = 0.1$).*

Dataset	NP-SVM		OC-SVM		Thin		Man	
	$P_F(f)$	$P_M(f)$	$P_F(f)$	$P_M(f)$	$P_F(f)$	$P_M(f)$	$P_F(f)$	$P_M(f)$
banana	.10	.13	.13	.49	.20	.21	.09	.26
breast-cancer	.11	.69	.08	.87	.12	.62	.15	.61
diabetes	.10	.50	.08	.80	.11	.67	.10	.71
flare-solar	.09	.56	.07	.89	.11	.57	.08	.83
waveform	.09	.13	.08	.86	.13	.90	.13	.63
image	.01	.00	.07	.66	.12	.63	.12	.33
splice	.02	.03	.08	.96	.12	.88	.00	1



(a)

(b)

Figure 5.2: *MV-set estimates applied to NP classification. (a) Classifier (MV-set estimate)* without *knowledge of “+” class, (b) Classifier* with *knowledge of “+” class.*

Chapter 6

Conclusion

While in some cases it may be reasonable to minimize the misclassification rate, in many important settings this is impractical. To tackle this problem we have considered two frameworks: minimax and NP classification. We have demonstrated that the 2ν-SVM is an efficient and effective algorithm for learning in both of these settings, and clearly outperforms the bias-shifting strategies commonly used in practice. An important observation from our study, which has placed a strong emphasis on using an objective and appropriate performance measure, is that error estimation for parameter estimation is crucial to a successful algorithm, with smoothing and coordinate descent strategies leading to significant gains in performance.

These techniques extend to the anomaly detection setting through the idea of MV-sets. We have observed that no method is clearly better than any other, but wild differences in performance are possible over different data sets. This leads us to believe that significant improvement should be possible through a better understanding of the reasons why an algorithm fails when it does so.

Appendix A

Proofs of Theorems

In [4] a detailed relationship between (D_ν) and (D_C) was established. We shall follow a similar course using the same technique. First we rescale (D_{2C}) by Cn in order to compare it with $(D_{2\nu})$. This gives us:

$$(D'_{2C}) \qquad \min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i\alpha_j y_i y_j k(\mathbf{x}_i,\mathbf{x}_j) - \frac{1}{Cn}\sum_{i=1}^{n}\alpha_i$$

$$\begin{aligned} \text{subject to} \qquad & 0 \le \alpha_i \le \frac{\gamma}{n} && \text{for } i \in I_+ \\ & 0 \le \alpha_i \le \frac{1-\gamma}{n} && \text{for } i \in I_- \\ & \sum_{i=1}^{n} \alpha_i y_i = 0. \end{aligned}$$

Rather than proving the theorems in Section 2.4 directly, we will take advantage of the relationship between (D_{2C}) and (D'_{2C}). We will establish equivalent theorems (which we denote Theorems A, B, and C) relating $(D_{2\nu})$ and (D'_{2C}), which are then trivially extended to the theorems stated in Section 2.4. We begin with the following lemma:

Lemma 1. *Fix $\gamma \in [0,1]$, $C > 0$, and $\nu \in [0,1]$. Assume (D'_{2C}) and ($D_{2\nu}$) share one optimal solution $\boldsymbol{\alpha}^C$ with $\sum_{i=1}^n \alpha_i^C = \nu$. Then any $\boldsymbol{\alpha}$ is an optimal solution of (D'_{2C}) if and only if it is an optimal solution of ($D_{2\nu}$).*

Proof. The analogue of this lemma for (D'_C) and (D_ν) is proved in [4]. The proof depends only on the form of the objective functions (specifically not taking the constraints into account) and on the analogue of Proposition 2. Since the objective function of (D_ν) is identical to that of $(D_{2\nu})$ and the objective function of (D'_C) is also identical to that of (D'_{2C}), we refer the reader to [4] and omit the proof. □

For the proofs of Theorems A and B, we will need to employ the Karush-Kuhn-Tucker (KKT) conditions. Essentially, the KKT conditions are necessary and sufficient conditions for $\boldsymbol{\alpha}$ to be an optimal solution to our optimization problem. Specifically, $\boldsymbol{\alpha}$ is an optimal solution of (D'_{2C}) if and only if there exist $b \in \mathbb{R}$ and $\boldsymbol{\lambda}, \boldsymbol{\xi} \in \mathbb{R}^n$ satisfying the KKT conditions:

$$\sum_{j=1}^n \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) - \frac{1}{Cn} + b y_i = \lambda_i - \xi_i \qquad \text{for } i = 1, \dots, n \tag{A.1}$$

$$\lambda_i \alpha_i = 0, \quad \lambda_i \geq 0, \quad \xi_i \geq 0 \qquad \text{for } i = 1, \dots, n \tag{A.2}$$

$$\xi_i \left(\frac{\gamma}{n} - \alpha_i \right) = 0, \quad 0 \leq \alpha_i \leq \frac{\gamma}{n} \qquad \text{for } i \in I_+ \tag{A.3}$$

$$\xi_i \left(\frac{1-\gamma}{n} - \alpha_i \right) = 0, \quad 0 \leq \alpha_i \leq \frac{1-\gamma}{n} \qquad \text{for } i \in I_- \tag{A.4}$$

$$\sum_{i=1}^n \alpha_i y_i = 0. \tag{A.5}$$

Similarly, $\boldsymbol{\alpha}$ is an optimal solution of $(D_{2\nu})$ if and only if there exist $b, \rho \in \mathbb{R}$ and

$\boldsymbol{\lambda}, \boldsymbol{\xi} \in \mathbb{R}^n$ satisfying the slightly different KKT conditions:

$$\sum_{j=1}^{n} \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) - \rho + b y_i = \lambda_i - \xi_i \qquad \text{for } i = 1, \ldots, n \qquad \text{(A.6)}$$

$$\lambda_i \alpha_i = 0, \quad \lambda_i \geq 0, \quad \xi_i \geq 0 \qquad \text{for } i = 1, \ldots, n \qquad \text{(A.7)}$$

$$\xi_i \left(\frac{\gamma}{n} - \alpha_i \right) = 0, \; 0 \leq \alpha_i \leq \frac{\gamma}{n} \qquad \text{for } i \in I_+ \qquad \text{(A.8)}$$

$$\xi_i \left(\frac{1-\gamma}{n} - \alpha_i \right) = 0, \; 0 \leq \alpha_i \leq \frac{1-\gamma}{n} \qquad \text{for } i \in I_- \qquad \text{(A.9)}$$

$$\sum_{i=1}^{n} \alpha_i y_i = 0, \quad \sum_{i=1}^{n} \alpha_i \geq \nu, \quad \rho \left(\sum_{i=1}^{n} \alpha_i - \nu \right) = 0. \qquad \text{(A.10)}$$

Notice that the two sets of conditions are mostly identical, except for the first and last two of the conditions for $(D_{2\nu})$. Using this observation, we can prove equivalent versions of Theorems 1 and 2.

Theorem A. *Fix $\gamma \in [0, 1]$. For any $C > 0$, let $\boldsymbol{\alpha}^C$ be any optimal solution of (D'_{2C}) and set $\nu = \sum_{i=1}^{n} \alpha_i^C$. Then any $\boldsymbol{\alpha}$ is an optimal solution of (D'_{2C}) if and only if it is an optimal solution of $(D_{2\nu})$.*

Proof. If $\boldsymbol{\alpha}^C$ is an optimal solution of (D'_{2C}) then it satisfies the KKT conditions for (D'_{2C}). By setting $\nu = \sum_{i=1}^{n} \alpha_i^C$ and $\rho = 1/(Cn)$, we see that $\boldsymbol{\alpha}^C$ also satisfies the KKT conditions for $(D_{2\nu})$ and thus is an optimal solution of $(D_{2\nu})$. From Lemma 1 we therefore have that, for any $\boldsymbol{\alpha}$, $\boldsymbol{\alpha}$ is an optimal solution of (D'_{2C}) if and only if it is an optimal solution of $(D_{2\nu})$. □

Theorem B. *Fix $\gamma \in [0, 1]$. For any $\nu \in (0, \nu_{\max}]$, assume $(D_{2\nu})$ has a nonzero* *optimal objective value, in which case $\rho > 0$, and set $C = 1/(\rho n)$. Then any $\boldsymbol{\alpha}$ is an optimal solution of (D'_{2C}) if and only if it is an optimal solution of ($D_{2\nu}$).*

Proof. If $\boldsymbol{\alpha}^\nu$ is an optimal solution of $(D_{2\nu})$ then it satisfies the KKT conditions for $(D_{2\nu})$. From condition (A.6) we have

$$\sum_{i=1}^{n}\left(\sum_{j=1}^{n}\alpha_j^\nu y_i y_j k(\mathbf{x}_i,\mathbf{x}_j) - \rho + b y_i\right)\alpha_i^\nu = \sum_{i=1}^{n}(\lambda_i - \xi_i)\alpha_i^\nu$$

which, by applying the first conditions of (A.7) and (A.8), reduces to

$$\sum_{i,j=1}^{n}\alpha_i^\nu\alpha_j^\nu y_i y_j k(\mathbf{x}_i,\mathbf{x}_j) - \rho\sum_{i=1}^{n}\alpha_i^\nu = -\frac{\gamma}{n}\sum_{i=1}^{n}\xi_i.$$

By assumption, $(D_{2\nu})$ has a nonzero optimal objective value. Thus from Proposition 2, $\sum_{i=1}^{n}\alpha_i^\nu = \nu$, and

$$\rho = \frac{1}{\nu}\left(\sum_{i,j=1}^{n}\alpha_i^\nu\alpha_j^\nu y_i y_j k(\mathbf{x}_i,\mathbf{x}_j) + \frac{\gamma}{n}\sum_{i=1}^{n}\xi_i\right) > 0.$$

Thus we can choose $C > 0$ such that $C = 1/(\rho n)$ and $\boldsymbol{\alpha}^\nu$ is a KKT point of (D'_{2C}), and from Lemma 1 we again have that for any $\boldsymbol{\alpha}$, $\boldsymbol{\alpha}$ is an optimal solution of (D'_{2C}) if and only if it is an optimal solution of $(D_{2\nu})$. □

We will need the following lemmas to prove Theorem C.

Lemma 2. *Fix $\gamma \in [0,1]$ and $\nu \in [0,1]$. If the optimal objective value of ($D_{2\nu}$) is zero and there is a $C > 0$ such that the optimal solution of (D'_{2C}), $\boldsymbol{\alpha}^C$ satisfies*

$\sum_{i=1}^{n} \alpha_i^C = \nu$, then $\nu = \nu_{\max}$ and any $\boldsymbol{\alpha}$ is an optimal solution of ($D_{2\nu}$) if and only if it is an optimal solution for all (D'_{2C}), $C > 0$.

Proof. By setting $\rho = 1/Cn$, $\boldsymbol{\alpha}^C$ is a KKT point of $(D_{2\nu})$. Therefore, if the optimal objective value of $(D_{2\nu})$ is zero, then $\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i^C \alpha_j^C y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) = 0$. Since k is a positive definite kernel, we also have $\sum_{j=1}^{n} \alpha_j^C y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) = 0$. In this case, conditions (A.1)/(A.6) become

$$-\frac{1}{Cn} + by_i = \lambda_i - \xi_i \quad \text{for} \quad i = 1, \dots, n,$$

or

$$\begin{aligned} -\frac{1}{Cn} + b &= \lambda_i - \xi_i \quad \text{for} \quad i \in I_+ \\ -\frac{1}{Cn} - b &= \lambda_i - \xi_i \quad \text{for} \quad i \in I_-. \end{aligned}$$

Assume first that $b \geq 0$, then

$$\lambda_i - \xi_i < 0 \quad \text{for} \quad i \in I_-.$$

This implies that $\xi_i > 0$ for all $i \in I_-$ since both λ_i and ξ_i are nonnegative. Therefore, in order for the first conditions of (A.3)/(A.8), $\xi_i((1-\gamma)/n - \alpha_i^C) = 0$, to hold, we need $\alpha_i^C = (1-\gamma)/n$ for all $i \in I_-$. From the first conditions of (A.5)/(A.10) we have that $\sum_{i \in I_+} \alpha_i^C = \sum_{i \in I_-} \alpha_i^C$. Therefore we need $\sum_{i \in I_+} \alpha_i^C = (1-\gamma)n_-/n \leq \gamma n_+/n$.

Hence, if $(1-\gamma)n_- > \gamma n_+$ we have reached a contradiction, in which case it must be that $b < 0$.

Therefore, assume without loss of generality that $b \geq 0$ (since we can always relabel the points so that this would be true), in which case $(1-\gamma)n_- \leq \gamma n_+$ and $\alpha_i^C = (1-\gamma)/n$ for all $i \in I_-$. There are three possibilities for $i \in I_+$:

1. $\lambda_i - \xi_i < 0$
2. $\lambda_i - \xi_i > 0$
3. $\lambda_i - \xi_i = 0$.

In case 1, where $\lambda_i - \xi_i < 0$, it must be that $\xi_i > 0$ for all $i \in I_+$. For the first conditions of (A.3)/(A.8), $\xi_i(\gamma/n - \alpha_i^C) = 0$, to hold, we need $\alpha_i^C = \gamma/n$ for all $i \in I_+$. The requirement that $\sum_{i \in I_+} \alpha_i^c = \sum_{i \in I_-} \alpha_i^c$ (from the first conditions of (A.5)/(A.10)) and the fact that $\alpha_i^C = (1-\gamma)/n$ for all $i \in I_-$ imply that $\sum_{i=1}^n \alpha_i^C = 2n_+\gamma/n = 2n_-(1-\gamma)/n = \nu_{\max}$. Furthermore, since the optimal objective value of $(D_{2\nu})$ is zero, the objective function for (D'_{2C}) in this case becomes

$$\min_{\boldsymbol{\alpha}} \quad -\frac{1}{Cn}\sum_{i=1}^n \alpha_i.$$

This is clearly minimized by $\boldsymbol{\alpha}^C$ (since $\sum_{i=1}^n \alpha_i^C = \nu_{\max}$) for all $C > 0$, thus $\boldsymbol{\alpha}^C$ is an optimal solution of (D'_{2C}) for all $C > 0$.

In case 2, where $\lambda_i - \xi_i > 0$, we have that $\lambda_i > 0$ for all $i \in I_-$. For the first conditions of (A.2)/(A.7), $\lambda_i \alpha_i^C = 0$, to hold, we need $\alpha_i^C = 0$ for all $i \in I_+$. However, the requirement that $\sum_{i \in I_+} \alpha_i^C = \sum_{i \in I_-} \alpha_i^C$ and the fact that $\alpha_i^C = (1-\gamma)/n$ for all $i \in I_-$ lead to a contradiction if I_- is nonempty. Hence all the training vectors are in the same class, and $\alpha_i^C = 0$ for all i. Thus, $\sum_{i=1}^n \alpha_i^C = 0 = \nu_{\max}$. Furthermore, if all the data are from the same class then $\boldsymbol{\alpha}^C = \mathbf{0}$ is an optimal solution of (D'_{2C}) for all $C > 0$.

In case 3, where $\lambda_i - \xi_i = 0$, we have that either $\lambda_i = \xi_i \neq 0$ or $\lambda_i = \xi_i = 0$ for each $i \in I_+$. However, $\lambda_i = \xi_i \neq 0$ leads to a contradiction because the conditions (A.2)/(A.7) and (A.3)/(A.8) require both $\alpha_i^C = 0$ and $\alpha_i^C = \gamma/n$. Thus, $\lambda_i = \xi_i = 0$ and the KKT conditions involving λ_i and ξ_i impose no conditions on α_i^C for $i \in I_+$. Since $\alpha_i^C = (1-\gamma)/n$ for all $i \in I_-$, and $(1-\gamma)n_- \leq \gamma n_+$, we can satisfy $\sum_{i \in I_+} \alpha_i^C = \sum_{i \in I_-} \alpha_i^C = (1-\gamma)n_+/n$. Thus, $\sum_{i=1}^n \alpha_i^C = \nu_{\max}$. Furthermore, by setting $b = 1/(Cn)$, $\boldsymbol{\alpha}^C$ is an optimal solution of (D'_{2C}) for all $C > 0$.

Therefore, in all three cases we have that $\nu = \nu_{\max}$ and that $\boldsymbol{\alpha}^C$ is an an optimal solution of (D'_{2C}), for all $C > 0$. Hence, if $\boldsymbol{\alpha}^C$ is an an optimal solution of (D'_{2C}) and for $\nu = \sum_{i=1}^n \alpha_i^C$ the optimal objective value of $(D_{2\nu})$ is zero, then $\nu = \nu_{\max}$ and $\boldsymbol{\alpha}^C$ is an an optimal solution of (D'_{2C}), for all $C > 0$. Combining this with Lemma 1 we get that any $\boldsymbol{\alpha}$ is an optimal solution of $(D_{2\nu})$ if and only if it is an optimal solution for all (D'_{2C}), $C > 0$. □

Lemma 3. *Assume $\boldsymbol{\alpha}^C$ is any optimal solution of (D'_{2C}), then $\sum_{i=1}^n \alpha_i^C$ is a continuous decreasing function of C on $(0, \infty)$.*

Proof. Again, the analogue of this lemma for (D'_C) is proved in [4]. Since the proof depends only on the form of the objective function and the analogues of Theorems A and B and Lemma 2, we omit the proof and refer the reader to [4]. □

Using these lemmas, we are now ready to prove the equivalent of the main theorem:

Theorem C. *Fix $\gamma \in [0,1]$ and let $\boldsymbol{\alpha}^C$ be any optimal solution of (D'_{2C}). Define*

$$\nu_* = \lim_{C\to\infty} \sum_{i=1}^{n} \alpha_i^C$$

and

$$\nu^* = \lim_{C\to 0} \sum_{i=1}^{n} \alpha_i^C .$$

Then $0 \le \nu_* \le \nu^* = \nu_{\max} \le 1$. For any $\nu > \nu^*$, ($D_{2\nu}$) is infeasible. For any $\nu \in (\nu_*, \nu^*]$, the optimal objective value of ($D_{2\nu}$) is strictly positive, and there exists at least one $C > 0$ such that any $\boldsymbol{\alpha}$ is an optimal solution of (D'_{2C}) if and only if it is an optimal solution of ($D_{2\nu}$). For any $\nu \in [0, \nu_*]$, ($D_{2\nu}$) is feasible with an optimal objective value of zero (and a trivial solution).

Proof. From Lemma 3, and the fact that for all C, $0 \le \sum_{i=1}^{n} \alpha_i^C \le \nu_{\max}$, we know that the above limits are well-defined and exist.

For any optimal solution of (D'_{2C}), we have that condition (A.1) holds:

$$\sum_{j=1}^{n} \alpha_j^C y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) - \frac{1}{Cn} + b = \lambda_i - \xi_i \quad \text{for} \quad i \in I_+$$

$$\sum_{j=1}^{n} \alpha_j^C y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) - \frac{1}{Cn} - b = \lambda_i - \xi_i \quad \text{for} \quad i \in I_- .$$

Assume first that $b \geq 0$. In this case, since $\boldsymbol{\alpha}^C$ is bounded, when C is sufficiently small, we will necessarily have $\lambda_i - \xi_i < 0$ for all $i \in I_+$. Pick such a C. Since ξ_i and λ_i are nonnegative, $\xi_i > 0$ for all $i \in I_+$, and from condition (A.3), $\alpha_i^C = \gamma/n$ for all $i \in I_+$. If $\gamma n_+/n \geq (1-\gamma)n_-/n$, then this $\boldsymbol{\alpha}^C$ is feasible and $\sum_{i=1}^n \alpha_i^C = \nu_{\max}$. However, if $\gamma n_+/n < (1-\gamma)n_-/n$ then we have a contradiction, and thus it must actually be that $b < 0$. In this case, for C sufficiently small, $\lambda_i - \xi_i < 0$ for all $i \in I_i$. As before, this now implies that $\alpha_i^C = (1-\gamma)/n$ for all $i \in I_-$, and thus $\sum_{i=1}^n \alpha_i^C = \nu_{\max}$. Hence, $\nu^* = \sum_{i=1}^n \alpha_i^C = \nu_{\max}$, and from Proposition 1 we immediately know that $(D_{2\nu})$ is infeasible if $\nu > \nu^*$.

For all $\nu \leq \nu^*$, from Proposition 1 $(D_{2\nu})$ is feasible. From Lemma 3 we know that $\sum_{i=1}^n \alpha_i^C$ is a continuous decreasing function. Thus for any $\nu \in (\nu_*, \nu^*]$, there is a $C > 0$ such that $\sum_{i=1}^n \alpha_i^C = \nu$, and by Lemma 1 any $\boldsymbol{\alpha}$ is an optimal solution of $(D_{2\nu})$ if and only if it is an optimal solution for (D'_{2C}).

Finally, we consider $\nu \in [0, \nu_*]$. If $\nu < \nu_*$, $(D_{2\nu})$ must have an optimal objective value of zero because otherwise, by the definition of ν_*, this would contradict Theorem B. If $\nu = \nu_* = 0$, the optimal objective value of $(D_{2\nu})$ is zero, as $\boldsymbol{\alpha}^\nu = \mathbf{0}$ is a feasible solution. If $\nu = \nu_* > 0$, the fact that feasible regions of $(D_{2\nu})$ are bounded by $0 \leq \alpha_i \leq \gamma/n$ for $i \in I_+$ and $0 \leq \alpha_i \leq (1-\gamma)/n$ for $i \in I_-$, and Proposition 2 imply that there exists a sequence $\{\boldsymbol{\alpha}^{\nu_j}\}$, $\nu_1 \leq \nu_2 \leq \cdots \leq \nu_*$ such that $\boldsymbol{\alpha}^{\nu_j}$ is an optimal solution of $(D_{2\nu})$ with $\nu = \nu_j$, $\sum_{i=1}^n \alpha_i^{\nu_j} = \nu_j$, and $\boldsymbol{\alpha}^* = \lim_{\nu_j \to \nu_*} \boldsymbol{\alpha}^{\nu_j}$ exists. Since $\sum_{i=1}^n \alpha_i^{\nu_j} = \nu_j$, $\sum_{i=1}^n \alpha_i^* = \lim_{\nu_j \to \nu_*} \sum_{i=1}^n \alpha_i^{\nu_j} = \nu_*$. We also have that $0 \leq \alpha_i^* \leq \gamma/n$

for $i \in I_+$, $0 \leq \alpha_i^* \leq (1-\gamma)/n$ for $i \in I_-$, and $\sum_{i=1}^n y_i \alpha_i^* = \lim_{\nu_j \to \nu_*} y_i \sum_{i=1}^n \alpha_i^{\nu_j} = 0$ so $\boldsymbol{\alpha}^*$ is feasible to $(D_{2\nu})$ for $\nu = \nu_*$. However, $\sum_{l,m=1}^n \alpha_l^* \alpha_m^* y_l y_m k(\mathbf{x}_l, \mathbf{x}_m) = \lim_{\nu_j \to \nu_*} \sum_{l,m=1}^n \alpha_l^{\nu_j} \alpha_m^{\nu_j} y_l y_m k(\mathbf{x}_l, \mathbf{x}_m) = 0$ as $\sum_{l,m=1}^n \alpha_l^{\nu_j} \alpha_m^{\nu_j} y_l y_m k(\mathbf{x}_l, \mathbf{x}_m) = 0$ for all ν_j. Therefore the optimal objective value of $(D_{2\nu})$ is zero if $\nu = \nu_*$. Thus the optimal objective value of $(D_{2\nu})$ is zero for all $\nu \in [0, \nu_*]$.

Now suppose for the sake of a contradiction that the optimal objective value of $(D_{2\nu})$ is zero but $\nu > \nu_*$. By Lemma 3 there exists a $C > 0$ such that, if $\boldsymbol{\alpha}^C$ is an optimal solution of (D'_{2C}), then $\sum_{i=1}^n \alpha_i^C = \nu$. From Lemma 2, $\nu = \nu_{\max} = \nu^* = \nu_*$, since $\sum_{i=1}^n \alpha_i^C$ is the same for all C. This contradicts the assumption that $\nu > \nu_*$. Thus the objective value of $(D_{2\nu})$ can be zero if and only if $\nu \leq \nu_*$. In this case, $\mathbf{w} = 0$ and we say that the solution is *trivial.* □

Bibliography

[1] S. Bengio, J. Mariéthoz, and M. Keller. The expected performance curve. In *Proc. Int. Conf. Machine Learning*, 2005. Bonn, Germany.

[2] A. P. Bradley. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159, 1997.

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

[4] C. C. Chang and C. J. Lin. Training ν-support vector classifiers: Theory and algorithms. *Neural Computation*, 13:2119–2147, 2001.

[5] H. G. Chew, R. E Bogner, and C. C. Lim. Target detection in radar imagery using support vector machines with training size biasing. In *Proc. Int. Conf. on Control, Automation, Robotics, and Vision (ICARCV)*, 2000.

[6] H. G. Chew, R. E. Bogner, and C. C. Lim. Dual-ν support vector machine with error rate and training size biasing. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Processing (ICASSP)*, pages 1269–1272, 2001.

[7] H. G. Chew, C. C. Lim, and R. E. Bogner. Dual-nu support vector machines and applications in multi-class image recognition. In *Proc. Int. Conf. on Optimization: Techniques and Applications (ICOTA)*, 2004.

[8] C. Cortes and M. Mohri. AUC optimization vs. error rate minimization. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16 (NIPS 2003)*, Cambridge, MA, 2004. MIT Press.

[9] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

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

[11] M. A. Davenport, R. G. Baraniuk, and C. D. Scott. Controlling false alarms with support vector machines. In *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Processing (ICASSP)*, 2006. Toulouse, France.

[12] M. A. Davenport, R. G. Baraniuk, and C. D. Scott. Learning minimum volume sets with support vector machines. In *Proc. IEEE Int. Work. Machine Learning for Signal Processing (MLSP)*, 2006. Maynooth, Ireland.

[13] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *J. Machine Learning Research*, 7:1–30, 2006.

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

[15] G. R. G. Lanckriet, L. El Ghaoui, C. Bhattacharyya, and M. I. Jordan. A robust minimax approach to classification. *J. Machine Learning Research*, 3:555–582, 2002.

[16] G. R. G. Lanckriet, L. El Ghaoui, and M. I. Jordan. Robust novelty detection with single-class MPM. In *Proc. Adv. in Neural Processing Systems (NIPS)*, 2003.

[17] Y. Lin, Y. Lee, and G. Wahba. Support vector machines for classification in nonstandard situations. Technical Report Technical Report No. 1016, University of Wisconsin, Dept. of Statistics, March, 2000.

[18] E. Osuna, R. Freund, and F. Girosi. Support vector machines: Training and applications. Technical Report A.I. Memo No. 1602, MIT Artificial Intelligence Laboratory, March 1997.

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

[20] F. Provost, T. Fawcett, and R. Kohavi. The case against accuracy estimation for comparing induction algorithms. In *Proc. Int. Conf. Machine Learning*, 1998. San Francisco, CA.

[21] S. Rosset. Model selection via the AUC. In R. Greiner and D. Schuurmans, editors, *Proc. Int. Conf. Machine Learning*, 2004.

[22] 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*, 13:1443–1471, 2001.

[23] B. Schölkopf, A. J. Smola, R. Williams, and P. Bartlett. New support vector algorithms. *Neural Computation*, 12:1083–1121, 2000.

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

[25] C. D. Scott and R. D. Nowak. A Neyman-Pearson approach to statistical learning. *IEEE Trans. Inform. Theory*, 51(11):3806–3819, 2005.

[26] C. D. Scott and R. D. Nowak. Learning minimum volume sets. *J. Machine Learning Research*, 7:665–704, 2006.

[27] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34:1–47, 2002.

[28] J. Shao. Linear model selection by cross-validation. *J. Amer. Statist. Assoc.*, 88(422):486–494, 1993.

[29] I. Steinwart. Support vector machines are universally consistent. *J. Complexity*, 18:768–791, 2002.

[30] I. Steinwart, D. Hush, and C. Scovel. A classification framework for anomaly detection. *J. Machine Learning Research*, 6:211–232, 2005.

[31] D. M. J. Tax and R. P. W. Duin. Support vector domain description. *Pattern Recognition Letters*, 20(26):1191–1199, 1999.

[32] D. M. J. Tax and R. P. W. Duin. Uniform object generation for optimizing one-class classifiers. *J. Machine Learning Research*, 2:155–713, 2001.

[33] J. Theiler and D. M. Cai. Resampling approach for anomaly detection in multispectral images. In *Proc. SPIE*, volume 5093, pages 230–240, 2003.

[34] H. Van Trees. *Detection, Estimation, and Modulation Theory: Part I.* John Wiley & Sons, New York, 2001.

[35] K. Veropoulos, N. Cristianini, and C. Campbell. Controlling the sensitivity of support vector machines. In *Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI)*, 1999.

[36] R. Vert and J. P. Vert. Consistency and convergence rates of one-class SVMs and related algorithms. *J. Machine Learning Research*, 7:817–854, 2006.

[37] R. S. Wagner, R. G. Baraniuk, S. Du, D. B. Johnson, and A. Cohen. An architecture for distributed wavelet analysis and processing in sensor networks. In *Proc. Int. Symp. Inform. Processing in Sensor Networks (IPSN)*, 2006. Nashville, TN.

[38] G. Walther. Granulometric smoothing. *Ann. Stat.*, 25:2273–2299, 1997.