Published in final edited form as:

J Comput Graph Stat. 2013 ; 22(2): . doi:10.1080/10618600.2012.680866.

Adaptively Weighted Large Margin Classifiers

Yichao Wu [Associate Professor] and

Department of Statistics, Temple University, Philadelphia, PA 19122 (yichao.wu@temple.edu)

Yufeng Liu [Associate Professor]

Department of Statistics and Operations Research, Carolina Center for Genome Sciences,
University of North Carolina, Chapel Hill, NC 27599 (yfliu@email.unc.edu)

Abstract

Large margin classifiers have been shown to be very useful in many applications. The Support Vector Machine is a canonical example of large margin classifiers. Despite their flexibility and ability in handling high dimensional data, many large margin classifiers have serious drawbacks when the data are noisy, especially when there are outliers in the data. In this paper, we propose a new weighted large margin classification technique. The weights are chosen adaptively with data. The proposed classifiers are shown to be robust to outliers and thus are able to produce more accurate classification results.

Key Words and Phrases

Binary classification; data adaptive learning; large margin; multcategory classification; SVM; weighted learning

1 Introduction

Classification is a very useful statistical means for information extraction from data. As a supervised learning technique, the goal of classification is to construct a classification rule based on a training set where both covariates and class labels are given. Once obtained, the classification rule can then be used for class prediction of new objects whose covariates are available. It can be understood as a special form of regression with the response variable being categorical. When the response variable is binary, the learning problem is known as binary classification. If there are more than two classes, we have multcategory classification.

There are a large number of methods for classification in the literature. Examples include Fisher Linear Discrimination Analysis, logistic regression, k-nearest neighbor, decision trees, neural networks, boosting, and many others. See Hastie et al. (2001) for more comprehensive reviews of various classification methods. Among numerous classification techniques, margin-based classifiers have attracted tremendous attentions in recent years due to their competitive performance and ability in handling high dimensional data. The Support Vector Machine (SVM) is one of the most well known large margin classifiers. Since its introduction, the SVM has gained much popularity in both machine learning and statistics. The seminal work by Vapnik (1995, 1998) has laid the foundation for the general statistical

Correspondence to: Yufeng Liu.

Supplementary materials

The Supplementary files for this article include a zip file (WSVMcodes.zip) of Matlab codes used in Section 5 and the Appendix which gives the proof of Theorem 1.

learning theory of the SVM, which furthermore inspired various extensions of the SVM. For other references on the binary SVM, see Cristianini and Shawe-Taylor (2000); Schölkopf and Smola (2002), and references therein. Besides the SVM, there are a number of other large margin classifiers introduced in the literature. Examples include the Penalized Logistic Regression (PLR) (Wahba, 1998; Lin et al., 2000; Zhu and Hastie, 2005), ψ -learning (Shen et al., 2003; Liu and Shen, 2006), the robust SVM (Wu and Liu, 2007), Distance-Weighted Discrimination (DWD) (Marron et al., 2007), and so on.

Despite progress on large margin classifiers, there are still drawbacks in certain situations. For example, when the data are noisy with possible outliers in the data, techniques such as the SVM may be affected. In particular, when there exist points far from their own classes, the SVM solution can be potentially pulled towards these outlying points due to the unbounded hinge loss function. To achieve robustness, Shen et al. (2003) proposed ψ -learning and Wu and Liu (2007) further extended ψ -learning to the robust SVM. Although these methods can yield more accurate classifiers than the original SVM when the data are noisy, the corresponding optimization problem is nonconvex and the computation can be much more difficult than that of the SVM.

In this article, we propose a different approach to gain robustness. Instead of tackling a nonconvex loss function as in ψ -learning and the RSVM, we consider weighted learning to achieve more accurate classifiers. In particular, we would like to use larger weights for points closer to the boundaries and smaller weights for points farther away. The impact of outliers will be reduced using smaller weights. As a result, the weighted classifiers can deliver better classification performance than the unweighted ones. For the choice of weights, we utilize the results of original unweighted classifiers to determine the relative locations of points to the decision boundary. The weighted classifiers will be able to refine the classification boundaries with robustness to outliers. Our focus in this paper is on the SVM. However, the idea of weighting and adaptive learning is applicable to general large margin classifiers, such as those considered in Zou et al. (2008).

We propose two different weighted SVMs in this paper, the one-step weighted SVM (OWSVM) and the iteratively weighted SVM (IWSVM). We show that both methods yield more accurate classifiers than the original SVM when the data are noisy. The OWSVM only uses one-step weighting and thus is more computationally efficient than the IWSVM, which requires iteration. Furthermore, we demonstrate that the IWSVM has a close relationship with the RSVM proposed by Wu and Liu (2007). In particular, the IWSVM is equivalent to the difference convex algorithm (DCA) of the RSVM as used in Wu and Liu (2007) in a certain sense (as developed in Section 3) and consequently, the IWSVM can be viewed as a different way of implementing the RSVM.

The rest of this article is organized as follows. We briefly review the SVM and then introduce its weighted extension in Section 2. The OWSVM is proposed. Section 3 introduces the IWSVM and discusses its connection with the RSVM. The extension to the multicategory case is given in Section 4. Both simulated and real data examples are used to demonstrate the performance of the proposed methods in Section 5. Some discussion is provided in Section 6. The appendix contains the proof of the theorem.

2 Binary SVM and Its Weighted Extension

For a classification problem, we are given a training sample $\{(\mathbf{x}_i, y_i) : i = 1, 2, \dots, n\}$ which is distributed according to some unknown probability distribution function $P(\mathbf{x}; y)$. Here, $\mathbf{x}_i \in \mathcal{S} \subset \mathbb{R}^d$ and y_i denote the input vector and output label respectively, where n is the sample

size, and d is the dimensionality of the input space. In this section, we first review the method of SVM and then introduce the weighted SVM (WSVM).

2.1 The Standard Binary SVM

For illustration, we first briefly describe the linear binary SVM. Let $y \in \{\pm 1\}$ and $f(\mathbf{x}) = \beta^T \mathbf{x} + \beta_0$. The standard SVM aims to find $f(\mathbf{x})$ so that $\hat{y} = \text{sign}(f(\mathbf{x}))$ can be used for prediction. In particular, for a separable problem, SVM solves the following optimization problem:

$$\max_{\beta, \beta_0} \frac{2}{\|\beta\|_2} \quad (1)$$

$$\text{subject to } y_i f(\mathbf{x}_i) \geq 1, \forall_i.$$

In (1), the quantity $\frac{2}{\|\beta\|_2}$, known as the geometric margin, measures the separation between the two classes using the Euclidean distance. The term $y f(\mathbf{x})$ reflects the correctness and strength of classification of f on $\mathbf{x}$ and is called the functional margin.

For a general problem in which (1) may not have any feasible solutions, slack variables ξ_i 's are used to measure the amount of violation of the constraints in problem (1). As a result, the corresponding optimization becomes

$$\min_{\beta, \beta_0} \frac{1}{2} \|\beta\|_2^2 + C \sum_{i=1}^n \xi_i \quad (2)$$

$$\text{subject to } y_i f(\mathbf{x}_i) \geq 1 - \xi_i, \xi_i \geq 0, \forall_i,$$

where $C > 0$ is a tuning parameter.

Optimization formulation in (2) is also known as the primal problem of SVM. Using the Lagrange multipliers, (2) can be converted into an equivalent dual problem as follows:

$$\max_{\alpha} \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle, \quad (3)$$

$$\text{subject to } \sum_{i=1}^n y_i \alpha_i = 0; 0 \leq \alpha_i \leq C, \forall_i,$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product operator. Once the solution of problem (3) is obtained, $\hat{\beta}$ can be calculated as $\sum_{i=1}^n y_i \hat{\alpha}_i \mathbf{x}_i$ and the intercept $\hat{\beta}_0$ can be computed using the Karush-Kuhn-Tucker (KKT) complementarity conditions of the optimization theory. Here we use the hat notations to denote the corresponding optimizer. If nonlinear learning is needed, one can apply the *kernel trick* by replacing the inner product $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$ by $K(\mathbf{x}_i, \mathbf{x}_j)$, where the kernel $K(\cdot, \cdot)$ is a non-negative definite function. This amounts to applying linear learning in the feature space induced by kernel $K(\cdot, \cdot)$ to achieve nonlinear learning in the original input

space. One example is the Gaussian kernel $K(\mathbf{x}_1, \mathbf{x}_2) = \exp(-\|\mathbf{x}_1 - \mathbf{x}_2\|^2 / \sigma^2)$ for some data width parameter $\sigma > 0$. More details about the SVM can be found in Cristianini and Shawe-Taylor (2000).

From problem (3), we can see that among the n training points, only points with $\alpha_i > 0$ will make an impact on the SVM classifier, namely the SVs. It can be shown that these points satisfy that $y_i f(\mathbf{x}_i) = 1$. Consequently, outliers that are far from their own classes will be included as SVs and influence the classifier. Our main contribution in this paper is to reduce the impact of outliers via weighting and deliver more robust classifiers.

It can be shown that the SVM classifier can be fit in the general regularization framework as follows

$$\min_f J(f) + C \sum_{i=1}^n H_1(y_i f(\mathbf{x}_i)). \quad (4)$$

Problem (4) is equivalent to problem (2) with the L_2 penalty $J(f) = \frac{1}{2} \|\beta\|_2^2$ and the hinge loss $H_1(u) = (1 - u)_+$, where $(u)_+ = u$ if $u > 0$ and 0 otherwise.

2.2 Weighted SVM

In weighted learning, we assume that there is a nonnegative weight w_i associated with each pair of observations $(\mathbf{x}_i, y_i)$ for $i = 1, 2, \dots, n$. The proposed weighted SVM solves the following optimization problem

$$\min_{f \in \mathcal{F}} J(f) + C \sum_{i=1}^n w_i H_1(y_i f(\mathbf{x}_i)) \quad (5)$$

to estimate a function $f(\cdot)$ whose sign gives a classification rule. When $w_i = 1$, it reduces to the standard SVM.

When linear learning is considered, the corresponding dual problem becomes

$$\min_{\alpha} \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle - \sum_{i=1}^n \alpha_i, \quad (6)$$

$$\begin{aligned} \text{subject to } & \sum_{i=1}^n \alpha_i y_i = 0, \\ & 0 \leq \alpha_i \leq C w_i, i=1, 2, \dots, n. \end{aligned}$$

Note that the above dual problem is still a quadratic programming problem. The only difference from the standard SVM problem in (3) is that the upper bound for α_i is changed to $C w_i$ in the constraint. Once (6) is solved, solution of the weighted linear SVM (5) can be recovered using $\hat{\beta} = \sum_{i=1}^n \hat{\alpha}_i y_i \mathbf{x}_i$ and $\hat{\beta}_0 = y_m - \mathbf{x}_m^T \hat{\beta}$ for any m such that $0 < \hat{\alpha}_m < C w_m$. Any observation $(\mathbf{x}_i, y_i)$ with $0 < \hat{\alpha}_i < C w_i$ is called a support vector (SV). Classifier function $f(\cdot)$ is estimated by $f(\mathbf{x}) = \hat{\beta}_0 + \mathbf{x}^T \hat{\beta}$.

Nonlinear learning is typically achieved using the kernel trick. Let $K(\cdot, \cdot)$ be a bivariate kernel. Due to the kernel representer theorem, nonlinear classifier function $f(\mathbf{x})$ can be

represented as $c_0 + \sum_{i=1}^n K(\mathbf{x}, \mathbf{x}_i) c_i$. The involved parameter $c_0, c_1, \dots, c_n$ can be estimated by solving

$$\min_{c_0, c_1, \dots, c_n} C \sum_{i=1}^n w_i H_1(y_i(c_0 + \sum_{j=1}^n K(\mathbf{x}_i, \mathbf{x}_j) c_j)) + \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n c_i K(\mathbf{x}_i, \mathbf{x}_j) c_j.$$

The corresponding dual problem is

$$\max_{\alpha_1, \alpha_2, \dots, \alpha_n} \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

$$\text{subject to } \sum_{i=1}^n \alpha_i y_i = 0 \\ 0 \leq \alpha_i \leq C w_i, \quad i=1, \dots, n.$$

Denote the optimizer of the dual problem by $\hat{\alpha}_i, i=1, 2, \dots, n$. Function $f(\cdot)$ is estimated by $\hat{f}(\mathbf{x}) = \hat{c}_0 + \sum_{i=1}^n \hat{\alpha}_i y_i K(\mathbf{x}_i, \mathbf{x})$, where $\hat{c}_0$ is given by $y_m - \sum_{i=1}^n \hat{\alpha}_i y_i K(\mathbf{x}_i, \mathbf{x}_m)$ for any m satisfying $0 < \alpha_m < C w_m$.

2.3 Choice of Weights

The standard SVM uses the same weight $w_i = 1$ for all pairs of observation. However when there are outliers, it is desirable to give smaller weights to observations that are outliers. One natural approach for weighting is to make use of the results from standard learning to understand relative locations of the training data points to the classification boundary. To that end, we first train the standard SVM with weight $w_i = 1$ in (5) using a properly tuned regularization parameter C . Denote the optimal solution as $f_{SVM}(\cdot)$. Next we define weight as $w_i = 1/(1 + |f_{SVM}(\mathbf{x}_i)|)$ for $i = 1, 2, \dots, n$ and train the weighted SVM (5) using this weight.

Our motivation of the weight function comes from the form of the hinge loss function. When $u \geq 0$, the hinge loss $H_1(u)$ is $(1 + |u|)$. To connect with the 0–1 loss $I(u \leq 0)$, we consider the weight form $1/(1 + |u|)$. With the standard learning solution $\hat{f}_{SVM}$, we use the weight $w_i = 1/(1 + |\hat{f}_{SVM}(\mathbf{x}_i)|)$ for the i -th point. The weight function is shown in Figure 1. Note that our proposed weighting scheme assigns a smaller weight for any observation $(\mathbf{x}_i, y_i)$ with a larger $|\hat{f}_{SVM}(\mathbf{x}_i)|$. This matches our goal. In particular, for any observation $(\mathbf{x}_i, y_i)$, it is misclassified if $y_i \hat{f}_{SVM}(\mathbf{x}_i) < 0$. In this case, the smaller $y_i \hat{f}_{SVM}(\mathbf{x}_i)$ is, the more likely for $(\mathbf{x}_i, y_i)$ to be an outlier, as it would indicate that it is far away from its own class. Our weight function will assign smaller weights for such observations in the weighted learning. On the other hand, the observation $(\mathbf{x}_i, y_i)$ is correctly classified if $y_i \hat{f}_{SVM}(\mathbf{x}_i) > 0$. When $y_i \hat{f}_{SVM}(\mathbf{x}_i) > 1$, the corresponding Lagrangian multiplier optimizer α_i is equal to zero and consequently does not contribute to the solution due to its solution representation. Thus, an observation $(\mathbf{x}_i, y_i)$ with a large positive $y_i \hat{f}_{SVM}(\mathbf{x}_i)$ will not affect the solution much and will be assigned a small weight. To further understand the weight function, we display the function $w(u)H_1(u) = \max\{0, 1 - u\} = (1 + |u|)$ in Figure 1, which can be viewed as the target loss through our weighted learning. Note that $w(u)H_1(u)$ is same as the 0–1 loss except the region of $[0, 1]$. The 0–1 loss has a discontinuous jump at 0 while $w(u)H_1(u)$ is a continuous function.

This weighted SVM procedure involves two steps: the first step is to apply the standard learning to calculate the weights for all training data points, and the second step is to apply the weighted learning. We call this procedure the one-step weighted SVM (OWSVM). As we will show in our numerical results, our proposed choice of weights works very well.

One natural question is whether one should continue to iterate the process after the proposed one-step weighted learning. We suggest not to iterate for this choice of weights. The main reason is that the choice of weights is defined based on the original hinge loss and it may not be reasonable anymore to weight using the OWSVM solution. Furthermore, there is no guarantee for convergence if we iterate. Consequently, we recommend this choice of weights for the one-step procedure OWSVM only. In Section 3, we consider another choice of weights and suggest an iterative procedure using the corresponding weights. We show some close connection of this iterative version of the WSVM with the DCA for the RSVM.

3 Connection of Binary RSVM with WSVM

To improve the SVM for noisy data, Wu and Liu (2007) proposed to truncate the hinge loss function and defined the truncated hinge loss $T_s(u) = H_1(u) - H_s(u)$, where $H_s(u) = [s - u]_+$ for $s \geq 0$. The truncated hinge loss function is non-convex and as a result, the corresponding optimization problem is much more challenging. Wu and Liu (2007) proposed to use the DCA to solve the corresponding optimization. They also recommended to use $s = -1/(k - 1)$, where k denotes the number of classes. We will follow this recommendation for the rest of this paper. In this section, we show some connection between the RSVM and WSVM. In particular, we propose an alternative algorithm for the RSVM using the iterative WSVM (IWSVM) implementation. Since each step of the IWSVM is to solve one WSVM, the proposed IWSVM is much simpler to implement. Surprisingly, we show that the proposed IWSVM is equivalent to the DCA in some sense, as we will discuss further.

For further illustration, we note that the RSVM solves

$$\min_f J(f) + C \sum_{i=1}^n T_s(y_i f(\mathbf{x}_i)). \quad (7)$$

Note that the only difference between the standard SVM problem (4) and the RSVM problem (7) is the change of loss from H_1 to T_s . To handle the nonconvexity of T_s , the DCA decomposes T_s as $T_s = H_1(u) - H_s(u)$, a difference of two convex functions. At each iteration with the given current solution, the DCA solves a convex minimization problem via keeping H_1 as it is and replacing H_s by its linear approximation at the current solution. Note that the truncated hinge loss function is flat over $(-\infty, s)$ and $(1, \infty)$. Due to the optimization theory, the optimizer of (7) does not depend on any observation falling inside either $(-\infty, s)$ or $(1, \infty)$ (Wu and Liu, 2007).

Motivating from the RSVM, in this section, we propose an alternative algorithm, IWSVM, for (7) by iteratively updating the weights and solving the corresponding WSVM.

The IWSVM algorithm can be summarized as follows:

Step 1 (Initialization): Solve the WSVM problem (5) with weights $w_i = 1$ for $i = 1, 2, \dots, n$ and denote the optimal solution as $f^{(0)}(\cdot)$.

Step 2 (Iteration): At the t th iteration with $t \geq 1$, define weights as $w_i^{(t)} = 1$ if $s - y_i f^{(t-1)}(\mathbf{x}_i) \geq 1$ and 0 otherwise for $i = 1, 2, \dots, n$. Solve the WSVM (5) with the updated weights $w_i^{(t)}$'s and denote the corresponding optimizer as $f^{(t)}(\cdot)$.

Step 3 (Convergence): Repeat Step 2 until convergence. We claim convergence when $w_i^{(t)} = w_i^{(t-1)}$ for all $i = 1, 2, \dots, n$.

Note that in Step 2, only observations with $s - y_i f^{(t-1)}(x_i) - 1$ have a positive weight 1. Thus the essential size of the corresponding optimization problem (6) is much smaller than n and consequently it is much more efficient to solve the corresponding optimization problem.

As a remark, we note that both the DCA and the proposed IWSVM algorithm solve the RSVM optimization problem (7). Since the corresponding objective function of the RSVM problem (7) is non-convex, both algorithms can only guarantee local optimizers. As typically there are multiple local minimums, we generally cannot guarantee that these two algorithms always lead to the same local solution. However we can show their equivalence in terms of *fixed points* as stated in the following theorem. For an iterative algorithm such as the DCA and IWSVM, a solution is called a *fixed point* if it satisfies the convergence criterion and no further iteration is needed.

Theorem 1. *The DCA and IWSVM algorithms are equivalent in terms of fixed points. Namely, the local solution of the DCA is a fixed point of the proposed IWSVM algorithm, and vice versa.*

So far, we have proposed two weighted SVM algorithms, one is the one-step algorithm OWSVM and the other is the iterative algorithm IWSVM. As we have shown, the IWSVM can be viewed as an algorithm for the RSVM. To further illustrate these two algorithms, we plot both weight functions in Figure 2 for further comparison. Note that the weight for the IWSVM is a step function with value 1 in the interval $[s, 1]$ and 0 elsewhere. In contrast, the weight for the OWSVM is a smooth concave function on both positive and negative sides of the horizontal axis. Another important difference between the two algorithms OWSVM and IWSVM is that the OWSVM algorithm is a one-step procedure and does not iterate. On the other hand, both the DCA and the proposed IWSVM algorithms for the RSVM involve iteration and thus require more computational time.

4 Extensions to Multicategory Learning

So far, we have been focusing on binary classification. In practice, multicategory problems are very common. In this section, we extend the proposed two algorithms, OWSVM and IWSVM, to multicategory cases. For multicategory learning, the response is labeled as $y_i \in \{1, 2, \dots, k\}$, where k denotes the number of classes. The goal is to estimate a function

vector $f(x) = (f_1(x), f_2(x), \dots, f_k(x))^T$. The argmax rule $\arg\max_{j=1,2,\dots,k} f_j(x)$ is used as the classification rule.

There are a number of multicategory SVM (MSVM) techniques proposed in the literature. To solve a multicategory problem using the SVM, typically there are two possible approaches. The first approach is to solve the multicategory problem via a sequence of binary problems, e.g., one-versus-rest and one-versus-one (Dietterich and Bakiri, 1995). The second approach is to generalize the binary SVM to a simultaneous multicategory formulation which deals with all classes at once (Vapnik, 1998; Weston and Watkins, 1999; Liu and Shen, 2006; Wang and Shen, 2007). Since the proposed binary weighted SVM can be directly applied for the first approach to solve multicategory problems, we will focus on the simultaneous multicategory formulation in this section. In particular, we use the MSVM proposed by Liu and Shen (2006) to illustrate the extensions of our proposed weighted SVM techniques for multicategory problems to gain robustness.

To handle multicategory classification, Liu and Shen (2006) proposed the generalized functional margin $\min \mathbf{g}(\mathbf{f}(\mathbf{x}_i), y_i)$, where $\mathbf{g}(\mathbf{f}(\mathbf{x}), y) = \{f_y(\mathbf{x}) - f_j(\mathbf{x}), j \neq y\}$. Using the generalized functional margin, the 0–1 loss can be represented as $I(\min \mathbf{g}(\mathbf{f}(\mathbf{x}_i), y_i) \leq 0)$. Liu and Shen (2006) suggested to replace the indicator function of the 0–1 loss by the convex hinge loss function to yield the MSVM. If we use the weight w_i for the i -th observation, we get the following weighted MSVM

$$\min_f \frac{1}{2} \sum_{j=1}^k J(f_j) + C \sum_{i=1}^n w_i H_1(\min \mathbf{g}(\mathbf{f}(\mathbf{x}_i), y_i)). \quad (8)$$

We first consider the problem with given weights. For linear learning with $f_j(\mathbf{x}) = \beta_{0j} + \mathbf{x}^T \beta_j$, the dual problem of (8) is given by

$$\min_{\alpha} \frac{1}{2} \sum_{j=1}^k \left\| \sum_{i: y_i=j} \alpha_{ij} \mathbf{x}_i - \sum_{i: y_i \neq j} \alpha_{ij} \mathbf{x}_i \right\|^2 - \sum_{i=1}^n \sum_{j \neq y_i} \alpha_{ij} \quad (9)$$

$$\begin{aligned} \text{subject to } & \sum_{i: y_i=j} \alpha_{ij} - \sum_{i: y_i \neq j} \alpha_{ij} = 0 \\ & 0 \leq \sum_{j \neq y_i} \alpha_{ij} \leq C w_i \\ & \alpha_{ij} \geq 0, j \neq y_i; i=1, \dots, n. \end{aligned}$$

Note that the weight w_i shows up in the upper bound of $\sum_j y_i \alpha_{ij}$ as $C w_i$. The smaller the weight w_i is, the smaller of the corresponding solution of α_{ij} will be. As a result, the impact of the i point will be reduced. Denote the optimizer by $\hat{\alpha}_{ij}$. Estimates of the coefficients $\hat{\beta}_j$ are given by $\hat{\beta}_j = \sum_{i: y_i=j} \sum_{j' \neq y_i} \hat{\alpha}_{ij'} \mathbf{x}_i - \sum_{i: y_i \neq j} \hat{\alpha}_{ij} \mathbf{x}_i$ for $j = 1, 2, \dots, k$. Once $\hat{\beta}_j$'s are determined, we denote $\tilde{f}_j(\mathbf{x}_i) = \mathbf{x}_i^T \hat{\beta}_j$ and estimate the intercepts by solving the following linear programming problem

$$\min_{\beta_{0j}} \sum_{i=1}^n w_i \eta_i$$

$$\begin{aligned} \text{subject to } & \eta_i \geq 0, i=1, 2, \dots, n, \\ & \eta_i \geq 1 - (\tilde{f}_{y_i}(\mathbf{x}_i) + \beta_{0y_i}) + \tilde{f}_j(\mathbf{x}_i) + \beta_{0j}, i=1, 2, \dots, n; j \neq y_i, \\ & \sum_{j=1}^k \beta_{0j} = 0 \end{aligned}$$

Denote the optimizer by $\hat{\beta}_{0j}$. The estimated function is given by $\hat{f}_j(\mathbf{x}) = \hat{\beta}_{0j} + \mathbf{x}^T \hat{\beta}_j$. The final classification rule is then given by $\arg\max_{j=1,2,\dots,k} \hat{f}_j(\mathbf{x})$.

When nonlinear learning is considered with kernel $K(\cdot, \cdot)$, the optimization is similar to the linear learning case with the objective function of the dual problem (9) replaced by the following objective function

$$\sum_{i=1}^n \sum_{j' \neq y_i} \alpha_{ij'} - \frac{1}{2} \sum_{j=1}^k \left\langle \sum_{i: y_i=j} \alpha_{ij} \mathbf{K}_i - \sum_{i: y_i \neq j} \alpha_{ij} \mathbf{K}_i, \sum_{i: y_i=j} \alpha_{ij} \mathbf{e}_i - \sum_{i: y_i \neq j} \alpha_{ij} \mathbf{e}_i \right\rangle, \quad (10)$$

where $\mathbf{e}_i = (0, \dots, 0, 1, 0, \dots, 0)^T$ denotes the standard basis of the n -dimension space and $\mathbf{K}_i = (K(\mathbf{x}_1, \mathbf{x}_i), K(\mathbf{x}_2, \mathbf{x}_i), \dots, K(\mathbf{x}_n, \mathbf{x}_i))^T$.

Next we consider the choice of weights for the weighted MSVM to gain robustness. Note that the standard MSVM uses weight $w_i = 1$. To extend the OWSVM we proposed for the binary case, we consider $w_i = 1/(1 + |\min \mathbf{g}(\hat{f}_{\text{SVM}}(\mathbf{x}_i), y_i)|)$ for the i -th point, where $\hat{f}_{\text{SVM}}$ is the solution from the standard MSVM. For the IWSVM, similar to the binary case, its extension to the multicategory case is also closely related to the robust truncated-hinge-loss MSVM (RMSVM). In particular, the RMSVM uses the truncated hinge loss $H_{T_s}(\cdot)$ for some $s \in [-1/(k-1), 0]$. Similar to the binary case, we extend the IWSVM algorithm using weighted learning to implement the RMSVM by iteratively updating weight w_i and solving (8) with updated weights. The algorithm is summarized as follows:

Step 1 (Initialization): Initialize by solving the weighted MSVM (8) with weights $w_i = 1$ for $i = 1, 2, \dots, n$ and denoting the estimated functions by $\hat{f}_j^{(0)}(\cdot)$.

Step 2 (Iteration): At the t th iteration, define weight as $w_i^{(t)} = 1$ if $s = \min \{\hat{f}_{y_i}(\mathbf{x}_i) - \hat{f}_j(\mathbf{x}_i)\} \geq 1$ and 0 otherwise for $i = 1, 2, \dots, n$. Solve the weighted MSVM (8) with the updated weights $w_i^{(t)}$ and denote the updated estimated functions by $\hat{f}_j^{(t)}(\cdot)$.

Step 3 (Convergence): Repeat Step 2 until convergence.

Similarly to the binary case, we can establish close relationship between the DCA for the RMSVM and the proposed IWSVM as follows.

Corollary 1. *The DCA and IWSVM algorithms are equivalent in terms of fixed points.*

5 Numerical Examples

In this session, we use both simulated examples in Sections 5.1 and 5.2 and a real data example in Section 5.3 to demonstrate the performance of the proposed OWSVM and IWSVM. Throughout our simulation, the training data set is generated with a sample of size 100. Independent tuning and test data sets are generated in the same way as the training data with sizes 100 and 500000, respectively, to tune the regularization parameter and evaluate performance. The regularization parameter C is tuned via a grid search using the tuning data set. We compare the performance of the OWSVM and IWSVM with the standard SVM, RSVM (with loss function $T_{-1/(k-1)}(\cdot)$, where k denotes the number of classes), and ψ -learning (with loss function $T_0(\cdot)$).

5.1 Linear Examples

We consider linear learning with a binary response or a three-class response. The two-dimension predictor $\mathbf{X} = (X_1, X_2)^T$ is generated from the uniform distribution over the unit disc $\{(x_1, x_2): x_1^2 + x_2^2 \leq 1\}$.

For binary learning, conditional on $\mathbf{X} = \mathbf{x} = (x_1, x_2)^T$, set $\tilde{y} = \text{sign}(x_2)$. To show robustness, a random flipping step is followed to generate the binary response. Denote the percentage of flipping by *perc*. The binary response y is set as $\tilde{y}$ with probability $1 - \text{perc}$ and $-\tilde{y}$ with probability *perc*. The points whose labels are flipped are potential outliers.

For the three-class case, conditional on $\mathbf{X} = \mathbf{x} = (x_1, x_2)^T$, denote θ to be the radian phase angle measured counterclockwise from the ray from $(0, 0)$ to $(1, 0)$ to another ray from $(0, 0)$

to (x_1, x_2) . In this case θ is valued inside interval $[0, 2\pi)$. Set $\tilde{y} = \lceil \frac{3\theta}{2\pi} \rceil$ where $\lceil \cdot \rceil$ denotes the ceiling operator. The response y is set as $\tilde{y}$ with probability $1 - \text{perc}$ and as one of the two values in $\{1, 2, 3\} \setminus \{\tilde{y}\}$, each with probability $\text{perc}/2$, where $\setminus$ denotes the set minus operator.

Four different flipping percentages are considered with $\text{perc} = 5\%, 10\%, 15\%$, and 20% . Test errors over 100 repetitions are summarized using boxplots in Figures 3 and 4, respectively. From the plots, we can see that both OWSVM and IWSVM work better than the SVM. Between the OWSVM and IWSVM, the IWSVM works better than that of the OWSVM. Furthermore, the IWSVM and DCA of the RSVM behave very similarly and they are slightly better than the performance of ψ -learning. Note that the OWSVM involves a full iteration till convergence while the OWSVM only uses one-step weighting. Thus, there is a trade-off between accuracy and computational complexity between the OWSVM and IWSVM. To compare the computational time involved for different methods, we report the computation time using boxplots in Figures 5 and 6. Indeed, we can see from the boxplots that the IWSVM takes longer to compute than the OWSVM.

So far the dimension of our examples is only two. Next we consider a higher dimensional example with 10 predictors. The predictors $\mathbf{X} = (X_1, X_2, \dots, X_{10})^T$ is uniformly distributed over the set $\{(x_1, x_2, \dots, x_{10}) : \sum_{j=1}^{10} |x_j| \leq 1\}$. The response is binary and generated in two steps similarly as the previous example. Conditional $\mathbf{X} = \mathbf{x} = (x_1, x_2, \dots, x_{10})^T$, we first set $\tilde{y} = 1$ if $\sum_{j=1}^{10} x_j > 0$ and 2 otherwise. Then a random flipping with $\text{perc} = 5\%$ is applied to $\tilde{y}$ to get the binary response y . The simulation results in terms of classification accuracy and computational time are summarized in Figure 7. A similar pattern is observed.

5.2 Nonlinear example

For the nonlinear learning, we consider a three-class example. The data generation scheme is similar to that of the linear examples. First, the two-dimensional predictor $\mathbf{X}$ is generated from the uniform distribution over the unit disc. Conditional on $\mathbf{X} = \mathbf{x}$, denote θ to be the radian phase angle measured counterclockwise from the ray from $(0, 0)$ to $(1, 0)$ to another ray from $(0, 0)$ to (x_1, x_2) . Set $\tilde{y} = 1$ if $\theta \in [0, \pi/3)$ or $[\pi, 4\pi/3)$; $\tilde{y} = 2$ if $\theta \in [\pi/3, \pi)$; $\tilde{y} = 3$ if $\theta \in [4\pi/3, 2\pi)$. The response y is set as $\tilde{y}$ with probability $1 - \text{perc}$ and as one of the two values in $\{1, 2, 3\} \setminus \{\tilde{y}\}$, each with probability $\text{perc}/2$.

To save space, we only consider two different flipping percentages $\text{perc} = 15\%$ and 20% . Test errors and computational time over 100 repetitions are reported in Figure 8. We can see that the improvement of the OWSVM and IWSVM over the SVM is less in terms of test errors compared to the linear examples. One potential reason is that the nonlinear kernel learning increases flexibility of the function. Consequently there are not many observations with functional margin $\min_{\mathbf{g}} (f(\mathbf{x}), y)$ being very small. Thus, the effect of weighting can be smaller than that of the linear learning case.

5.3 Wisconsin Breast Cancer Data

In this section, we apply the proposed methods to the Wisconsin Diagnostic Breast Cancer (WDBC) data. The goal of this breast cancer study is to use a digitized image of a fine needle aspirate of a breast mass to diagnose the corresponding breast cancer status. More details on the data are provided at the UCI Machine Learning Repository (<http://www.ics.uci.edu/~mllearn/MLRepository.html>). The WDBC dataset has 569 subjects. It has

a binary response of diagnosis as either malignant or benign. Furthermore, there are 30 real-valued input features available for each subject. We first preprocess the data by standardizing each input feature to have mean zero and standard deviation one. On each repetition, we randomly split the whole dataset into training, tuning, and test sets of sizes 150, 150, and 269, respectively. We report the average testing error over the test set across 100 random repetitions. For each repetition, to study robustness of different methods to outliers, we apply random flipping to the training set as in Wu and Liu (2007). We consider three different levels of flipping 0%, 5%, and 10%.

For the analysis, linear learning is applied on this dataset and the results are summarized in Table 1. From the table, we can see that when there is no data contamination, all five methods perform similarly. As the percentage of data contamination increases, the SVM performs much worse than other methods. In particular, the IWSVM delivers better performance than the OWSVM when there are outliers with more computational time. This is consistent with the simulation results.

6 Conclusion

In this article, we propose two different weighted extensions of the SVM, the OWSVM and IWSVM, to gain robustness for classification of noisy data. We show through numerical examples that both methods are effective in providing more accurate classification results than the standard SVM. Both binary and multiclass problems are considered. We also show the interesting connection of the IWSVM with the RSVM, which further justifies the robustness of the IWSVM. Numerical examples indicate that the IWSVM performs better than the OWSVM and costs more computational time. In general, we recommend the IWSVM if the computational time is not a concern. Although our focus in this paper is on the SVM, the idea can be directly generalized to other large margin classifiers. Further development and comparisons are necessary. Moreover, it will be interesting to extend the methods to other classification settings such as longitudinal or functional data as discussed in Müller (2005).

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgments

The authors would like to thank Professor Richard A. Levine, the associate editor, and two referees for their constructive comments and suggestions that led to significant improvement of the article. The authors are partially supported by NSF Grants DMS-0747575 (Liu), DMS-0905561 (Wu), and DMS-1055210 (Wu), NIH/NCI Grants R01 CA-149569 (Liu and Wu) and P01 CA142538 (Liu).

References

- Cristianini, N.; Shawe-Taylor, J. An introduction to support vector machines and other kernel-based learning methods. Cambridge University Press; 2000.
- Dietterich TG, Bakiri G. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*. 1995; 2:263–286.
- Hastie, T.; Tibshirani, R.; Friedman, J. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. New York: Springer-Verlag; 2001.
- Lin X, Wahba G, Xiang D, Gao F, Klein R, Klein B. Smoothing Spline ANOVA Models for Large Data Sets With Bernoulli Observations and the Randomized GACV. *The Annals of Statistics*. 2000; 28:1570–1600.

- Liu Y, Shen X. Multicategory ψ -learning. *Journal of the American Statistical Association*. 2006; 101:500–509.
- Marron JS, Todd M, Ahn J. Distance weighted discrimination. *Journal of the American Statistical Association*. 2007; 102:1267–1271.
- Müller H-G. Functional Modelling and Classification of Longitudinal Data. *Scandinavian Journal of Statistics*. 2005; 32:223–240.
- Schölkopf, B.; Smola, AJ. *Learning with kernels*. MIT Press; 2002.
- Shen X, Tseng GC, Zhang X, Wong WH. On ψ -learning. *Journal of the American Statistical Association*. 2003; 98:724–734.
- Vapnik, V. *The Nature of Statistical Learning Theory*. New York: Springer; 1995.
- Vapnik, V. *Statistical Learning Theory*. Wiley; 1998.
- Wahba, G. Support vector machines, reproducing kernel Hilbert spaces, and randomized GACV. In: Schölkopf, B.; Burges, CJC.; Smola, AJ., editors. *Advances in Kernel Methods: Support Vector Learning*. MIT Press; 1998. p. 125-143.
- Wang L, Shen X. On L_1 -Norm Multiclass Support Vector Machines: Methodology and Theory. *Journal of the American Statistical Association*. 2007; 102:583–594.
- Weston, J.; Watkins, C. Support vector machines for multi-class pattern recognition. In: Verleysen; Bruges, M., editors. *Proceedings of the 7th European Symposium on Artificial Neural Networks (ESANN-99)*. Belgium: 1999. p. 219-224.
- Wu Y, Liu Y. Robust truncated-hinge-loss support vector machines. *Journal of the American Statistical Association*. 2007; 102:974–983.
- Zhu J, Hastie T. Kernel Logistic Regression and the Import Vector Machine. *Journal of Computational and Graphical Statistics*. 2005; 14:185–205.
- Zou H, Zhu J, Hastie T. New Multicategory Boosting Algorithms Based on Multicategory Fisher-Consistent Losses. *Annals of Applied Statistics*. 2008; 2:1290–13061.

Figure 1.
Plot of the hinge loss H_1 , the weight functions for the OWSVM, and the product function $w(u)H_1(u)$ for the binary case with $k = 2$.

Figure 2.
Plot of the weight functions for the OWSVM and IWSVM for the binary case with $k = 2$.

Figure 3. Classification accuracy comparison among the SVM, OWSVM, IWSVM, RSVM, and ψ -learning (denoted by M1, M2, M3, M4, and M5, respectively) for the linear example of Section 5.1 with $K = 2$.

Figure 4. Classification accuracy comparison among the SVM, OWSVM, IWSVM, RSVM, and ψ -learning (denoted by M1, M2, M3, M4, and M5, respectively) for the linear example of Section 5.1 with $K = 3$.

Figure 5. Computational time comparison among the SVM, OWSVM, IWSVM, RSVM, and ψ -learning (denoted by M1, M2, M3, M4, and M5, respectively) for the linear example of Section 5.1 with $K = 2$.

Figure 6. Computational time comparison among the SVM, OWSVM, IWSVM, RSVM, and ψ -learning (denoted by M1, M2, M3, M4, and M5, respectively) for the linear example of Section 5.1 with $K = 3$.

Figure 7. Classification accuracy (left) and computational time (right) comparison among the SVM, OWSVM, IWSVM, RSVM, and ψ -learning (denoted by M1, M2, M3, M4, and M5, respectively) for the linear example of Section 5.1 with $d = 10$.

Figure 8. Classification accuracy (top two panels) and computational time (bottom two panels) comparisons among the SVM, OWSVM, IWSVM, RSVM, and ψ -learning (denoted by M1, M2, M3, M4, and M5, respectively) for the nonlinear example of Section 5.2 with $K = 3$.

Table 1

Classification accuracy of the SVM, OWSVM, IWSVM, RSVM, and ψ -learning on the WDBC data. The average test errors over 100 replications and the corresponding standard errors in parenthesis are reported in the table.

	0%	5%	10%
SVM	0.0382(0.0013)	0.0438(0.0014)	0.0521(0.0017)
OWSVM	0.0377(0.0013)	0.0401(0.0016)	0.0444(0.0018)
IWSVM	0.0381(0.0014)	0.0377(0.0015)	0.0413(0.0016)
RSVM	0.0381(0.0014)	0.0377(0.0015)	0.0413(0.0016)
Psi	0.0380(0.0013)	0.0379(0.0015)	0.0406(0.0016)