Robust Model-free Multiclass Probability Estimation

Yichao Wu, Hao Helen Zhang, and Yufeng Liu *

North Carolina State University and University of North Carolina

Abstract

Classical statistical approaches for multiclass probability estimation are typically based on regression techniques such as multiple logistic regression, or density estimation approaches such as linear discriminant analysis (LDA) and quadratic discriminant analysis (QDA). These methods often make certain assumptions on the form of probability functions or on the underlying distributions of subclasses. In this paper, we develop a model-free procedure to estimate multiclass probabilities based on large-margin classifiers. In particular, the new estimation scheme is employed by solving a series of weighted large-margin classifiers and then systematically extracting the probability information from these multiple classification rules. A main advantage of the proposed probability estimation technique is that it does not impose any strong parametric assumption on the underlying distribution and can be applied for a wide range of large-margin classification methods. A general computational algorithm is developed for class probability estimation. Furthermore, we establish asymptotic consistency of the probability estimates. Both simulated and real data examples are presented to illustrate competitive performance of the new approach and compare it with several other existing methods.

*Yichao Wu is Assistant Professor (E-mail: wu@stat.ncsu.edu); Hao Helen Zhang is Associate Professor (E-mail: hzhang2@stat.ncsu.edu), Department of Statistics, North Carolina State University, Raleigh NC 27695. Yufeng Liu is Associate Professor, Department of Statistics and Operations Research, Carolina Center for Genome Sciences University of North Carolina, Chapel Hill, NC 27599-3260 (E-mail: yfliu@email.unc.edu). The authors thank the Editor, the Associate Editor, and two referees for their helpful suggestions that lead to significant improvement of the paper. The authors are supported in part by NSF grants DMS-0905561 (Wu), DMS-0645293 (Zhang), DMS-0747575 (Liu), and DMS-0606577 (Liu), and NIH/NCI grant R01-CA-085848 (Zhang).

Key Words and Phrases: Fisher consistency, hard classification, multicategory classification, probability estimation, soft classification, SVM.

1 Introduction

Multiclass probability estimation is an important problem in statistics and data mining. Suppose we are given a sample $\{(\boldsymbol{x}_i, y_i), i = 1, 2, \cdots, n\}$ consisting of *i.i.d.* observations from some unknown probability distribution $P(\boldsymbol{X}, Y)$, where $\boldsymbol{x}_i \in \mathcal{S} \subset \Re^d$ denotes the input vector, $y_i \in \{1, 2, \cdots, K\}$ denotes the label, n is the sample size, d is the dimensionality of the input space, and K denotes the number of classes. The main goal is to estimate the conditional probabilities $p_k(\boldsymbol{x}) = P(Y = k|\boldsymbol{X} = \boldsymbol{x}), k = 1, \cdots, K$. This problem is also known as soft classification, since the estimated p_k's can be used to determine the classification boundary among K classes and to predict class labels for future samples collected from the same population.

Traditionally, the probability estimation problem is commonly tackled by regression techniques such as multiple logistic regression, or the density estimation approaches such as linear discriminant analysis (LDA) and quadratic discriminant analysis (QDA). Agresti and Coull (1998) gives a thorough review on these methods. These methods often make certain model assumptions on the function forms of p_k's (or their transformations) or on the underlying distributions of subclasses. For example, multiple logistic regression assumes that the logarithms of the odd ratios are linear in $\boldsymbol{x}$,

$$\log \frac{P(Y = k|\boldsymbol{X} = \boldsymbol{x})}{P(Y = 1|\boldsymbol{X} = \boldsymbol{x})} = \beta_{k0} + \boldsymbol{x}^T \boldsymbol{\beta}_k, \quad \forall k \geq 2,$$

where class 1 is chosen as the baseline class. On the other hand, both LDA and QDA assume that the covariates $\boldsymbol{X}$ associated with each subclass follow a multivariate Gaussian distribution and construct the probability estimates as

$$P(Y = k|\boldsymbol{X} = \boldsymbol{x}) = \frac{\phi(\boldsymbol{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\alpha_k}{\sum_{j=1}^{K} \phi(\boldsymbol{x}; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)\alpha_j}, \quad k = 1, \cdots, K,$$

where $\phi(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ is the density function of the multivariate Gaussian distribution associated with the mean $\boldsymbol{\mu}$ and the covariance $\boldsymbol{\Sigma}$, and $\alpha_k = P(Y = k)$ is known as the prior probability of class k. These methods are widely used in practice. In many real applications, however, it

is difficult to justify the assumption of the linear effects of covariates in the multiple logistic regression. Moreover, it is often difficult to validate the Gaussian assumption for multivariate data. If the distribution is very skewed, some proper transformation is needed to make data approximately Gaussian, which can be nontrivial for multivariate data. These issues become even more challenging for high dimensional data.

In this paper, we propose a new class of model-free methods for estimating multiclass probabilities. The new method does not make any assumption on the forms of p_k's or the distribution for each subclass. Different from the traditional methods, we tackle the soft classification problem by solving a series of hard classification problems and combining these decision rules to construct the probability estimates. The main difference between soft classification and hard classification is their estimation target, with the former directly estimating $p_k(\boldsymbol{x})$'s and the latter estimating $\arg\max_{k=1,\cdots,K} p_k(\boldsymbol{x})$. For many complicated problems, estimation of the classification rule $\arg\max_{k=1,\cdots,K} p_k(\boldsymbol{x})$ may be a relatively easier task than estimating the probability functions. Many successful large-margin classifiers such as the support vector machine (SVM) can estimate $\arg\max_{k=1,\cdots,K} p_k(\boldsymbol{x})$ with high accuracy without estimating $p_k(\boldsymbol{x})$'s at all. This motivates us to take advantage of good classification performance of hard classifiers and try to extract the probability information contained in them.

Wang, Shen and Liu (2008) recently explored class probability estimation for binary large-margin classifiers. In particular, they made use of the property that the theoretical minimizer of a consistent weighted binary large-margin loss function is $\text{sign}[p_1(\boldsymbol{x}) - \pi]$, where $\pi \in (0, 1)$. Although a particular weighted binary large-margin classifier only estimates whether $p_1(\boldsymbol{x})$ is larger than π or not, one can obtain a good estimate of $p_1(\boldsymbol{x})$ if a sequence of weighted classifiers are calculated for many different π's. As shown in Wang, Shen and Liu (2008), this method indeed works well for binary class probability estimation. However, the simultaneous generalization from $K = 2$ to $K \geq 3$ is nontrivial and largely unknown due to the increased level of problem complexity. Wu, Lin and Weng (2004) proposed a pairwise coupling method for multiclass probability estimation by solving many binary problems. In this paper, we develop a new multiclass probability estimation scheme by utilizing the proposed concept of *border weight* for large-margin classifiers. As a result, the $K - 1$ dimensional probability estimation reduces to the search of

border weight. We propose two estimation schemes for probability estimation, the direct scheme and the indirect scheme. Furthermore, we focus on the truncated hinge loss (Wu and Liu, 2007) for demonstration of our proposed probability estimation technique. The technique is, however, applicable to other large-margin classifiers as well.

The rest of our paper is structured as follows. Section 2 presents the idea of weighted classification and its Fisher consistency properties. Section 3 introduces the main methodology, along with two estimation schemes and the theoretical properties of the resulting probability estimator. Section 4 discusses the computational algorithm and tuning method. Section 5 and Section 6 contain numerous simulated and real examples to illustrate the numerical performance of the new approach, which is followed by the concluding section. The appendix collects proofs for the theoretical results as well as the derivation of our algorithm.

2 Weighted Classification and Fisher Consistency

In this section, we give a brief review on an important class of hard classifiers, Support Vector Machines (SVMs, Cortes and Vapnik, 1995; Vapnik, 1998). We start with the simple binary classification problems and then discuss the multiclass extensions. We will in particular discuss the extension of SVMs by minimizing a weighted loss function.

2.1 Weighted Binary Classification

When $K = 2$, the class label y is often coded as $\{-1, +1\}$ for notational convenience. The binary SVM classifier can be fit in the following regularization framework

$$\min_{f \in \mathcal{F}} n^{-1} \sum_{i=1}^{n} H_1(y_i f(\boldsymbol{x}_i)) + \lambda J(f), \tag{1}$$

where the function $H_1(z) = (1-z)_+ \equiv \max\{1-z, 0\}$ is the so-called hinge loss, $J(f)$ is a penalty term for model complexity, and $\mathcal{F}$ is some functional space. Let $p_1(\boldsymbol{x}) = P(Y = 1 | \boldsymbol{X} = \boldsymbol{x})$. Lin (2002) showed that the SVM solution $\hat{f}$ to (1) targets directly at sign $\left[p_1(\boldsymbol{x}) - \frac{1}{2}\right]$. Therefore, sign $\left[\hat{f}(\boldsymbol{x})\right]$ approximates the Bayes classification rule without estimating $p_1(\boldsymbol{x})$.

Since the SVM has shown good classification accuracy in many applications, a natural question to ask is whether it is possible to extract any information about $p_1(\boldsymbol{x})$ from the SVM solution.

Recently, Wang et al. (2008) proposed to train a series of binary SVMs by minimizing a weighted loss function, and then construct $\hat{p}_1(\boldsymbol{x})$ by combining multiple SVM classification rules. In particular, by assigning a weight π to all the samples from class -1 and assigning $1-\pi$ to all the samples from class $+1$, one can solve the regularization problem based on the weighted hinge loss

$$\min_{f\in\mathcal{F}} n^{-1}\left[(1-\pi)\sum_{y_i=1} H_1\{y_i f(\boldsymbol{x}_i)\} + \pi\sum_{y_i=-1} H_1\{y_i f(\boldsymbol{x}_i)\}\right] + \lambda J(f), \tag{2}$$

where $0 \le \pi \le 1$. Wang et al. (2008) showed that the minimizer to (2) is a consistent estimate of $\text{sign}\,[p_1(\boldsymbol{x})-\pi]$. Therefore, one can repeatedly solve (2) using different π values, say, $0 = \pi_1 < \cdots < \pi_{m+1} = 1$ and search $\hat{j}$ such that $\pi_{\hat{j}}$ and $\pi_{\hat{j}+1}$ satisfy $\text{sign}\left[p_1(\boldsymbol{x}) - \pi_{\hat{j}}\right] \neq \text{sign}\left[p_1(\boldsymbol{x}) - \pi_{\hat{j}+1}\right]$. The probability estimate can be estimated as $\hat{p}_1(\boldsymbol{x}) = \frac{1}{2}(\pi_{\hat{j}} + \pi_{\hat{j}+1})$. More technical details can be found in their paper.

2.2 Weighted Multiclass Classification

Now consider the multiclass problems with $K \ge 2$. In this setup, we code y as $\{1, 2, \cdots, K\}$. A classifier seeks the function vector $\boldsymbol{f} = (f_1, f_2, \cdots, f_K)$, where f_k is a map from the input domain $\mathcal{S}$ to $\Re$ (the set of all real numbers) representing the class k; $k = 1, \cdots, K$. To ensure uniqueness of the solution, a sum-to-zero constraint $\sum_{k=1}^{K} f_k = 0$ is usually employed. For any new input vector $\boldsymbol{x}$, its label is estimated via a decision rule $\hat{y} = \underset{k=1,2,\cdots,K}{\text{argmax}}\, f_k(\boldsymbol{x})$. Clearly, the argmax rule is equivalent to the sign function used in the binary case.

Various loss functions have been proposed to extend the binary SVM to multiclass problems, such as Weston and Watkins (1999), Lee et al. (2004), and Liu (2007). Here we focus on the notion of the $0-1$ loss. Note that a point $(\boldsymbol{x}, y)$ is misclassified by $\boldsymbol{f}$ if $y \neq \text{argmax}_k f_k(\boldsymbol{x})$, that is, if $\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}), y) \le 0$, where

$$\boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}), y) = \{f_y(\boldsymbol{x}) - f_k(\boldsymbol{x}), k \neq y\}.$$

The quantity $\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}), y)$ is known as the generalized functional margin and can be reduced to $yf(\boldsymbol{x})$ in the binary case with $y \in \{\pm 1\}$ (Liu and Shen, 2006). With the generalized functional margin, the $0-1$ loss can be expressed as $I(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}), y) \le 0)$. As in the binary case, one can replace the indicator function in the $0-1$ loss by some other loss ℓ. Typically, in order to

assure that a misclassified sample induces a larger loss than a correctly classified sample, the loss function ℓ is non-increasing and satisfies that $\ell'(0) < 0$. Once the loss $\ell(\cdot)$ is given, the decision vector can be obtained by solving the following regularization problem

$$\min_{\boldsymbol{f}} n^{-1} \sum_{i=1}^{n} \ell(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}_i), y_i)) + \lambda \sum_{k=1}^{K} J(f_k) \tag{3}$$
$$\text{subject to} \quad \sum_{k=1}^{K} f_k(\boldsymbol{x}) = 0.$$

Motivated by Wang et al. (2008), we propose a new approach to estimate the class probabilities by solving a series of weighted multiclass problems and then combining multiple classification rules. In the paper, we focus on the class of losses based on the functional margin $\ell(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{X}), Y))$, as they provide a natural extension from two-class to multiclass problems. For the weighted learning, we assign a weight $0 \leq \pi_k \leq 1$ to samples from class k, $k = 1, \cdots, K$, where $\pi_1 + \cdots + \pi_K = 1$ to assure identifiability. Define the unit K-cube hyperplane as

$$A_K = \{(\pi_1, \cdots, \pi_K) : \sum_{k=1}^{K} \pi_k = 1, \quad \pi_k \geq 0, \ k = 1, 2, \cdots, K\}.$$

For any given $\boldsymbol{\pi} \in A_K$, we can train a weighted hard classifier by minimizing the objective function using a weighted loss function

$$\min_{\boldsymbol{f}} n^{-1} \sum_{i=1}^{n} \pi_{y_i} \ell(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}_i), y_i)) + \lambda \sum_{k=1}^{K} J(f_k) \tag{4}$$
$$\text{subject to} \quad \sum_{k=1}^{K} f_k(\boldsymbol{x}) = 0.$$

Compared with the binary case, extracting the probability information from the constructed classifiers becomes much more challenging for $K > 2$. In particular, instead of estimating only one probability function as in $K = 2$, we need to estimate multiple functions $p_1(\boldsymbol{x}), \cdots, p_{K-1}(\boldsymbol{x})$ when $K > 2$. As a result, a substantially different formulation from the binary case is required for multiclass probability estimation.

In the binary case, the standard SVM is shown to be Fisher-consistent for estimating the Bayes classification rule $\text{sign}(p_1(\boldsymbol{x}) - \frac{1}{2})$. In order to estimate conditional class probabilities,

Wang et al. (2008)'s method requires that the weighted SVM (2) is Fisher-consistent for estimating weighted Bayes classification rule $\text{sign}(p_1(\boldsymbol{x}) - \pi)$. To proceed with the multicategory probability estimation, we need to extend the definition of weighted Fisher-consistency. In order to construct a good probability estimate from the classification rules, we require that the loss function l in (4) is consistent in the following sense.

Definition 1: A functional margin based loss ℓ is called weighted Fisher-consistent for the weighted classification problem if the minimizer $\boldsymbol{f}^*$ of $E[\pi_Y \ell(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{X}), Y))|\boldsymbol{X} = \boldsymbol{x}]$ satisfies

$$\underset{k=1,\cdots,K}{\text{argmax}}\, f_k^*(\boldsymbol{x}) = \underset{k=1,\cdots,K}{\text{argmax}}\, \pi_k p_k(\boldsymbol{x}), \quad \forall \boldsymbol{x}, \forall \boldsymbol{\pi} \in A_K.$$

In a standard multiclass classification problem, the misclassification costs are all equal, i.e., $C(Y, \boldsymbol{f}(\boldsymbol{X})) = I(Y \neq \boldsymbol{f}(\boldsymbol{X}))$, and the Bayes rule minimizing $E[C(Y, f(\boldsymbol{X}))]$ is $\text{argmax}_{k=1,2,\cdots,K}\, p_k(\boldsymbol{x})$. A loss ℓ is Fisher-consistent if the decision rule induced from $\boldsymbol{f}^* = \arg\min E[\ell(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{X}), Y))|\boldsymbol{X} = \boldsymbol{x}]$ is the same as the Bayes rule, i.e., $\text{argmax}_{k=1,\cdots,K}\, f_k^*(\boldsymbol{x}) = \text{argmax}_{k=1,\cdots,K}\, p_k(\boldsymbol{x})$ for all $\boldsymbol{x}$. For a weighted learning problem, the weighted loss $E[\pi_Y \ell(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{X}), Y))]$ implies that unequal costs $C_{\boldsymbol{\pi}}(Y, f(\boldsymbol{X})) = \pi_Y I(Y \neq f(\boldsymbol{X}))$ are used for incorrect decisions. It is straightforward to show that the Bayes rule minimizing $E[C_{\boldsymbol{\pi}}(Y, f(\boldsymbol{X})]$ is $\text{argmax}_{k=1,\cdots,K}\, \pi_k p_k(\boldsymbol{x})$. In this context, we say ℓ is weighted Fisher-consistent if $\arg\max_{k=1,\cdots,K} \boldsymbol{f}^*(\boldsymbol{x}) = \arg\max_{k=1,\cdots,K} \pi_k p_k(\boldsymbol{x})$ for all $\boldsymbol{\pi}$ and $\boldsymbol{x}$. This is also known as classification calibrated (Bartlett et al., 2006) and infinite-sample consistent (Zhang, 2004). Therefore, the weighted Fisher-consistency can be regarded as an equivalent formulation of Fisher-consistency for weighted classification problems.

It turns out that not all functional margin based loss $\ell(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}), y))$ satisfying $\ell'(0) < 0$ is weighted Fisher-consistent for multicategory problems, as shown in the next proposition.

Proposition 1. *Let $\ell(\cdot)$ be a non-increasing loss function satisfying $\ell'(0) < 0$. For any given positive weights $\boldsymbol{\pi} \in A_K$, the minimizer $\boldsymbol{f}^*$ of $E[\pi_Y \ell(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{X}), Y))|\boldsymbol{X} = \boldsymbol{x}]$ has the following properties:*

(a) If $\frac{\max_{k=1,\cdots,K} \pi_k p_k}{\sum_{k=1}^K \pi_k p_k} > 1/2$, then $\text{argmax}_k\, f_k^* = \text{argmax}_{k=1,\cdots,K}\, \pi_k p_k$.

(b) If $\ell(\cdot)$ is convex and $\frac{\max_{k=1,\cdots,K} \pi_k p_k}{\sum_{k=1}^K \pi_k p_k} \leq 1/2$, then $\boldsymbol{f}^* = \boldsymbol{0}$ is a minimizer.

Proposition 1 suggests that one sufficient condition for the weighted loss $\pi_y \ell(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}), y))$ to be weighted Fisher-consistent is $\frac{\max_k \pi_k p_k}{\sum_k \pi_k p_k} > 1/2$, *i.e.*, there exists a "dominating" class in the weighted sense. This condition is always satisfied for a binary problem except at the Bayes boundary $\{\boldsymbol{x} : \pi_1 p_1(\boldsymbol{x}) = \pi_2 p_2(\boldsymbol{x})\}$, but not for $K > 2$ as we require $\frac{\max_{k=1,\cdots,K} \pi_k p_k}{\sum_{k=1}^K \pi_k p_k} > 1/2$ to hold for all $\boldsymbol{\pi} \in A_K$. When $K > 2$ and $\frac{\max_{k=1,\cdots,K} \pi_k p_k}{\sum_{k=1}^K \pi_k p_k} \leq 1/2$, $\boldsymbol{f}^* = \boldsymbol{0}$ can be a minimizer and consequently $\operatorname{argmax}_{k=1,\cdots,K} f_k^*(\boldsymbol{x})$ is not uniquely determined. As a result, the weighted loss $\pi_y \ell(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}), y))$ is not weighted Fisher-consistent in such cases. By Theorem 1, the weighted hinge loss $\pi_y H_1(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}), y))$ is not weighted Fisher-consistent.

Interestingly, although the weighted loss $\pi_y \ell(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}), y))$ may not be weighted Fisher-consistent, the corresponding truncated version can be weighted Fisher-consistent. Specifically, for any $\ell(\cdot)$, we define its truncated loss at a location $s \leq 0$ by

$$\ell_{T_s}(\cdot) = \min(\ell(\cdot), \ell(s)).$$

The following theorem shows that the truncated loss ℓ_{T_s} is weighted Fisher-consistent.

Theorem 1. *Let $\ell(\cdot)$ be a non-increasing loss function satisfying $\ell'(0) < 0$. Then a sufficient condition for the weighted truncated loss $\pi_y \ell_{T_s}(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}), y))$ with $K > 2$ and $s \leq 0$ to be weighted Fisher-consistent for estimating $\operatorname{argmax}_j \pi_j p_j$ is that the truncation location s satisfies $\sup_{\{u: u \geq -s \geq 0\}} \frac{\ell(0) - \ell(u)}{\ell(s) - \ell(0)} \geq K - 1$. This condition is also necessary if $\ell(\cdot)$ is convex.*

Remark 1. As pointed out by one referee, the condition $\ell'(0) < 0$ requires differentiability of ℓ at 0 and thus excludes non-differentiable loss functions such as the ψ loss. In the following, we show how the condition can be relaxed for non-differentiable losses. Note that $\ell'(0)$ is used to assure

$$(\sum_{k \neq k_{\pi p}} \pi_k p_k(\boldsymbol{x}))(-K)\ell'(0) + \pi_{k_{\pi p}} p_{k_{\pi p}}(\boldsymbol{x}) K \ell'(0) < 0 \tag{5}$$

when $\pi_{k_{\pi p}} p_{k_{\pi p}}(\boldsymbol{x}) > \sum_{k \neq k_{\pi p}} \pi_k p_k(\boldsymbol{x})$. If $\ell'(0)$ does not exist, the term in (5) can be simply replaced by

$$(\sum_{k \neq k_{\pi p}} \pi_k p_k(\boldsymbol{x}))(-K)\ell'(0^-) + \pi_{k_{\pi p}} p_{k_{\pi p}}(\boldsymbol{x}) K \ell'(0^+) < 0, \tag{6}$$

where $\ell'(0^-)$ and $\ell'(0^+)$ denote the left and right derivatives, respectively. Therefore, the condition $\ell'(0) < 0$ can be relaxed as $\ell'(0^+) < \ell'(0^-) \leq 0$.

We now use two common loss examples to illustrate how to check the condition and find a proper truncating location s to assure the weighted Fisher-consistency of a truncated loss.

(a) The hinge loss, $\ell(u) = [1-u]_+$: In this case, $\sup_{\{u:u\geq -s\geq 0\}} \frac{\ell(0)-\ell(u)}{\ell(s)-\ell(0)} = \frac{1-0}{-s}$ and the condition becomes $s \in [-\frac{1}{K-1}, 0]$ by noting that $s \leq 0$.

(b) The logistic loss, $\ell(u) = \log(1+e^u)$: In this case, $\sup_{\{u:u\geq -s\geq 0\}} \frac{\ell(0)-\ell(u)}{\ell(s)-\ell(0)} = \frac{\log 2-0}{\log(1+e^{-s})-\log 2}$, which leads to condition $s \in [-\log(2^{K/(K-1)}-1), 0]$.

Both of these two loss functions satisfy $\ell'(0) < 0$. Although they are not weighted Fisher-consistent themselves, they can become weighted Fisher-consistent after truncating them at s (with s satisfying the above condition).

Proposition 1 and Theorem 1 are weighted extensions of the results of Wu and Liu (2007). Furthermore, we note that the truncating location s given in Theorem 1 depends on the class number K. The larger K is, the more truncation is needed to ensure Fisher consistency. This is due to the fact that the difficulty of no "dominating" class becomes more severe as K increases. The more truncation is, the closer of the truncated loss is to the $0-1$ loss. For the hinge loss $H_1(u)$, exponential loss e^{-u}, and logistic loss $\log(1+e^{-u})$, their truncated versions are guaranteed to be weighted Fisher-consistent for $s \in [-\frac{1}{K-1}, 0]$, $[\log(1-\frac{1}{K}), 0]$, and $[-\log(2^{K/(K-1)}-1), 0]$, respectively. Note that the ψ loss used in ψ-learning can be viewed as special examples of truncated loss functions (Shen et al., 2003; Liu and Shen, 2006). Theoretically different truncation may give different performance. Empirically, the numerical examples in Wu and Liu (2007) indicate that minimum truncation appears to work better for the unweighted case. In this paper we proceed with the minimum truncation required to achieve weighted Fisher-consistency. By minimal truncation, we mean truncation with the smallest s to make the corresponding truncated loss weighted Fisher-consistent: $s = -\frac{1}{K-1}$, $\log(1-\frac{1}{K})$, and $-\log(2^{K/(K-1)}-1)$ for the hinge loss, exponential loss, and logistic loss, respectively. In Figure 1 we plot these three truncated loss functions for $K = 3$ with the minimal truncation.

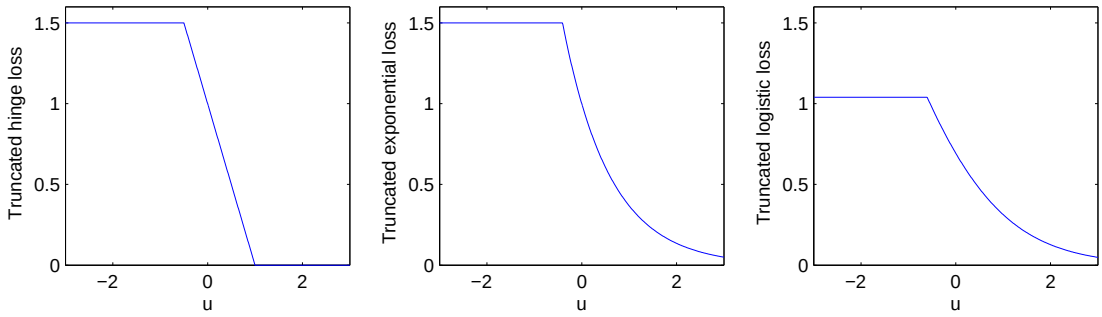

Figure 1: Plots of weighted Fisher-consistent truncated loss functions with minimal truncation for $K = 3$. The left, middle, and right panels correspond to the hinge, exponential, and logistic loss functions.

3 Methodology

In this section, we derive our methodology for multiclass probability estimation based on hard classifiers. In particular, we propose to train a series of weighted classifiers and use them to construct the probability estimates. For demonstration, we focus on the hinge and truncated hinge loss functions. However, our estimation schemes are applicable to general large-margin classifiers.

3.1 Direct Scheme for Probability Recovery

Define the truncated hinge loss as

$$H_{T_s}(u) = \min\left(H_1(s), H_1(u)\right),$$

where $s = -\frac{1}{K-1}$ corresponds to the minimum truncation required by Theorem 1 to guarantee H_{T_s} to be weighted Fisher-consistent. Denote $\widehat{\boldsymbol{f}}^{\boldsymbol{\pi}}$ as the solution of the $\boldsymbol{\pi}$-weighted truncated-hinge-loss SVM, obtained by solving the following optimization problem

$$\min_{\boldsymbol{f}} n^{-1} \sum_{i=1}^{n} \pi_{y_i} H_{T_s}(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}_i), y_i)) + \lambda \sum_{k=1}^{K} J(f_k) \tag{7}$$

$$\text{subject to} \quad \sum_{k=1}^{K} f_k(\boldsymbol{x}) = 0,$$

where $\boldsymbol{\pi} = (\pi_1, \cdots, \pi_K) \in A_K$. By Theorem 1, we have that $\text{argmax}_{k=1,\cdots,K} \hat{f}_k^{\boldsymbol{\pi}}$ converges to $\text{argmax}_{k=1,\cdots,K} \pi_k p_k$ as $n \to \infty$ and $\lambda \to 0$.

The following proposition gives a key result for estimating the probabilities for each $\boldsymbol{x} \in \mathcal{S}$.

Proposition 2. *For any given $\boldsymbol{x} \in \mathcal{S}$ satisfying $\min_k p_k(\boldsymbol{x}) > 0$, there exists a unique weight vector $\widetilde{\boldsymbol{\pi}}(\boldsymbol{x}) = (\pi_1(\boldsymbol{x}), \pi_2(\boldsymbol{x}), \cdots, \pi_K(\boldsymbol{x})) \in A_K$ such that*

$$\tilde{\pi}_1(\boldsymbol{x}) p_1(\boldsymbol{x}) = \tilde{\pi}_2(\boldsymbol{x}) p_2(\boldsymbol{x}) = \cdots = \tilde{\pi}_K(\boldsymbol{x}) p_K(\boldsymbol{x}).$$

Proposition 2 shows that for any $\boldsymbol{x} \in \mathcal{S}$ with $\min_k p_k(\boldsymbol{x}) > 0$, there is a unique weight vector so that the corresponding weighted probabilities $\tilde{\pi}_j(\boldsymbol{x}) p_j(\boldsymbol{x})$ are identical for all j. We call the point $\widetilde{\boldsymbol{\pi}}(\boldsymbol{x}) \in A_K$ as the *border weight* for $\boldsymbol{x}$, since the K weighted probabilities meet at this point.

Interestingly, the result in Proposition 2 can help us to estimate the conditional probabilities $p_k(\boldsymbol{x})$. In particular, for a given point $\boldsymbol{x}$, using the property of weighted Fisher-consistency, the corresponding Bayes rule is $\text{argmax}_{k=1,\cdots,K} \pi_k p_k$ for any $\boldsymbol{\pi} \in A_K$. Then one can vary the weight vector $\boldsymbol{\pi} \in A_K$ to search for the border weight. To illustrate this further, we consider a simple case of $K = 3$. In Figure 2, we plot the classification results of a particular point $\boldsymbol{x}$ for $K = 3$ when we change the weight vector. In this case, A_3 is an equilateral triangle with the three vertices being $(1, 0, 0)$, $(0, 1, 0)$, and $(0, 0, 1)$. Theoretically, for any $\boldsymbol{x}$, the weighted Bayes rule $\text{argmax}_k \pi_k p_k(\boldsymbol{x})$ assigns $\boldsymbol{x}$ to class k when π_k is close to one, and consequently the whole region A_3 can be divided into three subregions R_1, R_2, and R_3 with $R_k = \{\boldsymbol{\pi} \in A_3 : k = \text{argmax}_j \pi_j p_j(\boldsymbol{x})\}$ for $k = 1, 2, 3$. Since the vertex $(1, 0, 0)$ represents imposing the weight one to points from class 1 and the weight zero to points from the other classes, the region R_1 around $(1, 0, 0)$ corresponds to the set of $\boldsymbol{\pi}$ with prediction $\text{argmax}_k \pi_k p_k(\boldsymbol{x}) = 1$. The argument is similar for the other two vertices. Note that there is a special point in the center that borders all the three subregions. This is the border weight satisfying $\tilde{\pi}_1(\boldsymbol{x}) p_1(\boldsymbol{x}) = \tilde{\pi}_2(\boldsymbol{x}) p_2(\boldsymbol{x}) = \tilde{\pi}_3(\boldsymbol{x}) p_3(\boldsymbol{x})$.

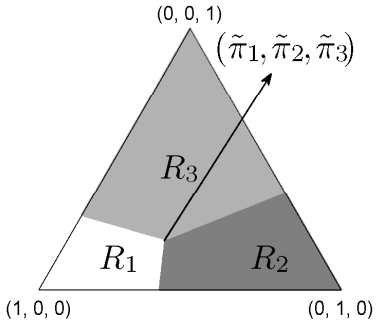

Figure 2: A plot of the weighted Bayes classification rule for all combinations of $\boldsymbol{\pi}$ for a certain point $\boldsymbol{x}$ when $K = 3$.

In order to estimate $p_j(\boldsymbol{x})$ it is enough to estimate $\tilde{\boldsymbol{\pi}}(\boldsymbol{x})$ because, once the estimate of $\tilde{\boldsymbol{\pi}}(\boldsymbol{x})$ is given, we can estimate $p_k(\boldsymbol{x})$ by the following proposition.

Proposition 3. *For any given $\boldsymbol{x} \in \mathcal{S}$, we assume that its associated border weight is estimated as $\widehat{\tilde{\boldsymbol{\pi}}}(\boldsymbol{x})$. Then its class probabilities can be estimated as*

$$\hat{p}_k(\boldsymbol{x}) = \frac{\hat{\tilde{\pi}}_k(\boldsymbol{x})^{-1}}{\hat{\tilde{\pi}}_1(\boldsymbol{x})^{-1} + \hat{\tilde{\pi}}_2(\boldsymbol{x})^{-1} + \cdots + \hat{\tilde{\pi}}_K(\boldsymbol{x})^{-1}}, \quad k = 1, \cdots, K.$$

Propositions 2 and 3 suggest that identifying the border weight for each $\boldsymbol{x}$ is a key step to estimate the conditional probabilities $p_k(\boldsymbol{x})$ for $k = 1, \cdots, K$. To that end, a general scheme is needed to search for $\widetilde{\boldsymbol{\pi}} \in A_K$ for each $\boldsymbol{x}$. Without loss of generality, we assume for the moment that the tuning parameter λ for (7) is properly chosen. In the following, we outline the probability estimation scheme for general cases.

Direct Scheme:

1. Define a fine grid of $\boldsymbol{\pi}$ within A_K. Let the grid size be d_π. Any grid point $\boldsymbol{\pi}$ takes the form $(m_1 d_\pi, m_2 d_\pi, \cdots, m_K d_\pi)$ with non-negative integers $m_1, m_2, \cdots, m_K$ satisfying $\sum_{k=1}^{K} m_k d_\pi = 1$.

2. Solve (7) over the above grid using the properly chosen tuning parameter λ.

3. Form all possible K-vertex polyhedrons of (side) length d_π using the available grid points. Here each K-vertex polyhedron corresponds to K adjacent grid points.

4. For any $\boldsymbol{x} \in \mathcal{S}$, identify the K-vertex polyhedron such that its K vertices all belong to K distinct classes. The average of the coordinates corresponding to these vertices is defined as the estimate of the border weight $\widetilde{\boldsymbol{\pi}}(\boldsymbol{x})$ for $\boldsymbol{x}$. The probability estimate can then be calculated using Proposition 3.

In the following, we demonstrate how the direct scheme works in the case of $K = 3$. To search for the border weight in Figure 2, we define a fine grid of $\boldsymbol{\pi}$ within the triangle as in Figure 3. Let the grid size be d_π with $1/d_\pi$ being an integer. To estimate probabilities at any point $\boldsymbol{x}$, we need to identify some $\boldsymbol{\pi}$ such that its three neighboring combinations are of the form

$$\{\boldsymbol{\pi}_1(\boldsymbol{x}) = (\pi_1, \pi_2, \pi_3), \boldsymbol{\pi}_2(\boldsymbol{x}) = (\pi_1 - d_\pi, \pi_2 + d_\pi, \pi_3), \boldsymbol{\pi}_3(\boldsymbol{x}) = (\pi_1 - d_\pi, \pi_2, \pi_3 + d_\pi)\} \quad (8)$$

which classify Y into three distinct classes as shown on the left panel of Figure 3.

3.1.1 Numerical Challenges in Implementing Direct Scheme

Now we provide some discussions on the numerical coherence of multiple decisions resulted from training multiple weighted classification problems. Let us start with two-class problems. Assume that π and $1 - \pi$ are the costs for the negative and positive classes, then it is known that the minimizer $\hat{f}_\pi(\boldsymbol{x})$ of Equation (2) gives a consistent estimator of $\text{sign}[P(Y = +1|\boldsymbol{X} = \boldsymbol{x}) - \pi]$. When an increasing sequence of weights $0 = \pi_1 < \pi_2 < \cdots < \pi_{m+1} = 1$ are used, we expect the decision sequence $\text{sign}[\hat{f}_{\pi_j}(\boldsymbol{x})]$ to be monotonically changed for a fixed $\boldsymbol{x}$ due to their consistency properties. Though this is true in theory (or when n goes to infinity), the monotonic property of $\text{sign}[\hat{f}_\pi]$ may not always hold in finite sampling situations, mainly due to numerical variations. In this case, the probability $p(x)$ can be estimated by taking the average of $\pi_* = \min\{\pi_j : \text{sign}[\hat{f}_{\pi_j}(\boldsymbol{x})] = -1\}$ and $\pi^* = \max\{\pi_j : \text{sign}[\hat{f}_{\pi_j}(\boldsymbol{x})] = 1\}$ (Wang et al., 2008).

For multiclass problems, the similar issue can occur even more frequently due to the increased complexity of the optimization problem. Take the three-class problem as an example. Each non-negative weight vector is $\boldsymbol{\pi} = (\pi_1, \pi_2, \pi_3)$ satisfying $\sum_{k=1}^{3} \pi_k = 1$. It is known that the minimization of Equation (5) satisfy that $\arg\max_k \hat{f}_k(\boldsymbol{x}) = \text{argmax}_k \, \pi_k p_k(\boldsymbol{x})$ asymptotically. This suggests that the weight vectors change (partially) monotonically, the decision rule $\arg\max_k \hat{f}_k(\boldsymbol{x})$ should satisfy some constraints. For example, for a given $\boldsymbol{x}$, if $\boldsymbol{\pi} = (\pi_1, \pi_2, \pi_3)$ satisfies $\pi_1 p_1(\boldsymbol{x}) >$

$\max(\pi_2 p_2(\boldsymbol{x}), \pi_3 p_3(\boldsymbol{x}))$, then we have $\arg\max_k \hat{f}_k(\boldsymbol{x}) = 1$ asymptotically. Now if the weight is changed to $\boldsymbol{\pi}' = (\pi_1', \pi_2', \pi_3') = (\pi_1 + d, \pi_2 - d, \pi_3)$, the inequality $\pi_1' p_1(\boldsymbol{x}) > \max(\pi_2' p_2(\boldsymbol{x}), \pi_3' p_3(\boldsymbol{x}))$ still holds, implying that $\arg\max_k \hat{f'}_k(\boldsymbol{x}) = 1$ asymptotically as well. Though this is true in theory, the relationship $\arg\max_k \hat{f}_k(\boldsymbol{x}) = \arg\max_k \hat{f'}_k(\boldsymbol{x})$ does not necessarily hold in finite-sample results. Therefore, in practice, with a finite sample, those three neighboring combinations do not always take the form (8). Other variations are possible. For example, it can also be of the form

$$\{\boldsymbol{\pi}_1(\boldsymbol{x}) = (\pi_1, \pi_2, \pi_3), \boldsymbol{\pi}_2(\boldsymbol{x}) = (\pi_1 - d_\pi, \pi_2, \pi_3 + d_\pi), \boldsymbol{\pi}_3(\boldsymbol{x}) = (\pi_1, \pi_2 - d_\pi, \pi_3 + d_\pi)\}$$

as shown on the right panel of Figure 3. This corresponds to the monotonicity violation in the binary case as discussed in the first paragraph of Section 2.2 of Wang et al. (2008). Our selection criterion is to select three neighboring combinations corresponding to three distinct classes. The average $(\boldsymbol{\pi}_1(\boldsymbol{x}) + \boldsymbol{\pi}_2(\boldsymbol{x}) + \boldsymbol{\pi}_3(\boldsymbol{x}))/3$ of these three neighboring combinations, denoted by $\hat{\boldsymbol{\pi}}(\boldsymbol{x}) = (\hat{\pi}_1(\boldsymbol{x}), \hat{\pi}_2(\boldsymbol{x}), \hat{\pi}_3(\boldsymbol{x}))$, serves as our estimate of the border weight. Using the estimated border weight $\hat{\boldsymbol{\pi}}(\boldsymbol{x})$, our estimate is given by

$$\hat{p}_k(\boldsymbol{x}) = \frac{\hat{\pi}_k(\boldsymbol{x})^{-1}}{\hat{\pi}_1(\boldsymbol{x})^{-1} + \hat{\pi}_2(\boldsymbol{x})^{-1} + \hat{\pi}_3(\boldsymbol{x})^{-1}}.$$

For the finite-sample case, it is possible to have more than one possibility of three neighboring combinations corresponding to three distinct classes. Each possibility leads to one estimated border weight. When this happens, averaging all the estimated border weights is necessary to proceed our probability estimation. The non-uniqueness of border weights encountered in practice adds some challenges in the implementation of the direct scheme. As the number of classes gets larger, this may become more severe. Furthermore, the border weights are identified through counting multiple decisions. The process is discrete and tends to be slow and unstable. These challenges motivate us to develop another scheme which is more continuous and of high stability.

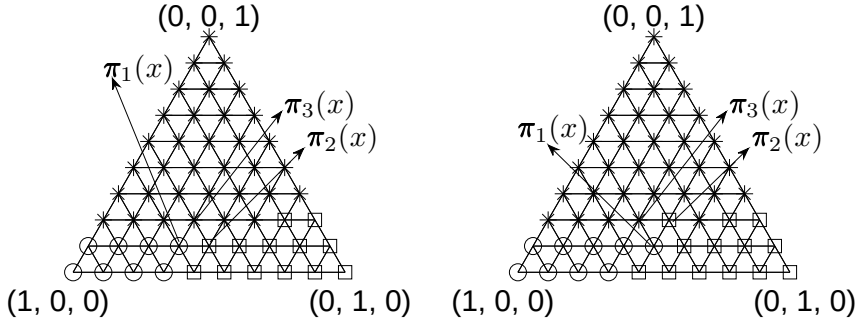

Figure 3: Left: Classification rule over a grid of $\boldsymbol{\pi}$ for a point $\boldsymbol{x}$ for $K = 3$, where the circle ○ denotes being classified as class 1, the square □ denotes being classified as class 2, and the asterisk ∗ denotes being classified as class 3. Right: Another possible configuration of neighboring three-class classifiers.

3.2 Indirect Scheme for Probability Estimation

In this section, we provide an alternative scheme to recover probabilities. Instead of directly targeting on the probabilities as the direct scheme does, the new scheme estimates some continuous functions of probabilities, which can be easier to estimate, and then inverts those functions to recover probabilities.

Note that the total volume (or area when $K = 3$) of A_K is given by

$$\int_0^1 d\pi_1 \int_0^{1-\pi_1} d\pi_2 \cdots \int_0^{1-\pi_1-\cdots-\pi_{K-2}} d\pi_{K-1}\sqrt{K} = \frac{\sqrt{K}}{(K-1)!}.$$

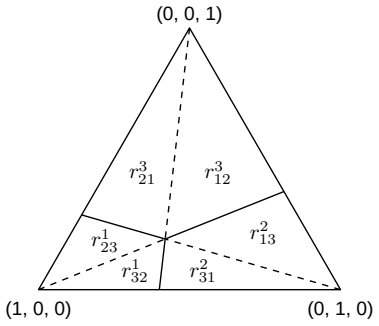

Figure 4: Demonstration of the partition of A_3.

The collection of π representing class k is given by $R_k = \{\boldsymbol{\pi} : \pi_k p_k \geq \pi_j p_j \text{ for } j \neq k\}$, which can be represented as

$$R_k = \cup\, r^k_{j_1 j_2 \cdots j_{K-1}},$$

where $r^k_{j_1 j_2 \cdots j_{K-1}} = \{\boldsymbol{\pi} : \pi_{j_1} p_{j_1} \leq \pi_{j_2} p_{j_2} \leq \cdots \leq \pi_{j_{K-1}} p_{j_{K-1}} \leq \pi_k p_k\}$ and the union is over all permutations $(j_1, j_2, \cdots, j_{K-1})$ of $\{1, 2, \cdots, K\} \setminus k$. When $K = 3$, Figure 4 demonstrates how A_3 is partitioned into different parts using the notation $r^k_{j_1 j_2 \cdots j_{K-1}}$ corresponding to Figure 2. For permutation $(j_1, j_2, \cdots, j_{K-1})$, the volume (or area) of $r^k_{j_1 j_2 \cdots j_{K-1}}$ is given by

$$\sqrt{K} \int_0^{B_{j_1}} d\pi_{j_1} \int_{p_{j_1}\pi_{j_1}/p_{j_2}}^{B_{j_2}} d\pi_{j_2} \cdots \int_{p_{j_{K-2}}\pi_{j_{K-2}}/p_{j_{K-1}}}^{B_{j_{K-1}}} d\pi_{j_{K-1}},$$

where $B_{j_i} = (1 - \sum_{m=1}^{k-1} \pi_{j_m})(1/p_{j_i})/((1/p_k) + \sum_{m=i}^{K-1}(1/p_{j_m})); i = 1, \ldots, K-1$ with the convention $\sum_{m=1}^{0} \pi_{j_m} = 0$. We can then sum over the area (volume) according to all permutations $(j_1, j_2, \cdots, j_{K-1})$ of $\{1, 2, \cdots, K\} \setminus k$ to give a formula for the volume (or area) of R_k.

Naturally the proportion of grid points of $\boldsymbol{\pi}$ leading to prediction of k, denoted by prop_k, estimates the volume (or area) ratio of $\frac{\text{Volume}(R_k)}{\text{Volume}(A_K)}$, which is a function of $p_1, p_2, \cdots, p_K$ and denoted by $h_k(p_1, p_2, \cdots, p_K)$. Then by solving the equation system of K equations:

$$\text{prop}_k = h_k(p_1, p_2, \cdots, p_K); \quad k = 1, 2, \cdots, K, \tag{9}$$

we can obtain the estimated probabilities. In particular when $K = 3$, area of A_3 is $\sqrt{3}/2$ and area of R_3 is $\frac{\sqrt{3}}{2} \frac{p_3^2(p_2p_3+p1p_3+2p_1p_2)}{(p_1+p_3)(p_2+p_3)(p_1p_2+p_1p_3+p_2p_3)}$. Consequently $h_3(p_1, p_2, p_3) = \frac{p_3^2(p_2p_3+p1p_3+2p_1p_2)}{(p_1+p_3)(p_2+p_3)(p_1p_2+p_1p_3+p_2p_3)}$.

The indirect scheme can be summarized as follows:

Indirect Scheme:

1-2. Same as those of the Direct Scheme.

3. For any $\boldsymbol{x} \in \mathcal{S}$, calculate the grid percentage prop_k for $k = 1, 2, \cdots, K$.

4. Solve the equation system (9) to recover the estimation of $(p_1(\boldsymbol{x}), p_2(\boldsymbol{x}), \cdots, p_K(\boldsymbol{x}))$.

In Section 5, we will illustrate the performance of both schemes. Our empirical results suggest that the indirect scheme is indeed faster and more accurate.

3.3 Theoretical Properties

The next theorem establishes the consistency of our class probability estimation.

Theorem 2. *For any non-increasing loss function $\ell(\cdot)$ with $\ell'(0) < 0$, if the truncation location s is chosen such that $\sup_{\{u:u\geq -s\geq 0\}} \frac{\ell(0)-\ell(u)}{\ell(s)-\ell(0)} \geq K-1$. When $\lambda \to 0$ and the grid size $d_\pi \to 0$ as $n \to \infty$, our estimate $\hat{p}_k(\boldsymbol{x})$ based on the truncated loss ℓ_{T_s} is asymptotically consistent, i.e., $\hat{p}_k(\boldsymbol{x}) \to p_k(\boldsymbol{x})$ for $k = 1, 2, \cdots, K$ as $n \to \infty$.*

The consistency result in Theorem 2 provides theoretical justification of our proposed method. It can be straightforward to extend the consistency to our indirect probability recovery scheme as Theorem 1 implies that prop_k is consistent for estimating $h_k(p_1, p_2, \cdots, p_K)$ and inversion will inherit the consistency. Although our probability estimation method is model-free, it converges to the true probability asymptotically. As shown in our simulation studies in Section 5, our method indeed provides competitive probability estimation compared to several other existing techniques.

4 Computation Algorithms

As shown on the right panel of Figure 5, the function $H_{T_s}(\cdot)$ is not convex, thus solving (7) involves a non-convex minimization problem. However, we note that $H_{T_s}(u)$ can be decomposed as the difference of two convex functions,

$$H_{T_s}(u) = \min\left(H_1(u), H_1(s)\right) = H_1(u) - H_s(u),$$

where $H_s(u) = (s-u)_+$. Figure 5 displays the three functions $H_1(u)$, $H_s(u)$, and $H_{T_s}(u)$.

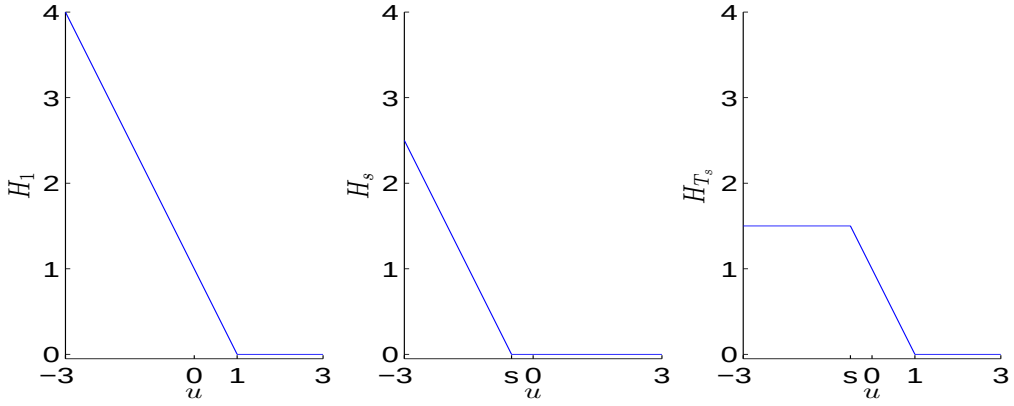

Figure 5: The left, middle, and right panels display functions $H_1(u)$, $H_s(u)$, and $H_{T_s}(u)$, respectively.

Using this property of the truncated hinge loss function, we apply the difference convex (d.c.) algorithm (An and Tao, 1997; Liu et al., 2005; Wu and Liu, 2007) to solve the nonconvex optimization problem of the weighted truncated-hinge-loss SVM. The d.c. algorithm solves the nonconvex minimization problem via minimizing a sequence of convex subproblems (see Algorithm 1). We derive the d.c. algorithm for linear learning in Section 4.1 and then generalize it to the case of nonlinear learning via kernel mapping in Section 4.2.

Algorithm 1: The Difference Convex Algorithm for minimizing $Q(\Theta) = Q_{vex}(\Theta) + Q_{cav}(\Theta)$

1. Initialize Θ_0.

2. Repeat $\Theta_{t+1} = \text{argmin}_\Theta(Q_{vex}(\Theta) + \langle Q'_{cav}(\Theta_t), \Theta - \Theta_t \rangle)$ until convergence of Θ_t.

4.1 Linear Learning

Let $f_k(\boldsymbol{x}) = \boldsymbol{w}_k^T \boldsymbol{x} + b_k$; $\boldsymbol{w}_k \in \Re^d$, $b_k \in \Re$, and $\boldsymbol{b} = (b_1, b_2, \cdots, b_K)^T \in \Re^K$, where $\boldsymbol{w}_k = (w_{1k}, w_{2k}, \cdots, w_{dk})^T$, and $\boldsymbol{W} = (\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_K)$. With $\ell = H_{T_s}$, (7) becomes

$$\min_{\boldsymbol{W},\boldsymbol{b}} \frac{1}{2} \sum_{k=1}^{K} \|\boldsymbol{w}_k\|_2^2 + C \sum_{i=1}^{n} \pi_{y_i} H_{T_s}(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}_i), y_i)) \tag{10}$$

$$\text{subject to} \quad \sum_{k=1}^{K} w_{jk} = 0; j = 1, 2, \cdots, d; \sum_{k=1}^{K} b_k = 0,$$

where the constraints are adopted to avoid non-identifiability issue of the solution. Note that (10) is equivalent to the other representation (4) by setting $C = 1/\lambda$. Thus we will use them interchangeably.

Denote Θ as $(\boldsymbol{W}, \boldsymbol{b})$. Applying the fact that $H_{T_s} = H_1 - H_s$, the objective function in (10) can be decomposed as

$$\begin{aligned} Q^s(\Theta) &= \frac{1}{2}\sum_{k=1}^{K}\|\boldsymbol{w}_k\|_2^2 + C\sum_{i=1}^{n}\pi_{y_i}H_1(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}_i), y_i)) - C\sum_{i=1}^{n}\pi_{y_i}H_s(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}_i), y_i)) \\ &= Q^s_{vex}(\Theta) + Q^s_{cav}(\Theta), \end{aligned}$$

where

$$Q^s_{vex}(\Theta) = \frac{1}{2}\sum_{k=1}^{K}\|\boldsymbol{w}_k\|_2^2 + C\sum_{i=1}^{n}\pi_{y_i}H_1(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}_i), y_i))$$

and

$$Q^s_{cav}(\Theta) = -C\sum_{i=1}^{n}\pi_{y_i}H_s(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}_i), y_i))$$

denote the convex and concave parts, respectively.

Define

$$\beta_{ik} = \begin{cases} C\pi_{y_i} & \text{if } k = \text{argmax}(f^t_{k'} : k' \neq y_i),\ f^t_{y_i} - f^t_k < s \\ 0 & \text{otherwise} \end{cases},$$

where $\boldsymbol{f}^t = (f^t_1(\cdot), f^t_2, \cdots, f^t_K)^T$ denotes the solution at the t-th iteration. It is shown in the appendix that the dual problem of the convex optimization at the $(t+1)$-th iteration, given the solution $\boldsymbol{f}^t$ at the t-th iteration, is as follows

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2}\sum_{k=1}^{K}\| \sum_{i:\ y_i=k}\sum_{k'\neq y_i}(\alpha_{ik'} - \beta_{ik'})\boldsymbol{x}_i^T - \sum_{i:\ y_i\neq k}(\alpha_{ik} - \beta_{ik})\boldsymbol{x}_i^T\|_2^2 - \sum_{i=1}^{n}\sum_{k'\neq y_i}\alpha_{ik'} \\ \text{subject to} \quad & \sum_{i:\ y_i=k}\sum_{k'\neq y_i}(\alpha_{ik'} - \beta_{ik'}) - \sum_{i:\ y_i\neq k}(\alpha_{ik} - \beta_{ik}) = 0,\ k = 1, 2, \cdots, K \\ & 0 \leq \sum_{j\neq y_i}\alpha_{ij} \leq C\pi_{y_i},\ i = 1, 2, \cdots, n \\ & \alpha_{ik} \geq 0,\ i = 1, 2, \cdots, n;\ k \neq y_i. \end{aligned}$$

This dual problem is a quadratic programming (QP) problem similar to that of the standard SVM and can be solved by many optimization software. Once the solution is obtained, the coefficients

$\boldsymbol{w}_k$'s can be recovered as follows,

$$\boldsymbol{w}_k = \sum_{i:\, y_i=k} \sum_{k' \neq y_i} (\alpha_{ik'} - \beta_{ik'}) \boldsymbol{x}_i - \sum_{i:\, y_i \neq k} (\alpha_{ik} - \beta_{ik}) \boldsymbol{x}_i. \tag{11}$$

It is interesting to note that representation of $\boldsymbol{w}_k$'s given in (11) automatically satisfies that $\sum_{k=1}^{K} w_{jk} = 0$ for each $1 \leq j \leq d$. Moreover, we can see that coefficients $\boldsymbol{w}_k$'s are determined only by those data points whose corresponding $\alpha_{ik} - \beta_{ik}$ is not zero for some $1 \leq k \leq K$ and these data points are the SVs of the weighted truncated-hinge-loss SVM. The set of SVs of the weighted truncated-hinge-loss SVM using the d.c. algorithm is only a subset of the set of SVs of the original weighted SVM. Basically the weighted truncated-hinge-loss SVM tries to remove points satisfying $f_{y_i}^t - f_k^t < s$ with $k = \text{argmax}(f_{k'}^t : k' \neq y_i)$ from the original set of SVs and consequently eliminate the effects of outliers. This provides an intuitive algorithmic explanation of the robustness of the weighted truncated-hinge-loss SVM to outliers. Similar conclusion was provided by Wu and Liu (2007) for the unweighted version.

After the solution of $\boldsymbol{W}$ is derived, $\boldsymbol{b}$ can be obtained via solving either a sequence of KKT conditions as used in the standard SVM or a linear programming (LP) problem. Denote $\tilde{f}_k(\boldsymbol{x}_i) = \boldsymbol{x}_i^T \boldsymbol{w}_k$. Then $\boldsymbol{b}$ can be obtained through the following LP problem:

$$\begin{aligned}
&\min_{\boldsymbol{\eta}, \boldsymbol{b}} C \sum_{i=1}^{n} \pi_{y_i} \eta_i + \sum_{k=1}^{K} \Big(\sum_{i:\, y_i=k} \sum_{k' \neq y_i} \beta_{ik'} - \sum_{i:\, y_i=k} \beta_{ik} \Big) b_k \\
\text{subject to} \quad & \eta_i \geq 0,\ i = 1, 2, \cdots, n \\
& \eta_i \geq 1 - (\tilde{f}_{y_i}(\boldsymbol{x}_i) + b_{y_i}) + \tilde{f}_k(\boldsymbol{x}_i) + b_k,\ i = 1, 2, \cdots, n;\ k \neq y_i \\
& \sum_{k=1}^{K} b_k = 0.
\end{aligned}$$

4.2 Nonlinear Learning

For nonlinear learning, each decision function $f_k(\boldsymbol{x})$ is represented by $h_k(\boldsymbol{x}) + b_k$ with $h_k(\boldsymbol{x}) \in \mathcal{H}_R$, where $\mathcal{H}_R$ is a reproducing kernel Hilbert space (RKHS). Here the kernel $R(\cdot, \cdot)$ is a positive definite function mapping from $\mathcal{S} \times \mathcal{S}$ to $\Re$. Due to the representer theorem of Kimeldorf and Wahba (1971) (also see Wahba, 1999), the nonlinear problem can be reduced to finding finite dimensional coefficients v_{ik}'s and $h_k(\boldsymbol{x})$ can be represented as $\sum_{i=1}^{n} R(\boldsymbol{x}, \boldsymbol{x}_i) v_{ik}$; $k = 1, 2, \cdots, K$.

Denote $\boldsymbol{v}_k = (v_{1k}, v_{2k}, \cdots, v_{nk})^T$, $\boldsymbol{V} = (\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_K)$, and $\boldsymbol{R}$ to be an $n \times n$ matrix whose (i_1, i_2) entry is $R(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2})$. Let $\boldsymbol{R}_i$ be the i-th column of $\boldsymbol{R}$, and denote the standard basis of the n-dimensional space by $e_i = (0, 0, \cdots, 1, \cdots, 0)^T$ with 1 for its i-th component and 0 for other components.

A similar derivation as in the linear case leads to the following dual problem for nonlinear learning

$$
\begin{aligned}
\min_{\boldsymbol{\alpha}} \quad & \frac{1}{2} \sum_{k=1}^{K} \Bigg\langle \sum_{i:\, y_i = k} \sum_{k' \neq y_i} (\alpha_{ik'} - \beta_{ik'}) \boldsymbol{R}_i - \sum_{i:\, y_i \neq k} (\alpha_{ik} - \beta_{ik}) \boldsymbol{R}_i, \\
& \qquad \sum_{i:\, y_i = j} \sum_{k' \neq y_i} (\alpha_{ik'} - \beta_{ik'}) e_i - \sum_{i:\, y_i \neq k} (\alpha_{ik} - \beta_{ik}) e_i \Bigg\rangle - \sum_{i=1}^{n} \sum_{k' \neq y_i} \alpha_{ik'} \\
\text{subject to} \quad & \sum_{i:\, y_i = k} \sum_{k' \neq y_i} (\alpha_{ik'} - \beta_{ik'}) - \sum_{i:\, y_i \neq k} (\alpha_{ik} - \beta_{ik}) = 0, \ k = 1, 2, \cdots, K \\
& 0 \leq \sum_{k \neq y_i} \alpha_{ik} \leq C \pi_{y_i}, \ i = 1, 2, \cdots, n \\
& \alpha_{ij} \geq 0, \ i = 1, 2, \cdots, n; \ k \neq y_i,
\end{aligned}
$$

where β_{ik}'s are defined similarly as in the linear case. After solving the above QP problem, we can recover the coefficients $\boldsymbol{v}_k$'s as follows

$$
\boldsymbol{v}_k = \sum_{i:\, y_i = k} \sum_{k' \neq y_i} (\alpha_{ik'} - \beta_{ik'}) e_i - \sum_{i:\, y_i \neq k} (\alpha_{ik} - \beta_{ik}) e_i.
$$

The intercepts b_k's can be solved using LP as in the linear learning.

4.3 Parameter Tuning

So far we assume that we have selected the optimal tuning parameter λ. In practice, the tuning parameter selection can be done using an independent validation set or cross validation. In this paper, we choose to select the parameter using an independent set of size $\tilde{n}$. Theoretically speaking, the larger $\tilde{n}$ is, the better tuning effect and the better classifier we may obtain. This is related to Shao (1993)'s results on cross validation in the context of linear model selection: the proportion of tuning set size over the size of all available data points, namely $\tilde{n}/(n + \tilde{n})$, should go to 1 as $(n + \tilde{n}) \to \infty$ to ensure the asymptotically correct selection. However, how to split the data set into the training part and the tuning part is always a trade-off between model

training and parameter tuning, since a large training set is also desired for better model fitting. A commonly accepted procedure is to use one half for training and the other half for tuning, i.e. $n = \tilde{n}$.

Now we detail the approach using an independent tuning set of size $\tilde{n}$. We first obtain probability estimates $\hat{p}_j^{(\lambda_m)}(\tilde{\boldsymbol{x}}_i)$, $j = 1, 2, \cdots, k$ for any $\tilde{\boldsymbol{x}}_i$ in the tuning set $\{(\tilde{\boldsymbol{x}}_i, \tilde{y}_i) : i = 1, 2, \cdots, \tilde{n}\}$ over a grid $\{\lambda_1, \lambda_2, \cdots, \lambda_M\}$ of the tuning parameter. Then we can evaluate the log-likelihood $L(\lambda_m) = \sum_{i=1}^{\tilde{n}} \log(\hat{p}_{y_i}^{(\lambda_m)}(\tilde{\boldsymbol{x}}_i))$ of the tuning set for each λ_m. Let $\hat{m} = \text{argmax}_m L(\lambda_m)$. The optimal tuning parameter is selected to be $\lambda_{\hat{m}}$.

5 Simulations

In this section we use four simulation examples to illustrate the methodological power of our new multiclass probability estimation scheme by comparing it to some existing methods. We consider five alternative methods: cumulative logit model (CLM), baseline logit model (BLM), kernel multi-category logistic regression (KMLR), classification tree (TREE), and random forest (RF). Both CLM and BLM make certain assumptions on the forms of the transformed probabilities. In particular, the CLM assumes that $\log \frac{\sum_{i=1}^{k} p_i(\boldsymbol{x})}{1-\sum_{i=1}^{k} p_i(\boldsymbol{x})} = \beta_{k0} + \boldsymbol{x}^T \boldsymbol{\beta}_k$ for $k = 1, 2, \cdots, K-1$ while the BLM assumes that $\log \frac{p_k(\boldsymbol{x})}{p_K(\boldsymbol{x})} = \beta_{k0} + \boldsymbol{x}^T \boldsymbol{\beta}_k$ for $k = 1, 2, \cdots, K-1$. KMLR refers to the one proposed by Zhu and Hastie (2005) with Gaussian kernel $R(\boldsymbol{x}_1, \boldsymbol{x}_2) = e^{-\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2^2/\sigma^2}$. Ten separate data sets are generated to tune the data width parameter σ among a grid of $\{1/4, 1/2, 3/4, 1, 5/4, 3/2\}\sigma_m$, where σ_m is the median pairwise Euclidean distance defined as median$\{\| \boldsymbol{x}_i - \boldsymbol{x}_j \| : y_i \neq y_j\}$. Among these methods, CLM and BLM are essentially parametric models while our methods, KMLR, Tree, and random forest are nonparametric. Denote the size of the training set by n. Five-fold cross validation is used to select the tuning parameter. For the TREE based method, we use the R package "Tree" and its build-in cross validation function is used to prune trees with fold number set as 10. Similarly we use the build-in tuning for RF provided in the R package.

In simulations, the true conditional probability functions $p_k(\cdot)$, $k = 1, 2, \cdots, K$ are known. In order to measure the estimation accuracy of the conditional probabilities, we use various scores

evaluated on the testing set (of size $10n$):

- 1-norm error $\frac{1}{10n}\sum_{i=1}^{10n}\sum_{k=1}^{K}|\hat{p}_k(\bar{\boldsymbol{x}}_i) - p_k(\bar{\boldsymbol{x}}_i)|$,
- 2-norm error $\frac{1}{10n}\sum_{i=1}^{10n}\sum_{k=1}^{K}(\hat{p}_k(\bar{\boldsymbol{x}}_i) - p_k(\bar{\boldsymbol{x}}_i))^2$,
- Empirical generalized Kullback-Leibler (EGKL) loss $\frac{1}{10n}\sum_{i=1}^{10n}\sum_{k=1}^{K} p_k(\bar{\boldsymbol{x}}_i)\log\frac{p_k(\bar{\boldsymbol{x}}_i)}{\hat{p}_k(\bar{\boldsymbol{x}}_i)}$.

Here $\bar{\boldsymbol{x}}_i$ denote the predictor vector of the ith observation in the testing set. The average errors over 100 replications and the corresponding standard deviations (in parentheses) are reported. Whenever appropriate, our method is employed with minimal truncation with $s = -1/(K-1)$. We implement our method using linear learning for Examples 1, 3, and 4 while using the Gaussian kernel $R(\boldsymbol{x}_1, \boldsymbol{x}_2) = e^{-\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2^2/\sigma^2}$ for Example 2. The grid size d_π is chosen to be 0.02 for our three-class examples and 0.05 for the five-class example. In addition to tuning parameter λ and truncation location s, different grid size gives different performance. See Table 5 for the effect of different grid sizes in the discussion following these numerical examples.

5.1 Numerical Examples

Example 1: We consider a three-class linear learning example. Our data is generated in two steps: 1) Y is uniformly distributed over $\{1, 2, 3\}$; 2) Conditional on $Y = y$, the two-dimension predictor $\boldsymbol{X}$ is generated from $N(\boldsymbol{\mu}(y), \boldsymbol{\Sigma})$, where $\boldsymbol{\mu}(y) = (\cos(2y\pi/3), \sin(2y\pi/3))^T$ and $\boldsymbol{\Sigma} = 0.7^2\boldsymbol{I}_2$ with $\boldsymbol{I}_2$ being the 2×2 identity matrix. The sample size n is 400. Table 1 reports the average test errors and the corresponding standard deviations (in parentheses) over 100 replications for various methods. Note that in this example, the BLM specifies the correct parametric model and hence fits the true (oracle) model, while the CLM corresponds to a model misspecification. A tuning of σ in Gaussian kernel for the KMLR selects σ_m as the best. As shown in Table 1, the oracle BLM performs the best while the CLM performs the worst. Except the oracle BLM, our method with either the direct probability recovery scheme or the indirect probability recovery scheme consistently outperforms all the other methods with significant improvement. Between these two different probability recovery schemes, the indirect scheme works much better. Hence in our later examples, we will only report results of our new method with

Table 1: Probability estimation errors on the test set for Example 1

	Our Method		CLM	KMLR	TREE	RF	BLM (Oracle)
	Direct	Indirect					
1-norm	18.46 (3.83)	11.03 (2.29)	57.35 (1.06)	52.92 (2.80)	27.48 (3.34)	24.02 (1.40)	6.19 (1.95)
2-norm	3.34 (1.74)	0.90 (0.34)	20.53 (0.19)	11.68 (1.20)	5.99 (1.21)	5.01 (0.64)	0.36 (0.23)
EGKL	6.73 (2.21)	2.56 (0.79)	31.28 (0.33)	23.72 (1.77)	Inf (NaN)	Inf (NaN)	0.78 (0.48)

Note: all table entries are multiplied by 100. Numbers in parentheses are the corresponding standard deviations. See the description at the end of Example 1 for the meaning and reasons for Inf (NaN). The same explanation applies to results of other examples.

the indirect scheme. Here the TREE based methods (TREE or RF) lead to infinity (denoted by Inf in Table 1) for EGKL because it returns zero probability for some point $\boldsymbol{x}$ and some classes. The corresponding standard deviation does not make sense and we denote by NaN, which stands for Not A Number. This is one property of TREE-type methods.

Example 2: In this example, we study a three-class nonlinear example. For any $\boldsymbol{x} = (x_1, x_2)^T$, define $f_1(\boldsymbol{x}) = -x_1 + 0.1x_1^2 - 0.05x_2^2 + 0.1$, $f_2(\boldsymbol{x}) = -0.2x_1^2 + 0.1x_2^2 - 0.2$, and $f_3(\boldsymbol{x}) = x_1 + 0.1x_1^2 - 0.05x_2^2 + 0.1$. Set $p_k(\boldsymbol{x}) = P(Y = k|\boldsymbol{X} = \boldsymbol{x}) = \exp(f_k(\boldsymbol{x}))/(\sum_{m=1}^{3} \exp(f_m(\boldsymbol{x})))$ for $k = 1, 2, 3$. Each pair of data point $(\boldsymbol{x}, y)$ is generated in two steps: we first generate $x_1 \sim$ Uniform$[-3, 3]$ and $x_2 \sim$ Uniform$[-6, 6]$; conditional on $\boldsymbol{X} = \boldsymbol{x}$, the class response Y takes value k with probability $p_k(\boldsymbol{x})$ for $k = 1, 2, 3$. The sample size is chosen to be $n = 100$. A similar example was previously used by Zhang et al. (2008).

In this example, we consider basis expansion for the parametric methods CLM and BLM by also including the quadratic terms x_1^2 and x_2^2. Consequently the BLM is again the oracle model. Results over 100 repetitions in the same format of Example 1 are reported in Table 2. Column Indirect corresponds to our method with the indirect probability recovery scheme. The tuning of σ in Gaussian kernel selects $5\sigma_m/4$ and $\sigma_m/2$ as the best for KMLR and our new method, respectively. Similar to Example 1, we again observe that the new method gives the smaller errors than all the other methods except RF for the 1-norm error and the oracle.

Example 3: In Examples 1 and 2, the BLM takes the true model form, so it is not surpris-

Table 2: Probability estimation errors on the test set for Example 2

	Indirect	CLM	KMLR	TREE	RF	BLM (Oracle)
1-norm	36.37 (4.35)	45.42 (2.38)	46.41 (9.82)	60.08 (10.57)	34.10 (3.68)	19.38 (5.21)
2-norm	7.85 (2.06)	13.46 (1.07)	10.08 (3.78)	22.23 (4.75)	8.84 (1.90)	3.19 (1.90)
EGKL	13.60 (2.91)	19.91 (2.04)	18.88 (6.00)	Inf (NaN)	Inf (NaN)	9.11 (9.23)

Note: all table entries are multiplied by 100.

Table 3: Probability estimation errors on the test set for Example 3

	Indirect	CLM	KMLR	TREE	RF	BLM
1-norm	21.78 (2.20)	67.88 (0.82)	59.31 (1.94)	24.44 (3.29)	24.47 (1.20)	31.02 (1.07)
2-norm	4.47 (1.04)	25.80 (0.29)	14.42 (0.88)	7.69 (1.35)	5.55 (0.54)	6.85 (0.27)
EGKL	11.79 (2.58)	38.51 (0.28)	28.48 (1.32)	Inf (NaN)	Inf (NaN)	12.72 (0.40)

Note: all table entries are multiplied by 100.

ing that the BLM shows better performance than our method. In this example, we design an experiment so that none of the parametric methods corresponds to the oracle. This will provide a fair comparison between them.

The two-dimension predictor $\boldsymbol{X}$ is uniformly distributed over the disc $\{\boldsymbol{x} : x_1^2 + x_2^2 \leq 100\}$. Define functions $h_1(\boldsymbol{x}) = -5x_1\sqrt{3} + 5x_2$, $h_2(\boldsymbol{x}) = -5x_1\sqrt{3} - 5x_2$, and $h_3(\boldsymbol{x}) = 0$. Apply a transformation $f_k(\boldsymbol{x}) = \Phi^{-1}(T_2(h_k(\boldsymbol{x})))$, where $\Phi(\cdot)$ and $T_2(\cdot)$ are the cumulative distribution functions of the standard normal distribution and t distribution with degrees of freedom 2, respectively. We set probabilities $p_k(\boldsymbol{x}) = P(Y = k|\boldsymbol{X} = \boldsymbol{x}) = \exp(f_k(\boldsymbol{x}))/(\sum_{j=1}^{3} \exp(f_j(\boldsymbol{x})))$ for $k = 1, 2, 3$ as in Example 2. Because of the nonlinear transformation $\Phi^{-1}(T_2(\cdot))$, BLM is no longer the oracle model. Our multiclass probability with linear kernel is not the oracle model either. The training set size is $n = 600$. The tuning of KMLR selects $\sigma = \sigma_m/4$ as the data width parameter. Table 3 shows clearly that our method is consistently better than the BLM and performs best among all the approaches under comparison.

Table 4: Probability estimation errors on the test set for Example 4

	Indirect	CLM	KMLR	TREE	RF	BLM (Oracle)
1-norm	22.61 (2.52)	81.71 (0.29)	57.91 (2.14)	38.05 (1.90)	42.54 (1.10)	7.33 (1.72)
2-norm	2.64 (0.64)	24.00 (0.25)	9.77 (0.74)	6.78 (0.70)	8.93 (0.50)	0.28 (0.13)
EGKL	7.36 (0.97)	49.05 (0.23)	27.02 (1.49)	Inf (NaN)	Inf (NaN)	0.63 (0.27)

Note: all table entries are multiplied by 100.

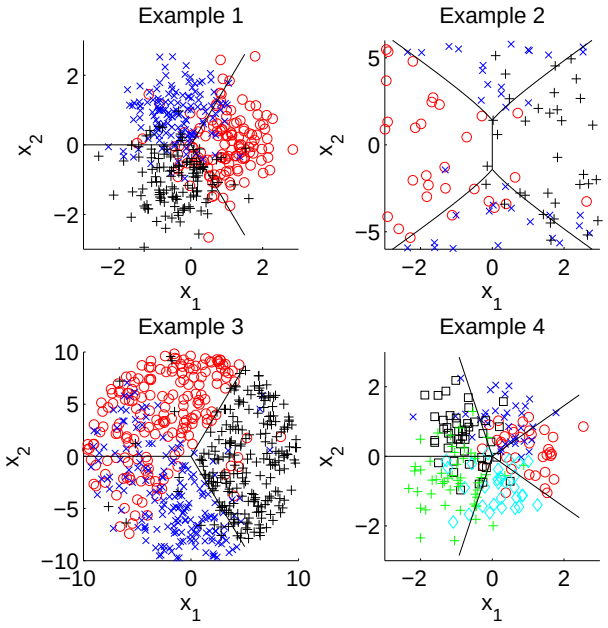

Figure 6: Plots of a randomly selected training set from each simulation example. The solid lines indicate Bayes boundaries for unweighted classification.

Example 4: In this five-class example, the data is generated similarly as in Example 1. Response Y is uniformly distributed over $\{1, 2, 3, 4, 5\}$. Conditional on $Y = y$, the two-dimension predictor $\boldsymbol{X}$ is generated from $N(\boldsymbol{\mu}(y), \boldsymbol{\Sigma})$, where $\boldsymbol{\mu}(y) = (\cos(2y\pi/5), \sin(2y\pi/5))^T$ and $\boldsymbol{\Sigma} = 0.7^2\boldsymbol{I}_2$. The sample size n is 1000. The tuning of KMLR selects $5\sigma_m/4$ as the best. Simulation results are reported in Table 4. Similar improvement is observed for our new method.

Among the six procedures considered above, BLM and CLM are parametric methods, while

our method, KMLR, TREE and RF are nonparametric procedures which do not make explicit assumptions on the form of the true probability functions. Our simulated results suggest that if the parametric assumption is correct, then the associated parametric estimator is essentially the oracle and performs best among all. This explains why BLM gives the smallest errors in Examples 1, 2, and 4. However, if the parametric assumption is incorrect, then the parametric estimators can perform poorly, as shown for the BLM in Example 3 and the CLM in all the settings. By contrast, model-free methods do not rely on the model assumption and show more robust performance. For complicated problems, some of the nonparametric methods can outperform the parametric ones. As shown in Example 3, our method and RF are the top two performers. Furthermore, it is noticed that our method performs competitively among the three nonparametric procedures.

In practice, sometimes it is difficult to determine or validate the parametric assumption on the function forms, especially when data is complicated or high dimensional, then a good nonparametric procedure will provide a useful alternative tool for estimating multiclass probabilities.

5.2 Empirical Computation Cost

The total computation cost of the proposed procedure is mainly determined by three factors: the computation cost of solving one weighted optimization problem, the number of optimization problems corresponding to different weight vectors, and the scheme for recovering probabilities from multiple decision rules. As shown in the paper, each optimization problem involves a non-convex minimization problem, and the proposed DCA-based algorithm seems quite efficient. For example, it takes 0.4827, 5.4086, 0.5118, 2.3549 seconds on average to solve an individual optimization problem for Examples 1, 2, 3, and 4 respectively. Since Example 2 deals with more complicated nonlinear classification problems, it takes a little longer.

The second factor is controlled by the size of d_π. In a three-class problem, if $d_\pi = 0.02$, we need to solve 1,176 optimization problems; and if $d_\pi = 0.1$, we only need to solve 24 problems. The effects of d_π is important: the smaller d_π is, the better estimation result will be. On the other hand, this accuracy gain is obtained at the cost of computational time. To illustrate the effects of d_π on the procedure, we now present the performance of our procedure in Example 1

Table 5: Probability estimation errors on the test set for Example 1 with different d_π

	$d_\pi = 0.1$	$d_\pi = 0.04$	$d_\pi = 0.02$	$d_\pi = 0.01$
1-norm	17.26 (2.55)	12.49 (2.50)	11.03 (2.29)	11.76 (2.18)
2-norm	2.37 (0.81)	1.31 (0.56)	0.90 (0.34)	0.93 (0.33)
EGKL	12.50 (2.67)	4.81 (1.63)	2.56 (0.79)	2.43 (0.64)

with different values of $d_\pi = 0.1, 0.04, 0.02.0.01$ used.

From Table 5, it is clear that smaller d_π values in general lead to better accuracy in probability estimation. However, this accuracy gain levels off as d_π becomes very small. For example, the accuracy improvement from $d_\pi = 0.1$ to $d_\pi = 0.04$ is substantial, but the difference among $d_\pi = 0.04, 0.02, 0.01$ is quite small. It is worth to point out that the computational time grows fast as d_π gets smaller. For example, the computation time for $d_\pi = 0.01$ is about twenty-five folds of that for $d_\pi = 0.04$ and about four folds of that for $d_\pi = 0.02$. So there is a trade-off between computational cost and estimation accuracy when choosing d_π. In our simulations, we find $d_\pi = 0.02$ works pretty well in various three-class problem settings.

To recover the probabilities, we propose two schemes in the paper. The numerical results suggest that the indirect scheme is faster and produces better estimation accuracy. In practice, we recommend to use the indirect scheme.

To conclude our simulation studies, we plot in Figure 6 a randomly chosen training set from each example to show how our training data look like. Note that the sample size is 1000 for Example 4. However to make Figure 6 look nicer, we only use a random sample of size 200 while plotting the right bottom panel for Example 4.

6 Real data

In this section, we apply our new multiclass probability estimation scheme to the wine data by comparing it to those four alternative methods considered in the previous section. The wine data is available online at the UCI Machine Learning Repository by following the URL `http://archive.ics.uci.edu/ml/datasets/Wine`. In addition to the categorical response vari-

Table 6: Results of Wine Example

	Direct	Indirect	CLM	KMLR	TREE	RF	BLM
Test log-likelihood	−9.5487	−6.4817	−Inf	−26.7858	−Inf	−11.2876	−12.7390
Test error	2/58	1/58	8/58	0/58	5/58	0/58	3/58

able Wine Type, it has 13 attributes available. They are Alcohol, Malic acid, Ash, Alcalinity of ash, Magnesium, Total phenols, Flavanoids, Nonflavanoid phenols, Proanthocyanins, Color intensity, Hue, OD280/OD315 of diluted wines, and Proline. All these 13 attributes are continuous. Before applying any probability estimation scheme, we standardize each attribute to have mean zero and standard deviation one. Wine Type belongs to one of three classes with class distribution of 59 in class 1, 71 in class 2, and 48 in class 3. The total number of observations is $n = 178$. We randomly select 19 observations from class 1, 23 from class 2, and 16 from class 3 to be set aside as the testing set. The remaining 120 observations are used as the training data set. We randomly divide those 120 observations in the training set into 8 folds with each fold containing 5 observations in class 1, 6 in class 2, and 4 in class 3 so that an eight-fold cross validation is used to select any tuning parameter over a grid if necessary. The tuning selects σ_m as the best data width parameter for KMLR. For simplicity, our method is implemented with the linear kernel.

For any estimate $\hat{p}_j(\cdot)$, we define its log-likelihood over the testing set as $\sum_{i=1}^{58} \log \hat{p}_{y_i}(\boldsymbol{x}_i)$, where $(\boldsymbol{x}_i, y_i)$, $i = 1, 2, \cdots, 58$ denote observations of the testing set. The corresponding test error is defined by $\sum_{i=1}^{58} I(y_i \neq \text{argmax}_j \hat{p}_j(\boldsymbol{x}_i))$, where $I(.)$ is the indicator function taking value 1 if its argument is true and 0 otherwise. We report both the log-likelihood of the testing set and the test error in Table 6 for all five methods we include for comparison. Same reason as in simulation examples applies to why both CLM and TREE lead to negative infinity. According to Table 6, our new method with linear learning performs competitively in terms of either the log-likelihood of the testing set or the test error.

7 Conclusion

In this work, we propose a model-free multiclass probability estimation approach. It is achieved by solving a series of weighted hard classification problems and then combining these decision rules to construct the probability estimates. Both theoretical and numerical results are provided to demonstrate the competitive performance of our estimation procedure. Our numerical results show favorable performance of our new probability estimation procedure in comparison to several other existing approaches.

Our probability estimation procedure requires computation of weighted classifiers over a fine grid of the K-vertex polyhedron. The computational cost can be high when the class number K gets large. To further improve the computational efficiency, one possible solution is to investigate an efficient solution path over the grid. Further investigation is needed.

Appendix

Proofs of Proposition 1 and Theorem 1. For any $\boldsymbol{x}$ and any $\boldsymbol{\pi} \in A_K$, define $\tilde{p}_k = \pi_k p_k(\boldsymbol{x})/(\sum_{m=1}^K \pi_k p_k(\boldsymbol{x}))$. Then $\tilde{p}_k$ satisfies $\tilde{p}_k \geq 0$ and $\sum_{k=1}^K \tilde{p}_k = 1$. Thus we can treat $\tilde{p}_k$ as new conditional class probability and, as a result, Fisher-consistency implies weighted Fisher-consistency in that $\tilde{p}_k$ covers all possibilities as we vary $\boldsymbol{x}$ in the whole domain. Proofs of Proposition 1 and Theorem 1 will be parallel to the proofs of Proposition 2 and Theorem 1 of Wu and Liu (2007). Thus we skip them to save space. □

Proof of Proposition 2. The unique vector $\tilde{\boldsymbol{\pi}}(\boldsymbol{x})$ is given by $(\tilde{\pi}_1(\boldsymbol{x}), \tilde{\pi}_2(\boldsymbol{x}), \cdots, \tilde{\pi}_K(\boldsymbol{x}))$ with $\tilde{\pi}_k(\boldsymbol{x}) = 1/p_k(\boldsymbol{x})/(\sum_{m=1}^K 1/p_m(\boldsymbol{x}))$. □

Proof of Proposition 3. The result is straightforward by Proposition 2. □

Proof of Theorem 2. Note that, as $n \to \infty$ and $\lambda \to \infty$, the penalty consequently does not contribute to the objective function. Thus asymptotically we are solving

$$\min_{\boldsymbol{f}} \frac{1}{n} \sum_{i=1}^n \pi_{y_i} \ell_{T_s}(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}_i), y_i)) \text{ subject to } \sum_{k=1}^K f_k(\boldsymbol{x}) = 0$$

for each $\boldsymbol{\pi}$.

Note first that, for any $\boldsymbol{x}$ with positive probability density within a small neighborhood $B(\boldsymbol{x}, r) = \{\tilde{\boldsymbol{x}} :\| \tilde{\boldsymbol{x}} - \boldsymbol{x} \| \leq r\}$ of radius $r > 0$ (*i.e.*, $P_{\boldsymbol{X}}(\tilde{\boldsymbol{x}}) > 0$ for any $\tilde{\boldsymbol{x}} \in B(\boldsymbol{x}, r)$), the average of $\pi_{y_i} \ell_{T_s}(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}_i), y_i))$ over $\boldsymbol{x}_i \in B(\boldsymbol{x}, r)$ converges to $E(\pi_Y \ell_{T_s}(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{X}), Y))|\boldsymbol{X} = \boldsymbol{x})$ as $n \to \infty$ and the radius r shrinks to zero. Thus, by Theorem 1, it is guaranteed that there exists a set of neighboring $\boldsymbol{\pi}_1(\boldsymbol{x}) = (\pi_1(\boldsymbol{x}), \pi_2(\boldsymbol{x}), \cdots, \pi_K(\boldsymbol{x}))$, $\boldsymbol{\pi}_2(\boldsymbol{x}) = (\pi_1(\boldsymbol{x}) - d_\pi, \pi_2(\boldsymbol{x}) + d_\pi, \cdots, \pi_K(\boldsymbol{x}))$, $\cdots$, $\boldsymbol{\pi}_K(\boldsymbol{x}) = (\pi_1(\boldsymbol{x}) - d_\pi, \pi_2(\boldsymbol{x}), \cdots, \pi_K(\boldsymbol{x}) + d_\pi)$ such that the weighted truncated large-margin classifiers with $\boldsymbol{\pi}_1(\boldsymbol{x})$, $\boldsymbol{\pi}_2(\boldsymbol{x})$, $\cdots$, and $\boldsymbol{\pi}_K(\boldsymbol{x})$ classify $\boldsymbol{x}$ to class 1, 2, $\cdots$, and K, respectively, for some $\boldsymbol{\pi}(\boldsymbol{x}) = (\pi_1(\boldsymbol{x}), \pi_2(\boldsymbol{x}), \cdots, \pi_K(\boldsymbol{x}))$. Consequently our $\hat{\boldsymbol{\pi}}(\boldsymbol{x}) = \sum_{k=1}^K \boldsymbol{\pi}_k(\boldsymbol{x})/K$ is well defined for any $\boldsymbol{x}$.

For any $\boldsymbol{\pi} = (\pi_1, \pi_2, \cdots, \pi_K)$, $\| \boldsymbol{\pi} \|_1 = \sum_{k=1}^K |\pi_k|$ denotes its 1-norm. Next we prove consistency by contradiction. If there exists an $\boldsymbol{x}$ such that $\hat{\boldsymbol{\pi}}(\boldsymbol{x})$ does not converge to $\tilde{\boldsymbol{\pi}}(\boldsymbol{x})$ satisfying $\tilde{\pi}_k(\boldsymbol{x}) p_k(\boldsymbol{x}) = \tilde{\pi}_{k'}(\boldsymbol{x}) p_{k'}(\boldsymbol{x})$ for $1 \leq k \neq k' \leq K$, then, as $d_\pi \to 0$ and $n \to \infty$, the classification rules for $\boldsymbol{\pi}_1(\boldsymbol{x}), \boldsymbol{\pi}_2(\boldsymbol{x}), \cdots, \boldsymbol{\pi}_K(\boldsymbol{x})$ do not union to the set $\{1, 2, \cdots, K\}$ due to the consistency established in Theorem 1 and the fact that $\max_{k=1}^K \| \boldsymbol{\pi}_k(\boldsymbol{x}) - \hat{\boldsymbol{\pi}}(\boldsymbol{x}) \|_1 = d_\pi / K \to 0$. This violates our criterion on selecting $\boldsymbol{\pi}_1(\boldsymbol{x})$, $\boldsymbol{\pi}_2(\boldsymbol{x})$, $\cdots$, $\boldsymbol{\pi}_K(\boldsymbol{x})$. As a result $\hat{\boldsymbol{\pi}}(\boldsymbol{x}) \to \tilde{\boldsymbol{\pi}}(\boldsymbol{x})$ for any $\boldsymbol{x}$, which in turn implies that $\hat{p}_k(\boldsymbol{x}) \to p_k(\boldsymbol{x})$ for $k = 1, 2, \cdots, K$ for any $\boldsymbol{x}$. □

Derivation of the Dual Problem in Section 4.1:

Note that $\frac{\partial}{\partial \boldsymbol{w}_k} Q^s_{cav}(\Theta)$ and $\frac{\partial}{\partial b_k} Q^s_{cav}(\Theta)$ can be written respectively as follows

$$-C \left[\sum_{i:\, y_i = k} \pi_{y_i} (-I_{\{\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}_i), y_i) < s\}}) \boldsymbol{x}_i^T + \sum_{i:\, y_i \neq k} \pi_{y_i} (I_{\left\{ j = \text{argmax}(f_{k'}(\boldsymbol{x}_i):\, k' \neq y_i),\, f_{y_i}(\boldsymbol{x}_i) - f_k(\boldsymbol{x}_i) < s \right\}}) \boldsymbol{x}_i^T \right],$$

$$-C \left[\sum_{i:\, y_i = k} \pi_{y_i} (-I_{\{\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}_i), y_i) < s\}}) + \sum_{i:\, y_i \neq k} \pi_{y_i} (I_{\left\{ k = \text{argmax}(f_{k'}(\boldsymbol{x}_i):\, k' \neq y_i),\, f_{y_i}(\boldsymbol{x}_i) - f_k(\boldsymbol{x}_i) < s \right\}}) \right],$$

where $I_{\{A\}} = 1$ if event A is true, and 0 otherwise.

Using the definition of β_{ij}, we have

$$\frac{\partial}{\partial \boldsymbol{w}_k} Q^s_{cav}(\Theta) = \sum_{i:\, y_i = k} (\sum_{k' \neq y_i} \beta_{ik'}) \boldsymbol{x}_i^T - \sum_{i:\, y_i \neq k} \beta_{ik} \boldsymbol{x}_i^T,$$

and

$$\frac{\partial}{\partial b_k} Q^s_{cav}(\Theta) = \sum_{i:\ y_i=k} \Big(\sum_{k' \neq y_i} \beta_{ik'} \Big) - \sum_{i:\ y_i \neq k} \beta_{ik}.$$

Applying the first order approximation to the concave part, the objective function at step $(t+1)$ becomes

$$\begin{aligned} Q^s(\Theta) &= \frac{1}{2} \sum_{k=1}^{K} \|\boldsymbol{w}_k\|_2^2 + C \sum_{i=1}^{n} \pi_{y_i} H_1(\min \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}_i), y_i)) \\ &+ \sum_{k=1}^{K} \left\langle \frac{\partial}{\partial \boldsymbol{w}_k} Q^s_{cav}(\Theta_t), \boldsymbol{w}_k \right\rangle + \sum_{k=1}^{K} b_k \frac{\partial}{\partial b_k} Q^s_{cav}(\Theta_t), \end{aligned}$$

where Θ_t is the current solution.

Using slack variable ξ_i's for the hinge loss function, the optimization problem at step $(t+1)$ becomes

$$\begin{aligned} \min_{\boldsymbol{W},\boldsymbol{b},\boldsymbol{\xi}} \quad & \frac{1}{2} \sum_{k=1}^{K} \|\boldsymbol{w}_k\|_2^2 + C \sum_{i=1}^{n} \pi_{y_i} \xi_i + \sum_{k=1}^{K} \left\langle \frac{\partial}{\partial \boldsymbol{w}_k} Q^s_{cav}(\Theta_t), \boldsymbol{w}_k \right\rangle + \sum_{k=1}^{K} b_k \frac{\partial}{\partial b_k} Q^s_{cav}(\Theta_t) \\ \text{subject to} \quad & \xi_i \geq 0 \ \ i = 1, 2, \cdots, n \\ & \xi_i \geq 1 - \left[\boldsymbol{x}_i^T \boldsymbol{w}_{y_i} + b_{y_i} \right] + \left[\boldsymbol{x}_i^T \boldsymbol{w}_k + b_k \right], \ i = 1, 2, \cdots, n; \ \ k \neq y_i. \end{aligned}$$

The corresponding Lagrangian is

$$\begin{aligned} L(\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{\xi}) = & \frac{1}{2} \sum_{k=1}^{K} \|\boldsymbol{w}_k\|_2^2 + C \sum_{i=1}^{n} \pi_{y_i} \xi_i - \sum_{i=1}^{n} u_i \xi_i \\ & - \sum_{i=1}^{n} \sum_{k' \neq y_i} \alpha_{ik'} (\boldsymbol{x}_i^T \boldsymbol{w}_{y_i} + b_{y_i} - \boldsymbol{x}_i^T \boldsymbol{w}_{k'} - b_{k'} + \xi_i - 1) \\ & + \sum_{k=1}^{K} \left\langle \frac{\partial}{\partial \boldsymbol{w}_k} Q^s_{cav}(\Theta_t), \boldsymbol{w}_k \right\rangle + \sum_{k=1}^{K} b_k \frac{\partial}{\partial b_k} Q^s_{cav}(\Theta_t), \end{aligned} \tag{8}$$

subject to

$$\frac{\partial}{\partial \boldsymbol{w}_k} L = \boldsymbol{w}_k^T - \left[\sum_{i:\ y_i=k} \sum_{k' \neq y_i} (\alpha_{ik'} - \beta_{ik'}) \boldsymbol{x}_i^T - \sum_{i:\ y_i \neq k} (\alpha_{ik} - \beta_{ik}) \boldsymbol{x}_i^T \right] = 0 \tag{9}$$

$$\frac{\partial}{\partial b_k} L = - \left[\sum_{i:\ y_i=k} \sum_{k' \neq y_i} (\alpha_{ik'} - \beta_{ik'}) - \sum_{i:\ y_i \neq k} (\alpha_{ik} - \beta_{ik}) \right] = 0 \tag{10}$$

$$\frac{\partial}{\partial \xi_i} L \;=\; C\pi_{y_i} - u_i - \sum_{k \neq y_i} \alpha_{ik} = 0, \tag{11}$$

where the Lagrangian multipliers are $u_i \geq 0$ and $\alpha_{ik'} \geq 0$ for any $i = 1, 2, \cdots, n$, $k' \neq y_i$. Substituting (9)-(11) into (8) yields the desired dual problem in Section 4.1.

References

Agresti, A. and Coull, B. (1998). Approximate is better than 'exact' for interval estimation of binomial proportions. *The American Statistician*, **52** 119–126.

An, L. T. H. and Tao, P. D. (1997). Solving a class of linearly constrained indefinite quadratic problems by d.c. algorithms. *Journal of Global Optimization*, **11** 253–285.

Bartlett, P. L., Jordan, M. I. and Mcauliffe, J. D. (2006). Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, **101** 138–156.

Cortes, C. and Vapnik, V. (1995). Support vector networks. *Machine Learning*, **20** 273–297.

Kimeldorf, G. and Wahba, G. (1971). Some results on Tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications*, **33** 82–95.

Lee, Y., Lin, Y. and Wahba, G. (2004). Multicategory support vector machines, theory, and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association*, **99** 67–81.

Lin, Y. (2002). Support vector machines and the bayes rule. In *Classification, Data Minning and Knoweldge Discovery*. 259–275.

Liu, Y. (2007). Fisher consistency of multicategory support vector machines. *Eleventh International Conference on Artificial Intelligence and Statistics* 289–296.

Liu, Y. and Shen, X. (2006). Multicategory ψ-learning. *Journal of the American Statistical Association*, **101** 500–509.

Liu, Y., Shen, X. and Doss, H. (2005). Multicategory ψ-learning and support vector machine: computational tools. *Journal of Computational and Graphical Statistics*, **14** 219–236.

Shao, J. (1993). Linear model selection by cross-validation. *Journal of the American Statistical Association*, **88** 486–494.

Shen, X., Tseng, G., Zhang, X. and Wong, W. (2003). On ψ-learning. *Journal of the American Statistical Association*, **98** 724–734.

Vapnik, V. (1998). *Statistical Learning Theory*. Wiley, New York.

Wahba, G. (1999). Support vector machines, reproducing kernel hilbert spaces and the randomized GACV. In *Advances in Kernel Methods Support Vector Learning* (B. Schoelkopf, C. Burges and A. Smola, eds.). MIT Press, 69–88.

Wang, J., Shen, X. and Liu, Y. (2008). Probability estimation for large margin classifiers. *Biometrika*, **95** 149–167.

Weston, J. and Watkins, C. (1999). Support vector machines for multi-class pattern recognition. In *Proceedings of the 7th European Symposium on Artificial Neural Networks (ESANN-99)* (M. Verleysen, ed.). Bruges, Belgium, 219–224.

Wu, T. F., Lin, C. J. and Weng, R. C. (2004). Probability estimates for multi-class classification by pairwise coupling. *Journal of Machine Learning Research*, **5** 975–1005.

Wu, Y. and Liu, Y. (2007). Robust truncated-hinge-loss support vector machines. *Journal of the American Statistical Association*, **102** 974–983.

Zhang, H. H., Liu, Y., Wu, Y. and Zhu, J. (2008). Variable selection for the multicategory svm via adaptive sup-norm regularization. *Eletronic Journal of Statistics*, **2** 149–167.

Zhang, T. (2004). Statistical analysis of some multi-category large margin classification methods. *Journal of Machine Learning Research*, **5** 1225–1251.

Zhu, J. and Hastie, T. (2005). Kernel logistic regression and the import vector machine. *Journal of Computational and Graphical Statistics*, **14** 185–205.