Technometrics

ISSN: 0040-1706 (Print) 1537-2723 (Online) Journal homepage: https://www.tandfonline.com/loi/utch20

A Multicategory Kernel Distance Weighted Discrimination Method for Multiclass Classification

Boxiang Wang & Hui Zou

To cite this article: Boxiang Wang & Hui Zou (2019) A Multicategory Kernel Distance Weighted Discrimination Method for Multiclass Classification, Technometrics, 61:3, 396-408, DOI: 10.1080/00401706.2018.1529629

To link to this article: https://doi.org/10.1080/00401706.2018.1529629

Accepted author version posted online: 20 Dec 2018.
Published online: 22 Mar 2019.

TECHNOMETRICS
2019, VOL. 61, NO. 3, 396–408
https://doi.org/10.1080/00401706.2019.1529629

Taylor & Francis
Taylor & Francis Group

A Multicategory Kernel Distance Weighted Discrimination Method for Multiclass Classification

Boxiang Wang[a] and Hui Zou[b]

[a]Department of Statistics and Actuarial Science, University of Iowa, Iowa City, IA; [b]School of Statistics, University of Minnesota, Minneapolis, MN

ABSTRACT

Distance weighted discrimination (DWD) is an interesting large margin classifier that has been shown to enjoy nice properties and empirical successes. The original DWD only handles binary classification with a linear classification boundary. Multiclass classification problems naturally appear in various fields, such as speech recognition, satellite imagery classification, and self-driving vehicles, to name a few. For such complex classification problems, it is desirable to have a flexible multicategory kernel extension of the binary DWD when the optimal decision boundary is highly nonlinear. To this end, we propose a new multicategory kernel DWD, that is, defined as a margin-vector optimization problem in a reproducing kernel Hilbert space. This formulation is shown to enjoy Fisher consistency. We develop an accelerated projected gradient descent algorithm to fit the multicategory kernel DWD. Simulations and benchmark data applications are used to demonstrate the highly competitive performance of our method, as compared with some popular state-of-the-art multiclass classifiers.

ARTICLE HISTORY
Received August 2017
Accepted September 2018

KEYWORDS
Distance weighted discrimination; Fisher consistency; Multicategory classification; Nesterov's acceleration; Projected gradient descent; Reproducing kernel Hilbert space

1. Introduction

Classification is a task of identifying observations to one of several pre-defined categories, and its applications are extremely diverse, ranging from daily life to frontiers of science and engineering. Two classic examples are detecting spam e-mail based on the message content and categorizing tissues as tumor or benign based on DNA microarray data. Many real-world problems have multicategory responses. Speech recognition has been formulated as a multicategory classification problem to analyze voice input, which enables the translation of spoken language into text and has many promising applications in as court reporting, mobile e-mail, and robotics (Rabiner 1989; Rabiner and Juang 1993; Hansen and Hasan 2015; Yu and Deng 2016). Speech recognition has also greatly helped people with hearing disturbances (Chen et al. 2016; Takashima et al. 2017; Wang 2017). Image classification (Haralick and Shanmugam 1973; Krizhevsky, Sutskever, and Hinton 2012; Russakovsky et al. 2015) is another hot application, referring to detection of an object in digital images: for instance, satellite remote images have been used to successfully predict earthquakes (Dong and Shan 2013; Lillesand, Kiefer, and Chipman 2014; Maulik and Chakraborty 2017), vision-based road detection inspires the study of self-driving vehicles (Chen et al. 2015; Xu et al. 2016; Bojarski et al. 2017), and facial expression extraction facilitates interactions between humans and machines (Liu et al. 2012; Barsoum et al. 2016). Besides engineering applications, binary and multicategory classifications are also abundant in biology, climatology, geology, economics, and finance, among many others.

For binary classification, the support vector machine (SVM, Vapnik 1995) is a commonly used large-margin classifier. Another large margin classifier is the distance weighted discrimination (DWD) proposed by Marron, Todd, and Ahn (2007). Although the SVM and DWD are originally designed for binary classification, they can be generalized to multicategory classification problems. Two simple approaches are one-versus-one and one-versus-rest that decompose multicategory classification into a set of multiple binary classification problems (Hastie and Tibshirani 1998; Hsu and Lin 2002). In particular, one-versus-one approach solves each of the pairwise two-class problem and predicts the class that wins the most comparisons, but it may suffer from the tie-in-vote issue. One-versus-rest approach alternatively treats each class as positive and all the other classes as negative; however, this approach has shown to be inconsistent in many situations (Lee et al. 2004; Liu 2007). In addition, error-correcting coding is an information-theoretic approach that turns the multicategory response into a coding matrix; details are seen in Dietterich and Bakiri (1995), James and Hastie (1998), and Allwein, Schapire, and Singer, (2000). Instead of reducing multicategory classification to binary problems, another approach is to propose a unified framework that considers all classes at once. With such a simultaneous fashion, there are several multicategory SVMs developed in Vapnik (1998), Weston and Watkins (1999), and Lee et al. (2004), as well as multicategory extension of other large-margin classifiers including import vector machine (Zhu and Hastie 2005), ψ-learning (Liu and Shen 2006), large-margin unified machines (Zhang and Liu 2013), and angle-based large-margin classification (Zhang and Liu 2014; Zhang et al. 2016).

In the context of DWD, Huang et al. (2013) proposed a multiclass generalization of linear DWD. From methodological and theoretical viewpoint, the linear classifier will be inadequate because the optimal Bayes rule can often be nonlinear.

CONTACT Hui Zou zouxx019@umn.edu School of Statistics, University of Minnesota, Minneapolis, MN 55455.

However, it is unclear how to generalize the linear multiclass DWD (Huang et al. 2013) to its kernel counterpart. The same difficulty appeared in the development of the original binary linear DWD (Marron, Todd, and Ahn 2007), and a kernel binary DWD, that is, computationally efficient and theoretically justified, was only recently proposed in Wang and Zou (2018). Moreover, the kernel binary DWD in Wang and Zou (2018) has been shown to enjoy very competitive classification performance against popular classifiers such as the SVM, random forest, gradient boosting, and k-nearest neighbors, etc. Given its excellent performance for binary classification, it will be interesting and natural to ask whether the DWD idea could also be competitive for multiclass classification. Hence, it is necessary to derive the kernel version of the multicategory DWD in order to handle multiclass classification problems with complex nonlinear decision boundaries.

In this article, we develop a multicategory kernel DWD by formulating the multicategory DWD in a reproducing kernel Hilbert space (RKHS). We used the concept *margin vector* introduced by Zou, Zhu, and Hastie (2008), where the margin vector is defined to be a multicategory generalization of the margin in binary classification and can be regarded as a proxy of the conditional class probability. With the device of margin vector, we propose multicategory kernel DWD, and we then demonstrate that our proposal is multicategory Fisher-consistent, in the sense that the class with the largest conditional class probability always has the largest margin. To compute the multicategory kernel DWD, we present a multicategory representer theorem, and we develop a projected gradient descent algorithm. We further implement the Nesterov's acceleration to improve the rate of convergence, thereby reducing the number of iterations effectively. Note that our formulation of multicategory DWD is completely different from the approach (Huang et al. 2013) that generalizes the linear DWD by involving the pairwise differences in terms of the discriminate functions. We shall review the formulation of Huang et al. (2013) in Section 2.

To give a quick illustration, Figure 1 delineates the decision boundaries of multicategory kernel DWD and the Bayes rule for a simulation example based on mixture Gaussian distributions. Figure 1 shows that the Bayes rule has a nonlinear decision boundary and our method resembles the Bayes rule. This example clearly reveals the inadequacy of the multicategory linear DWD as well as the excellent performance of our new method.

The rest of the article is organized as follows. In Section 2, we briefly review DWD in binary classification and multicategory linear DWD proposed in Huang et al. (2013). Section 3 describes our proposal of multicategory kernel DWD and we explore its multicategory Fisher consistency. In Section 4, we derive an efficient convex optimization algorithm to solve the proposed classifier. Simulations and benchmark data examples are presented in Section 5.

2. Review of DWD

Before introducing the multicategory kernel DWD, it is necessary to review the basic idea of the original binary DWD. Suppose that a training dataset consists of n pairs of observations, $\{\boldsymbol{x}_i, y_i\}_{i=1}^n$, where $\boldsymbol{x}_i \in \mathbb{R}^p$ and $y_i \in \{-1, 1\}$. Linear DWD seeks a hyperplane $\{\boldsymbol{x} : \hat{\beta}_0 + \boldsymbol{x}^\top \hat{\boldsymbol{\beta}} = 0\}$ where

$$\left(\hat{\beta}_0, \hat{\boldsymbol{\beta}}\right) = \underset{\beta_0, \boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^{n} \left(\frac{1}{r_i} + C\xi_i\right),$$
$$\text{subject to } r_i = y_i(\beta_0 + \boldsymbol{x}_i^\top \boldsymbol{\beta}) + \xi_i \geq 0, \xi_i \geq 0, \forall i, \|\boldsymbol{\beta}\|_2^2 = 1, \tag{2.1}$$

where C is a tuning parameter controlling the slack variables ξ_i. DWD predicts the class label of a new observation $\boldsymbol{x}_{\text{new}}$ by $\text{sgn}(\hat{\beta}_0 + \boldsymbol{x}_{\text{new}}^\top \hat{\boldsymbol{\beta}})$. The problem (2.1) was originally solved by second-order cone programming (Marron, Todd, and Ahn 2007). Other developments of linear DWD include weighted DWD (Qiao et al. 2010), distance weighted SVM (Qiao and

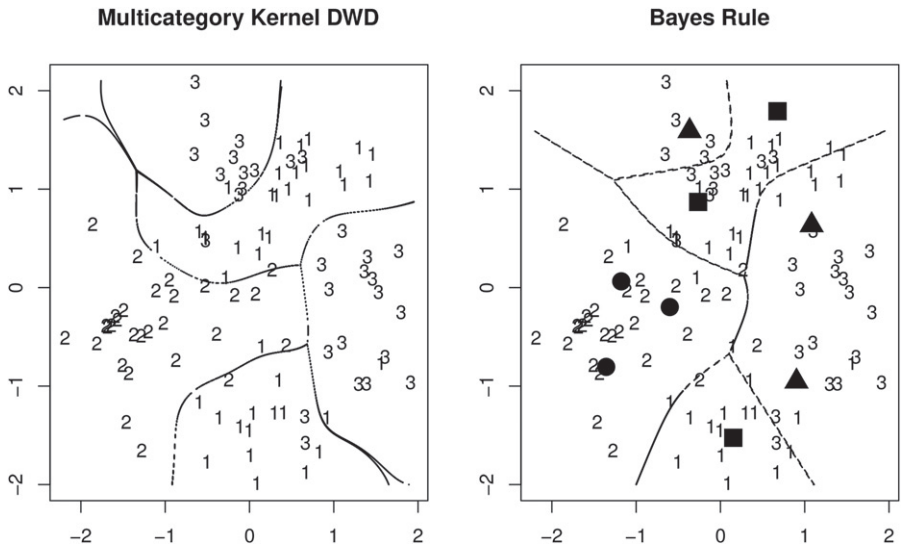

Figure 1. Decision boundaries of multicategory kernel DWD (left panel) and the Bayes rule (right panel). We simulated three classes, each of which follows a mixture Gaussian distribution $\frac{1}{3}\sum_{i=1}^{3} N(\boldsymbol{\mu}_i, \tau^2 \boldsymbol{I})$, where $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \boldsymbol{\mu}_3$ are three centers independently drawn from standard normal distribution. In the right panel, the centers of class 1, 2, and 3 are depicted as squares, circles, and triangles, respectively. We set $\tau = 0.4$ and Bayes error is 13.4% in this example. The proposed multicategory kernel DWD is fit based on 100 training data and its misclassification error rate is 15.9%. In contrast, the multicategory linear DWD (Huang et al. 2013) has a misclassification error rate of 34.5%.

Zhang 2015a), flexible assortment machines (Qiao and Zhang 2015b), and sparse DWD (Wang and Zou 2016).

Huang et al. (2013) proposed a multicategory linear DWD. Suppose the response of a training dataset $\{\boldsymbol{x}_i, y_i\}_{i=1}^n$ has k categories, that is, $y_i \in \{1, \ldots, k\}$. A vector of discriminant functions $\boldsymbol{f} = (f_1, \ldots, f_k)$ is introduced, where each element corresponds to one class. For any new observation $\boldsymbol{x}_{\text{new}}$, the label is predicted by $\hat{y}_{\text{new}} = \text{argmax}_j \hat{f}_j(\boldsymbol{x}_{\text{new}})$, where each $\hat{f}_j(\boldsymbol{x}) = \hat{\beta}_0 + \boldsymbol{x}^\top \hat{\boldsymbol{\beta}}$ and $(\hat{\beta}_{0j}, \hat{\boldsymbol{\beta}}_j)$ are estimated by

$$\begin{aligned} &\min_{\beta_{0j}, \boldsymbol{\beta}_j} \sum_{i=1}^n \sum_{j \neq l} \left(\frac{1}{r_i^{(jl)}} + C\xi_i^{(jl)} \right), \\ &\text{subject to } r_i^{(jl)} = f_j(\boldsymbol{x}_i) - f_l(\boldsymbol{x}_i) + \xi_i^{(jl)}, \text{ for } y_i = j, l \neq j, \\ &\qquad f_j(\boldsymbol{x}_i) = \beta_{0j} + \boldsymbol{x}_i^\top \boldsymbol{\beta}_j, \\ &\qquad r_i^{(jl)} \geq 0, \xi_i^{(jl)} \geq 0, \sum_{j=1}^k \beta_{0j} = 0, \\ &\qquad \sum_{j=1}^k \boldsymbol{\beta}_j = \boldsymbol{0}, \|\boldsymbol{\beta}_j\|_2^2 \leq 1. \end{aligned} \tag{2.2}$$

Like the binary linear DWD, the problem (2.2) is solved by second-order cone programming. However, it is unclear how to extend the formulation (2.2) to a reproducing kernel Hilbert space so that one can fit a nonlinear kernel classifier. It is difficult even when the problem (2.2) degenerates to the binary DWD, when $k = 2$. Only recently, Wang and Zou (2018) derived a kernel DWD based on a different formulation of linear DWD.

3. A New Multicategory Kernel DWD

In this section, we develop a multicategory DWD in an RKHS, and we elucidate its Fisher-consistent property.

3.1. Statistical View of DWD

Wang and Zou (2018) showed that the linear DWD classifier can be equivalently derived from a regularized empirical risk minimization approach as

$$\left(\hat{\beta}_0, \hat{\boldsymbol{\beta}}\right) = \underset{\beta_0, \boldsymbol{\beta}}{\text{argmin}} \left[\frac{1}{n} \sum_{i=1}^n \phi \left\{ y_i(\beta_0 + \boldsymbol{x}_i^\top \boldsymbol{\beta}) \right\} + \lambda \boldsymbol{\beta}^\top \boldsymbol{\beta} \right],$$

where

$$\phi(u) = \begin{cases} 1 - u, & \text{if } u \leq 1/2, \\ 1/(4u), & \text{if } u > 1/2, \end{cases} \tag{3.1}$$

and the DWD classifer is $\text{sgn}(\hat{\beta}_0 + \boldsymbol{x}^\top \hat{\boldsymbol{\beta}})$. The loss function $\phi(u)$ has also appeared in Qiao et al. (2010), Liu, Zhang, and Wu (2011), and Wang and Zou (2016). For the kernel DWD, Wang and Zou (2018) formulated kernel DWD as $\text{sgn}(\hat{f}(\boldsymbol{x}))$, where $\hat{f}$ is given by

$$\hat{f} = \underset{f \in \mathcal{H}_K}{\text{argmin}} \left[\frac{1}{n} \sum_{i=1}^n \phi \left\{ y_i f(\boldsymbol{x}_i) \right\} + \lambda \|f\|_{\mathcal{H}_K}^2 \right], \tag{3.2}$$

in which $\mathcal{H}_K$ is an RKHS generated by a positive definite kernel function K. The popular kernel functions include the Gaussian kernel and the polynomial kernel. By Mercer's theorem, kernel K has an eigen-expansion $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sum_{t=1}^\infty \gamma_t \varphi_t(\boldsymbol{x}_i) \varphi_t(\boldsymbol{x}_j)$ with $\gamma_t \geq 0$ and $\sum_{t=1}^\infty \gamma_t^2 < \infty$. The function f in the space $\mathcal{H}_K$ has an expansion in terms of eigen-functions, $f(\boldsymbol{x}) = \sum_{t=1}^\infty c_t \varphi_t(\boldsymbol{x})$, where $\|f\|_{\mathcal{H}_K}^2 \equiv \sum_{t=1}^\infty c_t^2/\gamma_t < \infty$.

By the representer theorem (Wahba 1990), the solution of problem (3.2) has a finite form

$$\hat{f}(\boldsymbol{x}) = \sum_{i=1}^n \hat{\boldsymbol{\alpha}}_i K(\boldsymbol{x}, \boldsymbol{x}_i).$$

Then the reproducing property of the RKHS (Wahba 1990) implies $\|\hat{f}\|_{\mathcal{H}_K}^2 = \sum_{i=1}^n \sum_{i'=1}^n \hat{\alpha}_i \hat{\alpha}_{i'} K(\boldsymbol{x}_i, \boldsymbol{x}_{i'})$, and problem (3.2) becomes

$$\hat{\boldsymbol{\alpha}} = \underset{\boldsymbol{\alpha} \in \mathbb{R}^n}{\text{argmin}} \left[\frac{1}{n} \sum_{i=1}^n \phi \left\{ y_i \boldsymbol{K}_i^\top \boldsymbol{\alpha} \right\} + \lambda \boldsymbol{\alpha}^\top \boldsymbol{K} \boldsymbol{\alpha} \right], \tag{3.3}$$

where $\boldsymbol{K}$ is an $n \times n$ matrix whose (i, i')th element is $K(\boldsymbol{x}_i, \boldsymbol{x}_{i'})$. Problem (3.3) can be efficiently solved based on the MM principle (Wang and Zou 2018), which is much faster than the second-order cone programming algorithm. The reproducing property of the RKHS largely facilitates the computation algorithms, as the implicit and infinite-dimensional problem (3.2) reduces to an explicit and finite-dimensional problem (3.3). The explicit feature map of the RKHS induced by the Gaussian kernel has been studied in Steinwart, Hush, and Scovel (2006).

3.2. Our Proposal

The empirical loss minimization formulation of the original DWD is the first step toward the multicategory DWD. In the literature many efforts have been devoted to the multiclass generalization of the binary large margin classifier that can be formulated as an empirical loss minimization problem. For example, the multicategory SVM (Lee et al. 2004), the multicategory ψ-learning (Liu and Shen 2006), and so on. Here, we take a different approach from the existing multicategory large margin classifiers in the literature. Specifically, we take advantage of the concept of *margin vector*, which is introduced by Zou, Zhu, and Hastie (2008) and is conceptually identical to the binary margin. In binary classification, the margin is defined as yf, which assigns margin $f(\boldsymbol{x}_i)$ to a data point $(\boldsymbol{x}_i, y_i)$ from positive class and assigns margin $-f(\boldsymbol{x}_i)$ to datum from negative class. The binary margin definition explicitly uses the special $1, -1$ coding of the class label. For a k class problem, a margin vector has the form of $\boldsymbol{f} = (f_1, \ldots, f_k)^\top$ with a sum-to-zero constraint $\sum_{j=1}^k f_j = 0$. Data point $(\boldsymbol{x}_i, y_i)$ belonging to class y_i has margin $f_{y_i}(\boldsymbol{x}_i)$, where $y_i \in \{1, 2, \ldots, k\}$. When $k = 2$, by the sum-to-zero constraint we have $f_1(\boldsymbol{x}_i) = -f_2(\boldsymbol{x}_i)$. Thus, when we use $1, -1$ to code the classes 1 and 2, we have $f = f_1$ and the margin for $(\boldsymbol{x}_i, y_i)$ is $y_i f(\boldsymbol{x}_i)$, which is the definition of the margin.

Now we replace the margin $y_i f(\boldsymbol{x}_i)$ with the margin vector $f_{y_i}(\boldsymbol{x}_i)$ in problem (3.2) and end up with the formulation

$$\hat{\boldsymbol{f}} = \underset{f_j \in \mathcal{H}_K}{\operatorname{argmin}} \left[\frac{1}{n} \sum_{i=1}^{n} \phi \left\{ f_{y_i}(\boldsymbol{x}_i) \right\} + \lambda \sum_{j=1}^{k} ||f_j||^2_{\mathcal{H}_K} \right], \quad \text{subject to } \sum_{j=1}^{k} f_j = 0, \tag{3.4}$$

where ϕ is the DWD loss (3.1) and $\mathcal{H}_K$ is an RKHS generated by a positive definite kernel K. The multicategory DWD classifier is $\hat{y} = \operatorname{argmax}_{j \in \{1,2,\ldots,k\}} \hat{f}_j(\boldsymbol{x})$. For the actual multiclass classification problem with $k \geq 3$, the formulation (3.4) is fundamentally different from the binary case in problem (3.2) in terms of computational and theoretical treatments. Thus, the multicategory kernel DWD is not a trivial extension of the binary kernel DWD. The computation of $\hat{\boldsymbol{f}}$ is discussed in Section 4, and its competitive performance is demonstrated in Section 5.

To appreciate the formulation (3.4), let us consider the ideal case when n is infinity and $\lambda = 0$. Define $p_j(\boldsymbol{x}) = P(y = j|\boldsymbol{x}), j \in \{1, \ldots, k\}$. Note that $\sum_{i=1}^{n} \phi\{f_{y_i}(\boldsymbol{x}_i\}/n$ becomes $\sum_{j=1}^{k} \phi\{f_j(\boldsymbol{x})\} p(y = j|\boldsymbol{x})$. Thus, the problem (3.4) becomes

$$\boldsymbol{f}^\star(\boldsymbol{x}) = \underset{f}{\operatorname{argmin}} \left[\sum_{j=1}^{k} \phi\{f_j(\boldsymbol{x})\} p_j(\boldsymbol{x}) \right], \text{ subject to } \sum_{j=1}^{k} f_j(\boldsymbol{x}) = 0. \tag{3.5}$$

The population multicategory DWD classifier is $\hat{y} = \operatorname{argmax}_{j \in \{1,2,\ldots,k\}} f_j^\star(\boldsymbol{x})$.

Conceptually speaking, the population multicategory DWD classifier is the target of the proposed multicategory DWD classifier $\hat{\boldsymbol{f}}$. In the next theorem, we show that the population multicategory DWD classifier is actually the Bayes rule, which indicates that the proposed multicategory DWD classifier estimates the right target for the multiclass classification problem. Such a property is called Fisher consistency (Lin 2004).

Theorem 1. (Multicategory fisher consistency). Assume that for each x (or with measure one) there is a most likely label $j^\star$ such that $p_{j^\star}(\boldsymbol{x}) > p_j(\boldsymbol{x})\ \forall j \neq j^\star$, and there is the least possible label j_* such that $p_j(\boldsymbol{x}) > p_{j_*}(\boldsymbol{x})\ \forall j \neq j_*$. Then the solution of the problem (3.5) is given by

$$f_j^\star(\boldsymbol{x}) = \begin{cases} \frac{1}{2} \sqrt{\frac{p_j(\boldsymbol{x})}{p_{j_\star}(\boldsymbol{x})}}, & j \neq j_*, \\ -\frac{1}{2} \sum_{l \neq j_*} \sqrt{\frac{p_l(\boldsymbol{x})}{p_{j_\star}(\boldsymbol{x})}}, & j = j_*. \end{cases} \tag{3.6}$$

Consequently, Theorem 1 indicates that $\operatorname{argmax}_{j \in \{1,2,\ldots,k\}} f_j^\star(\boldsymbol{x}) = \operatorname{argmax}_{j \in \{1,2,\ldots,k\}} p_j(\boldsymbol{x})$, that is, the population multicategory DWD is identical to the Bayes rule.

Huang et al. (2013) also proved the Fisher consistency of their multicategory DWD, that is, based on pairwise differences in discriminant functions. However, their method only considered the linear DWD but not the more flexible kernel DWD. Based on its meaning, Fisher consistency is much more relevant when the classifier can be flexible and nonlinear. In Section 5 the multicategory linear DWD is shown to be inconsistent in some simulation examples.

Although (3.4) is defined as a functional optimization problem in a possibly infinite dimensional functional space, the nice reproducing property of RKHS makes the computation of $\hat{\boldsymbol{f}}$ in problem (3.4) to be carried out in a finite-dimensional vector space.

Theorem 2. (Multicategory representer theorem). If $\mathcal{H}_K$ is generated by a positive definite kernel function K, then the solution of (3.4), $\hat{\boldsymbol{f}} = (\hat{f}_1, \ldots, \hat{f}_k)$, has a finite form,

$$\hat{f}_j(\boldsymbol{x}) = \sum_{i=1}^{n} \hat{\alpha}_{ij} K(\boldsymbol{x}, \boldsymbol{x}_i),\ j = 1, \ldots, k, \tag{3.7}$$

and

$$\sum_{j=1}^{k} \hat{\alpha}_{ij} = 0, \quad \forall i = 1, \ldots, n.$$

Define $\boldsymbol{K}$ to be the kernel matrix whose (i, i')th element is $K(\boldsymbol{x}_i, \boldsymbol{x}_{i'})$ and let $\boldsymbol{K}_i$ be the ith column vector. For each class j, Theorem 2 implies that there exists $\hat{\boldsymbol{\alpha}}_j$ such that $\hat{f}_j(\boldsymbol{x}_i) = \boldsymbol{K}_i^\top \boldsymbol{\alpha}_j$, where $\hat{\boldsymbol{\alpha}}_j = (\hat{\alpha}_{1j}, \ldots, \hat{\alpha}_{nj})^\top$. Now we only need to compute $\hat{\boldsymbol{\alpha}}_j$ for $j = 1, \ldots, k$.

By the reproducing property, it can be further obtained that

$$\|\hat{f}_j\|^2_{\mathcal{H}_K} = \sum_{i=1}^{n} \sum_{i'=1}^{n} \hat{\alpha}_{ij} \hat{\alpha}_{i'j} K(\boldsymbol{x}_i, \boldsymbol{x}_{i'}) = \hat{\boldsymbol{\alpha}}_j^\top \boldsymbol{K} \hat{\boldsymbol{\alpha}}_j. \tag{3.8}$$

Note that $f_{y_i}(\boldsymbol{x}_i) = \boldsymbol{K}_i^\top \boldsymbol{\alpha}_{y_i}$. Then, we can rephrase the optimization problem (3.4) as follows

$$\min_{\boldsymbol{\alpha}_j \in \mathbb{R}^n} \left[\frac{1}{n} \sum_{i=1}^{n} \phi \left\{ \boldsymbol{K}_i^\top \boldsymbol{\alpha}_{y_i} \right\} + \lambda \sum_{j=1}^{k} \boldsymbol{\alpha}_j^\top \boldsymbol{K} \boldsymbol{\alpha}_j \right], \quad \text{subject to } \sum_{j=1}^{k} \boldsymbol{\alpha}_j = \boldsymbol{0}. \tag{3.9}$$

In Section 4, we shall derive an efficient algorithm to solve the problem (3.9).

3.3. Related Methods

To connect our method with other simultaneous multiclass large-margin classifiers in the literature, we formulate the loss of our proposal (3.5) as

$$\min_{f} E_{\boldsymbol{x}y} \phi\{f_y(\boldsymbol{x})\}, \text{ subject to } \sum_{j=1}^{k} f_j(\boldsymbol{x}) = 0.$$

Vapnik (1998), Bredensteiner and Bennett (1999), and Weston and Watkins (1999) proposed multiclass SVMs, all of which, as shown by Guermeur (2002), can be written equivalently as

$$\min_{f} E_{\boldsymbol{x}y} \sum_{j \neq y} \left[1 - (f_y(\boldsymbol{x}) - f_j(\boldsymbol{x})) \right]_+, \tag{3.10}$$

where $[w] = \max(w, 0)$. Crammer and Singer (2001) presented another multiclass SVM as

$$\min_{f} E_{\boldsymbol{x}y} \left[1 - \min_{j \neq y} (f_y(\boldsymbol{x}) - f_j(\boldsymbol{x})) \right]_+. \tag{3.11}$$

Lee et al. (2004) developed multiclass SVM as

$$\min_{f} E_{xy} \sum_{j \neq y} \left[1 + f_j(\boldsymbol{x})\right]_+, \text{ subject to } \sum_{j=1}^{k} f_j(\boldsymbol{x}) = 0, \quad (3.12)$$

and they showed that their proposal is Fisher consistent but the methods (3.10) and (3.11) are not. Liu and Shen (2006) introduced multicategory ψ-learning

$$\min_{f} E_{xy} \left[1 - \left(\min_{j \neq y} (f_y(\boldsymbol{x}) - f_j(\boldsymbol{x})) \right)_+ \right]_+ \quad (3.13)$$

by replacing the convex SVM hinge loss with a nonconvex ψ-loss. Liu and Yuan (2011) proposed reinforced multiclass SVM, on the basis of the linear combination of the SVM hinge loss and the loss function in Lee et al. (2004):

$$\min_{f} E_{xy} \sum_{j \neq y} \left[(1-\gamma)(1 - f_y(\boldsymbol{x}))_+ + \gamma(1 + f_j(\boldsymbol{x}))_+ \right],$$
$$\text{subject to } \sum_{j=1}^{k} f_j(\boldsymbol{x}) = 0, \quad (3.14)$$

which is shown to enjoy the Fisher consistency when $\gamma \in [1/2, 1]$.

Among the aforementioned approaches, the sum-to-zero constraint $\sum_{j=1}^{k} f_j(\boldsymbol{x}) = 0$ is enforced in the methods (3.12) and (3.14) and can be also imposed in others to ensure the uniqueness of the optimal solution. To avoid the explicit sum-to-zero constraint of those methods, Zhang and Liu (2014) proposed a novel angle-based approach, fitting a model based on the angles between data and each vertex vector of a k-simplex. We take as an example the reinforced angle-based multiclass SVM (RAMSVM, Zhang et al. 2016), which is developed by applying the angle-based approach to the reinforced multiclass SVM (3.14). Specifically, the angle-based approach first finds a k-simplex that consists of k unit-norm vertices $\{\boldsymbol{W}_j\}_{j=1}^{k} \in \mathbb{R}^{k-1}$ such that the angles between the pairs $(\boldsymbol{W}_j, \boldsymbol{W}_{j'})$ are the same. The model is then fitted by replacing each functional margin $f_j(\boldsymbol{x})$ in (3.14) by $\langle \boldsymbol{f}, \boldsymbol{W}_j \rangle$:

$$\min_{f} E_{xy} \sum_{j \neq y} \left[\frac{1}{2}(1 - \langle \boldsymbol{f}(\boldsymbol{x}), \boldsymbol{W}_y \rangle)_+ + \frac{1}{2}(1 + \langle \boldsymbol{f}(\boldsymbol{x}), \boldsymbol{W}_j \rangle)_+ \right].$$

The prediction is made according to $\hat{y} = \operatorname{argmax}_j \langle \boldsymbol{f}(x), \boldsymbol{W}_j \rangle$. Since $\sum_{j=1}^{k} \langle \boldsymbol{f}(\boldsymbol{x}), \boldsymbol{W}_j \rangle = 0$ always holds, the sum-to-zero constraint is dismissed. Other applications of the angle-based approaches are seen in Sun, Craig, and Zhang (2017), Zhang et al. (2017), Fu, Zhang, and Liu (2018), and Liu, Liu, and Zhu (2018). Compared with the angle-based method, our method has the explicit sum-to-zero constraint. As will be shown in Section 4, our algorithm handles such constraint quite naturally and efficiently.

4. Computation Algorithm

The multicategory kernel DWD problem (3.9) is more sophisticated than the binary kernel DWD problem (3.2) due to the sum-to-zero constraint. In this section, we derive an accelerated projected gradient descent (PGD) algorithm to solve problem (3.9).

4.1. Projected Gradient Descent Algorithm

We first derive the projected gradient descent algorithm and then derive its accelerated version.

Notation. $\boldsymbol{A} \otimes \boldsymbol{B}$ denotes the *Kronecker product* of an $m \times n$ matrix $\boldsymbol{A}$ and a $p \times q$ matrix $\boldsymbol{B}$ is the $mp \times nq$ matrix

$$\boldsymbol{A} \otimes \boldsymbol{B} = \begin{pmatrix} a_{11}\boldsymbol{B} & a_{12}\boldsymbol{B} & \dots & a_{1n}\boldsymbol{B} \\ a_{21}\boldsymbol{B} & a_{22}\boldsymbol{B} & \dots & a_{2n}\boldsymbol{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}\boldsymbol{B} & a_{m2}\boldsymbol{B} & \dots & a_{mn}\boldsymbol{B} \end{pmatrix}.$$

The *vectorization* of a $m \times n$ matrix $\boldsymbol{A}$ converts the matrix into a mn-column vector by stacking the first, second, ..., nth columns $\boldsymbol{a}_1, \boldsymbol{a}_2, \dots, \boldsymbol{a}_n$ of $\boldsymbol{A}$ one under the other:

$$\operatorname{vec}(\boldsymbol{A}) = (\boldsymbol{a}_1^\top, \boldsymbol{a}_2^\top, \dots, \boldsymbol{a}_n^\top)^\top.$$

Consider a constrained minimization problem over a convex set $\mathcal{A}$:

$$\min F(\boldsymbol{\alpha}), \text{ subject to } \boldsymbol{\alpha} \in \mathcal{A},$$

where F is a continuously differentiable and strongly convex function. For $t = 0, 1, 2, \dots$, the PGD algorithm updates

$$\boldsymbol{\alpha}^{(t+1)} = \operatorname{proj}_{\mathcal{A}} \left(\boldsymbol{\alpha}^{(t)} - d_t \nabla F(\boldsymbol{\alpha}^{(t)}) \right)$$
$$\equiv \operatorname*{argmin}_{\boldsymbol{\alpha} \in \mathcal{A}} \left\| \boldsymbol{\alpha} - \left(\boldsymbol{\alpha}^{(t)} - d_t \nabla F(\boldsymbol{\alpha}^{(t)}) \right) \right\|^2, \quad (4.1)$$

where d_t is a step size that we shall determine. If the algorithm converges to $\boldsymbol{\alpha}^\star$ such that

$$\boldsymbol{\alpha}^\star = \operatorname{proj}_{\mathcal{A}} \left(\boldsymbol{\alpha}^\star - d_t \nabla F(\boldsymbol{\alpha}^\star) \right),$$

then one can observe that $\boldsymbol{\alpha}^\star \in \mathcal{A}$ and $\nabla F(\boldsymbol{\alpha}^\star) = \boldsymbol{0}$ by differentiating Equation (4.1) in terms of $\boldsymbol{\alpha}$. Hence, $\boldsymbol{\alpha}^\star$ is a global minimizer of F on the set $\mathcal{A}$.

We next apply the PGD algorithm to solve the optimization problem (3.9). Suppose $\boldsymbol{A}$ is an $n \times k$ matrix whose jth column is $\boldsymbol{\alpha}_j$, then $\boldsymbol{A}\boldsymbol{e}_j = \boldsymbol{\alpha}_j$, where $\boldsymbol{e}_j$ a k-vector whose elements are 0 except that the jth element is 1. We observe that the constraint $\sum \boldsymbol{\alpha}_j = 0$ in problem (3.9) amounts to $\boldsymbol{A}\boldsymbol{1}_k = \boldsymbol{0}$, where $\boldsymbol{1}_k$ is the k-vector of 1's. Let $\boldsymbol{\alpha} = \operatorname{vec}(\boldsymbol{A})$, we have

$$\boldsymbol{A}\boldsymbol{e}_j = \operatorname{vec}(\boldsymbol{A}\boldsymbol{e}_j) = (\boldsymbol{e}_j^\top \otimes \boldsymbol{I}_n)\boldsymbol{\alpha},$$
$$\boldsymbol{A}\boldsymbol{1}_k = \operatorname{vec}(\boldsymbol{A}\boldsymbol{1}_k) = (\boldsymbol{1}_k^\top \otimes \boldsymbol{I}_n)\boldsymbol{\alpha}.$$

Accordingly, the optimization problem (3.9) can be written as

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^{nk}} F(\boldsymbol{\alpha}) = \min_{\boldsymbol{\alpha} \in \mathbb{R}^{nk}} \left[\frac{1}{n} \sum_{i=1}^{n} \phi \left\{ \boldsymbol{K}_i^\top (\boldsymbol{e}_{y_i}^\top \otimes \boldsymbol{I}_n)\boldsymbol{\alpha} \right\} \right.$$
$$\left. + \lambda \sum_{j=1}^{k} \boldsymbol{\alpha}^\top (\boldsymbol{e}_j \otimes \boldsymbol{I}_n) \boldsymbol{K} (\boldsymbol{e}_j^\top \otimes \boldsymbol{I}_n)\boldsymbol{\alpha} \right], \quad (4.2)$$
$$\text{subject to } (\boldsymbol{1}_k^\top \otimes \boldsymbol{I}_n)\boldsymbol{\alpha} = \boldsymbol{0}.$$

Let $\boldsymbol{B} = \boldsymbol{1}_k^\top \otimes \boldsymbol{I}_n$. By the PGD algorithm introduced in Equation (4.1), problem (4.2) can be solved as

$$\boldsymbol{\alpha}^{(t+1)} = \operatorname*{argmin}_{\{\boldsymbol{\alpha}: \boldsymbol{B}\boldsymbol{\alpha} = \boldsymbol{0}\}} \|\boldsymbol{\alpha} - \tilde{\boldsymbol{\alpha}}\|^2$$
$$= \tilde{\boldsymbol{\alpha}} - \boldsymbol{B}^\top (\boldsymbol{B}\boldsymbol{B}^\top)^{-1} \boldsymbol{B}\tilde{\boldsymbol{\alpha}} = \tilde{\boldsymbol{\alpha}} - ((\boldsymbol{1}_k \boldsymbol{1}_k^\top) \otimes \boldsymbol{I}_n)\tilde{\boldsymbol{\alpha}},$$

Algorithm 1 Projected Gradient Descent Algorithm for Multicategory Kernel DWD (4.2)

Initialize $\boldsymbol{\alpha}^{(0)}$, step size $d = 1$, $\eta = 0.5$, $\boldsymbol{B} = \mathbf{1}_k^\top \otimes \boldsymbol{I}_n$, and $t = 0$
repeat
  Compute $\nabla F(\boldsymbol{\alpha}^{(t)})$ as in (4.3)
  repeat
    Set $d = d\eta$
    Compute $\tilde{\boldsymbol{\alpha}} = \boldsymbol{\alpha}^{(t)} - d\nabla F(\boldsymbol{\alpha}^{(t)})$ and $\boldsymbol{\alpha}^+ = \tilde{\boldsymbol{\alpha}} - \boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}\tilde{\boldsymbol{\alpha}}$
  until the condition (4.4) is satisfied
  Set $\boldsymbol{\alpha}^{(t+1)} = \boldsymbol{\alpha}^+$
  Set $t = t + 1$
until a convergence condition is met

where $\tilde{\boldsymbol{\alpha}} = \boldsymbol{\alpha}^{(t)} - d_t\nabla F(\boldsymbol{\alpha}^{(t)})$, and

$$\nabla F(\boldsymbol{\alpha}^{(t)}) = \frac{1}{n}\sum_{i=1}^{n}\phi'\left\{\boldsymbol{K}_i^\top(\boldsymbol{e}_{y_i}^\top \otimes \boldsymbol{I}_n)\boldsymbol{\alpha}^{(t)}\right\}(\boldsymbol{e}_{y_i} \otimes \boldsymbol{I}_n)\boldsymbol{K}_i + 2\lambda(\boldsymbol{I}_k \otimes \boldsymbol{K})\boldsymbol{\alpha}^{(t)}. \tag{4.3}$$

We use a linear search method to determine the step size d_t. Specifically, at each iteration t, with a predefined constant $\eta < 1$, we find the smallest nonnegative integer b such that $d_t = \eta^b d_{t-1}$ and

$$F(\boldsymbol{\alpha}^+) \le F(\boldsymbol{\alpha}^{(t)}) + \nabla F(\boldsymbol{\alpha}^{(t)})(\boldsymbol{\alpha}^+ - \boldsymbol{\alpha}^{(t)}) + \frac{1}{2d_t}||\boldsymbol{\alpha}^+ - \boldsymbol{\alpha}^{(t)}||_2^2, \tag{4.4}$$

where $\tilde{\boldsymbol{\alpha}} = \boldsymbol{\alpha}^{(t)} - d_t\nabla F(\boldsymbol{\alpha}^{(t)})$ and $\boldsymbol{\alpha}^+ = \tilde{\boldsymbol{\alpha}} - \boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}\tilde{\boldsymbol{\alpha}}$. Then, we set $F(\boldsymbol{\alpha}^{(t+1)}) = F(\boldsymbol{\alpha}^+)$.

Algorithm 1 summarizes the details of the PGD algorithm. The rate of convergence is given in Proposition 1.

Proposition 1. Let $\boldsymbol{\alpha}^{(t)}$ be the sequence generated by Algorithm 1 and $\boldsymbol{\alpha}^\star$ is the global minimizer of problem (4.2). Then for any $t \ge 1$,

$$F(\boldsymbol{\alpha}^{(t)}) - F(\boldsymbol{\alpha}^\star) \le \frac{c_1}{t}\|\boldsymbol{\alpha}^{(0)} - \boldsymbol{\alpha}^\star\|^2,$$

in which $c_1 = 2\eta\tilde{\sigma}/n + \eta\lambda\sigma$, $\tilde{\sigma} = \max_j \tilde{\sigma}_j$ where each $\tilde{\sigma}_j$ is the largest eigenvalue of $\sum_{\{i:y_i=j\}} \boldsymbol{K}_i\boldsymbol{K}_i^\top$, and σ is the largest eigenvalue of $\boldsymbol{K}$.

Proposition 1 implies that $\mathcal{O}(1/\varepsilon)$ iterations are needed to reach $F(\boldsymbol{\alpha}^{(t)}) - F(\boldsymbol{\alpha}^\star) < \varepsilon$.

4.2. PGD Algorithm With the Nesterov's Acceleration

In this section, we further develop an accelerated PGD algorithm by employing the Nesterov's acceleration (Beck and Teboulle 2009; Nesterov 2013). The improved algorithm has a much faster rate of converge.

The accelerated PGD algorithm generates a number sequence δ_t such that

$$\delta_{t+1} = \frac{1 + \sqrt{1 + 4\delta_t^2}}{2},$$

Algorithm 2 Accelerated Projected Gradient Descent for Multicategory Kernel DWD (4.2)

Initialize $\boldsymbol{\alpha}^{(0)} = \boldsymbol{\beta}^{(1)}$, step size $d = 1$, $\eta = 0.5$, $\boldsymbol{B} = \mathbf{1}_k^\top \otimes \boldsymbol{I}_n$, $\delta_1 = 1$, and $t = 1$
repeat
  Compute $\nabla F(\boldsymbol{\beta}^{(t)})$ as in (4.3)
  repeat
    Set $d = d\eta$
    Compute $\tilde{\boldsymbol{\alpha}} = \boldsymbol{\beta}^{(t)} - d\nabla F(\boldsymbol{\beta}^{(t)})$ and $\boldsymbol{\alpha}^+ = \tilde{\boldsymbol{\beta}} - \boldsymbol{B}^\top(\boldsymbol{B}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}\tilde{\boldsymbol{\beta}}$
  until the condition (4.4) is satisfied
  Set $\boldsymbol{\alpha}^{(t)} = \boldsymbol{\alpha}^+$
  Compute $\delta_{t+1} = \frac{1+\sqrt{1+4\delta_t^2}}{2}$.
  Set $\boldsymbol{\beta}^{(t+1)} = \boldsymbol{\alpha}^{(t)} + \frac{\delta_t - 1}{\delta_{t+1}}(\boldsymbol{\alpha}^{(t)} - \boldsymbol{\alpha}^{(t-1)})$
  Set $t = t + 1$
until a convergence condition is met

and a sequence of $\boldsymbol{\beta}^{(t)}$ along with $\alpha^{(t)}$. With $\boldsymbol{\beta}^{(0)} = \boldsymbol{\alpha}^{(0)}$ initialized and $\delta_1 = 1$, the algorithm updates

$$\boldsymbol{\alpha}^{(t+1)} = \underset{\{\boldsymbol{\alpha}:\boldsymbol{B}\boldsymbol{\alpha}=\mathbf{0}\}}{\operatorname{argmin}} \left\|\boldsymbol{\alpha} - \left(\boldsymbol{\beta}^{(t)} - d_t\nabla F(\boldsymbol{\beta}^{(t)})\right)\right\|^2,$$

$$\boldsymbol{\beta}^{(t+1)} = \boldsymbol{\alpha}^{(t+1)} + \frac{\delta_t - 1}{\delta_{t+1}}(\boldsymbol{\alpha}^{(t)} - \boldsymbol{\alpha}^{(t-1)}).$$

The accelerated PGD algorithm is summarized in Algorithm 2. The rate of convergence is $\mathcal{O}(1/t^2)$, as presented in Proposition 2.

Proposition 2. Let $\boldsymbol{\alpha}^{(t)}$ be the sequence generated by Algorithm 2 and $\boldsymbol{\alpha}^\star$ is the global minimizer of problem (4.2). Then for any $t \ge 1$,

$$F(\boldsymbol{\alpha}^{(t)}) - F(\boldsymbol{\alpha}^\star) \le \frac{c_2}{t^2}\|\boldsymbol{\alpha}^{(0)} - \boldsymbol{\alpha}^\star\|^2,$$

in which $c_2 = 8\eta\tilde{\sigma}/n + 4\eta\lambda\sigma$, $\tilde{\sigma} = \max_j \tilde{\sigma}_j$, where each $\tilde{\sigma}_j$ is the largest eigenvalue of $\sum_{\{i:y_i=j\}} \boldsymbol{K}_i\boldsymbol{K}_i^\top$, and σ is the largest eigenvalue of $\boldsymbol{K}$.

Proposition 2 implies that, by the Nesterov's acceleration, $\mathcal{O}(1/\sqrt{\varepsilon})$ iterations are needed to reach $F(\boldsymbol{\alpha}^{(t)}) - F(\boldsymbol{\alpha}^\star) < \varepsilon$.

4.3. Implementation

We have implemented the accelerated PGD algorithm for solving multicategory kernel DWD in an R package `mdwd`. Users can choose the kernel function and use cross-validation to select the regularization parameter λ.

5. Numerical Studies

In this section, we use simulations and benchmark data applications to compare our multicategory kernel DWD with the multicategory DWD proposed by Huang et al. (2013) and implemented in the R package `DWD` (Huang et al. 2012). We also compare our method with off-the-shelf multicategory classifiers in R: multiclass kernel SVM in the R package `SMSVM` (Lee et al. 2004), reinforced angle-based multiclass SVM in the R package

RAMSVM (Zhang et al. 2016), random forest in the R package randomForest (Liaw and Wiener 2002), gradient boosting machines in the R package gbm (Ridgeway 2017), and k-nearest neighbors in the R package class (Venables and Ripley 2002). The Gaussian kernels are employed for mdwd, SMSVM, and RAMSVM, and the tuning parameters are estimated by 5-fold cross-validations. For the k-nearest neighbors, k is chosen from $\{3, 4, \ldots, 9\}$.

5.1. Simulations

We followed the simulation setting that was used in Section 1. Example 1 is a three-category classification, that is, $k = 3$, and the dimension $p = 5$. For each class $j = 1, 2, 3$, we independently generated three centers $\boldsymbol{\mu}_{j,1}, \boldsymbol{\mu}_{j,2}$, and $\boldsymbol{\mu}_{j,3}$ following $N(\mathbf{0}, \boldsymbol{I}_{3\times 3})$. We generated each data point $(\boldsymbol{x}_i, y_i)$ by first assigning a class label to y_i with equal conditional class probabilities and then randomly having a center $\boldsymbol{\mu}_{y_i,l}$, where $l = 1, 2, 3$ to draw $\boldsymbol{x}_i$ from $N(\boldsymbol{\mu}_{y_i,l}, \tau^2 \boldsymbol{I}_{p\times p})$. In Example 1, we set $\tau = 0.5$ and the corresponding Bayes error rate is 6.39%. We assembled the training data with the sample size varying over 100, 200, 400, 600, and 800. We trained and tuned each method on the training data, and we investigated the prediction error on a test set consisting of 10,000 independently generated observations.

Table 1 exhibits the mean misclassification rates and the standard errors, averaged by 100 replicates. For Example 1, we observe that multicategory kernel DWD delivers the least prediction error among the seven multicategory classifiers. The SVM and k-nearest neighbors have slightly worse performance than our proposal. We discover that the prediction error of our method decreases and also approaches the Bayes error as the sample size increases, which indicates that multicategory kernel DWD loss has a right target function and is thus, Fisher consistent. Multicategory linear DWD does not work well and is far from the Bayes rate.

Examples 2–4 adopt the same simulation settings as Example 1, except that Example 2 sets $k = 3$, $p = 20$, and $\tau = 1.5$ yielding the Bayes error of 9.39%, Example 3 has $k = 3$, $p = 2$, $\tau = 0.4$, and the Bayes error of 25.27%, and Example 4 contains four categories in the response, $k = 4$, and it has the Bayes error of 8.78%. As seen in Table 1, multicategory kernel DWD has the best prediction accuracy in all those three examples, two variants of SVM have slightly worse but very competitive behaviors, whereas multicategory linear DWD has the worst accuracy in general.

In Example 5, we generated each class center $\boldsymbol{\mu}_y$ from the standard normal distribution, and we then drew each data point from $N(\boldsymbol{\mu}_{y_i}, 2^2 \boldsymbol{I}_{20\times 20})$. The Bayes error is 11.45%, and the true decision boundaries between the classes are actually linear. From Table 1, we see that the classification error of multicategory linear DWD is the lowest and approaches the Bayes error. The performance of our proposal and the angle-based method follows intimately as well.

Table 1. Prediction error (%) on mixture Gaussian simulation examples.

	Prediction error (%) for the following methods:													
n	DWD WZ		DWD HLD		Multiclass SVM		Angle-based MSVM		Random forest		Gradient boosting		k-nearest neighbors	
Example 1: $k = 3, p = 5$, Bayes error: 6.39														
100	*10.06*	(0.48)	25.60	(0.94)	11.68	(0.52)	12.00	(0.52)	14.08	(0.51)	18.86	(0.58)	11.34	(0.54)
200	*8.92*	(0.44)	24.79	(0.95)	9.62	(0.45)	9.93	(0.48)	11.35	(0.45)	16.58	(0.56)	9.38	(0.48)
400	*7.88*	(0.41)	24.32	(0.92)	8.46	(0.41)	8.99	(0.44)	9.75	(0.41)	15.37	(0.54)	8.51	(0.45)
600	*7.59*	(0.39)	24.27	(0.92)	8.05	(0.39)	8.87	(0.43)	9.08	(0.40)	15.01	(0.53)	8.24	(0.43)
800	*7.38*	(0.38)	24.21	(0.93)	7.78	(0.39)	8.76	(0.43)	8.69	(0.40)	14.80	(0.53)	7.94	(0.41)
Example 2: $k = 3, p = 20$, Bayes error: 9.39														
100	*22.40*	(0.40)	27.30	(0.44)	23.07	(0.38)	24.51	(0.42)	27.58	(0.40)	30.31	(0.41)	26.26	(0.43)
200	*18.08*	(0.32)	24.78	(0.40)	18.48	(0.29)	20.41	(0.37)	23.18	(0.31)	27.03	(0.36)	21.44	(0.38)
400	*14.91*	(0.28)	23.33	(0.40)	15.42	(0.26)	17.61	(0.33)	19.95	(0.30)	24.95	(0.33)	18.16	(0.34)
600	*13.82*	(0.27)	22.77	(0.38)	14.30	(0.26)	16.41	(0.30)	18.61	(0.28)	24.30	(0.33)	16.92	(0.32)
800	*13.09*	(0.25)	22.50	(0.38)	13.66	(0.24)	16.04	(0.32)	17.84	(0.27)	23.97	(0.32)	16.01	(0.30)
Example 3: $k = 3, p = 2$, Bayes error: 25.27														
100	*29.82*	(0.80)	42.24	(0.96)	30.18	(0.77)	30.93	(0.78)	31.96	(0.79)	33.80	(0.76)	30.87	(0.83)
200	*27.97*	(0.78)	41.75	(0.98)	28.32	(0.76)	29.23	(0.78)	30.52	(0.81)	31.53	(0.77)	29.29	(0.81)
400	*26.97*	(0.74)	41.46	(0.97)	27.12	(0.74)	27.90	(0.75)	29.58	(0.78)	30.63	(0.75)	28.36	(0.78)
600	*26.76*	(0.74)	41.49	(0.96)	26.81	(0.74)	27.80	(0.73)	29.45	(0.80)	30.29	(0.75)	28.28	(0.80)
800	*26.55*	(0.73)	41.41	(0.94)	26.61	(0.72)	27.67	(0.73)	29.32	(0.79)	29.99	(0.72)	28.21	(0.78)
Example 4: $k = 4, p = 5$, Bayes error: 8.78														
100	*15.04*	(0.52)	34.83	(0.91)	17.38	(0.55)	17.31	(0.52)	19.98	(0.52)	27.27	(0.61)	16.84	(0.54)
200	*12.16*	(0.44)	33.62	(0.89)	13.65	(0.49)	14.41	(0.50)	15.62	(0.46)	23.78	(0.56)	13.21	(0.48)
400	*10.86*	(0.39)	33.33	(0.92)	11.86	(0.43)	12.99	(0.46)	13.47	(0.40)	22.26	(0.54)	11.67	(0.42)
600	*10.50*	(0.39)	33.32	(0.91)	11.21	(0.40)	12.69	(0.43)	12.55	(0.39)	21.77	(0.52)	11.33	(0.42)
800	*10.16*	(0.38)	33.29	(0.91)	10.72	(0.38)	12.44	(0.44)	11.95	(0.39)	21.50	(0.54)	10.80	(0.40)
Example 5: $k = 3, p = 20$, Bayes error: 11.45 (linear decision boundary)														
100	16.39	(0.38)	*15.30*	(0.36)	17.33	(0.36)	16.82	(0.40)	19.34	(0.38)	20.11	(0.36)	22.71	(0.47)
200	14.18	(0.36)	*13.39*	(0.32)	14.95	(0.33)	14.07	(0.35)	17.13	(0.36)	17.50	(0.34)	19.85	(0.45)
400	13.18	(0.32)	*12.54*	(0.30)	13.74	(0.31)	12.91	(0.31)	15.60	(0.32)	16.04	(0.31)	17.92	(0.41)
600	12.77	(0.32)	*12.20*	(0.30)	13.39	(0.31)	12.59	(0.31)	15.11	(0.32)	15.71	(0.31)	17.19	(0.40)
800	12.54	(0.31)	*12.07*	(0.29)	12.98	(0.30)	12.34	(0.30)	14.90	(0.31)	15.60	(0.30)	16.81	(0.39)

NOTES: Our multicategory kernel DWD (denoted by WZ) are compared with DWD (denoted by HLD, Huang et al. 2013), multiclass kernel SVM (R package SMSVM), reinforcement angle-based multiclass SVM (R package ramsvm), random forest (R package randomForest), gradient boosting machines (R package gbm), and k-nearest neighbors (R package class). The results are averaged by 100 independent runs, and the standard error of the mean prediction error is given in parentheses. For each case, the method incurring the lowest error is marked by italics.

Table 2. Prediction error (%) on benchmark data applications.

n_{train}	DWD WZ		DWD HLD		Multiclass SVM		Angle-based MSVM		Random forest		Gradient boosting		k-nearest neighbors	
	Prediction error (%) for the following methods:													
abalone: $k = 3, p = 9, N = 4177, n_{\text{test}} = 3377$														
100	40.05	(0.36)	*39.55*	(0.21)	39.69	(0.35)	40.20	(0.37)	40.78	(0.29)	39.74	(0.26)	42.84	(0.26)
200	37.88	(0.18)	38.20	(0.17)	*37.80*	(0.21)	38.15	(0.30)	39.00	(0.19)	38.34	(0.20)	41.93	(0.28)
300	*37.02*	(0.21)	37.65	(0.16)	37.20	(0.20)	37.52	(0.32)	38.16	(0.15)	37.84	(0.18)	40.93	(0.19)
400	*36.35*	(0.18)	37.35	(0.14)	36.63	(0.21)	36.61	(0.25)	37.52	(0.15)	37.53	(0.19)	40.14	(0.21)
600	35.96	(0.11)	37.38	(0.12)	*35.64*	(0.13)	36.09	(0.19)	36.98	(0.12)	37.24	(0.15)	40.00	(0.15)
800	35.33	(0.11)	36.91	(0.10)	*35.21*	(0.09)	35.73	(0.12)	36.58	(0.09)	36.76	(0.10)	39.21	(0.16)
covtype: $k = 3, p = 10, N = 73,631, n_{\text{test}} = 72,831$														
100	25.25	(0.26)	28.09	(0.33)	23.21	(0.19)	23.69	(0.26)	*21.91*	(0.21)	22.83	(0.27)	27.90	(0.23)
200	22.87	(0.21)	27.09	(0.28)	22.62	(0.21)	22.00	(0.23)	*20.41*	(0.19)	21.62	(0.16)	25.23	(0.16)
300	21.33	(0.15)	27.00	(0.24)	21.44	(0.22)	20.88	(0.20)	*18.90*	(0.12)	21.00	(0.15)	23.87	(0.19)
400	20.61	(0.10)	26.55	(0.19)	20.44	(0.19)	20.31	(0.16)	*18.32*	(0.10)	20.56	(0.12)	22.80	(0.12)
600	19.44	(0.11)	26.48	(0.15)	19.52	(0.20)	19.94	(0.13)	*16.92*	(0.08)	20.12	(0.12)	21.58	(0.13)
800	18.50	(0.10)	26.13	(0.15)	18.79	(0.14)	19.73	(0.12)	*16.22*	(0.09)	19.92	(0.08)	20.67	(0.09)
pendigits: $k = 10, p = 16, N = 10990, n_{\text{test}} = 10190$														
100	10.03	(0.21)	21.02	(0.35)	11.41	(0.28)	*9.69*	(0.26)	13.53	(0.23)	25.18	(0.41)	13.91	(0.30)
200	6.48	(0.15)	19.47	(0.29)	7.45	(0.23)	*5.80*	(0.18)	8.59	(0.16)	18.66	(0.20)	9.08	(0.16)
300	4.84	(0.11)	*	(*)	5.14	(0.15)	*4.41*	(0.11)	6.49	(0.12)	16.52	(0.21)	6.69	(0.13)
400	*3.91*	(0.07)	*	(*)	4.00	(0.15)	4.00	(0.09)	5.29	(0.10)	14.92	(0.16)	5.47	(0.09)
600	*3.05*	(0.07)	*	(*)	*	(*)	3.19	(0.08)	4.17	(0.08)	13.86	(0.13)	3.97	(0.06)
800	*2.44*	(0.05)	*	(*)	*	(*)	2.91	(0.07)	3.45	(0.07)	13.20	(0.12)	3.21	(0.06)
satimage: $k = 6, p = 36, N = 6435, n_{\text{test}} = 5835$														
100	*17.12*	(0.17)	20.44	(0.24)	18.48	(0.10)	18.34	(0.23)	17.29	(0.11)	19.87	(0.21)	19.05	(0.24)
200	*14.88*	(0.12)	20.22	(0.17)	17.21	(0.11)	16.54	(0.18)	15.11	(0.15)	17.57	(0.18)	16.67	(0.11)
300	*13.93*	(0.10)	20.14	(0.14)	16.67	(0.11)	16.17	(0.17)	14.05	(0.11)	16.91	(0.12)	15.60	(0.12)
400	13.27	(0.09)	20.27	(0.13)	16.15	(0.08)	15.69	(0.15)	*12.97*	(0.10)	16.09	(0.10)	14.76	(0.11)
600	12.27	(0.10)	*	(*)	15.22	(0.11)	14.97	(0.11)	*12.13*	(0.08)	15.57	(0.10)	13.97	(0.11)
800	11.91	(0.08)	*	(*)	14.47	(0.12)	14.61	(0.11)	*11.69*	(0.07)	15.47	(0.08)	13.13	(0.09)

NOTES: Our multicategory kernel DWD (denoted by WZ) are compared with DWD (denoted by HLD, Huang et al. 2013), multiclass kernel SVM (R package `SMSVM`), reinforcement angle-based multiclass SVM (R package `ramsvm`), random forest (R package `randomForest`), gradient boosting machines (R package `gbm`), and k-nearest neighbors (R package `class`). The results are averaged by 40 independent runs, and the standard error of the mean prediction error is given in parentheses. For each dataset, k and p are the number of classes and dimensions, and the total sample size is N. For each case, the method incurring the lowest error is marked by italics. Cases are marked as "*" when the algorithm did not converge within 10 hours.

To sum up, the simulation examples have clearly conveyed the following messages:

- the proposed multicategory kernel DWD works much better than the multicategory linear DWD when the underlying Bayes rule is nonlinear;
- the proposed multicategory kernel DWD delivers lower classification error that approaches the Bayes error when the training size increases;
- the proposed multicategory kernel DWD has very competitive performance against other popular off-the-shelf multiclass classifiers, although none dominates the rest.

5.2. Benchmark Data Applications

We examined the performance of multicategory kernel DWD on eight benchmark datasets that were downloaded at University of California at Irvine Machine Learning Repository (Dua and Karra Taniskidou 2017). We compared our method with the same methods that were used in Section 5.1. In the tables of this section, the total sample size of each dataset, the number of categories in the response, and the dimension are denoted by N, k, and p, respectively. For each dataset, we held out $n_{\text{test}} = N - 800$ observations as test data, and we randomly selected n_{train} observations as training data to train and tune each method, where the sample size of the training data n_{train} varied over 100, 200, 300, 400, 600, and 800. Averaged over 40 independent random splits, the misclassification error and computation time are summarized in Tables 2–4. The computation time in Table 4 includes fitting the models and tuning the parameters. We ranked the error and time of each method in Tables 5 and 6.

From the prediction error in Tables 2 and 3 and the ranks in Table 5, we find that our proposal has the lowest prediction error on two benchmark data, `vowel` and `waveform`, as well as the second lowest error on three examples, `abalone`, `pendigits`, and `satimage`. Multicategory linear DWD suffers from the worst accuracy on five datasets so it appears inadequate for these real applications. From Table 5, we see that the overall performance of our proposal on these benchmark data outperforms the SVM and the angle-based approach but it is worse than random forest. In terms of computation time, Tables 4 and 6 show that our implementation `mdwd` is the fastest among the four large-margin classifiers. The algorithms implemented in the packages `SMSVM` and `DWD` did not even converge within ten hours in several cases. We discover that random forest and gradient boosting are faster than these large-margin classifiers. We further notice the computation time of our proposal is relatively unaffected when there are many categories in the response, for example, the pendigits and vowel data; nonetheless, the computing speed of other large-margin classifiers is dramatically degraded as the number k increases. Although the prediction accuracy of k-nearest neighbors is among the worse, it runs the fastest in all examples.

According to the performance on the eight benchmark data, it is clear that our multicategory kernel DWD is much better

Table 3. Prediction error (%) on benchmark data applications.

	Prediction error (%) for the following methods:													
n_{train}	DWD WZ		DWD HLD		Multiclass SVM		Angle-based MSVM		Random forest		Gradient boosting		k-nearest neighbors	
`segmentation`: $k = 7, p = 19, N = 2310, n_{\text{test}} = 1510$														
100	18.00	(0.30)	23.09	(0.31)	17.93	(0.31)	17.34	(0.38)	*13.60*	(0.32)	16.10	(0.41)	21.89	(0.34)
200	14.04	(0.26)	21.50	(0.25)	14.04	(0.21)	14.67	(0.28)	*9.99*	(0.23)	12.15	(0.21)	15.92	(0.25)
300	12.46	(0.26)	21.30	(0.25)	11.33	(0.28)	13.21	(0.22)	*7.87*	(0.16)	11.13	(0.18)	12.70	(0.28)
400	11.15	(0.18)	21.01	(0.22)	9.72	(0.19)	12.54	(0.17)	*6.85*	(0.16)	10.57	(0.16)	11.27	(0.17)
600	9.47	(0.14)	*	(*)	7.77	(0.16)	11.85	(0.16)	*5.36*	(0.11)	9.79	(0.12)	8.82	(0.12)
800	8.44	(0.11)	*	(*)	*	(*)	11.26	(0.16)	*4.51*	(0.09)	9.38	(0.11)	7.64	(0.12)
`sensorless`: $k = 3, p = 48, N = 15957, n_{\text{test}} = 15157$														
100	0.99	(0.13)	0.16	(0.01)	0.51	(0.04)	0.60	(0.06)	*0.03*	(0.00)	0.21	(0.05)	12.73	(0.49)
200	0.31	(0.05)	0.08	(0.00)	0.31	(0.02)	0.33	(0.02)	*0.02*	(0.00)	0.09	(0.01)	6.26	(0.21)
300	0.17	(0.01)	0.06	(0.00)	0.27	(0.02)	0.33	(0.03)	*0.02*	(0.00)	0.08	(0.01)	3.88	(0.18)
400	0.14	(0.01)	0.06	(0.00)	0.25	(0.02)	0.28	(0.02)	*0.01*	(0.00)	0.08	(0.01)	2.87	(0.15)
600	0.12	(0.01)	0.06	(0.00)	0.22	(0.02)	0.20	(0.01)	*0.01*	(0.00)	0.08	(0.01)	1.84	(0.08)
800	0.10	(0.01)	0.05	(0.00)	0.19	(0.01)	0.19	(0.02)	*0.01*	(0.00)	0.07	(0.01)	1.29	(0.06)
`vowel`: $k = 11, p = 11, N = 990, n_{\text{test}} = 190$														
100	40.83	(0.69)	60.34	(0.63)	49.72	(0.68)	45.82	(0.89)	*36.46*	(0.75)	48.63	(0.72)	57.70	(0.73)
200	23.63	(0.53)	55.57	(0.61)	40.86	(0.73)	33.54	(0.64)	*21.82*	(0.61)	40.28	(0.58)	41.89	(0.61)
300	*14.93*	(0.54)	57.89	(0.12)	29.87	(0.74)	30.45	(0.63)	15.71	(0.51)	37.11	(0.60)	30.43	(0.52)
400	*9.05*	(0.38)	*	(*)	21.70	(0.68)	26.84	(0.62)	11.29	(0.48)	36.43	(0.56)	20.80	(0.51)
600	*4.03*	(0.28)	*	(*)	*	(*)	23.51	(0.44)	5.75	(0.31)	34.80	(0.50)	9.79	(0.45)
800	*2.03*	(0.17)	*	(*)	*	(*)	22.32	(0.50)	3.96	(0.27)	34.33	(0.46)	5.28	(0.25)
`waveform`: $k = 3, p = 40, N = 4999, n_{\text{test}} = 4199$														
100	*16.86*	(0.15)	18.74	(0.20)	22.55	(0.28)	22.44	(0.40)	18.74	(0.25)	18.83	(0.24)	28.37	(0.40)
200	*15.40*	(0.10)	16.46	(0.11)	19.33	(0.17)	18.75	(0.23)	16.91	(0.11)	17.06	(0.12)	25.39	(0.26)
300	*14.95*	(0.08)	15.66	(0.09)	17.60	(0.13)	17.91	(0.17)	16.29	(0.09)	16.26	(0.08)	24.28	(0.22)
400	*14.54*	(0.07)	15.12	(0.11)	16.60	(0.11)	17.35	(0.16)	15.84	(0.09)	15.97	(0.08)	23.82	(0.21)
600	*14.21*	(0.06)	14.77	(0.09)	15.62	(0.08)	16.54	(0.12)	15.42	(0.09)	15.54	(0.09)	22.69	(0.15)
800	*14.23*	(0.07)	14.72	(0.07)	15.21	(0.08)	16.24	(0.13)	15.26	(0.07)	15.56	(0.06)	22.35	(0.10)

NOTE: Our multicategory kernel DWD (denoted by WZ) are compared with DWD (denoted by HLD, Huang et al. 2013), multiclass kernel SVM (R package `SMSVM`), reinforcement angle-based multiclass SVM (R package `ramsvm`), random forest (R package `randomForest`), gradient boosting machines (R package `gbm`), and k-nearest neighbors (R package `class`). The results are averaged by 40 independent runs, and the standard error of the mean prediction error is given in parentheses. For each dataset, k and p are the number of classes and dimensions, and the total sample size is N. For each case, the method incurring the lowest error is marked by italics. Cases are marked as "*" when the algorithm did not converge within 10 hours.

Table 4. Mean computation time on benchmark data applications.

n_{train}	100	200	300	400	600	800	100	200	300	400	600	800
	`abalone`: $k = 3, p = 9, N = 4177, n_{\text{test}} = 3377$						`covtype`: $k = 3, p = 10, N = 73,631, n_{\text{test}} = 72,831$					
DWD (WZ)	3.9	15.2	42.9	100.3	462.1	967.3	0.8	3.5	7.7	13.9	63.9	154.6
DWD (HLD)	6.6	12.4	315.5	719.9	2603.9	6226.6	7.7	21.1	529.5	1111.1	3613.4	8670.1
Multiclass SVM	3.2	26.2	99.1	256.8	946.0	2342.2	5.3	23.1	79.1	204.2	771.5	1966.8
Angle-based SVM	4.3	10.2	19.8	37.9	80.2	175.6	43.1	50.6	62.8	83.4	135.2	245.8
Random forest	0.2	0.4	0.5	0.7	1.0	1.2	4.4	4.5	4.7	4.7	4.7	5.0
Gradient boosting	0.2	0.3	0.4	0.4	0.6	0.7	3.4	3.4	3.4	3.5	3.6	3.8
k-nearest neighbors	0.1	0.1	0.1	0.1	0.1	0.2	0.4	0.6	0.8	1.0	1.4	1.7
	`pendigits`: $k = 10, p = 16, N = 10990, n_{\text{test}} = 10190$						`satimage`: $k = 6, p = 36, N = 6435, n_{\text{test}} = 5835$					
DWD (WZ)	0.9	2.6	5.6	10.2	58.1	221.5	2.5	5.2	11.9	23.4	99.5	271.3
DWD (HLD)	1080.9	11,139.8	*	*	*	*	311.1	2512.5	8194.5	27,861.0	*	*
Multiclass SVM	98.8	1083.9	4482.8	12268.9	*	*	25.2	213.6	784.6	2009.3	7563.4	20,737.1
Angle-based SVM	75.9	270.4	595.7	1054.8	2358.3	4269.0	21.3	70.1	163.5	294.6	659.6	1236.0
Random forest	0.6	0.9	1.0	1.2	1.5	1.8	0.3	0.5	0.8	1.1	1.7	2.4
Gradient boosting	1.6	2.0	2.3	2.6	3.3	3.9	0.8	1.1	1.4	1.7	2.2	2.9
k-nearest neighbors	0.1	0.1	0.2	0.2	0.3	0.5	0.1	0.1	0.2	0.3	0.4	0.7
	`segmentation`: $k = 7, p = 19, N = 2310, n_{\text{test}} = 1510$						`sensorless`: $k = 3, p = 48, N = 15957, n_{\text{test}} = 15,157$					
DWD (WZ)	1.1	2.9	6.4	11.4	76.0	288.5	2.5	10.5	26.6	50.3	299.0	1085.7
DWD (HLD)	285.4	3734.8	12,850.7	26,370.3	*	*	6.9	17.6	632.7	1377.4	6591.7	14,212.0
Multiclass SVM	43.3	401.1	1606.8	4299.3	21,251.3	*	2.2	14.8	53.0	135.5	579.7	1489.1
Angle-based SVM	24.6	100.0	224.6	398.6	898.7	1672.2	9.7	15.6	24.4	38.7	79.5	164.3
Random forest	0.2	0.4	0.6	0.7	1.1	1.3	1.6	1.7	1.8	1.8	2.4	2.2
Gradient boosting	0.4	0.6	0.9	1.2	1.7	2.2	0.9	1.1	1.3	1.5	1.9	2.4
k-nearest neighbors	0.1	0.1	0.1	0.1	0.2	0.3	0.2	0.4	0.6	0.8	1.3	1.9

(*continued*)

Table 4. (Continued)

n_{train}	100	200	300	400	600	800	100	200	300	400	600	800
			vowel: $k = 11, p = 11, N = 990, n_{\text{test}} = 190$						waveform: $k = 3, p = 40, N = 4999, n_{\text{test}} = 4199$			
DWD (WZ)	1.6	4.9	10.7	19.0	61.9	143.0	2.7	12.3	32.0	60.6	335.3	959.1
DWD (HLD)	1458.4	13,342.3	26,280.3	*	*	*	7.8	19.3	626.7	1301.8	4039.3	9705.7
Multiclass SVM	102.7	1272.7	5416.9	17,436.9	*	*	1.8	10.9	39.5	93.8	338.4	848.9
Angle-based SVM	81.6	322.1	717.6	1279.4	2879.1	5214.6	4.6	10.6	20.0	37.2	78.6	170.1
Random forest	0.2	0.4	0.5	0.7	1.0	1.4	0.3	0.6	0.9	1.2	2.1	3.0
Gradient boosting	0.4	0.6	1.0	1.3	1.9	2.5	0.4	0.6	0.7	0.9	1.3	1.7
k-nearest neighbors	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.2	0.3	0.4	0.7

NOTES: We compare our multicategory kernel DWD (denoted by WZ) with DWD (denoted by HLD (Huang et al. 2013)), multiclass kernel SVM (R package `SMSVM`), reinforcement angle-based multiclass SVM (R package `ramsvm`), random forest (R package `randomForest`), gradient boosting machines (R package `gbm`), and k-nearest neighbors (R package `class`). The results are averaged by 40 independent runs. The computation time of each method includes the parameter tunes. Computations were conducted on a single-processor Intel(R) Xeon(R) central processor unit E5-2660 at 2.60 GHz. Cases when the algorithm did not converge within 10 hours are marked as "*."

Table 5. Rank of the prediction accuracy of each method on benchmark data applications.

	k	p	DWD WZ	DWD HLD	Multiclass SVM	Angle-based MSVM	Random forest	Gradient boosting	k-nearest neighbors
`abalone`	3	9	2	4	1	3	6	5	7
`covtype`	3	10	5	7	2	4	1	3	6
`pendigits`	10	16	2	7	3	1	4	6	5
`satimage`	6	36	2	7	5	4	1	6	3
`segmentation`	7	19	4	7	3	6	1	2	5
`sensorless`	3	48	5	2	4	6	1	3	7
`vowel`	11	11	1	7	5	4	2	6	3
`waveform`	3	40	1	2	5	6	3	4	7
overall			2	6	3	4	1	5	6

NOTES: For each dataset, the rank is given based on the average prediction accuracy of different methods on different training sizes: 100, 200, 300, 400, 600, 800. For each dataset, k and p are the number of classes and dimensions. For each method, the ranks of the prediction accuracy on different datasets are averaged to yield the overall rank.

Table 6. Rank of the computation time of each method on benchmark data applications.

	k	p	DWD WZ	DWD HLD	Multiclass SVM	Angle-based MSVM	Random forest	Gradient boosting	k-nearest neighbors
`abalone`	3	9	5	7	6	4	3	2	1
`covtype`	3	10	4	7	6	5	3	2	1
`pendigits`	10	16	4	6	7	5	2	3	1
`satimage`	6	36	4	7	6	5	2	3	1
`segmentation`	7	19	4	7	6	5	2	3	1
`sensorless`	3	48	5	7	6	4	3	2	1
`vowel`	11	11	4	7	6	5	2	3	1
`waveform`	3	40	6	7	5	4	3	2	1
overall			4	7	6	5	2	2	1

NOTES: For each dataset, the rank is given based on the average prediction accuracy of different methods on different training sizes: 100, 200, 300, 400, 600, 800. For each dataset, k and p are the number of classes and dimensions. For each method, the ranks of the computation time on different datasets are averaged to yield the overall rank.

than multicategory linear DWD by the `DWD` package and also highly competitive with other popular classifiers. The random forest has the overall best performance. We also implemented the polynomial kernel and found its performance is worse than that by the Gaussian kernel in these examples. For sake of space we do not include the result of DWD with polynomial kernel here.

6. Discussion

In this article, we have proposed a new multicategory kernel DWD for multiclass classification. Our method is able to capture potential highly nonlinear structure of the Bayes rule and hence, is more desirable than the restrictive multicategory linear DWD. We have derived an efficient accelerated PGD algorithm for fitting the multicategory kernel DWD. Extensive simulations and benchmark data examples have shown that the multicategory kernel DWD is a worthy competitor against the popular off-the-shelf multiclass classifiers, including the SVM, random forest, gradient boosting, and k-nearest neighbors. Therefore, we view the multicategory kernel DWD as a valuable addition to the classification toolbox for real applications.

The classification problem in this article has equal weights on different classes. In many applications, we may face the non-standard classification problems such as imbalanced class size or unequal cost. Nonstandard SVM and DWD have previously studied in Lee et al. (2004) and Qiao et al. (2010). In the future research, it will be interesting to generalize the proposal in this paper to the nonstandard case.

We have implemented our method in an R package `mdwd`. The URL link of the R package is

`http://users.stat.umn.edu/~zouxx019/ftpdir/code/mdwd/.`

Appendix A. Appendix

A.1. Proof of Theorem 1

For simplicity we write $p_j = p_j(\boldsymbol{x})$. Using the Lagrangian multiplier method, we define

$$L(\boldsymbol{f}) = p_1\phi(f_1) + \cdots + p_k\phi(f_k) + \mu(f_1 + \cdots + f_k).$$

Then for each $j = 1, \ldots, k$,

$$\partial L(f)/\partial f_j = \phi'(f_j)p_j + \mu = 0,$$

where $\phi'(f_j) = -1$ if $f_j \le 1/2$ or $\phi'(f_j) = -1/(4f_j^2)$ if $f_j > 1/2$. We notice that $-1 \le \phi' < 0$.

Without loss of generality, assume $p_1 > p_2 \ge p_3 \ge \cdots \ge p_{k-1} > p_k$. We observe that $\mu \le p_j$, $\forall j$. If assume that $\mu < p_k < p_j$, then for any $1 \le j \le k$, $\phi'(f_j) = -1/(4f_j^2)$, which gives that

$$f_j = \frac{1}{2}\sqrt{\frac{p_j}{\mu}} > 0, \tag{A.1}$$

contradicting the fact that $\sum_{j=1}^k f_j = 0$, so $\mu = p_k$ and $\mu < p_j$ for any $j < k$. Hence, by considering the constraint $\sum_{j=1}^k f_j = 0$ and using the same argument as obtaining the solution (A.1), we have that

$$f_j^\star = \begin{cases} \frac{1}{2}\sqrt{\frac{p_j}{p_k}}, & j < k, \\ -\frac{1}{2}\sum_{j=1}^{k-1}\sqrt{\frac{p_j}{p_k}}, & j = k. \end{cases} \tag{A.2}$$

A.2. Proof of Theorem 2

Consider any feasible solution $\boldsymbol{f}$. For each $j = 1, \ldots, k$, we write

$$f_j(\boldsymbol{x}) = \sum_{i=1}^n \alpha_{ij}K(\boldsymbol{x}_i, \boldsymbol{x}) + \rho_j(\boldsymbol{x}),$$

where ρ_j is orthogonal to the span of $\{K(\boldsymbol{x}_i, \boldsymbol{x})\}$. By the sum-to-zero constraint, we have

$$\sum_{j=1}^k \left[\sum_{i=1}^n \alpha_{ij}K(\boldsymbol{x}_i, \boldsymbol{x}) + \rho_j(\boldsymbol{x})\right] = 0,$$

or equivalently

$$\left[\sum_{i=1}^n \left(\sum_{j=1}^k \alpha_{ij}\right) K(\boldsymbol{x}_i, \boldsymbol{x})\right] + \left[\sum_{j=1}^k \rho_j(\boldsymbol{x})\right] = 0.$$

Since $\sum_{j=1}^k \rho_j(\boldsymbol{x})$ is orthogonal to the span of $\{K(\boldsymbol{x}_i, \boldsymbol{x})\}$ and K is a positive definite kernel, the above identity holds if and only if

$$\sum_{j=1}^k \alpha_{ij} = 0$$

and

$$\sum_{j=1}^k \rho_j(\boldsymbol{x}) = 0.$$

Define $g_j(\boldsymbol{x}) = \sum_{i=1}^n \alpha_{ij}K(\boldsymbol{x}_i, \boldsymbol{x})$. So we can write $f_j = g_j + \rho_j$ and $\sum_{j=1}^k g_j(\boldsymbol{x}) = 0$, which means $(g_1, \ldots, g_k)$ is another feasible solution.

By the orthogonality of ρ_j and g_j, we have $\|f_j\|_{\mathcal{H}_K}^2 = \|g_j\|_{\mathcal{H}_K}^2 + \|\rho_j\|_{\mathcal{H}_K}^2$. On the other hand, we show $g_j(\boldsymbol{x}_i) = f_j(\boldsymbol{x}_i)$ for all i. For that, we use the reproducing property (Wahba 1990) and have $\rho_j(\boldsymbol{x}_i) = \langle \rho_j(x), K(\boldsymbol{x}_i, \boldsymbol{x})\rangle_{\mathcal{H}_K} = 0$.

To sum up, for every feasible solution $\boldsymbol{f}$, we can find a better (or at least no worse) feasible solution $\boldsymbol{g}$ such that the two feasible solutions have the same empirical loss value but the latter also has a smaller penalty term. The two feasible solutions are identical if and only if $\rho_j = 0$ for all j. This completes the proof.

A.3. Proof of Proposition 1

Let ϕ be the DWD loss (3.1). We observe that for any $u_1 \ne u_2$,

$$|\phi'(u_1) - \phi'(u_2)| < 4|u_1 - u_2|.$$

Therefore, we see that

$$\begin{aligned} &\|\nabla F(\boldsymbol{\alpha}) - \nabla F(\boldsymbol{\alpha}')\| \\ \le &\left\| \frac{1}{n}\sum_{i=1}^n \left(\phi'\left\{\boldsymbol{K}_i^\top(\boldsymbol{e}_{y_i}^\top \otimes \boldsymbol{I}_n)\boldsymbol{\alpha}\right\} - \phi'\left\{\boldsymbol{K}_i^\top(\boldsymbol{e}_{y_i}^\top \otimes \boldsymbol{I}_n)\boldsymbol{\alpha}'\right\}\right)\right. \\ &\left. \times (\boldsymbol{e}_{y_i} \otimes \boldsymbol{I}_n)\boldsymbol{K}_i + 2\lambda(\boldsymbol{I}_k \otimes \boldsymbol{K})\left(\boldsymbol{\alpha} - \boldsymbol{\alpha}'\right)\right\| \qquad \text{(A.3)} \\ \le &\left\| \left(\frac{4}{n}\sum_{i=1}^n (\boldsymbol{e}_{y_i} \otimes \boldsymbol{I}_n)\boldsymbol{K}_i\boldsymbol{K}_i^\top(\boldsymbol{e}_{y_i}^\top \otimes \boldsymbol{I}_n) + 2\lambda(\boldsymbol{I}_k \otimes \boldsymbol{K})\right)\left(\boldsymbol{\alpha} - \boldsymbol{\alpha}'\right)\right\| \\ \le &L(F)\left\|\left(\boldsymbol{\alpha} - \boldsymbol{\alpha}'\right)\right\|, \end{aligned}$$

where $L(F)$ is the largest eigenvalue of the matrix

$$\boldsymbol{T} \equiv \frac{4}{n}\sum_{i=1}^n (\boldsymbol{e}_{y_i} \otimes \boldsymbol{I}_n)\boldsymbol{K}_i\boldsymbol{K}_i^\top(\boldsymbol{e}_{y_i}^\top \otimes \boldsymbol{I}_n) + 2\lambda(\boldsymbol{I}_k \otimes \boldsymbol{K}),$$

which is an $nk \times nk$ block diagonal matrix whose jth block is an $n \times n$ matrix

$$\boldsymbol{T}_j = \frac{4}{n}\sum_{\{i:y_i=j\}} \boldsymbol{K}_i\boldsymbol{K}_i^\top + 2\lambda\boldsymbol{K}.$$

The largest eigenvalue of $\boldsymbol{T}_j$ is $4\tilde{\sigma}_j/n + 2\lambda\sigma$, then $L(F)$, the largest eigenvalue of $\boldsymbol{T}$, is $4\tilde{\sigma}/n + 2\lambda\sigma$. The proposition is then proved by applying Theorem 3.1 of Beck and Teboulle (2009).

A.4. Proof of Proposition 2

The proposition follows inequality (A.3) and Theorem 4.4 of Beck and Teboulle (2009).

Acknowledgments

We thank the Editor Professor Daniel Apley, an Associate Editor, and three reviewers for their helpful comments that greatly improved this work.

Funding

Zou's research was partially supported by National Science Foundation grant DMS-1505111.

References

Allwein, E. L., Schapire, R. E., and Singer, Y. (2000), "Reducing Multiclass to Binary: A Unifying Approach for Margin Classifiers," *Journal of Machine Learning Research*, 1, 113–141. [396]

Barsoum, E., Zhang, C., Ferrer, C. C., and Zhang, Z. (2016), "Training Deep Networks for Facial Expression Recognition With Crowd-Sourced Label Distribution," in *Proceedings of the 18th ACM International Conference on Multimodal Interaction*, pp. 279–283. [396]

Beck A., and Teboulle, M. (2009), "A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems," *SIAM Journal on Imaging Sciences*, 2, 183–202. [401,406]

Bojarski, M., Yeres, P., Choromanska, A., Choromanski, K., Firner, B., Jackel, L., and Muller, U. (2017), "Explaining How a Deep Neural Network trained With End-to-End Learning Steers a Car," *https://arxiv.org/pdf/1704.07911*. [396]

Bredensteiner, E. J., and Bennett, K. P. (1999), "Multicategory Classification by Support Vector Machines," *Computational Optimization and Applications*, 12, 53–79. [399]

Chen, C., Seff, A., Kornhauser, A., and Xiao, J.(2015), "Deepdriving: Learning Affordance for Direct Perception in Autonomous Driving," in *IEEE International Conference on Computer Vision (ICCV)*, pp. 2722–2730. [396]

Chen, J., Wang, Y., Yoho, S.E., Wang, D., and Healy, E.W. (2016), "Large-Scale Training to Increase Speech Intelligibility for Hearing-Impaired Listeners in Novel Noises," *Journal of the Acoustical Society of America*, 139, 2604–2612. [396]

Crammer, K., and Singer, Y. (2001), "On the Algorithmic Implementation of Multiclass Kernel-Based Vector Machines," *Journal of Machine Learning Research*, 2, 265–292. [399]

Dietterich, T. G., and Bakiri, G. (1995), "Solving Multiclass Learning Problems via Error-Correcting Output Codes," *Journal of Artificial Intelligence Research*, 2, 263–286. [396]

Dong, L. and Shan, J. (2013), "A Comprehensive Review of Earthquake-Induced Building Damage Detection With Remote sensing Techniques," *ISPRS Journal of Photogrammetry and Remote Sensing*, 84, 85–99. [396]

Dua, D., and Karra Taniskidou, E. (2017), "UCI Machine Learning Repository," School of Information and Computer Science, University of California at Irvine, Irvine. *http://archive.ics.uci.edu/ml*. [403]

Fu, S., Zhang, S., and Liu, Y. (2018), "Adaptively Weighted Large-Margin Angle-Based Classifiers," *Journal of Multivariate Analysis*, 166, 282–299. [400]

Guermeur, Y. (2002), "Combining Discriminant Models With New Multi-Class SVMs," *Pattern Analysis & Applications*, 5, 168–179. [399]

Hansen, J. H., and Hasan, T. (2015), "Speaker Recognition by Machines and Humans: A Tutorial Review," *IEEE Signal Processing Magazine*, 32, 74–99. [396]

Haralick, R. M., and Shanmugam, K. (1973), "Textural Features for Image Classification," *IEEE Transactions on Systems, Man, and Cybernetics*, 6, 610–621. [396]

Hastie, T., and Tibshirani, R. (1998), "Classification by Pairwise Coupling," *Annals of Statistics*, 26, 451–471. [396]

Hsu, C., and Lin, C. (2002), "A Comparison of Methods for Multiclass Support Vector Machines," *IEEE Transactions on Neural Networks*, 13, 415–425. [396]

Huang, H., Liu, Y., Du. Y., Perou, C., Hayes, D., Todd, M., and Marron, J.S. (2013), "Multiclass Distance-Weighted Discrimination," *Journal of Computational and Graphical Statistics*, 22, 953–969. [396,397,398,399,401,402,403,404,405]

Huang, H., Lu, X., Liu, Y., Haaland, P., and Marron, J.S. (2012), "R/DWD: Distance-Weighted Discrimination for Classification, Visualization and Batch Adjustment," *Bioinformatics*, 28, 1182–1183. [401]

James, G., and Hastie, T. (1998), "The Error Coding Method and PICTs," *Journal of Computational and Graphical Statistics*, 7, 377–387. [396]

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012), "Imagenet Classification With Deep Convolutional Neural Networks," *Advances in Neural Information Processing Systems*, 25, 1097–1105. [396]

Lee, Y., Lin, Y., 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. [396,398,400,401,405]

Liaw, A., and Wiener, M. (2002), "Classification and Regression by randomForest," *R News*, 2, 18–22. [402]

Lillesand, T., Kiefer, R., and Chipman, J. (2014), *Remote Sensing and Image Interpretation* (2nd ed.), Hoboken, NJ: Wiley. [396]

Lin, Y. (2004), "A Note on Margin-Based Loss Functions in Classification," *Statistics & Probability Letters*, 68, 73–82. [399]

——— (2007), "Fisher Consistency of Multicategory Support Vector Machines," in *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, pp. 291–298. [396]

Liu, L., Liu, Y., and Zhu, H. (2018), "SMAC: Spatial Multi-Category Angle-Based Classifier for High-Dimensional Neuroimaging Data," *NeuroImage*, 175, 230–245. [400]

Liu, P., Han, S., Meng, Z., and Tong, Y. (2012), "Facial Expression Recognition via a Boosted Deep Belief Network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1805–1812. [396]

Liu, Y., and Yuan, M. (2011), "Reinforced Multicategory Support Vector Machines," *Journal of Computational and Graphical Statistics*, 20, 901–919. [400]

Liu, Y., and Shen, X. (2006), "Multicategory ψ-Learning," *Journal of the American Statistical Association*, 101, 500–509. [396,398,400]

Liu, Y., Zhang, H., and Wu, Y. (2011), "Hard or Soft Classification? Large-Margin Unified Machines," *Journal of the American Statistical Association*, 106, 166–177. [398]

Marron, J.S., Todd, M., and Ahn, J. (2007), "Distance Weighted Discrimination," *Journal of the American Statistical Association*, 102, 1267–1271. [396,397]

Maulik, U., and Chakraborty, D. (2017), "Remote Sensing Image Classification: A Survey of Support-Vector-Machine-Based Advanced Techniques," *IEEE Geoscience and Remote Sensing Magazine*, 5, 33–52. [396]

Nesterov, Y. (2013), "Gradient Methods for Minimizing Composite Functions," *Mathematical Programming*, 140, 125–161. [401]

Qiao, X., and Zhang, L. (2015a), "Distance-Weighted Support Vector Machine," *Statistics and Its Interface*, 8, 331–345. [398]

——— (2015b), "Flexible High-Dimensional Classification Machines and Their Asymptotic Properties," *Journal of Machine Learning Research*, 16, 1547–1572. [398]

Qiao, X., Zhang, H., Liu, Y., Todd, M., Marron, J.S. (2010), "Weighted Distance Weighted Discrimination and Its Asymptotic Properties," *Journal of the American Statistical Association*, 105, 401–414. [397,398,405]

Rabiner, L. (1989), "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proceedings of the IEEE*, 77, 257–286. [396]

Rabiner, L., and Juang, B. (1993), *Fundamentals of Speech Recognition*, Englewood Cliffs, NJ: Prentice Hall. [396]

Ridgeway, G. (2017), "gbm: Generalized Boosted Regression Models," R package 2.9.0. *https://cran.r-project.org/web/packages/gbm/index.html*. [402]

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., and Fei-Fei, L. (2015), "Imagenet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision*, 115, 211–252. [396]

Steinwart, I., Hush, D., and Scovel, C. (2006), "An Explicit Description of the Reproducing Kernel Hilbert Spaces of Gaussian RBF Kernels," *IEEE Transactions on Information Theory*, 52, 4635–4643. [398]

Sun, H., Craig, B., and Zhang, L. (2017), "Angle-Based Multicategory Distance-Weighted SVM," *Journal of Machine Learning Research*, 18, 1–21. [400]

Takashima, Y., Takiguchi, T., Ariki, Y., and Omori, K. (2017), "Audio-Visual Speech Recognition for a Person With Severe Hearing Loss Using Deep Canonical Correlation Analysis," in *Proceedings of the 1st Conference on Challenges in Hearing Assistive Technology*, Stockholm, Sweden. [396]

Vapnik, V. (1995), *The Nature of Statistical Learning Theory*, Berlin: Springer. [396]

——— (1998), *Statistical Learning Theory*, Chichester: Wiley. [396,399]

Venables, W., and Ripley, B. (2002), *Modern Applied Statistics With S* (4th ed.), Berlin: Springer. [402]

Wahba, G. (1990), *Spline Models for Observational Data*, Philadelphia, PA: Society for Industrial and Applied Mathematics. [398,406]

Wang, B., and Zou, H. (2016), "Sparse Distance Weighted Discrimination," *Journal of Computational and Graphical Statistics*, 25, 826–838. [398]

——— (2018), "Another Look at Distance-Weighted Discrimination," *Journal of Royal Statistical Society*, Series B, 80, 177–198. [397,398]

Wang, D. (2017), "Deep Learning Reinvents the Hearing Aid," *IEEE Spectrum*, 54, 32–37. [396]

Weston, J., and Watkins, C. (1999), "Support Vector Machines for Multi-Class Pattern Recognition," in *Proceedings of the Seventh European Symposium on Artificial Neural Networks*, pp. 219–224. [396,399]

Xu, H., Gao, Y., Yu, F., and Darrel, T. (2016), "End-to-End Learning of Driving Models from Large-Scale Video Datasets," *https://arxiv.org/abs/1612.01079*. [396]

Yu, D., and Deng, L. (2016), *Automatic Speech Recognition*, London: Springer. [396]

Zhang, C., and Liu, Y. (2013), "Multicategory Large-Margin Unified Machines," *Journal of Machine Learning Research*, 14, 1349–1386. [396]

——— (2014), "Multicategory Angle-Based Large-Margin Classification," *Biometrika*, 101, 625–640. [396,400]

Zhang, C., Liu, Y., Wang, J., and Shen, X. (2016), "Reinforced Angel-Based Multicategory Support Vector Machines," *Journal of Computational and Graphical Statistics*, 25, 806–825. [396,400,402]

Zhang, C., Pam, M., Fu, S., and Liu, Y. (2017), "Robust Multicategory Support Vector Machines Using Difference Convex Algorithm," *Mathematical Programming*, 169, 277–305. [400]

Zhu, J., and Hastie, T. (2005), "Kernel Logistic Regression and the Import Vector Machines," *Journal of Computational and Graphical Statistics*, 14, 185–205 [396]

Zou, H., Zhu, J., Hastie, T. (2008), "New Multicategory Boosting Algorithms Based on Multicategory Fisher-Consistent Losses," *Annals of Applied Statistics*, 2, 1290–1306. [397,398]