Published in final edited form as:

Stat Anal Data Min. 2010 August ; 3(4): 272–286. doi:10.1002/sam.10081.

Multicategory Composite Least Squares Classifiers

Seo Young Park[Ph.D Student]

Department of Statistics and Operations Research, University of North Carolina, Chapel Hill, NC 27599

Yufeng Liu[Associate Professor]

Department of Statistics and Operations Research, Carolina Center for Genome Sciences, University of North Carolina, Chapel Hill, NC 27599

Dacheng Liu and Paul Scholl*

Boehringer Ingelheim Pharmaceuticals, Inc.

Abstract

Classification is a very useful statistical tool for information extraction. In particular, multicategory classification is commonly seen in various applications. Although binary classification problems are heavily studied, extensions to the multicategory case are much less so. In view of the increased complexity and volume of modern statistical problems, it is desirable to have multicategory classifiers that are able to handle problems with high dimensions and with a large number of classes. Moreover, it is necessary to have sound theoretical properties for the multicategory classifiers. In the literature, there exist several different versions of simultaneous multicategory Support Vector Machines (SVMs). However, the computation of the SVM can be difficult for large scale problems, especially for problems with large number of classes. Furthermore, the SVM cannot produce class probability estimation directly. In this article, we propose a novel efficient multicategory composite least squares classifier (CLS classifier), which utilizes a new composite squared loss function. The proposed CLS classifier has several important merits: efficient computation for problems with large number of classes, asymptotic consistency, ability to handle high dimensional data, and simple conditional class probability estimation. Our simulated and real examples demonstrate competitive performance of the proposed approach.

Keywords

Fisher Consistency; Multicategory Classification; Regression; Squared loss; Support Vector Machine

1 Introduction

Classification is a very useful statistical tool for information extraction. There are numerous applications for classification in various fields, such as handwritten digit recognition, disease classification, and drug discovery. In the literature, many classification techniques are proposed, ranging from the traditional methods such as Fisher linear discriminant analysis (LDA)(Mardia et al., 1979) to modern machine learning techniques such as the celebrated Support Vector Machine (SVM) (Boser et al., 1992; Vapnik, 1998) and ensemble-based

* (paul.scholl@boehringer-ingelheim.com).
(seoyoung@uchicago.edu).
(yfliu@email.unc.edu).
(dacheng.liu@boehringer-ingelheim.com)

strategies, such as Random Forests (Breiman, 2001). Interested readers are referred to a comprehensive book on statistical learning by Hastie et al. (2003) as well as a book on the SVM by Cristianini and Shawe-Taylor (2000). Many interesting real examples can be found there.

Binary classification problems are heavily studied, while in contrast, the attentions on multiclass problems are much less so. To solve a multiclass problem, there are two major groups of approaches. One is to employ multiple binary classifiers and then combine the results. The one-versus-rest and one-versus-one approaches are common examples of this type. Although the extension is simple to implement, there are drawbacks with these approaches. The one-versus-one approach may not work well when the numbers of observations in some classes are small. For the one-versus-rest approach, a serious drawback is that the approach may not be consistent when there is no dominating class, i.e., when the maximum class conditional probability is less than 0.5. The other group of multiclass approaches is to use simultaneous multiclass formulations. For example, Vapnik (1998); Crammer and Singer (2001); Lee et al. (2004) proposed various SVM techniques for simultaneous multiclass classification. One limitation of these approaches is that the corresponding computational complexity grows very rapidly when the number of classes gets large.

Despite progress in multiclass classification, many challenges have yet to be solved. In particular, given the abundance of high-dimensional complex data, it is desirable to have multiclass classification techniques that are

- based on simultaneous formulation with sound theoretical properties;
- able to handle high dimensional data;
- efficient to compute even when the class number is large;
- able to estimate conditional class probabilities.

In this paper, we propose a novel simultaneous multiclass technique, namely the Multiclass Composite Least Squares (CLS) Classifier. The proposed CLS classifier possesses all four aforementioned properties. Motivated from multiclass SVMs, the CLS classifier is based on a simultaneous formulation by using all data at once to produce a multiclass classifier. It has the desirable consistency. Similar to the SVM, it has the ability to handle high dimensional data. In contrast to the challenging optimization of multiclass SVM, the CLS classifier is very efficient to compute. Surprisingly, although it makes use of a simultaneous formulation, its computation can be decoupled as multiple smaller optimization problems as in the one-versus-rest approach. Consequently, computation complexity of the CLS classifier grows with the class number *linearly*, thus it is feasible even for problems with very large number of classes.

The CLS classifier is closely related to the SVM. Instead of using the multiclass hinge loss function as in the SVM, it makes use of the proposed composite squared loss which yields very efficient computation. More specifically, the CLS classifier uses a nontrivial *convex combination* of two different types of squared loss functions, with one of the two being the loss of the Proximal SVM (PSVM) (Suykens and Vandewalle, 1999; Fung and Mangasarian, 2001; Tang and Zhang, 2006). The combination is shown to be necessary as the performance of the combined loss is much better than either of the uncombined loss functions. Another important advantage of the CLS classifier is its ability to produce class probability estimation, while in contrast, the SVM cannot. Due to the special form of the loss function for the CLS classifier, we are able to derive closed-form solutions and the formulae to predict class probability.

The rest of this paper is organized as follows. In Section 2, we give a brief background on multiclass classification and review some existing multiclass SVM techniques. In Section 3, we propose the multiclass CLS classifier, study its property, and provide probability estimation. Section 4 discusses the computational algorithm. Numerical results through simulated examples and real data applications are presented in Section 5 and Section 6, respectively. Some discussion is given in Section 7. Proofs of theoretical results are included in the Appendix.

2 Background on Multiclass Classification

Suppose our training dataset is $\{x_i, y_i\}_{i=1, \dots, n}$. Here, x_i is the d -dimensional covariate, and $y_i \in \{1, \dots, k\}$ represents a k -class label with $k > 2$. Our goal is to build a classifier based on the training data so that we can predict the class membership of new observations.

2.1 Sequence of Binary Classifiers

To solve a multiclass problem, a natural and direct way is to implement multiple binary classifiers. For example, one can use the one-versus-rest or one-versus-one approach. The one-versus-rest approach relabels the training data in the class j as the positive class and data which are not in the class j as the negative class, for each $j = 1, \dots, k$. Then, one can employ a sequence of k binary classifiers for the membership of each data point, which can possibly give contradictory results among the k binary classifiers. The one-versus-one approach applies a given binary classifier to a binary problem of the class j_1 and the class j_2 for each

of all possible pairs $j_1, j_2 \in \{1, \dots, k\}$. Overall, $\binom{k}{2}$ binary classifications are performed. For each binary problem, the dataset can be very small.

When there is no dominating class, in the SVM context, the one-versus-rest approach can be self-contradicted (Lee et al., 2004) and Fisher consistency is not guaranteed (Liu, 2007). Thus, it is necessary to generalize binary classification methods to multiclass versions which consider all classes simultaneously, whilst retaining the desirable properties of the original methods.

2.2 Simultaneous methods

In contrast to the approach using a sequence of binary classifiers, one can obtain the decision functions for all classes simultaneously and compare them all at once to predict class membership. More specifically, let

$\mathbf{f}(x) = (f_1(x), f_2(x), \dots, f_k(x)) \in \prod_{j=1}^k (\{1\} + \mathcal{H}_K)$ be the decision function vector, where $\mathcal{H}_K$ denotes a reproducing kernel Hilbert space generated by the kernel K . Here we use the same $\mathcal{H}_K$ for different f_j although one can make it more general. Once the value of $\mathbf{f}$ is obtained, the class membership of any new data point x is estimated by $\hat{y} = \arg\max_{j=1,2,\dots,k} f_j(x)$. To remove redundancy in solutions, we use the zero-sum constraint, $\sum_{j=1}^k f_j(x) = 0$. This formulation becomes equivalent to the binary problem when $k = 2$. Here, f_j represents class j and is typically not the class probability $P_j(x) = P(Y = j|x)$. In general, points in class l tend to have high f_l values and yield correct classification if f_l is the maximum among $f_j, j = 1, \dots, k$. In certain cases, the theoretically optimal f_j can be represented as a function of $P_1, \dots, P_k$ and as a result, one can use transformation of f_j to derive estimation of P_j .

With a sensible loss function V given, the multiclass large margin classifier solves

$$\min_{f \in \mathcal{F}} \sum_{i=1}^n V(\mathbf{f}(x_i), y_i) + \lambda \sum_{j=1}^k J(f_j), \text{ subject to } \sum_{j=1}^k f_j(x) = 0 \text{ for } i=1, \dots, n. \quad (1)$$

Since x is misclassified when $y \neq \operatorname{argmax}_j f_j(x)$, a good loss function should force f_k to be the maximum among $f_1, \dots, f_k$.

Fisher consistency is an important property of classification. It requires that a classifier approximates the Bayes rule when the sample size is sufficiently large. Thus in our formulation, Fisher consistency requires that $\operatorname{argmax}_j f_j^* = \operatorname{argmax}_j P_j$, where

$\mathbf{f}^*(x) = (f_1^*(x), \dots, f_k^*(x))$ denotes the minimizer of $E[V(\mathbf{f}(x), y)|X=x]$. When we select a loss function V , it is necessary to study its Fisher consistency.

2.3 Existing Multicategory SVMs

In this section, we focus on different versions of simultaneous multicategory SVMs. In the literature, there are several different ways to extend the binary hinge loss of the SVM to the multicategory case. Here we list several commonly used versions for the SVM:

1. (Naive hinge loss) $[1 - f_y(x)]_+$;
2. (Vapnik, 1998) $\sum_{j \neq y} [1 - (f_y(x) - f_j(x))]_+$;
3. (Crammer and Singer, 2001; Liu and Shen, 2006) $\sum_{j \neq y} [1 - \min_j (f_y(x) - f_j(x))]_+$;
4. (Lee et al., 2004) $\sum_{j \neq y} [1 + f_j(x)]_+$.

In these loss formulations, x is the covariate of each data point, and y is its corresponding class membership. Loss 1 is a simple extension of the binary hinge loss. We call this loss function as the naive hinge loss. ? showed that the minimizer $\mathbf{f}^*$ of $E[[1 - f_y(x)]_+ | X=x]$ subject to $\sum_{j=1}^k f_j(x) = 0$ is $f_j^*(x) = -(k-1)$ if $j = \operatorname{argmin}_j P_j(x)$ and 1 otherwise, which implies the naive hinge loss function is not Fisher consistent. He also showed the cases when Loss 2 and Loss 3 are not Fisher consistent. In contrast to these three loss functions, Loss 4 is Fisher consistent as the minimizer of $\mathbf{f}^*$ of $E[\sum_{j \neq y} [1 + f_j(x)]_+ | X=x]$ subject to $\sum_{j=1}^k f_j(x) = 0$ is $f_j^*(x) = k-1$ if $j = \operatorname{argmax}_j P_j(x)$ and -1 otherwise.

Recently, Liu and Yuan (2009) proposed the Reinforced Multicategory Support Vector Machine (RMSVM), which uses a new group of the multicategory SVM loss functions called the reinforced hinge loss,

$$V(\mathbf{f}(x), y) = \gamma [(k-1) - f_y(x)]_+ + (1-\gamma) \sum_{j \neq y} [1 + f_j(x)]_+. \quad (2)$$

It is the convex combination of the naive hinge loss and the Loss 4 by Lee et al. (2004) with weights $(\gamma, 1-\gamma)$. Interestingly, even though the naive hinge loss function is not Fisher consistent, the reinforced hinge loss function is Fisher consistent when $0 \leq \gamma \leq 1/2$. ? showed that the loss function (2) with $\gamma = 1/2$ gives the best classification performance among $\gamma \in [0, 1]$.

One computational difficulty of the hinge loss is that it is not differentiable. As a result, the minimizer of the hinge loss does not attain all information of the conditional class probability. To improve this, the squared loss function can be employed. The computation is much simpler and its theoretical minimizer contains information about the class probability,

which enables us to estimate the class probability. Tang and Zhang (2006) proposed the multiclass proximal SVM, which employs the squared loss function for the simultaneous multicategory classification framework. More specifically, they used the loss function

$$V(\mathbf{f}(\mathbf{x}), y) = \sum_{j \neq y} (1 + f_j(\mathbf{x}))^2. \quad (3)$$

Note that this loss function is essentially the same extension with Loss 4, except it uses the squared loss instead of the hinge loss. Tang and Zhang (2006) showed the loss function of the multiclass proximal SVM is always Fisher consistent.

3 Multicategory Composite Least Squares Classifier

Despite the good theoretical property and numerical performance on simulated and real data of the reinforced multicategory SVM (RMSVM) by Liu and Yuan (2009), it has two limitations. The first drawback is that the computation involves a quadratic programming (QP) of size $nk \times nk$. Since the computational complexity of QP is polynomial, the RMSVM can be infeasible to implement when k is relatively large. In this section, we propose to use the following new family of composite squared loss functions

$$V(\mathbf{f}(\mathbf{x}), y) = \gamma \left[(k-1) - f_y(\mathbf{x}) \right]^2 + (1-\gamma) \sum_{j \neq y} \left[1 + f_j(\mathbf{x}) \right]^2, \quad (4)$$

subject to $\sum_{j=1}^k f_j(\mathbf{x}) = 0$, where $\gamma \in [0; 1]$. We call problem (1) with the composite squared loss, the Multicategory Composite Least Squares (CLS) Classifier. It can be shown that the multicategory CLS classifier with $\gamma = 0$ is equivalent to the multiclass proximal SVM of Tang and Zhang (2006) with all misclassification costs equal.

To further understand the proposed loss family (4), we express the composite squared loss function using the multiple comparison vector representation proposed by ?. In particular, they defined the comparison vector $\mathbf{g}(\mathbf{f}(\mathbf{x}), y) = (f_y(\mathbf{x}) - f_1(\mathbf{x}), \dots, f_y(\mathbf{x}) - f_{y-1}(\mathbf{x}), f_y(\mathbf{x}) - f_{y+1}(\mathbf{x}), \dots, f_y(\mathbf{x}) - f_k(\mathbf{x}))$. Recall $(\mathbf{x}, y)$ is misclassified if $f_y(\mathbf{x})$ is not the maximum among $f_1(\mathbf{x}), \dots, f_k(\mathbf{x})$ due to the argmax classification rule. Then, $\min(\mathbf{g}(\mathbf{f}(\mathbf{x}), y)) \leq 0$ if and only if an observation $(\mathbf{x}, y)$ is misclassified. This is because that the data point $(\mathbf{x}, y)$ is misclassified when there is at least one $f_j(\mathbf{x})$ with $j \neq y$ which is greater than $f_y(\mathbf{x})$. Let $\mathbf{u} = \mathbf{g}(\mathbf{f}(\mathbf{x}), y)$. Using this notation, the 0 – 1 loss function becomes $I\{\min_j u_j \leq 0\}$. The composite squared loss function can be written as

$$\gamma \left[(k-1) - \sum_{l=1}^{k-1} u_l / k \right]^2 + (1-\gamma) \sum_{j=1}^{k-1} \left[1 + \sum_{l=1}^{k-1} u_l / k - u_j \right]^2, \quad (5)$$

and we plot (5) in Figure 1 with $\gamma = 0, 0.5, 1$, as well as the 0 – 1 loss function. We can see that as γ increases, the value of the loss function increases when both u_1 and u_2 are negative, while the loss decreases when only one of u_l 's is negative.

The behavior of γ in the multicategory CLS classifier can be very different from that of the RMSVM as shown in the numerical examples in Section 5. In particular, $\gamma = 0.5$ does not always show the best performance, although $\gamma = 0.5$ is near optimal in many cases. Thus the choice of γ should depend on the data. One might consider selecting γ using cross validation.

To reduce the computational cost, we recommend to tune γ among $\{0, 0.5, 1\}$. This scheme is demonstrated in the numerical examples in Section 5.

Unlike the RMSVM, the multicategory CLS classifier offers class probability estimation which enables one to better understand the nature of the data.

3.1 Properties of the multicategory CLS classifier

The following theorem establishes Fisher consistency of the composite loss function in a general form which includes the composite squared loss function as a special case.

Theorem 1—Suppose a function $\ell(u)$ is twice differentiable, $\ell'(u) < 0$, and $\ell''(0) > 0$. Let $V(f(x), y) = \gamma \ell(g_y(x)) + (1-\gamma) \sum_{j \neq y} \ell(-\ell_j(x))$. Then, the minimizer f^* of $E[V(f(x), y)]$, subject to $\sum_j f_j(x) = 0$, satisfies the following: $\arg\max_j P_j(x) = \arg\max_j f_j(x)$.

Remark 1—The conclusion of the Theorem 1 holds if the assumption on $\ell(u)$ is reduced to that $\ell'(u) < 0$ and $\ell'(u)$ is strictly increasing.

Remark 2—The reinforced hinge loss function is Fisher consistent only when $0 \leq \gamma \leq 1/2$. However, the composite squared loss function is always Fisher consistent for all values of $\gamma \in [0, 1]$.

3.2 Probability Estimation

It can be shown that the theoretical minimizer $f^*(x)$ of $E[V(f(X), Y)|X = x]$ is a function of the conditional class probabilities. Hence, we can use $\hat{f}^*(x)$ to predict class probabilities. The following theorem shows the exact form of $f^*(x)$.

Theorem 2—The minimizer $f^*(x)$ of $E[V(f(X), Y)|X = x]$ with V in (4), subject to $\sum_j f_j(x) = 0$, is $(f_1^*(x), \dots, f_k^*(x))$, with

$$f_j^*(x) = \left(\frac{\sum_{l=1}^k a_l b_l}{\sum_{l=1}^k a_l} + b_j \right) a_j,$$

where $a_l = 1/[2\gamma P_l(x) + 2(1-\gamma)(1-P_l(x))]$ and $b_l = 2\gamma(k-1)P_l(x) - 2(1-\gamma)(1-P_l(x))$ for $l = 1, \dots, k$.

Due to the complicated structure of $f_j^*(x)$, it is difficult to recover $(P_1(x), \dots, P_k(x))$ from $(f_1(x), \dots, f_k(x))$ for a general γ . However, for $\gamma = 0, 0.5$, and 1 , we are able to make $a_l b_l$ simple to recover conditional class probabilities $(P_1(x), \dots, P_k(x))$ from $(f_1(x), \dots, f_k(x))$. The formulae to estimate class probabilities using $(f_1(x), \dots, f_k(x))$ for $\gamma = 0, 0.5$, and 1 are given as follows:

$$\text{For } \gamma=0, \quad \widehat{P}_j(x) = 1 - (k-1) \frac{1/[1+\widehat{f}_j(x)]}{\sum_{l=1}^k 1/[1+\widehat{f}_l(x)]}; \quad (6)$$

$$\text{For } \gamma=0.5, \quad \widehat{P}_j(x) = \frac{1}{k} (1 + \widehat{f}_j(x)); \quad (7)$$

$$\text{For } \gamma=1, \quad \widehat{P}_j(x) = \frac{1/\left[\widehat{f}_j(x) - (k-1)\right]}{\sum_{l=1}^k 1/\left[\widehat{f}_l(x) - (k-1)\right]}. \quad (8)$$

Notice that the proposed estimators of class probabilities $\widehat{P}_j(x)$ sum up to 1, that is, $\sum_{j=1}^k \widehat{P}_j(x) = 1$. However, individual estimator $\widehat{P}_j(x)$ in (6)-(8) may be outside of $[0, 1]$. To ensure the estimated probabilities are proper, we propose to rescale the probability estimates using the following formulae,

$$\widehat{P}_j^{\text{scaled}}(x) = \frac{\widehat{P}_j(x) - \min_{l=1, \dots, k} \widehat{P}_l(x)}{\sum_{l=1}^k [\widehat{P}_l(x) - \min_{m=1, \dots, k} \widehat{P}_m(x)]}.$$

4 Computational Algorithm

Since $f_j \in (\{1\} + \mathcal{H}_K)$, we can write $f_j(x) = \beta_{j0} + \sum_{i=1}^n \beta_{ji} K(x_i, x)$ using the representative theorem (?). The L_2 penalty commonly used is $J(f_j) = \|f_j\|_{\mathcal{H}_K}^2 = \beta_j^T \mathbf{K} \beta_j$, where $\beta_j = (\beta_{j1}, \dots, \beta_{jn})^T$ and $\mathbf{K}$ is the $n \times n$ matrix with ij -th entry $K(x_i, x_j)$. For simplicity of calculation, we use the penalty term $J(f_j) = \|f_j\|_{\mathcal{H}_K}^2 + \beta_{j0}^2 = \beta_j^T \mathbf{K} \beta_j + \beta_{j0}^2$, which results in a closed form solution. This makes computation simpler and gives similar results with the same problem using the original L_2 penalty. In addition, we use a new class coding to replace y with z defined as

$$z_{ij} = \begin{cases} 1 & \text{if } y_i = j \\ -\frac{1}{k-1} & \text{otherwise.} \end{cases} \quad (9)$$

Let

$$w_{ij} = \begin{cases} \gamma & \text{if } y_i = j \\ 1 - \gamma & \text{otherwise.} \end{cases} \quad (10)$$

Then minimizing $\sum_{i=1}^n V(f(x_i), y_i)$ with V in (4) becomes equivalent to minimizing

$$\begin{aligned} & \sum_{i=1}^n \left[\gamma \left(1 - \tilde{f}_{y_i}(x_i) \right)^2 + (1 - \gamma) \sum_{j \neq y_i} \left(-\frac{1}{k-1} - \tilde{f}_j(x_i) \right)^2 \right] \\ &= \sum_{i=1}^n \sum_{j=1}^k w_{ij} \left(z_{ij} - \tilde{f}_j(x_i) \right)^2 \\ &= \sum_{j=1}^k \left[z_j - \tilde{\mathbf{X}} \tilde{\beta}_j \right]^T \mathbf{W}_j \left[z_j - \tilde{\mathbf{X}} \tilde{\beta}_j \right], \end{aligned} \quad (11)$$

where $\tilde{f}_j = f_j/(k-1)$, $z_j = (z_{1j}, \dots, z_{nj})^T$, $1_n = (1, \dots, 1)^T$, $\tilde{\mathbf{X}} = [1_n, \mathbf{K}]$, $\tilde{\beta}_j = (\beta_{j0}, \beta_j^T)^T / (k-1)$ and $\mathbf{W}_j = \text{diag}\{w_{1j}, \dots, w_{nj}\}$. Let $\mathbf{0}_n = (0, \dots, 0)^T$,

$$\mathbf{G} = \begin{pmatrix} 1 & 0_n^T \\ 0_n & \mathbf{K} \end{pmatrix},$$

$\tilde{\mathbf{X}}_j = \mathbf{W}_j^{1/2} \tilde{\mathbf{X}} \mathbf{G}^{-1/2}$, $\mathbf{z}_j^* = \mathbf{W}_j^{1/2} \mathbf{z}_j$, and $\beta_j^* = \mathbf{G}^{1/2} \tilde{\beta}_j$. Then, (11) can be written as

$\sum_{j=1}^k \left[\mathbf{z}_j^* - \tilde{\mathbf{X}}_j \beta_j^* \right]^T - \left[\mathbf{z}_j^* - \tilde{\mathbf{X}}_j \beta_j^* \right]$. Hence, solving (1) with the loss function V in (4) and the penalty term $J(\mathbf{f}_j) = \beta_j^T \mathbf{K} \beta_j + \beta_{j0}^2$ is equivalent to minimizing

$$\sum_{j=1}^k \left[\mathbf{z}_j^* - \tilde{\mathbf{X}}_j \beta_j^* \right]^T \left[\mathbf{z}_j^* - \tilde{\mathbf{X}}_j \beta_j^* \right] + \lambda \beta_j^{*T} \beta_j^*$$

subject to $\sum_{j=1}^k \beta_j^* = 0_{n+1}$. To solve this, we consider its dual problem with the Lagrange multiplier vector $2\mathbf{u} \in \mathbf{R}^{n+1}$,

$$L(\beta, \mathbf{u}) = \sum_{j=1}^k \left[\left(\mathbf{z}_j^* - \tilde{\mathbf{X}}_j \beta_j^* \right)^T \left(\mathbf{z}_j^* - \tilde{\mathbf{X}}_j \beta_j^* \right) + \lambda \beta_j^{*T} \beta_j^* \right] - 2\mathbf{u}^T \sum_{j=1}^k \beta_j^*. \quad (12)$$

Setting derivatives of (12) to zero gives

$$\frac{\partial L}{\partial \beta_j^*} = 2 \left[\tilde{\mathbf{X}}_j^T \left(\mathbf{z}_j^* - \tilde{\mathbf{X}}_j \beta_j^* \right) + \lambda \beta_j^* - \mathbf{u} \right] = 0_{n+1}, \quad (13)$$

$$\frac{\partial L}{\partial \mathbf{u}} = \sum_{j=1}^k \beta_j^* = 0_{n+1}. \quad (14)$$

From (13), we have $\beta_j^* = \left(\tilde{\mathbf{X}}_j^T \tilde{\mathbf{X}}_j + \lambda \mathbf{I} \right)^{-1} \left(\tilde{\mathbf{X}}_j^T \mathbf{z}_j^* + \mathbf{u} \right)$. Combining with (14), we have

$\mathbf{u} = - \left[\sum_{j=1}^k \tilde{\mathbf{B}}_j \right]^{-1} \left[\sum_{j=1}^k \tilde{\mathbf{B}}_j \tilde{\mathbf{X}}_j^T \mathbf{z}_j^* \right]$, where $\tilde{\mathbf{B}}_j = \left(\tilde{\mathbf{X}}_j^T \tilde{\mathbf{X}}_j + \lambda \mathbf{I} \right)^{-1}$. Plugging this into (13), together with the definition of β_j^* , gives

$$\tilde{\beta}_j = \mathbf{A}_j \left[\tilde{\mathbf{X}}^T \mathbf{W}_j \mathbf{z}_j - \left(\sum_{j=1}^k \mathbf{A}_j \right)^{-1} \left(\sum_{j=1}^k \mathbf{A}_j \tilde{\mathbf{X}}^T \mathbf{W}_j \mathbf{z}_j \right) \right], \quad (15)$$

where $\mathbf{A}_j = (\tilde{\mathbf{X}}^T \mathbf{W}_j \tilde{\mathbf{X}} + \lambda \mathbf{G})^{-1}$.

So far, we have focused on the standard case which treats all samples with equal weight. However, there could be situations that we want to give different weights on different subjects. For example, it could be more severe to misclassify an observation to a certain class than to other classes. Then it is natural to put a higher cost for that certain type of

misclassification. This can be achieved by putting different weights on observations in different classes.

The multicategory weighted CLS classifier can be directly implemented with a simple modification. Let π_i be the weight we want to impose on the i -th observation. Then the loss function in (11) remains the same, except w_{ij} is replaced by $w_{ij}^* = \pi_i w_{ij}$. The rest of the algorithm remains the same.

5 Simulation Studies

To explore the performance of our proposed multicategory CLS classifier, we carry out some numerical analysis on the following multi-class examples, as illustrated in the Figure 2. For Example 1, which is a 3-class problem used in ?, we generate 50 observations for training, 50 observations for tuning, and 10^6 observations for testing. For Examples 2 and 3, which have 6 and 4 classes respectively, we generate 100 observations for each of training and tuning sets to ensure each training set has reasonable number of observations for every class. For testing, we generate 10^6 observations similarly to Example 1. A model is developed based on the training set, then the tuning set is used to choose λ among the set $\{2^{-16}, 2^{-15}, \dots, 2^{15}\}$. With the selected model and λ , the misclassification rate is calculated based on the testing set, as the proportion of misclassified testing points by the classifier. We repeat this procedure 100 times with $\gamma = 0.0, 0.1, 0.2, \dots, 1.0$ to examine the effect of γ . We also included the results of the case when γ is tuned among $\{0, 0.5, 1\}$ together with γ . For probability estimation, we train and tune the model in the same manner for $\gamma = 0, 0.5, 1$, then use the formulae (6)-(8) to estimate class probability. The probability estimation error,

$\frac{1}{nk} \sum_{i=1}^n \sum_{j=1}^k \left| \widehat{P}_j(x) - P_j(x) \right|$, is calculated based on the testing set.

The three examples are used to illustrate the performance of the multicategory CLS classifier as well as the effect of γ . The first two examples show the case where higher γ works better. For Example 3, the case with $\gamma = 1$ gives the worst performance. These examples demonstrate that γ should be chosen in a data-adaptive fashion.

Example 1 We generate three-class data with

$$\begin{aligned} P(Y=j) &= 1/3, \text{ for } j=1, \dots, 3, \\ P(\mathbf{X}|Y=j) &\sim \mathcal{N}(\mu_j, 1.5^2 \mathbf{I}_2), \text{ for } j=1, \dots, 3, \\ \mu_j &= \begin{pmatrix} 0 \\ 2 \end{pmatrix}, \begin{pmatrix} -\sqrt{3} \\ -1 \end{pmatrix}, \begin{pmatrix} \sqrt{3} \\ -1 \end{pmatrix}, \text{ for } j=1, \dots, 3, \text{ respectively.} \end{aligned} \quad (16)$$

Since the Bayes boundary of this example is piecewise linear, linear learning can be sufficient. However, we added the results using the polynomial kernel of order 2 as well to further illustrate the behavior of the multicategory CLS classifier. Moreover, we also report the performance of the RMSVM for comparison.

The results are summarized in Table 1 and Figure 3. When the linear kernel is used, the test error decreases as γ increases. On the other hand, the polynomial kernel of order 2 gives the similar results except the test error becomes very high when $\gamma = 1.0$. Overall, the results are consistently good when γ is around 0.5. The probability estimation error has a similar pattern: it becomes the smallest when $\gamma = 0.5$. These results indicate that the multicategory CLS classifier gives the best result when γ is near 0.5, and the combination of two different versions of the squared loss functions is indeed better than both uncombined loss functions.

The performance of the RMSVM has a similar pattern to that of the multicategory CLS classifier in the sense that γ in the middle works the best overall. Furthermore, the multicategory CLS classifier outperforms the RMSVM on this example.

Example 2 In this example, we generate six-class data with

$$\begin{aligned} P(Y=j) &= 1/6, \text{ for } j=1, \dots, 6, \\ P(\mathbf{X}|Y=j) &\sim \mathcal{N}(\mu_j, \sigma^2 \mathbf{I}_2), \text{ for } j=1, \dots, 6, \\ \mu_j &= \begin{pmatrix} 1 \\ \sqrt{3} \end{pmatrix}, \begin{pmatrix} 2 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ -\sqrt{3} \end{pmatrix}, \begin{pmatrix} -1 \\ -\sqrt{3} \end{pmatrix}, \begin{pmatrix} -2 \\ 0 \end{pmatrix}, \begin{pmatrix} -1 \\ \sqrt{3} \end{pmatrix}, \text{ for } j=1, \dots, 6, \text{ respectively,} \end{aligned} \quad (17)$$

for $\sigma = 0.5$ and $\sigma = 0.7$. Similar to Example 1, the Bayes boundary is piecewise linear, but we report the results of the multicategory CLS classifier with both of linear and polynomial kernel of order 2 in Table 2 and Figure 4. Since the RMSVM runs into numerical problems on this example due to the large number of classes, we do not report the results of the RMSVM and the Two-step MCLS classifier.

Clearly, higher γ works better in this example regardless of the kernel choice. This indicates that the multicategory CLS classifier indeed improves the existing multi-class proximal SVM. The tuned methods give very reasonable performance.

Example 3 we generate four-class data with

$$\begin{aligned} P(Y=j) &= 1/4, \text{ for } j=1, \dots, 4, \\ P(\mathbf{X}|Y=1) &\sim 0.5N\left(\begin{pmatrix} 1 \\ \sqrt{3} \end{pmatrix}, \sigma^2 \mathbf{I}_2\right) + 0.5N\left(\begin{pmatrix} -1 \\ -\sqrt{3} \end{pmatrix}, \sigma^2 \mathbf{I}_2\right) \\ P(\mathbf{X}|Y=2) &\sim 0.5N\left(\begin{pmatrix} 2 \\ 0 \end{pmatrix}, \sigma^2 \mathbf{I}_2\right) + 0.5N\left(\begin{pmatrix} -2 \\ 0 \end{pmatrix}, \sigma^2 \mathbf{I}_2\right) \\ P(\mathbf{X}|Y=j) &\sim \mathcal{N}(\mu_j, \sigma^2 \mathbf{I}_2), \text{ for } j=3, 4, \\ \mu_3 &= \begin{pmatrix} 1 \\ -\sqrt{3} \end{pmatrix}, \mu_4 = \begin{pmatrix} -1 \\ \sqrt{3} \end{pmatrix}, \end{aligned} \quad (18)$$

for $\sigma = 0.5$ and $\sigma = 0.7$. Due to the structure of this dataset, linear learning will not be suitable, thus we consider the Gaussian kernel and the polynomial kernel of order 2.

The results are summarized in Table 3 and Figure 5. For each kernel, higher γ generally works better, but the highest $\gamma = 1$ gives the worst performance. Hence, we can conclude that it is safe to use γ that is somewhere in the middle. Furthermore, tuning γ can be a reasonable way to choose γ .

Regarding to performance of the cases where $\sigma = 0.5$ and $\sigma = 0.7$, the misclassification errors for $\sigma = 0.7$ is around 2 times of that for $\sigma = 0.5$. This is not surprising in view of their corresponding Bayes errors 0.0434 versus 0.1450.

In summary, the examples illustrate the behavior of the multicategory CLS classifier with different values of γ . The performance heavily depends on the choice of γ . One sensible way is to tune γ among candidate values $\{0, 0.5, 1\}$. If one wants to select a fixed value for γ , the multicategory CLS classifier $\gamma = 0.5$ appears to yield a robust and accurate classifier in most cases.

6 Real Application

To further demonstrate the performance of the multicategory CLS classifiers, we use the Leukemia data set described in Golub et al. (1999) and the data from a Phase I clinical trial sponsored by Boehringer Ingelheim.

6.1 Application to Leukemia Data

The Leukemia data set in Golub et al. (1999) consists of 7129 gene expression values of 38 examples in the training set and 34 examples in the testing set. The original paper and many other papers that analyze this dataset classify subjects into two types of leukemias, ALL (acute lymphoblastic leukemia) and AML (acute myeloid leukemia). Since ALL type can be further divided into B-cell and T-cell ALLs (ALLB and ALLT), we can perform three-category classification on this data set. The training set contains 19, 8, 11 subjects of ALLB, ALLT, and AML types, and the testing set has 19, 1, 14 subjects of ALLB, ALLT, and AML types. Out of 7129 genes in the training set, we choose 40 genes by prescreening procedure using the ratios of between-groups to within-groups sum of squares in the training set as in Dudoit et al. (2002). We also tried other numbers of genes and the results are similar. The tuning parameter was selected by 5-fold cross validation on the training set.

Similar to the results of the original binary problem, the number of misclassified testing points is at most 2, regardless of the value of γ . Hence rather than looking at the misclassification rate, it might be more interesting to look at the performance of the class probability estimation. As shown in Figure 6, the estimated class probabilities agree with the true memberships of the observations, except when $\gamma = 1$, two patients are misclassified (ALLT patient and the first AML patient in the graph). According to heatmap of this data set with selected 40 genes in Figure 7, these two patients are the ones that are hard to classify. Their estimated probabilities reflect that the gene expressions of these patients are not very distinguishable. In real application, estimated class probabilities such as the ones shown here could be more helpful than just class membership prediction.

6.2 Application to Phase I Clinical Trial Data

In this data set, 84 healthy volunteers were randomly given one of 4 different doses (0, 10mg, 20mg, or 40mg) of Prednisolone, and values of a panel of pharmacodynamic biomarkers were observed at several planned time points. Our goal is to classify patients into dose groups using the values of the different biomarkers.

First, we choose the tuning parameter λ by 10-fold Cross Validation (CV) for each $\gamma \in \{0, 0.1, \dots, 1\}$. The CV errors for each γ with the best λ are reported in Figure 8. We can conclude that $\lambda = 0.5$ might not be the best choice, but it gives a reasonable performance compared to $\gamma = 0$ or $\gamma = 1$. For $\gamma = 0, 0.5$, and 1, we performed multicategory CLS classifier with the λ selected by CV and estimated class probabilities. The results are summarized in Figure 9. When $\gamma = 0$, lots of subjects have positive estimated probabilities only for three classes and zero probability for one class. As γ goes higher, more subjects have positive estimated probabilities for all four classes. Overall, when $\gamma = 0.5$, the multicategory CLS classifier shows pretty stable performance, especially for the 0mg and the 40mg group.

7 Summary and Discussion

In this paper, we propose the multicategory CLS classifier which makes use of a convex combination of two square loss functions. The CLS classifier is shown to have Fisher consistency, conditional probability estimation, efficient computation, ability to handle data with high dimension and large number of classes. Through simulated examples, the CLS classifier is shown to outperform proximal SVMs and multicategory SVMs.

The current CLS classifier makes use of the L_2 penalty as in the ridge regression. Although L_2 penalty works well overall, it does not perform automatic variable selection. For high dimensional data with many noise variables, other penalties such as L_1 penalty (Zhu et al., 2004; Wang and Shen, 2007) can be more useful to deliver sparse models. It will be interesting to explore sparse CLS classifiers for high dimensional data analysis.

Acknowledgments

The research of Park and Liu was supported in part by NSF grant DMS-0747575 and NIH grant IR01CA149569-01.

Appendix

Proof of Theorem 1

First, observe that

$$\begin{aligned} E[V(f(x), y) | X=x] &= \gamma \sum_{j=1}^k \ell(f_j(x)) P_j(x) + (1-\gamma) \sum_{l=1}^k \left[\left(\sum_{j=1}^k \ell(-f_j(x)) \right) - \ell(-f_l(x)) \right] P_l(x) \\ &= \gamma \sum_{j=1}^k \ell(f_j(x)) P_j(x) + (1-\gamma) \sum_{j=1}^k \ell(-f_j(x)) - (1-\gamma) \sum_{l=1}^k \ell(-f_l(x)) P_l(x) \\ &= \gamma \sum_{j=1}^k \ell(f_j(x)) P_j(x) + (1-\gamma) \sum_{j=1}^k (1 - P_j(x)) \ell(-f_j(x)). \end{aligned} \quad (19)$$

To obtain the minimizer of (19) subject to $\sum_{j=1}^k f_j(x) = 0$, we use the Lagrange multiplier method. For convenience, let $f_j = f_j(x)$. The corresponding Lagrange primal is

$$L(f, \alpha) = \gamma \sum_{j=1}^k \ell(f_j) P_j(x) + (1-\gamma) \sum_{j=1}^k (1 - P_j(x)) \ell(-f_j) - \alpha \sum_{j=1}^k f_j.$$

Setting the first derivatives to zero gives,

$$\begin{aligned} \frac{\partial L}{\partial f_1} &= \gamma P_1(x) \ell'(f_1) - (1-\gamma)(1 - P_1(x)) \ell'(-f_1) - \alpha = 0 \\ &\vdots \\ \frac{\partial L}{\partial f_k} &= \gamma P_k(x) \ell'(f_k) - (1-\gamma)(1 - P_k(x)) \ell'(-f_k) - \alpha = 0 \\ \frac{\partial L}{\partial \alpha} &= \sum_{j=1}^k f_j = 0. \end{aligned}$$

Therefore, we have

$$-\gamma P_j(x) \ell'(f_j) + (1-\gamma)(1 - P_j(x)) \ell'(-f_j) = -\gamma P_l(x) \ell'(f_l) + (1-\gamma)(1 - P_l(x)) \ell'(-f_l) \text{ for any } j \neq l \quad (20)$$

Without loss of generality, we can assume that $\arg\max_j P_j(x) = 1$. Suppose $f_j \geq f_1$ for some $j \neq 1$. Since $\ell'(u) < 0$ and $\ell''(u) > 0$, we have $\ell'(f_1) \leq \ell'(f_j) < 0$ and $0 > \ell'(-f_1) \geq \ell'(-f_j)$. This implies $\gamma P_1(x) \ell'(f_1) < \gamma P_j(x) \ell'(f_j)$ and $(1-\gamma)(1 - P_1(x)) \ell'(-f_1) > (1-\gamma)(1 - P_j(x)) \ell'(-f_j)$, which gives $-\gamma P_1(x) \ell'(f_1) + (1-\gamma)(1 - P_1(x)) \ell'(-f_1) > -\gamma P_j(x) \ell'(f_j) + (1-\gamma)(1 - P_j(x)) \ell'(-f_j)$. This contradicts to (20). Thus we conclude $f_1 > f_j$ for any $j \neq 1$ when $\arg\max_j P_j(x) = 1$. This completes the proof.

Proof of Theorem 2

With V in (4), we have

$$\begin{aligned}
& E[V(f(x), y) | \mathbf{X} = \mathbf{x}] \\
&= \gamma \sum_{j=1}^k \{k-1-f_j(\mathbf{x})\}^2 P_j(\mathbf{x}) + (1-\gamma) \sum_{l=1}^k \left[\sum_{j=1}^k \{1+f_j(\mathbf{x})\}^2 - \{1+f_l(\mathbf{x})\}^2 \right] P_l(\mathbf{x}) \\
&= \gamma \sum_{j=1}^k \{k-1-f_j(\mathbf{x})\}^2 P_j(\mathbf{x}) + (1-\gamma) \sum_{j=1}^k \{1-P_j(\mathbf{x})\} \{1+f_j(\mathbf{x})\}^2
\end{aligned} \tag{21}$$

because $\sum_{l=1}^k P_l(\mathbf{x}) = 1$. Since we have a constraint $\sum_j f_j(\mathbf{x}) = 0$, we consider its dual problem with Lagrange multiplier τ ,

$$L = \gamma \sum_{j=1}^k \{k-1-f_j(\mathbf{x})\}^2 P_j(\mathbf{x}) + (1-\gamma) \sum_{j=1}^k \{1-P_j(\mathbf{x})\} \{1+f_j(\mathbf{x})\}^2 - \tau \sum_{j=1}^k f_j(\mathbf{x}) \tag{22}$$

Setting derivatives of (22) to zero gives,

$$\frac{\partial L}{\partial f_j(\mathbf{x})} = -\gamma 2 \{k-1-f_j(\mathbf{x})\} = 0, \text{ for } j=1 \cdots k, \tag{23}$$

$$\frac{\partial L}{\partial \tau} = -\sum_{j=1}^k f_j(\mathbf{x}) = 0. \tag{24}$$

From 23, we have

$$f_j(\mathbf{x}) = (\tau + b_j) a_j, \tag{25}$$

where $a_j = 1/\{2\gamma P_j(\mathbf{x}) + 2(1-\gamma)(1-P_j(\mathbf{x}))\}$ and $b_j = 2\gamma(k-1)P_j(\mathbf{x}) - 2(1-\gamma)\{1-P_j(\mathbf{x})\}$. From (24) and (25), we have

$$0 = \sum_{j=1}^k f_j(\mathbf{x}) = \tau \sum_{j=1}^k a_j + \sum_{j=1}^k a_j b_j, \tag{26}$$

which implies,

$$\tau = -\frac{\sum_{j=1}^k a_j b_j}{\sum_{j=1}^k a_j} \tag{27}$$

From (25), we have

$$f_j(\mathbf{x}) = \left(-\frac{\sum_{l=1}^k a_l b_l}{\sum_{l=1}^k a_l} + b_j \right) a_j. \tag{28}$$

This completes the proof.

Figure 1. Plot of the 0 - 1 loss function and the composite squared loss functions with $\gamma = 0, 0.5, 1$.

Figure 2.
Scatter plots of typical datasets of Examples 1, 2, and 3.

Figure 3.

Left: Plot of the average test errors of the multicategory CLS classifier based on 100 replications with $\gamma = 0.0, 0.1, 0.2, \dots, 1.0$ for Example 1. Right: Plot of the average probability estimation errors of the multicategory CLS classifier based on 100 replications with $\gamma = 0.0, 0.5$, and 1.0 for Example 1.

Figure 4.

Left: Plot of the average test errors of the multicategory CLS classifier based on 100 replications with $\gamma = 0.0, 0.1, 0.2, \dots, 1.0$ for Example 2. Here, the results with ‘tuned γ ’ are the results when γ is tuned among $\{0, 0.5, 1\}$ along with γ . Right: Plot of the average probability estimation errors of the multicategory CLS classifier based on 100 replications with $\gamma = 0.0, 0.5$, and 1.0 for Example 2.

Figure 5.

Left: Plot of the average test errors of the multicategory CLS classifier based on 100 replications with $\gamma = 0.0, 0.1, 0.2, \dots, 1.0$ for Example 3. Right: Plot of the average probability estimation errors of the multicategory CLS classifier based on 100 replications with $\gamma = 0.0, 0.5$, and 1.0 for Example 3.

Figure 6. Plot of the estimated class probabilities for subjects in the testing set of the Leukemia data. The heights of cyan, bright yellow, and dark green bars stand for the estimated probability of ALLB, ALLT, and AML, respectively.

Figure 7.

Heat maps of the Leukemia data. The left panel is for the training set and the right panel is for the testing set. The red and green colors represent high and low expression values respectively. The subjects are displayed in the same order as the Figure 4.

Figure 8.
Plot of CV error for each $\gamma \in \{0, 0.1, \dots, 1\}$ with the λ which is chosen by CV.

Figure 9.

Plot of the estimated class probabilities for subjects in clinical trial data. The height of white, cyan, bright blue, and dark blue stands for the estimated probability of placebo, 10mg, 20mg and 40mg group, respectively.

Table 1

Estimated test errors based on 100 replications for Example 1. The numbers in parenthesis represent the estimated standard errors of the misclassification rates. The rows with tuned 1 and tuned 2 show the results when λ is tuned at the same time with γ among $\{0, 0.1, 0.2, \dots, 1.0\}$, and among $\{0, 0.5, 1\}$, respectively. The Bayes error is 0.2043.

γ	Multicategory CLS		RMSVM	
	Linear	Poly	Linear	Poly
0.0	0.2275(0.0019)	0.2218(0.0016)	0.2821(0.0094)	0.2273 (0.0023)
0.1	0.2268(0.0017)	0.2226(0.0019)	0.2672(0.0080)	0.2286 (0.0024)
0.2	0.2248(0.0015)	0.2227(0.0018)	0.2527(0.0062)	0.2291 (0.0024)
0.3	0.2234(0.0015)	0.2230(0.0019)	0.2425(0.0051)	0.2315 (0.0028)
0.4	0.2211(0.0013)	0.2237(0.0019)	0.2370(0.0043)	0.2323 (0.0029)
0.5	0.2196(0.0012)	0.2248(0.0020)	0.2312(0.0036)	0.2337 (0.0031)
0.6	0.2182(0.0011)	0.2261(0.0021)	0.2282(0.0031)	0.2363 (0.0034)
0.7	0.2171(0.0013)	0.2270(0.0021)	0.2282(0.0032)	0.2390 (0.0034)
0.8	0.2162(0.0010)	0.2269(0.0018)	0.2285(0.0032)	0.2426 (0.0035)
0.9	0.2169(0.0011)	0.2296(0.0019)	0.2307(0.0037)	0.2546 (0.0051)
1.0	0.2180(0.0014)	0.2399(0.0033)	0.2493(0.0061)	0.3101 (0.0066)
tuned 1	0.2204(0.0013)	0.2255(0.0022)	0.2288(0.0029)	0.2344 (0.0032)
tuned 2	0.2215(0.0015)	0.2252(0.0022)	0.2340(0.0036)	0.2331 (0.0035)

Table 2

Estimated test errors based on 100 replications for Example 2. The numbers in parenthesis represent the estimated standard errors of the misclassification rates. The rows with tuned 1 and tuned 2 show the results when λ is tuned at the same time with γ among $\{0, 0.1, 0.2, \dots, 1.0\}$, and among $\{0, 0.5, 1\}$, respectively. The Bayes error is 0.0459 and 0.1538 when $\sigma = 0.5$ and $\sigma = 0.7$, respectively.

	$\sigma = 0.5$		$\sigma = 0.7$	
λ	Linear	Poly	Linear	Poly
0.0	0.2280 (0.0078)	0.1137 (0.0046)	0.3074 (0.0062)	0.2304 (0.0045)
0.1	0.2276 (0.0078)	0.1035 (0.0043)	0.3072 (0.0063)	0.2229 (0.0041)
0.2	0.2274 (0.0078)	0.0927 (0.0038)	0.3078 (0.0063)	0.2139 (0.0036)
0.3	0.2269 (0.0079)	0.0848 (0.0034)	0.3075 (0.0063)	0.2046 (0.0033)
0.4	0.2244 (0.0079)	0.0772 (0.0029)	0.3061 (0.0063)	0.1961 (0.0029)
0.5	0.2177 (0.0078)	0.0712 (0.0025)	0.3009 (0.0062)	0.1898 (0.0027)
0.6	0.1984 (0.0075)	0.0668 (0.0020)	0.2873 (0.0059)	0.1841 (0.0022)
0.7	0.1595 (0.0064)	0.0633 (0.0015)	0.2594 (0.0051)	0.1793 (0.0018)
0.8	0.1172 (0.0050)	0.0612 (0.0012)	0.2255 (0.0041)	0.1761 (0.0015)
0.9	0.0822 (0.0034)	0.0604 (0.0010)	0.1957 (0.0030)	0.1743 (0.0014)
1.0	0.0660 (0.0012)	0.0590 (0.0011)	0.1839 (0.0018)	0.1737 (0.0014)
tuned 1	0.0670 (0.0012)	0.0626 (0.0010)	0.1871 (0.0020)	0.1775 (0.0016)
tuned 2	0.0660 (0.0012)	0.0619 (0.0011)	0.1849 (0.0019)	0.1775 (0.0016)

Table 3

Estimated test errors based on 100 replications for Example 3. The numbers in parenthesis represent the estimated standard errors of the misclassification rates. The rows with tuned 1 and tuned 2 show the results when λ is tuned at the same time with γ among $\{0, 0.1, 0.2, \dots, 1.0\}$, and among $\{0, 0.5, 1\}$, respectively. The Bayes error is 0.0434 and 0.1450 when $\sigma = 0.5$ and $\sigma = 0.7$, respectively.

	$\sigma = 0.5$		$\sigma = 0.7$	
γ	Linear	Poly	Linear	Poly
0.0	0.0916 (0.0031)	0.0963 (0.0038)	0.2056 (0.0029)	0.2065 (0.0038)
0.1	0.0839 (0.0027)	0.0882 (0.0032)	0.2011 (0.0028)	0.2010 (0.0035)
0.2	0.0779 (0.0023)	0.0821 (0.0027)	0.1951 (0.0027)	0.1955 (0.0032)
0.3	0.0732 (0.0020)	0.0771 (0.0023)	0.1899 (0.0024)	0.1891 (0.0028)
0.4	0.0700 (0.0018)	0.0724 (0.0020)	0.1855 (0.0022)	0.1850 (0.0025)
0.5	0.0673 (0.0016)	0.0686 (0.0016)	0.1822 (0.0021)	0.1793 (0.0021)
0.6	0.0655 (0.0015)	0.0656 (0.0015)	0.1788 (0.0020)	0.1761 (0.0019)
0.7	0.0628 (0.0013)	0.0633 (0.0013)	0.1763 (0.0016)	0.1721 (0.0016)
0.8	0.0614 (0.0012)	0.0611 (0.0012)	0.1742 (0.0016)	0.1695 (0.0014)
0.9	0.0606 (0.0012)	0.0597 (0.0010)	0.1772 (0.0016)	0.1689 (0.0015)
1.0	0.1476 (0.0031)	0.0599 (0.0010)	0.2571 (0.0035)	0.1662 (0.0013)
tuned 1	0.0634 (0.0014)	0.0619 (0.0012)	0.1795 (0.0022)	0.1715 (0.0015)
tuned 2	0.0699 (0.0017)	0.0615 (0.0011)	0.1851 (0.0021)	0.1714 (0.0014)