Taylor & Francis
Taylor & Francis Group

Journal of Computational and Graphical Statistics

ISSN: 1061-8600 (Print) 1537-2715 (Online) Journal homepage: http://www.tandfonline.com/loi/ucgs20

Sparse Distance Weighted Discrimination

Boxiang Wang & Hui Zou

To cite this article: Boxiang Wang & Hui Zou (2016) Sparse Distance Weighted Discrimination, Journal of Computational and Graphical Statistics, 25:3, 826-838, DOI: 10.1080/10618600.2015.1049700

To link to this article: http://dx.doi.org/10.1080/10618600.2015.1049700

Accepted author version posted online: 27 May 2015.
Published online: 05 Aug 2016.

SHORT TECHNICAL NOTE

Sparse Distance Weighted Discrimination

Boxiang WANG and Hui ZOU

Distance weighted discrimination (DWD) was originally proposed to handle the data piling issue in the support vector machine. In this article, we consider the sparse penalized DWD for high-dimensional classification. The state-of-the-art algorithm for solving the standard DWD is based on second-order cone programming, however such an algorithm does not work well for the sparse penalized DWD with high-dimensional data. To overcome the challenging computation difficulty, we develop a very efficient algorithm to compute the solution path of the sparse DWD at a given fine grid of regularization parameters. We implement the algorithm in a publicly available R package `sdwd`. We conduct extensive numerical experiments to demonstrate the computational efficiency and classification performance of our method.

Key Words: DWD; High-dimensional classification; SVM.

1. INTRODUCTION

The support vector machine (SVM; Vapnik 1995) is a widely used modern classification method. In the standard binary classification problem, training dataset consists of n pairs, $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, where $\mathbf{x}_i \in \mathbb{R}^p$ and $y_i \in \{-1, 1\}$. The linear SVM seeks a hyperplane $\{\mathbf{x} : \beta_0 + \mathbf{x}^T\boldsymbol{\beta} = 0\}$, which maximizes the smallest margin of all data points:

$$\begin{aligned} \arg\max_{\beta_0,\boldsymbol{\beta}} \quad & \min_i d_i, \\ \text{subject to} \quad & d_i = y_i(\beta_0 + \mathbf{x}_i^T\boldsymbol{\beta}) + \eta_i \geq 0, \ \forall i, \\ & \eta_i \geq 0, \ \forall i, \ \sum_{i=1}^n \eta_i \leq c, \ ||\boldsymbol{\beta}||_2^2 = 1, \end{aligned} \tag{1.1}$$

where d_i is defined as the *margin* of the ith data point, η_i's are slack variables introduced to ensure all margins nonnegative, and $c > 0$ is a tuning parameter controlling the overlap. By using a kernel trick, the SVM can also produce nonlinear decision boundaries by fitting an optimal separating hyperplane in the extended kernel feature space. The readers are

Boxiang Wang (E-mail: *wang3660@umn.edu*) is a PhD student and Hui Zou (E-mail: *zouxx019@umn.edu*) is a Professor in School of Statistics at University of Minnesota, Minneapolis, MN 55455.

Journal of Computational and Graphical Statistics, Volume 25, Number 3, Pages 826–838
DOI: 10.1080/10618600.2015.1049700

referred to Hastie, Tibshirani, and Friedman (2009) for a more detailed explanation of the SVM.

Marron, Todd, and Ahn (2007) noticed that when the SVM is applied on some data with $n < p$, many data points lie on two hyperplanes parallel to the decision boundary. Marron, Todd, and Ahn (2007) referred to this phenomenon as *data pilling* and claimed that the data pilling can "affect the generalization performance of SVM." To overcome this issue, Marron, Todd, and Ahn (2007) proposed a new method called the distance weighted discrimination (DWD), which finds a separating hyperplane minimizing the sum of the inverse margins of all data points:

$$\begin{aligned} &\arg\max_{\beta_0,\boldsymbol{\beta}} \quad \sum_i 1/d_i, \\ &\text{subject to} \quad d_i = y_i(\beta_0 + \mathbf{x}_i^T\boldsymbol{\beta}) + \eta_i \geq 0, \ \forall i, \\ &\qquad\qquad \eta_i \geq 0, \ \forall i, \ \sum_i \eta_i \leq c, \ ||\boldsymbol{\beta}||_2^2 = 1. \end{aligned} \tag{1.2}$$

The initial version of Marron, Todd, and Ahn (2007) also mentioned that the sum of the inverse margins $\sum_i 1/d_i$ could be also replaced by $\sum_i 1/d_i^q$, the qth power of the inverse margins, and this generalized version was used as the definition of the DWD in Hall, Marron, and Neeman (2005). Marron, Todd, and Ahn (2007) asserted that the DWD can avoid the data piling and thereby improve the generalizability. One example (see the group 2 of Figure 3 in Marron, Todd, and Ahn 2007) shows that the DWD has about 5% prediction error whereas the SVM does 15%. Enhancement of the DWD over the SVM can also be exemplified in Hall, Marron, and Neeman (2005) through a novel geometric view. As for the computation of the DWD, Marron, Todd, and Ahn (2007) observed that the DWD is an application of the second-order cone programming and thus can be solved by the primal-dual interior-point methods. The algorithm has been implemented in both Matlab code *http://www.unc.edu/~marron/marron_software.html* and an R package `DWD` (Huang et al. 2012). Other notable developments on DWD include the weighted DWD (Qiao et al. 2010), the multiclass DWD (Huang et al. 2012), and the distance weighted SVM (Qiao and Zhang 2015) which is a combination of DWD and SVM.

In this article, we focus on classification with high-dimensional data where the number of covariates is much larger than the sample size. The standard SVM and DWD are not suitable tools for high-dimensional classification for two reasons. First, based on the scientific hypothesis that only a few important variables affect the outcome, a good classifier for high-dimensional classification should have the ability to select important variables and discard irrelevant ones. However, the standard SVM and DWD use all variables and do not conduct variable selection. Second, because these two classifiers use all variables, they may have very poor classification performance. As explained by Fan and Fan (2008), the bad performance is caused by the error accumulation when estimating too many noise variables in the classifier. Owing to these two considerations, sparse classifiers are generally preferred for high-dimensional classification. In the literature, some penalties have been applied to the SVM to produce sparse SVMs such as the ℓ_1 SVM (Bradley and Mangasarian 1998; Zhu et al. 2004), the SCAD SVM (Zhang et al. 2006), and the elastic-net penalized SVM (Wang, Zhu, and Zou 2006).

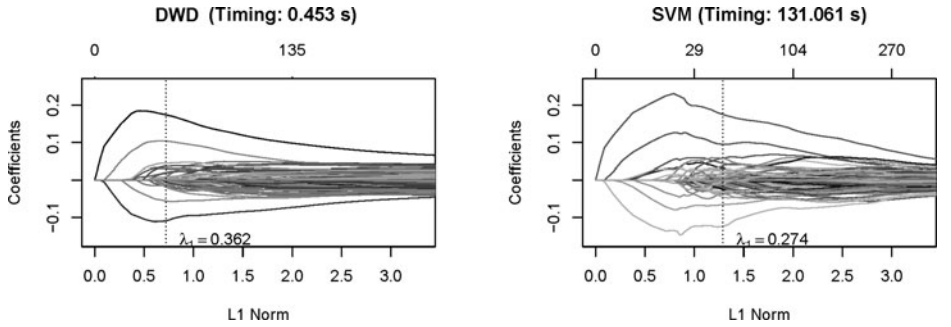

Figure 1. The solution paths for the prostate data ($n = 102$, $p = 6033$) using the elastic-net DWD and the elastic-net SVM. In every method, λ_2 is fixed to be 1. The dashed vertical lines indicate the λ_1 selected by the five-fold cross-validation. Both timings are averaged over 10 runs.

In this work, we consider sparse penalized DWD for high-dimensional classification. The standard DWD uses the ℓ_2 penalty and can be solved by the second-order cone programming. However, the sparse DWD is computationally more challenging and requires a different computing algorithm. To cope with the computational challenges associated with the sparse penalty and high-dimensionality, we derive an efficient algorithm to solve the sparse DWD by combining majorization-minimization principle and coordinate-descent. We have implemented the algorithm in an R package `sdwd`. To give a quick demonstration here, we use the prostate cancer data (Singh et al. 2002, 102 observations and 6033 genes) as an example. The left panel of Figure 1 depicts the solution paths of the elastic-net penalized DWD, and `sdwd` only took 0.453 sec to compute the whole solution path. As comparison, we also used the code in Wang, Zhu, and Zou (2006) to compute the solution path of the elastic-net penalized SVM. We observed that the timing of the sparse SVM was about 290 times larger than that of the sparse DWD.

2. SPARSE DWD

In this section, we present several sparse penalized DWDs. Our formulation follows the ℓ_1 SVM (Zhu et al. 2004). Thus, we first review the derivation process of the ℓ_1 SVM. The standard SVM (1.1) is often rephrased as the following quadratic programming problem (Hastie, Tibshirani, and Friedman 2009):

$$\begin{aligned} \underset{\beta_0,\,\boldsymbol{\beta}}{\arg\min} \quad & ||\boldsymbol{\beta}||_2^2 \\ \text{subject to} \quad & y_i(\beta_0 + \mathbf{x}_i^T\boldsymbol{\beta}) + \eta_i \geq 1,\ \forall i, \\ & \eta_i \geq 0,\ \forall i,\ \sum_{i=1}^{n} \eta_i \leq c. \end{aligned}$$

Moreover, the above constrained minimization problem has an equivalent *loss+penalty* formulation (Hastie, Tibshirani, and Friedman 2009):

$$\underset{\beta_0,\,\boldsymbol{\beta}}{\arg\min}\ \frac{1}{n}\sum_{i=1}^{n}\left[1 - y_i(\beta_0 + \mathbf{x}_i^T\boldsymbol{\beta})\right]_+ + \frac{\lambda_2}{2}||\boldsymbol{\beta}||_2^2.$$

The loss function $[1-t]_+ = \max(1-t,\ 0)$ is the so-called hinge loss in the literature. For the high-dimensional setting, the standard SVM uses all variables because of the ℓ_2 norm penalty used therein. As a result, its performance can be very poor. Zhu et al. (2004) proposed the ℓ_1-norm SVM to fix this issue:

$$\underset{\beta_0,\,\boldsymbol{\beta}}{\arg\ \min}\ \frac{1}{n}\sum_{i=1}^{n}\left[1-y_i(\beta_0+\mathbf{x}_i^T\boldsymbol{\beta})\right]_+ + \lambda_1||\boldsymbol{\beta}||_1.$$

Similarly, we can propose the ℓ_1 penalized DWD. It has been shown that the standard DWD also has a *loss+penalty* formulation (Liu, Zhang, and Wu 2011):

$$\underset{\beta_0,\,\boldsymbol{\beta}}{\arg\ \min}\ \frac{1}{n}\sum_{i=1}^{n} V\left(y_i(\beta_0+\mathbf{x}_i^T\boldsymbol{\beta})\right) + \frac{\lambda_2}{2}||\boldsymbol{\beta}||_2^2,$$

where the loss function is given by

$$V(u) = \begin{cases} 1-u, & \text{if } u \le 1/2,\\ 1/(4u), & \text{if } u > 1/2. \end{cases}$$

Similar to the ℓ_1 SVM, we replace the ℓ_2 norm penalty with the ℓ_1 norm penalty to achieve sparsity in the DWD classifier. Hence, the ℓ_1 DWD is defined by

$$\left(\hat{\beta}_0(\text{lasso}),\,\hat{\boldsymbol{\beta}}(\text{lasso})\right) = \arg\max_{\beta_0,\boldsymbol{\beta}}\frac{1}{n}\sum_{i=1}^{n} V\left(y_i(\beta_0+\mathbf{x}_i^T\boldsymbol{\beta})\right) + \lambda_1||\boldsymbol{\beta}||_1. \tag{2.1}$$

The lasso penalized DWD classification rule is $\text{Sign}(\hat{\beta}_0(\text{lasso}) + \mathbf{x}^T\hat{\boldsymbol{\beta}}(\text{lasso}))$. The above *loss+penalty* formulation of sparse DWD is not new. For example, Zhang and Lin (2013) wrote a review article of classification methods and mentioned the sparse DWD idea in Section 4.4 of their article, although no further technical details were given.

Besides the ℓ_1 norm penalty, we also consider the elastic-net penalty (Zou and Hastie 2005). It is now well known that the elastic-net often outperforms the lasso (ℓ_1 norm penalty) in prediction. Wang, Zhu, and Zou (2006) studied the elastic-net penalized SVM (DrSVM) and showed that the DrSVM performs better than the ℓ_1 norm SVM. Similarly, we propose the elastic-net penalized DWD:

$$\left(\hat{\beta}_0(\text{enet}),\,\hat{\boldsymbol{\beta}}(\text{enet})\right) = \underset{\beta_0,\,\boldsymbol{\beta}}{\arg\ \min}\ \frac{1}{n}\sum_{i=1}^{n} V(y_i(\beta_0+\mathbf{x}_i^T\boldsymbol{\beta})) + P_{\lambda_1,\lambda_2}(\boldsymbol{\beta}), \tag{2.2}$$

where

$$P_{\lambda_1,\lambda_2}(\boldsymbol{\beta}) = \sum_{j=1}^{p}\left(\lambda_1|\beta_j| + \frac{\lambda_2}{2}\beta_j^2\right).$$

The elastic-net penalized DWD classification rule is $\text{Sign}(\hat{\beta}_0(\text{enet}) + \mathbf{x}^T\hat{\boldsymbol{\beta}}(\text{enet}))$. Both λ_1 and λ_2 are important tuning parameters for regularization. In practice, λ_1 and λ_2 are chosen from finite grids by validation or cross-validation.

A further refinement of the elastic-net penalty is the adaptive elastic-net penalty (Zou and Zhang 2009) where we replace the ℓ_1 (lasso) penalty with the adaptive ℓ_1 (lasso)

penalty (Zou 2006). The adaptive lasso penalty produces estimators with the oracle properties. The adaptive elastic-net enjoys the benefits of elastic-net and adaptive lasso. After fitting the elastic-net penalized DWD, we further consider the adaptive elastic-net penalized DWD:

$$\left(\hat{\beta}_0(\text{aenet}), \hat{\boldsymbol{\beta}}(\text{aenet})\right) = \underset{\beta_0,\,\boldsymbol{\beta}}{\arg\min} \frac{1}{n}\sum_{i=1}^{n} V(y_i(\beta_0 + \mathbf{x}_i^T\boldsymbol{\beta})) + \sum_{j=1}^{p}\left(\lambda_1\hat{\omega}_j|\beta_j| + \frac{\lambda_2}{2}\beta_j^2\right), \tag{2.3}$$

and the adaptive weights are computed by

$$\hat{\omega}_j = (|\hat{\beta}_j(\text{enet})| + 1/n)^{-1},$$

where $\hat{\beta}_j(\text{enet})$ is the solution of β_j in (2.2). The adaptive elastic-net penalized DWD classification rule is $\text{Sign}(\hat{\beta}_0(\text{aenet}) + \mathbf{x}^T\hat{\boldsymbol{\beta}}(\text{aenet}))$.

A referee mentioned that Lingsong Zhang, and Xihong Lin gave a presentation at JSM 2010 on the oracle property of sparse DWD. The abstract of their presentation is available at *https://www.amstat.org/meetings/jsm/2010/onlineprogram/AbstractDetails.cfm?abstractid=306064*. Through private communications with Zhang, we learned that the sparse DWD in his talk was the adaptive elastic-net penalized DWD.

3. COMPUTATION

In this section, we propose an intuitive but efficient algorithm for computing the solution paths of the sparse DWD. Our algorithm uses the generalized coordinate descent (GCD) proposed by Yang and Zou (2013). We introduce the algorithm in Section 3.1, the implementation in Section 3.2, and the strict descent property in Section 3.3. The same algorithm solves all the ℓ_1, the elastic-net, and adaptive elastic-net penalized DWDs, while only the elastic-net is focused in the discussion for the sake of presentation.

3.1 Derivation of the Algorithm

Without loss of generality, we assume that the variables $\mathbf{x}_j$ are standardized: $\sum_{i=1}^n x_{ij} = 0$, $\frac{1}{n}\sum_{i=1}^n x_{ij}^2 = 1$, for $j = 1, \ldots, p$. We fix λ_1 and λ_2 and let $u_i = y_i(\tilde{\beta}_0 + \mathbf{x}_i^T\tilde{\boldsymbol{\beta}})$. We focus on β_j's first. For each β_j, we define the coordinate-wise update function:

$$F(\beta_j|\tilde{\boldsymbol{\beta}}, \tilde{\beta}_0) = \frac{1}{n}\sum_{i=1}^{n} V\left(u_i + y_i x_{ij}(\beta_j - \tilde{\beta}_j)\right) + p_{\lambda_1,\lambda_2}(\beta_j). \tag{3.1}$$

Then the standard coordinate descent algorithm suggests cyclically updating

$$\hat{\beta}_j = \underset{\beta_j}{\arg\min}\, F(\beta_j|\tilde{\beta}_0, \tilde{\boldsymbol{\beta}}) \tag{3.2}$$

for each $j = 1, \ldots, p$. However, (3.2) does not have a closed-form solution. The GCD algorithm solves this issue by adopting the MM principle (Hunter and Lange 2004). We

approximate the F function by a quadratic function

$$Q(\beta_j|\tilde{\boldsymbol{\beta}}, \tilde{\beta}_0) = \frac{\sum_{i=1}^n V(u_i)}{n} + \frac{\sum_{i=1}^n V'(u_i) y_i x_{ij}}{n}(\beta_j - \tilde{\beta}_j) + 2(\beta_j - \tilde{\beta}_j)^2 + p_{\lambda_1,\lambda_2}(\beta_j). \tag{3.3}$$

Then we update $\tilde{\beta}_j$ by $\tilde{\beta}_j^{\text{new}}$, the closed-form minimizer of (3.3):

$$\tilde{\beta}_j^{\text{new}} = \frac{S\left(M\tilde{\beta}_j - \frac{1}{n}\sum_{i=1}^n V'(u_i) y_i x_{ij}, \lambda_1\right)}{4 + \lambda_2}, \tag{3.4}$$

where $S(z, r) = \text{sign}(z)(|z| - r)_+$ is the soft-thresholding operator (Donoho and Johnston 1994) and $\omega_+ = \max(\omega, 0)$ is the positive part of ω.

With the intercept similarly updated, Algorithm 1 summarizes the details of the GCD algorithm.

Algorithm 1 *The GCD algorithm for the sparse DWD*

1. Initialize $(\tilde{\beta}_0, \tilde{\boldsymbol{\beta}})$.
2. Cyclic coordinate descent, for $j = 1, 2, \ldots, p$:
   (a) Compute $u_i = y_i(\tilde{\beta}_0 + \mathbf{x}_i^\intercal \tilde{\boldsymbol{\beta}})$.
   (b) Compute $\tilde{\beta}_j^{\text{new}} = \frac{1}{4+\lambda_2} \cdot S\left(4\tilde{\beta}_j - \frac{1}{n}\sum_{i=1}^n V'(u_i) y_i x_{ij}, \lambda_1\right)$.
   (c) Set $\tilde{\beta}_j = \tilde{\beta}_j^{\text{new}}$.
3. Update the intercept term:
   (a) Compute $u_i = y_i(\tilde{\beta}_0 + \mathbf{x}_i^T \tilde{\boldsymbol{\beta}})$.
   (b) Compute $\tilde{\beta}_0^{\text{new}} = \tilde{\beta}_0 - \sum_{i=1}^n V'(u_i) y_i/(4n)$.
   (c) Set $\tilde{\beta}_0 = \tilde{\beta}_0^{\text{new}}$.
4. Repeat Steps 2–3 until convergence of $(\tilde{\beta}_0, \tilde{\boldsymbol{\beta}})$.

3.2 IMPLEMENTATION

We have implemented Algorithm 1 in an R package `sdwd`. We exploit the warm-start, the strong rule, and the active set trick to increase the algorithm speeding. In our implementation, λ_2 is prechosen and we compute the solution path as λ_1 varies.

First, we adopt the warm-start to lead to a faster and more stable algorithm (Friedman et al. 2007). We compute the solutions at a grid of K decreasing λ_1 values, starting at the smallest λ_1 value such that $\tilde{\boldsymbol{\beta}} = 0$. Denote these grid points by $\lambda_1^{[1]}, \ldots, \lambda_1^{[K]}$. With the warm-start trick, we can use the solution at $\lambda_1^{[k]}$ as the initial value (the warm-start) to compute the solution at $\lambda_1^{[k+1]}$.

Specifically, to find $\lambda_1^{[1]}$, we fit a model with a sufficiently large λ_1 and thus $\tilde{\boldsymbol{\beta}} = 0$. Let $\hat{\beta}_0$ be the estimate of the intercept. By the Karush-Kuhn-Tucker (KKT) conditions, $\frac{1}{n}\max_j \left|\sum_{i=1}^n V'(\hat{\beta}_0) y_i x_{ij})\right| \le \lambda_1$, so we can choose

$$\lambda_1^{[1]} = \frac{1}{n}\max_j \left|\sum_{i=1}^n V'(\hat{\beta}_0) y_i x_{ij})\right|.$$

Generally, we use $K = 100$, and $\lambda_1^{[100]} = \epsilon\lambda_1^{[1]}$, where $\epsilon = 10^{-4}$ when $n < p$ and $\epsilon = 10^{-2}$ otherwise. All the other grid points are placed to uniformly distribute on a log scale.

Second, we follow the strong rule (Tibshirani et al. 2010) to improve the computational speed. Suppose $\tilde{\boldsymbol{\beta}}^{[k]}$ and $\tilde{\beta}_0^{[k]}$ are the solutions at $\lambda_1^{[k]}$. After we solve $\tilde{\boldsymbol{\beta}}^{[k]}$ and $\tilde{\beta}_0^{[k]}$, the strong rule claims that any $j \in \{1, \ldots, p\}$ satisfying

$$\left|\frac{1}{n}\sum_{i=1}^n V'(y_i(\hat{\beta}_0^{[k]} + x_i^T \hat{\boldsymbol{\beta}}^{[k]}))y_i x_{ij}\right| < 2\lambda_1^{[k+1]} - \lambda_1^{[k]} \tag{3.5}$$

is likely to be inactive at $\lambda_1^{[k+1]}$, that is, $\hat{\beta}_j^{[k+1]} = 0$. Let $\mathcal{D}$ be the collection of j that satisfies (3.5), and its compliment $\mathcal{D}^C = \{1, \ldots, p\}\backslash\mathcal{D}$. We call $\mathcal{D}^C$ the survival set. If the strong rule guesses correctly, the variables contained in $\mathcal{D}$ are discarded, and we only apply Algorithm 1 to repeat the coordinate descent in the survival set $\mathcal{D}^C$. After computing the solution $\hat{\beta}_0$ and $\hat{\boldsymbol{\beta}}$, we need to check whether some variables are incorrectly discarded. We check this by the KKT condition,

$$\left|\frac{1}{n}\sum_{i=1}^n V'(y_i(\hat{\beta}_0 + x_i^T \hat{\boldsymbol{\beta}}))y_i x_{ij}\right| \le \lambda_1. \tag{3.6}$$

If no $j \in \mathcal{D}$ violates (3.6), $\hat{\beta}_0$ and $\hat{\boldsymbol{\beta}}$ are the solutions at $\lambda_1^{[k+1]}$. We rephrase them as $\tilde{\beta}_0^{[k+1]}$ and $\tilde{\boldsymbol{\beta}}^{[k+1]}$. Otherwise, any incorrectly discarded variable should be added to the survival set $\mathcal{D}^C$. We update $\mathcal{D}$ by $\mathcal{D} = \mathcal{D}/U$, where

$$U = \left\{j : j \in \mathcal{D} \text{ and } \left|\frac{1}{n}\sum_{i=1}^n V'(y_i(\hat{\beta}_0 + x_i^T \hat{\boldsymbol{\beta}}))y_i x_{ij}\right| > \lambda_1\right\}.$$

After each update of $\mathcal{D}$, some incorrectly discarded variables are added back to the survival set.

Third, the active set is also used to boost the algorithm speed. After we apply Algorithm 1 on the survival set $\mathcal{D}^C$, we only apply the coordinate descent on a subset S of $\mathcal{D}^C$ till convergence, where $S = \{j : j \in \mathcal{D}^C \text{ and } \beta_j \neq 0\}$. Then another cycle of coordinate descent is run on $\mathcal{D}^C$ to investigate if the active set S changes. We finish the algorithm if no changes in S; otherwise, we update the active set S and repeat the process.

In Algorithm 1, the margin u_i can be updated conveniently: if β_j is updated by β_j^{new}, we update u_i by $u_i + y_i x_{ij}(\beta_j^{\text{new}} - \beta_j)$.

Last, the default convergence rule in `sdwd` is $4(\tilde{\beta}_j^{\text{new}} - \tilde{\beta}_j)^2 < 10^{-8}$ for all $j = 0, 1, \ldots, p$.

3.3 THE STRICT DESCENT PROPERTY OF ALGORITHM 1

Yang and Zou (2013) showed that the GCD algorithm enjoys descent property. In this section, we also show the GCD algorithm has a stronger statement, the strict descent property, when the GCD is used to solve the sparse DWD. We first elaborate the following majorization result, whose proof is deferred in the appendix.

Lemma 1. $F(\beta_j|\tilde{\boldsymbol{\beta}}, \tilde{\beta}_0)$ is the coordinate-wise update function defined in (3.1), and $Q(\beta_j|\tilde{\boldsymbol{\beta}}, \tilde{\beta}_0)$ is the surrogate function defined in (3.3). We have (3.7) and (3.8):

$$F(\beta_j|\tilde{\boldsymbol{\beta}}, \tilde{\beta}_0) = Q(\beta_j|\tilde{\boldsymbol{\beta}}, \tilde{\beta}_0), \text{ if } \beta_j = \tilde{\beta}_j, \tag{3.7}$$

$$F(\beta_j|\tilde{\boldsymbol{\beta}}, \tilde{\beta}_0) < Q(\beta_j|\tilde{\boldsymbol{\beta}}, \tilde{\beta}_0), \text{ if } \beta_j \neq \tilde{\beta}_j. \tag{3.8}$$

Given $\tilde{\beta}_j^{\text{new}} = \underset{\beta_j}{\arg\min}\, Q(\beta_j|\tilde{\beta}_0, \tilde{\boldsymbol{\beta}})$, and assuming $\tilde{\beta}_j^{\text{new}} \neq \tilde{\beta}_j$, (3.7) and (3.8) imply the strict descent property of the GCD algorithm: $F(\tilde{\beta}_j^{\text{new}}|\tilde{\boldsymbol{\beta}}, \tilde{\beta}_0) < F(\tilde{\beta}_j|\tilde{\boldsymbol{\beta}}, \tilde{\beta}_0)$. It is because $F(\tilde{\beta}_j^{\text{new}}|\tilde{\boldsymbol{\beta}}, \tilde{\beta}_0) < Q(\tilde{\beta}_j^{\text{new}}|\tilde{\boldsymbol{\beta}}, \tilde{\beta}_0) < Q(\tilde{\beta}_j|\tilde{\boldsymbol{\beta}}, \tilde{\beta}_0) = F(\tilde{\beta}_j|\tilde{\boldsymbol{\beta}}, \tilde{\beta}_0)$. Note that the original GCD article only showed $F(\tilde{\beta}_j^{\text{new}}|\tilde{\boldsymbol{\beta}}, \tilde{\beta}_0) \leq F(\tilde{\beta}_j|\tilde{\boldsymbol{\beta}}, \tilde{\beta}_0)$.

The arguments above prove that the objective function F strictly decreases after updating all variables in a cycle, unless the solution does not change after each update. If this is the case, the algorithm stops. We show that the algorithm must stop at the right answer. Assuming $\tilde{\beta}_j = \tilde{\beta}_j^{\text{new}}$ for all j, (3.4) implies:

$$\tilde{\beta}_j = \frac{S(4\tilde{\beta}_j - \frac{1}{n}\sum_{i=1}^n V'(u_i)y_i x_{ij}, \lambda_1)}{4 + \lambda_2}.$$

A straightforward algebra can show that for all j,

$$\frac{1}{n}\sum_{i=1}^n V'(u_i)y_i x_{ij} + \lambda_1 \text{sign}(\beta_j) + \lambda_2\beta_j = 0, \text{ if } \beta_j \neq 0;$$

$$\left|\frac{1}{n}\sum_{i=1}^n V'(u_i)y_i x_{ij}\right| \leq \lambda_1, \text{ if } \beta_j = 0,$$

which is exactly the KKT conditions of the original objective function (2.2). In conclusion, if the objective function does not change after a cycle, the algorithm necessarily converges to the correct solution satisfying the KKT condition.

4. SIMULATION

The simulation in this section aims to support the following three points: (i) the sparse DWD has highly competitive prediction accuracy with the sparse SVM and the sparse logistic regression; (ii) the adaptive elastic-net penalized DWD performs the best in variable selection; and (iii) for the prediction accuracy, no single method among the ℓ_1, the elastic-net, and the adaptive elastic-net penalized DWDs dominate the others in all situations.

In this section, the response variables of all the data are binary. The dimension p of the variables $\mathbf{x}_i$ is always 3000. Within each example, our simulated data consist of a training set, an independent validation set, and an independent test set. The training set contains 50 observations: 25 of them are from the positive class and the other 25 from the negative class. Models are fitted on the training data only, and we use an independent validation set of 50 observations to select the tuning parameters: λ_2 is selected from 10^{-4}, 10^{-3}, 10^{-2}, 0.1, 1, 5, and 10; λ_1 is searched along the solution paths. We compared the prediction accuracy (in percentage) on another independent test dataset of 20,000 observations.

We followed Marron, Todd, and Ahn (2007) to generate the first two examples. In Example 1, the positive class is a random sample from $N_p(\boldsymbol{\mu}_+, \boldsymbol{I}_p)$, where $\boldsymbol{I}_p$ is the p by p identity matrix and $\boldsymbol{\mu}_+$ has all zeros except for 2.2 at the first dimension; the negative class is from $N_p(\boldsymbol{\mu}_-, \boldsymbol{I}_p)$ with $\boldsymbol{\mu}_- = -\boldsymbol{\mu}_+$. In Example 2, 80% of the data are generated from the same distributions as Example 1; for the other 20% of the data, the positive class is drawn from $N_p(\boldsymbol{\mu}_+, \boldsymbol{I}_p)$ and negative class $N_p(-\boldsymbol{\mu}_+, \boldsymbol{I}_p)$, where $\boldsymbol{\mu}_+ = (100, 500, 0, \ldots, 0)$. We obtained the other three examples following Wang, Zhu, and Zou (2006). In Example 3, the positive class has a normal distribution with mean $\boldsymbol{\mu}_+$ and covariance $\boldsymbol{\Sigma} = \boldsymbol{I}_{p\times p}$, where $\boldsymbol{\mu}_+$ has 0.7 in the first five covariates and 0 in others; the negative class has the same distribution except for a different mean $\boldsymbol{\mu}_- = -\boldsymbol{\mu}_+$. In Examples 4 and 5, we consider the cases where the relevant variables are correlated. Two classes have the same distributions except for the covariance,

$$\boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}^{\star}_{5\times 5} & \mathbf{0}_{5\times(p-5)} \\ \mathbf{0}_{(p-5)\times 5} & \boldsymbol{I}_{(p-5)\times(p-5)} \end{pmatrix}.$$

In Example 4, the diagonal elements of $\boldsymbol{\Sigma}^{\star}$ are 1 and the off-diagonal elements are all equal to 0.7. In Example 5, the (i, j)th element of $\boldsymbol{\Sigma}^{\star}$ equals $0.7^{|i-j|}$.

We compared the sparse DWD with the sparse SVM and the sparse logistic regression. Both the DWD and the logistic regression use the ℓ_1, the elastic-net, and the adaptive elastic-net penalties. We used R packages `sdwd` and `gcdnet` (Yang and Zou 2013) to compute the sparse DWDs and the sparse logistic regressions, respectively. The ℓ_1 and the elastic-net SVMs were solved by using the code from Wang, Zhu, and Zou (2006), which does not handle the adaptive elastic-net penalty. Table 1 presents the prediction accuracy results. In the first two examples, the ℓ_1 DWD and the ℓ_1 logistic regression perform the best. We attribute this good performance to the only one nonzero variable in the data, despite 20% of outliers in Example 2. In Examples 3–5, we increase the number of nonzero variables to five. For all models, the elastic-net and the adaptive elastic-net penalties have similar performance, and both of them dominate the ℓ_1 penalties. The elastic-net DWD produces the least prediction error in Examples 4 and 5. Table 2 compares the variable selection. In all cases, the adaptive elastic-net penalties address all relevant variables with relatively few mistakes. The ℓ_1 penalties share similar performance in the first two examples.

5. REAL DATA EXAMPLES

In this section, we analyze four benchmark data. The data Arcene was obtained from Lichman (2013), the breast cancer data from Graham et al. (2010), the LSVT data from

Table 1. Comparisons of mis-classification percentage on 300 training data, 300 validation data, and 20,000 test data, based on 200 replicates. The numbers in parentheses are the standard errors. For each example, the methods with the best performance are marked by black boxes

	DWD			SVM		Logistic		
	ℓ_1	enet	aenet	ℓ_1	enet	ℓ_1	enet	aenet
Example 1	**1.42**	1.47	1.44	1.46	1.50	**1.42**	1.46	1.44
Bayes: 1.39	(0.01)	(0.02)	(0.01)	(0.01)	(0.02)	(0.01)	(0.02)	(0.02)
Example 2	1.14	1.15	1.13	1.16	1.16	**1.11**	1.14	1.15
Bayes: 1.11	(0.01)	(0.01)	(0.01)	(0.01)	(0.01)	(0.01)	(0.01)	(0.02)
Example 3	6.41	6.25	6.21	6.45	**6.15**	6.40	6.21	6.22
Bayes: 5.88	(0.03)	(0.03)	(0.03)	(0.04)	(0.03)	(0.03)	(0.03)	(0.03)
Example 4	22.05	**21.48**	21.54	22.03	21.56	22.00	21.54	21.64
Bayes: 21.10	(0.07)	(0.07)	(0.05)	(0.06)	(0.05)	(0.06)	(0.06)	(0.06)
Example 5	18.91	**18.74**	18.75	18.84	18.78	18.81	18.80	18.77
Bayes: 18.03	(0.07)	(0.05)	(0.05)	(0.06)	(0.05)	(0.06)	(0.05)	(0.05)

Tsanas et al. (2014), and the prostate cancer was from Singh et al. (2002). We randomly split each data with a ratio 1:1 into a training set and a test set. On the training set, we fit the sparse DWD with imposing the elastic-net and the adaptive elastic-net penalties. With the same tuning parameter candidates in the simulation, we used a five-fold cross-validation to find the best pair of (λ_1, λ_2) incurring the least misclassification rate. Then we investigated the prediction accuracy of the selected model on the test set. As comparisons, we considered the sparse SVM and the sparse logistic regression. Every method was trained and tuned in the same way as the sparse DWD. All numerical experiments were carried out on an Intel Core i7-3770 (3.40 GHz) processor.

In Table 3, we reported the average misclassification percentage on the test set from 200 independent splits. We observe that the classifiers achieving the least error in these four datasets are the adaptive elastic-net logistic regression, the elastic-net SVM, the elastic-net, and the adaptive elastic-net DWDs. We also find all the differences are not quite large. For

Table 2. Comparisons of the variable selection. C is the number of selected nonzero variables, and IC is the number of zero variables incorrectly selected into the model. The results are the medians over 200 replicates

	DWD						SVM				Logistic					
	ℓ_1		enet		anet		ℓ_1		enet		ℓ_1		enet		aenet	
	C	IC	C	IC	C	IC	C	IC	C	IC	C	IC	C	IC	C	IC
Example 1	1	0	1	2	1	0	1	0	1	4	1	0	1	4.5	1	0
Example 2	1	0	1	0	1	0	1	0	1	1	1	0	1	0	1	0
Example 3	5	0	5	5	5	0	5	0	5	2.5	5	1	5	7	5	0
Example 4	4	1	5	8.5	5	1.5	4	0	5	7	4	1	5	14	5	2
Example 5	4	1	5	3.5	5	0	4	0	5	2	4	1	5	6.5	5	0

Table 3. The mean misclassification percentage and timings (in seconds) for four benchmark datasets. All the timings include the five-fold cross-validation. The timings of adaptive elastic-net methods include computing the weights. The numbers in parentheses are the standard errors. For each data, the methods with the best prediction accuracy are marked by black boxes

	Arcene $n = 100$, $p = 10,000$		Breast $n = 42$, $p = 22,283$		LSVT $n = 126$, $p = 309$		Prostate $n = 102$, $p = 6033$	
	Error	Time	Error	Time	Error	Time	Error	Time
enet DWD	34.43	123.41	26.50	58.40	16.01	8.28	**10.22**	28.18
	(0.56)	(5.16)	(1.00)	(1.90)	(0.34)	(0.23)	(0.30)	(0.95)
aenet DWD	34.60	200.19	26.86	116.12	**15.92**	13.72	10.26	39.25
	(0.57)	(9.24)	(1.00)	(3.78)	(0.34)	(0.29)	(0.26)	(1.24)
enet logistic	34.16	211.18	24.67	145.35	16.96	10.73	10.65	102.19
	(0.58)	(3.40)	(1.00)	(0.74)	(0.37)	(0.18)	(0.29)	(1.56)
aenet logistic	**34.15**	393.03	25.12	290.31	16.93	17.02	10.75	189.44
	(0.57)	(6.52)	(0.87)	(1.47)	(0.37)	(0.29)	(0.29)	(2.84)
enet SVM	35.10	7410.09	**23.95**	567.43	16.27	63.10	10.56	2508.94
	(0.67)	(1465.68)	(1.00)	(15.19)	(0.37)	(0.77)	(0.36)	(0.77)

the sparse DWD, we get the same message as Marron, Todd, and Ahn (2007) concluded for the standard DWD: "it very often is competitive with the best of the others and sometimes is better." We also notice that the computation of the sparse DWD is the fastest in almost all cases. The timing of the SVM is much longer than other methods. A possible explanation is that the SVM uses the nondifferentiable hinge loss function, which makes the GCD algorithm not suitable for solving the sparse SVM. So far, the best algorithm for the sparse SVM is a LARS-type algorithm (Wang, Zhu, and Zou 2006), which is very different from the GCD algorithm for the sparse DWD and logistic regression. It has been observed that coordinate descent may be faster than the LARS algorithm for solving the lasso penalized least squares (Friedman et al. 2007).

6. DISCUSSION

In this article, we have proposed the sparse DWD for high-dimensional classification and developed an efficient algorithm to compute its solution path. We have shown that the sparse DWD has competitive prediction performance with the sparse SVM and the sparse logistic regression and is often faster to compute with the help of our algorithm. Thus, the sparse DWD is a valuable addition to the toolbox for high-dimensional classification.

The generalized DWD defined by Hall, Marron, and Neeman (2005) minimizes the qth power of the inverse margins. When $q = 1$, it reduces to the usual DWD. For computation considerations, Marron, Todd, and Ahn (2007) chose to fix $q = 1$, because it leads to a second-order cone programming problem. We have found that our algorithm can be readily used to solve the sparse generalized DWD with any positive q. In our numerical study, we tried the generalized DWD with $q = 0.5, 1, 2, 5, 100$ and also tried to use cross-validation to select a data-driven q value. Our numeric results indicated that using different q values

does not lead to significant differences in performance. We opt to leave those results to the technical report version of this article.

APPENDIX: PROOF

Proof of Lemma 1. (3.7) is trivial. To prove (3.8), it suffices to show for any $a \neq b \in \mathbb{R}$,

$$V(a) < V(b) + V'(b)(a-b) + 2(a-b)^2. \tag{A.1}$$

First, it is not hard to check that the first-order derivative $V'(\cdot)$ is Lipschitz continuous, that is, for any $a \neq b$,

$$|V'(a) - V'(b)| < 4|a-b|. \tag{A.2}$$

Let $g(a) = 2a^2 - V(a)$, then (A.2) shows $g'(a) \equiv 4a - V'(a)$ is strictly increasing. Therefore, $g(a)$ is a strictly convex function, and its first-order condition leads to (A.1) directly.

ACKNOWLEDGMENTS

The authors thank the editor, the associate editor, and two referees for their helpful comments and suggestions.

[Received June 2014. Revised March 2015.]

REFERENCES

Bradley, P., and Mangasarian, O. (1998), "Feature Selection Via Concave Minimization and Support Vector Machines," *Machine Learning Proceedings of the Fifteenth International Conference (ICML'98)*, pp. 82–90. [827]

Donoho, D. L., and Johnston, I. M. (1994), "Ideal Spatial Adaptation by Wavelet Shrinkage," *Biometrika*, 81, 425–455. [831]

Fan, J., and Fan, Y. (2008), "High Dimensional Classification Using Featured Annealed Independence Rules," *The Annals of Statistics*, 36, 2605–2637. [827]

Friedman, J., Hastie, T., Höfling, H., and Tibshirani, R. (2007), "Pathwise Coordinate Optimization," *The Annals of Applied Statistics*, 1, 302–332. [831,836]

Graham, K., de las Morenas, A., Tripathi, A., King, C., Kavanah, M., Mendez, J., Stone, M., Slama, J., Miller, M., Antoine, G., Willers, H., Sebastiani, P., and Rosenberg, C. (2010), "Gene Expression in Histologically Normal Epithelium from Breast Cancer Patients and from Cancer-Free Lemmahylactic Mastectomy Patients Shares a Similar Profile," *British Journal of Cancer*, 102, 1284–1293. [834]

Hall, P., Marron, S., and Neeman, A. (2005), "Geometric Representation of High Dimensions, Low Sample Size Data," *Journal of the Royal Statistical Society*, Series B, 67, 427–444. [827,836]

Hastie, T., Tibshirani, R., and Friedman, J. (2009), *The Elements of Statistical Learning: Prediction, Inference, and Data Mining* (2nd ed.), New York: Springer-Verlag. [827,828]

Huang, H., Liu, Y., Du, Y., Perou, C., Hayes, D., Todd, M., and Marron, J. (2013), "Multiclass Distance-Weighted Discrimination," *Journal of Computational and Graphical Statistics*, 222, 953–969. [827]

Huang, H., Lu, X., Liu, Y., Haaland, P., and Marron, J. (2012), "R/DWD: Distance-Weighted Discrimination for Classification, Visualization and Batch Adjustment," *Bioinformatics*, 28, 1182–1183. [827]

Hunter, D., and Lange, K. (2004), "A Tutorial on MM Algorithms," *The American Statistician*, 58, 30–37. [830]

Lichman, M. (2013), "UCI Machine Learning Repository," Irvine, CA: University of California, School of Information and Computer Science. Available at *http://archive.ics.uci.edu/ml*. [834]

Liu, Y., Zhang, H., and Wu, Y. (2011), "Hard or Soft Classification? Large-Margin Unified Machines," *Journal of American Statistical Association*, 106, 166–177. [829]

Marron, J., Todd, M., and Ahn, J. (2007), "Distance-Weighted Discrimination," *Journal of American Statistical Association*, 102, 1267–1271. [827,834,836]

Qiao, X., Zhang, H., Liu, Y., Todd, M., and Marron, J. (2010), "Weighted Distance Weighted Discrimination and Its Asymptotic Properties," *Journal of American Statistical Association*, 105, 401–414. [827]

Qiao, X., and Zhang, L. (2015), "Distance Weighted Support Vector Machine," *Statistics and Its Interface*, 8, 331–345. [827]

Singh, D., Febbo, P., Ross, K., Jackson, D., Manola, J., Ladd, C., Tamayo, P., Renshaw, A., D'Amico, A., and Richie, J. (2002), "Gene Expression Correlates of Clinical Prostate Cancer Behavior," *Cancer Cell*, 1, 203–209. [828,835]

Tibshirani, R., Bien, J., Friedman, J., Hastie, T., Simon, N., Taylor, J., and Tibshirani, R. J. (2010), "Strong Rules for Discarding Predictors in Lasso-Type Problems," *Journal of the Royal Statistical Society*, Series B, 74, 245–266. [832]

Tsanas, A., Little, M., Fox, C., and Ramig, L. (2014), "Objective Automatic Assessment of Rehabilitative Speech Treatment in Parkinson's Disease," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 22, 181–190. [835]

Vapnik, V. (1995), *The Nature of Statistical Learning Theory*, New York: Springer-Verlag. [826]

Wang, L., Zhu, J., and Zou, H. (2006), "The Doubly Regularized Support Vector Machine," *Statistica Sinica*, 16, 589–616. [827,829,834,836]

Yang, Y., and Zou, H. (2013), "An Efficient Algorithm for Computing the HHSVM and its Generalizations," *Journal of Computational and Graphical Statistics*, 22, 396–415. [830,833,834]

Zhang, H. H., Ahn, J., Lin, X., and Park, C. (2006), "Gene Selection Using Support Vector Machines With Nonconvex Penalty," *Bioinformatics*, 22, 88–95. [827]

Zhang, L., and Lin, X. (2013), "Some Considerations of Classification for High Dimension Low-Sample Size Data," *Statistical Methods in Medical Research*, 22, 536–549. [829]

Zhu, J., Rosset, S., Hastie, T., and Tibshirani, R. (2004), "1-Norm Support Vector Machines," *The Annual Conference on Neural Information Processing Systems*, 16, 49–56. [827,828]

Zou, H. (2006), "The Adaptive Lasso and Its Oracle Properties," *Journal of American Statistical Association*, 101, 1418–1429. [830]

Zou, H., and Hastie, T. (2005), "Regularization and Variable Selection Via the Elastic-net," *Journal of the Royal Statistical Society*, Series B, 67, 301–320. [829]

Zou, H., and Zhang, H. H. (2009), "On the Adaptive Elastic-Net With a Diverging Number of Parameters," *Annals of Statistics*, 37, 1733–1751. [829]