Statistica Sinica **33** (2023), 1343-1364
doi:https://doi.org/10.5705/ss.202021.0028

AN EFFICIENT GREEDY SEARCH ALGORITHM FOR HIGH-DIMENSIONAL LINEAR DISCRIMINANT ANALYSIS

Hannan Yang, Danyu Lin and Quefeng Li

University of North Carolina at Chapel Hill

Abstract: High-dimensional classification is an important statistical problem with applications in many areas. One widely used classifier is the linear discriminant analysis (LDA). In recent years, many regularized LDA classifiers have been proposed to solve the problem of high-dimensional classification. However, these methods rely on inverting a large matrix or solving large-scale optimization problems in order to render classification rules, making them computationally prohibitive when the dimension is ultrahigh. With the emergence of big data, it has become increasingly important that we develop more efficient algorithms to solve high-dimensional LDA problems. In this paper, we propose an efficient greedy search algorithm that depends solely on closed-form formulae to learn a high-dimensional LDA rule. We establish a theoretical guarantee of its statistical properties in terms of variable selection and error rate consistency. In addition, we provide an explicit interpretation of the extra information brought by an additional feature in an LDA problem under some mild distributional assumptions. We demonstrate that the computational speed of the new algorithm is significantly better than that of other high-dimensional LDA methods, while maintaining comparable or even better classification performance.

Key words and phrases: Greedy search, high-dimensional classification, linear discriminant analysis, Mahalanobis distance, variable selection.

1. Introduction

Classification–assigning a subject to one of several classes based on certain features–is an important statistical problem. However, the recent emergence of big data poses great challenges, for it requires the efficient use of many features for classification. A simple classifier, namely, linear discriminant analysis (LDA) was widely used before the big data era (Anderson (1962)). However, as Bickel and Levina ((2004) have shown, when the number of features exceeds the sample size, a traditional LDA is no longer applicable, owing to the accumulation of errors when estimating the unknown parameters. To deal with the high-dimensional

Corresponding author: Quefeng Li, Department of Biostatistics, University of North Carolina at Chapel Hill, Chapel Hill, NC 27599, USA. E-mail: quefeng@email.unc.edu.

LDA problem, a number of regularization methods have been proposed (Clemmensen et al. (2011); Witten and Tibshirani (2011); Shao et al. (2011); Cai and Liu (2011); Fan, Feng and Tong (2012); Mai, Zou and Yuan (2012); Han, Zhao and Liu (2013)). Early works by Clemmensen et al. (2011) and Witten and Tibshirani (2011) proposed solving the regularized Fisher's discriminant problem using sparsity-induced penalties. However, at that stage, there was little theory on the statistical properties of such classifiers. These properties were subsequently studied in more detail after additional regularized LDA classifiers (Shao et al. (2011); Cai and Liu (2011); Fan, Feng and Tong (2012); Mai, Zou and Yuan (2012); Han, Zhao and Liu (2013)) had been proposed to deal with the high-dimensional LDA problem.

In general, these methods showed that as long as the unknown population parameters satisfy some sparsity assumptions, building a regularized LDA classifier can yield a consistent classification rule, in the sense that its misclassification error converges to the Bayes error. For example, Shao et al. (2011) showed that if the difference between the population means and the covariance matrices are sparse, using thresholding estimators (Bickel and Levina (2008a)) can still yield a consistent rule. On the other hand, Cai and Liu (2011), Fan, Feng and Tong (2012), and Mai, Zou and Yuan (2012) separately developed distinct consistent rules while assuming that the slope of the Bayes rule is sparse. Han, Zhao and Liu (2013) relaxed the normality assumption on these rules, extending them to more general distributions using a Gaussian copula method.

Although these rules are guaranteed to be consistent, learning the rules is computationally difficult: the rule proposed by Shao et al. (2011) must invert a high-dimensional covariance matrix, and the other aforementioned rules must solve large-scale optimization problems. In particular, it takes a long time to learn these rules when the dimension is ultrahigh. Therefore, we propose a computationally efficient classifier that can be learned without needing to invert large matrices or solve large-scale optimization problems. Our proposed classifier is based solely on closed-form formulae.

Our method is motivated by a recent study (Li and Li (2018)) on the Bayes error of the LDA problem. Li and Li (2018) showed that the Bayes error always decreases when new features are added to the Bayes rule, and that this decrease is fully characterized by the increment of the Mahalanobis distance between the two classes. We therefore develop an efficient greedy search algorithm to learn the increment of the Mahalanobis distance. Unlike many other methods, this algorithm does not estimate all population parameters; instead, it selects discriminative features in a sequential way, and computes the classification rule as it

does so. Our method is therefore scalable for ultrahigh-dimensional LDA problems. We show that the proposed method admits both variable selection and error rate consistency when the classes follow some general distributions.

To prove these theoretical properties, we first establish a concentration result for the estimated increment of the Mahalanobis distance and the true increment. This result characterizes the trade-off between the gains of using more features for classification and the additional estimation error it produces. We also offer an explicit interpretation of how much information a new feature adds to the LDA problem; this interpretation holds for a general class of distributions, and is new to the LDA literature. We then show that if the slope of the Bayes rule is exactly sparse, our method can asymptotically recover its nonzero elements, and that our method's misclassification error converges to the Bayes error. These results also hold under a general class of elliptical distributions. We demonstrate numerically that our method achieves comparable or even better classification performance with a much shorter training time than other LDA-based methods.

The rest of the paper is organized as follows. Section 2 presents our efficient greedy search algorithm. Section 3 describes the statistical properties of the proposed method in terms of its variable selection and error rate consistency. Section 4 relaxes the normality assumption, and shows that the statistical properties hold for a general class of distributions. Section 5 presents extensive numerical studies that compare the proposed method with other existing methods, demonstrating the proposed method's superiority in terms of both computational efficiency and classification performance under various scenarios. In Section 6, we apply our method to microarray data to classify cancer subtypes, and show that our method renders a more meaningful classification rule. All technical proofs are given in the Supplementary Material.

2. An Efficient Greedy Search Algorithm

Consider a binary classification problem where the class label $Y \in \{0, 1\}$ has a prior distribution of $P(Y = k) = \pi_k$, for $k = 0, 1$. Suppose $\boldsymbol{x}_k \in \mathbb{R}^p$ denotes a p-dimensional vector of features from the kth class that follows the normal distribution $N(\boldsymbol{\mu}_k, \boldsymbol{\Sigma})$, where $\boldsymbol{x}_0$ and $\boldsymbol{x}_1$ are assumed to be independent. The Bayes rule of this classification problem is given by $D_{Bayes}(\boldsymbol{x}) = I(\boldsymbol{\delta}^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}) \leq \log(\pi_1/\pi_0))$, where $\boldsymbol{\delta} = \boldsymbol{\mu}_0 - \boldsymbol{\mu}_1$, $\boldsymbol{\mu} = (1/2)(\boldsymbol{\mu}_0 + \boldsymbol{\mu}_1)$, and $\boldsymbol{x}$ is a new observation. The corresponding Bayes error is given by $R_{Bayes} = \Phi(-\sqrt{\Delta_p}/2)$, where $\Delta_p = \boldsymbol{\delta}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\delta}$ is the Mahalanobis distance between the centroids of the two classes and Φ is the cumulative distribution function of the standard normal

distribution.

In practice, the Bayes rule is unknown. A classification rule is learned from the training data $\boldsymbol{X} = \{\boldsymbol{x}_{ki}; k = 0, 1; i = 1, \ldots, n_k\}$, where $\boldsymbol{x}_{ki}$ are independent and identically distributed (i.i.d) samples from the kth class and n_k is the sample size of the kth class, with $n = n_0 + n_1$. Then, the rule is applied to classify a new observation $\boldsymbol{x}$, which is assumed to be independent of the training data. For the classical LDA method, the unknown parameters in the Bayes rule are replaced with their maximum likelihood estimators; this LDA rule has the form

$$D_{LDA}(\boldsymbol{x}) = I\left(\widehat{\boldsymbol{\delta}}^T \widehat{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{x} - \widehat{\boldsymbol{\mu}}) \le \log\left(\frac{\widehat{\pi}_1}{\widehat{\pi}_0}\right)\right),$$

where

$$\widehat{\pi}_k = \frac{n_k}{n}, \quad \widehat{\boldsymbol{\mu}}_k = \frac{1}{n_k}\sum_{i=1}^{n_k} \boldsymbol{x}_{ki}, \quad \widehat{\boldsymbol{\delta}} = \widehat{\boldsymbol{\mu}}_0 - \widehat{\boldsymbol{\mu}}_1, \tag{2.1}$$

$$\widehat{\boldsymbol{\mu}} = \frac{1}{2}(\widehat{\boldsymbol{\mu}}_0 + \widehat{\boldsymbol{\mu}}_1), \quad \widehat{\boldsymbol{\Sigma}} = \frac{1}{n}\sum_{k=0}^{1}\sum_{i=1}^{n_k}(\boldsymbol{x}_{ki} - \widehat{\boldsymbol{\mu}}_k)(\boldsymbol{x}_{ki} - \widehat{\boldsymbol{\mu}}_k)^T. \tag{2.2}$$

However, in the high-dimensional setting, where $p > n$, the classical LDA method is no longer feasible, because $\widehat{\boldsymbol{\Sigma}}$ is not invertible. Even if we replace $\widehat{\boldsymbol{\Sigma}}^{-1}$ with a generalized matrix inverse, Bickel and Levina ((2004) showed that the resulting rule has an asymptotic misclassification error of 1/2, which is as bad as random guessing. This is essentially due to the error accumulation when estimating the high-dimensional parameters in the classifier. To avoid this issue in the high-dimensional setting, many regularized methods have been proposed (Clemmensen et al. (2011); Witten and Tibshirani (2011); Shao et al. (2011); Cai and Liu (2011); Fan, Feng and Tong (2012); Mai, Zou and Yuan (2012); Han, Zhao and Liu (2013)). In particular, Shao et al. (2011) proposed the sparse linear discriminant analysis (SLDA) rule

$$D_{SLDA}(\boldsymbol{x}) = I\left(\widetilde{\boldsymbol{\delta}}^T \widetilde{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{x} - \widehat{\boldsymbol{\mu}}) \le \log\left(\frac{\widehat{\pi}_1}{\widehat{\pi}_0}\right)\right),$$

where $\widetilde{\boldsymbol{\delta}}$ and $\widetilde{\boldsymbol{\Sigma}}$ are the thresholding estimators. Here, $\widetilde{\boldsymbol{\delta}} = (\widetilde{\delta}_j)$, with $\widetilde{\delta}_j = \widehat{\delta}_j I(|\widehat{\delta}_j| > t_\delta)$ and $\widehat{\delta}_j$ is the jth element of $\widehat{\boldsymbol{\delta}}$, and $\widetilde{\boldsymbol{\Sigma}} = (\widetilde{\sigma}_{ij})$, with $\widetilde{\sigma}_{ii} = \widehat{\sigma}_{ii}$, $\widetilde{\sigma}_{ij} = \widehat{\sigma}_{ij} I(|\widehat{\sigma}_{ij}| > t_\sigma)$, for $i \ne j$, and $\widehat{\sigma}_{ij}$ is the (i, j)th element of $\widehat{\boldsymbol{\Sigma}}$. They showed that the SLDA's misclassification error still converges to the Bayes error, given that the thresholds t_δ and t_σ are chosen properly and given some sparsity conditions on both $\boldsymbol{\delta}$ and $\boldsymbol{\Sigma}$. Instead of separately estimating $\boldsymbol{\delta}$ and $\boldsymbol{\Sigma}$, as the

SLDA does, two other methods directly estimate the slope of the Bayes rule $\boldsymbol{\beta} = \boldsymbol{\Sigma}^{-1}\boldsymbol{\delta}$ by solving convex optimization problems. For example, the linear programming discriminant (LPD) method (Cai and Liu (2011)) estimates $\boldsymbol{\beta}$ by solving

$$\widehat{\boldsymbol{\beta}}_{LPD} = \underset{\boldsymbol{\beta}\in\mathbb{R}^p}{\operatorname{argmin}}\|\boldsymbol{\beta}\|_1 \text{ subject to } \|\widehat{\boldsymbol{\Sigma}}\boldsymbol{\beta} - \widehat{\boldsymbol{\delta}}\|_\infty \le \lambda,$$

where λ is a tuning parameter and $(\widehat{\boldsymbol{\delta}}, \widehat{\boldsymbol{\Sigma}})$ is defined in (2.1) and (2.2). The regularized optimal affine discriminant (ROAD) method (Fan, Feng and Tong (2012)) estimates $\boldsymbol{\beta}$ by solving

$$\widehat{\boldsymbol{\beta}}_{ROAD} = \underset{\boldsymbol{\beta}\in\mathbb{R}^p}{\operatorname{argmin}} \ \frac{1}{2}\boldsymbol{\beta}^T\widehat{\boldsymbol{\Sigma}}\boldsymbol{\beta} + \lambda\|\boldsymbol{\beta}\|_1 + \frac{\gamma}{2}(\boldsymbol{\beta}^T\widehat{\boldsymbol{\delta}} - 1)^2,$$

where λ and γ are tuning parameters and $(\widehat{\boldsymbol{\delta}}, \widehat{\boldsymbol{\Sigma}})$ is defined in (2.1) and (2.2). Then, replacing $\widetilde{\boldsymbol{\Sigma}}^{-1}\widetilde{\boldsymbol{\delta}}$ in the SLDA rule with $\widehat{\boldsymbol{\beta}}_{LPD}$ or $\widehat{\boldsymbol{\beta}}_{ROAD}$ gives the corresponding LPD or ROAD rule. Both papers showed that the resulting misclassification errors converge asymptotically to the Bayes error, given some sparsity condition on $\boldsymbol{\beta}$.

However, these methods all rely on evaluating and inverting a large matrix or solving a large-scale optimization problem. When p is huge, it can become computationally expensive to run these methods. Instead, we propose an efficient greedy search algorithm that does not require inverting a matrix or solving an optimization problem. Moreover, our method does not need to evaluate the whole sample covariance matrix in advance, but rather computes its elements as it goes, depending on which features enter the classification rule. We compare the computational complexity of these methods with that of ours later in this section. As shown in the numerical studies, our method has a much shorter learning time than these methods do, and even better classification performance.

The Bayes error of the LDA problem is fully characterized by the Mahalanobis distance Δ_p. Recently, Li and Li (2018) proved that Δ_p is a monotonically increasing function of p, which implies that the Bayes error always decreases when more features are involved. Therefore, we propose a greedy search algorithm that operates by learning the increment of the Mahalanobis distance. At each step of our algorithm, we seek the variable that results in the largest increment of the Mahalanobis distance. Such a variable can be regarded as the most informative, given those selected in the previous steps. We terminate the iterations when the increment is smaller than a predefined threshold. We show that the iterations are based on closed-form formulae, and the algorithm does not need to compute

the whole covariance matrix; therefore, it is computationally efficient.

Let S be an arbitrary subset of $\{1, \ldots, p\}$, and s be the size of S. Let $\Delta_s = \boldsymbol{\delta}_S^T \boldsymbol{\Sigma}_{SS}^{-1} \boldsymbol{\delta}_S$ be the Mahalanobis distance involving only variables in S, where $\boldsymbol{\delta}_S$ is a subvector of $\boldsymbol{\delta}$ and $\boldsymbol{\Sigma}_{SS}$ is a submatrix of $\boldsymbol{\Sigma}$ with indices in S. For an arbitrary $c \notin S$, let

$$\Delta_{s+1} = \begin{pmatrix} \boldsymbol{\delta}_S^T & \delta_c \end{pmatrix} \begin{pmatrix} \boldsymbol{\Sigma}_{SS} & \boldsymbol{\Sigma}_{Sc} \\ \boldsymbol{\Sigma}_{Sc}^T & \sigma_{cc} \end{pmatrix}^{-1} \begin{pmatrix} \boldsymbol{\delta}_S \\ \delta_c \end{pmatrix}$$

be the Mahalanobis distance by adding a new variable indexed by c, and let $\theta_{Sc} = \Delta_{s+1} - \Delta_s$ be the increment of the Mahalanobis distance. Using an argument analogous to Proposition 1 of Li and Li (2018), we can show that

$$\theta_{Sc} = \frac{(\delta_c - \boldsymbol{\Sigma}_{Sc}^T \boldsymbol{\Omega}_{SS} \boldsymbol{\delta}_S)^2}{\sigma_{cc} - \boldsymbol{\Sigma}_{Sc}^T \boldsymbol{\Omega}_{SS} \boldsymbol{\Sigma}_{Sc}} \geq 0, \tag{2.3}$$

where $\boldsymbol{\Omega}_{SS} = \boldsymbol{\Sigma}_{SS}^{-1}$. A proof of (2.3) is given in the Appendix. Moreover, under the normality assumption, we find that θ_{Sc} has a clear interpretation. Let $\boldsymbol{z} = \boldsymbol{x}_0 - \boldsymbol{x}_1$. Using the conditional distribution of the multivariate normal distribution, we can easily show that

$$\theta_{Sc} = \frac{2\{\mathrm{E}(z_c | \boldsymbol{z}_S = \boldsymbol{0})\}^2}{\mathrm{Var}(z_c | \boldsymbol{z}_S = \boldsymbol{0})},$$

where z_c is the cth element of $\boldsymbol{z}$ and $\boldsymbol{z}_S$ is the subvector of $\boldsymbol{z}$ with indices in S. This result shows that the contribution of a new variable does not depend on its marginal difference between the two classes (i.e., $\mathrm{E}(z_c)$), but rather on its effect size, conditional on other variables (i.e., the standardized $\mathrm{E}(z_c | \boldsymbol{z}_S = \boldsymbol{0})$). In the extreme case in which there is no difference in the cth variable between the two classes (i.e., $\mathrm{E}(z_c) = 0$), adding such a variable to those in S can still reduce the Bayes error if $\mathrm{E}(z_c | \boldsymbol{z}_S = \boldsymbol{0}) \neq 0$. This interpretation seems to be new in the LDA literature. In Section 4, we show that this interpretation not only holds for the normal distribution, but also holds for all elliptical distributions.

In practice, θ_{Sc} is unknown. We propose a greedy search algorithm based on learning θ_{Sc} from the training data. From (2.3), if we replace $\boldsymbol{\delta}$ and $\boldsymbol{\Sigma}$ with the corresponding estimators $\widehat{\boldsymbol{\delta}}$ and $\widehat{\boldsymbol{\Sigma}}$ in (2.1) and (2.2), we can easily obtain an estimator of θ_{Sc}. However, this naive method requires the computation of all elements of $\widehat{\boldsymbol{\Sigma}}$ and the inversion of its submatrices. We show that there is a more efficient method of computing $\widehat{\theta}_{Sc}$ that computes elements of $\widehat{\boldsymbol{\Sigma}}$ as it goes, with no need to invert a matrix.

At the initial step, we set the selected set $\widehat{S}_0 = \emptyset$. For all $1 \leq c \leq p$, we

calculate $\widehat{\delta}_c^2/\widehat{\sigma}_{cc}$, where $\widehat{\delta}_c$ is the cth element of $\widehat{\boldsymbol{\delta}}$ and $\widehat{\sigma}_{cc}$ is the (c,c)th element of $\widehat{\boldsymbol{\Sigma}}$. We choose $\widehat{s}_1$ to be the index such that $\widehat{\delta}_c^2/\widehat{\sigma}_{cc}$ is maximized, and set the selected set $\widehat{S}_1 = \{\widehat{s}_1\}$ and the candidate set $\widehat{C}_1 = \{1,\ldots,p\}\backslash\{\widehat{s}_1\}$. To simplify the calculations in the subsequent steps, we compute and store $\widehat{\boldsymbol{\Omega}}_1 = \widehat{\sigma}^{-1}_{\widehat{s}_1\widehat{s}_1}$ and the submatrix $\widehat{\boldsymbol{\Sigma}}_{\widehat{S}_1\widehat{C}_1}$, that is, the sample covariance of the selected and candidate variables. At this step, $\widehat{\boldsymbol{\Omega}}_1$ and $\widehat{\boldsymbol{\Sigma}}_{\widehat{S}_1\widehat{C}_1}$ are a scalar and a vector of $p-1$ elements, respectively. As shown below, storing these two matrices is the key to enabling a fast computation. At the kth step, we compute

$$\widehat{\theta}_{\widehat{S}_{k-1}c} = (\widehat{\delta}_c - \widehat{\boldsymbol{\Sigma}}^T_{\widehat{S}_{k-1}c}\widehat{\boldsymbol{\Omega}}_{k-1}\widehat{\boldsymbol{\delta}}_{\widehat{S}_{k-1}})^2(\widehat{\sigma}_{cc} - \widehat{\boldsymbol{\Sigma}}^T_{\widehat{S}_{k-1}c}\widehat{\boldsymbol{\Omega}}_{k-1}\widehat{\boldsymbol{\Sigma}}_{\widehat{S}_{k-1}c})^{-1},$$

for all $c \in \widehat{C}_{k-1}$. Because $\widehat{\delta}_c$, $\widehat{\sigma}_{cc}$, $\widehat{\boldsymbol{\Sigma}}_{\widehat{S}_{k-1}\widehat{C}_{k-1}}$, and $\widehat{\boldsymbol{\Omega}}_{k-1}$ have all been stored in previous steps, computing $\widehat{\theta}_{\widehat{S}_{k-1}c}$ is fast. Then, we select $\widehat{s}_k$ to be the index that maximizes $\widehat{\theta}_{\widehat{S}_{k-1}c}$ for all $c \in \widehat{C}_{k-1}$, and let $\widehat{S}_k = \widehat{S}_{k-1}\cup\{\widehat{s}_k\}$ and $\widehat{C}_k = \widehat{C}_{k-1}\backslash\{\widehat{s}_k\}$. Next, we update $\widehat{\boldsymbol{\Omega}}_k$, the estimated precision matrix of all selected variables in $\widehat{S}_k$. Using the Woodbury matrix identity, we have

$$\widehat{\boldsymbol{\Omega}}_k = \widehat{\boldsymbol{\Sigma}}^{-1}_{\widehat{S}_k\widehat{S}_k} = \begin{pmatrix} \widehat{\boldsymbol{\Omega}}_{k-1} + \widehat{\alpha}_k\widehat{\boldsymbol{\rho}}_k\widehat{\boldsymbol{\rho}}_k^T & -\widehat{\alpha}_k\widehat{\boldsymbol{\rho}}_k \\ -\widehat{\alpha}_k\widehat{\boldsymbol{\rho}}_k^T & \widehat{\alpha}_k \end{pmatrix},$$

where $\widehat{\boldsymbol{\rho}}_k = \widehat{\boldsymbol{\Omega}}_{k-1}\widehat{\boldsymbol{\Sigma}}_{\widehat{S}_{k-1}\widehat{s}_k}$ and $\widehat{\alpha}_k = (\widehat{\sigma}_{\widehat{s}_k\widehat{s}_k} - \widehat{\boldsymbol{\Sigma}}^T_{\widehat{S}_{k-1}\widehat{s}_k}\widehat{\boldsymbol{\Omega}}_{k-1}\widehat{\boldsymbol{\Sigma}}_{\widehat{S}_{k-1}\widehat{s}_k})^{-1}$. Note that $\widehat{\sigma}_{\widehat{s}_k\widehat{s}_k}$, $\widehat{\boldsymbol{\Omega}}_{k-1}$, and $\widehat{\boldsymbol{\Sigma}}_{\widehat{S}_{k-1},\widehat{s}_k}$ can all be read directly from previously stored objects, allowing $\widehat{\boldsymbol{\Omega}}_k$ to be computed efficiently. Next, we update $\widehat{\boldsymbol{\Sigma}}_{\widehat{S}_k\widehat{C}_k}$ by letting $\widehat{\boldsymbol{\Sigma}}_{\widehat{S}_k\widehat{C}_k} = (\widehat{\boldsymbol{\Sigma}}_{\widehat{C}_k\widehat{S}_{k-1}}\ \widehat{\boldsymbol{\Sigma}}_{\widehat{C}_k\widehat{s}_k})^T$. This quantity is needed to calculate $\widehat{\theta}_{\widehat{S}_kc}$ in the next iteration. Note that $\widehat{\boldsymbol{\Sigma}}_{\widehat{C}_k\widehat{S}_{k-1}}$ is the submatrix of $\widehat{\boldsymbol{\Sigma}}_{\widehat{C}_{k-1}\widehat{S}_{k-1}}$ without its $\widehat{s}_k$th row, which has been stored in the $(k-1)$th iteration. Therefore, we need only compute $\widehat{\boldsymbol{\Sigma}}_{\widehat{C}_k\widehat{s}_k} \in \mathbb{R}^{p-k}$, which is the sample covariance between the newly selected variable $\widehat{s}_k$ and the candidate variables in $\widehat{C}_k$. We calculate the number of operations computed at the kth iteration. First, it takes $O(k^2(p-k+1)) = O(k^2p)$ operations to calculate $\widehat{\theta}_{\widehat{S}_{k-1}c}$. Then, we obtain $\widehat{\boldsymbol{\Sigma}}_{\widehat{C}_k\widehat{s}_k}$ at a cost of $O(np)$ operations. Finally, we update $\widehat{\boldsymbol{\Omega}}_k$ at a cost of $O(k^2)$ operations. Thus, at the kth iteration, our algorithm costs $O(np)$ operations. Therefore, up to the kth iteration, the total computational cost is $O(knp)$. As discussed in Section 3, the total number of iterations is close to K, which is the number of nonzero elements in $\boldsymbol{\beta}$ and is much smaller than n. Thus, the total computational cost of our algorithm is $O(Knp)$.

On the other hand, both the SLDA and the LPD need to first compute $\widehat{\mathbf{\Sigma}}$, which requires $O(np^2)$ operations. For the SLDA, it requires additional operations to obtain the regularized estimators $\widetilde{\mathbf{\Sigma}}$ and $\widetilde{\boldsymbol{\delta}}$. Futhermore, inverting $\widetilde{\mathbf{\Sigma}}$ costs another $O(p^{2+\epsilon})$ operations for some $\epsilon \in (0, 1]$, depending on the algorithm used to invert the matrix. Finally, computing the product $\widetilde{\mathbf{\Sigma}}^{-1}\widetilde{\boldsymbol{\delta}}$ costs $O(p^2)$ operations. Thus, the total computational cost of the SLDA is at least $\max\{O(np^2), O(p^{2+\epsilon})\}$, which is much slower than ours when p is big. For the LPD, the optimization problem can be solved using the primal-dual interior-point method (Candes and Tao (2007)). As shown by Candes and Tao (2007), when $p \gg n$, each iteration requires solving an $n \times n$ linear system ($O(n^2)$) and updating the matrix for the system ($O(np^2)$), which also requires evaluating $\widehat{\mathbf{\Sigma}}$. Such an evaluation already takes $O(np^2)$ operations. Therefore, let T be the number of iterations for the interior-point method to converge. The total computational cost for the LPD is $\max\{O(Tnp^2), O(Tn^2)\}$, which is clearly slower than ours. For the ROAD, if one chooses to evaluate $\widehat{\mathbf{\Sigma}}$ first and then solve the optimization problem, the computational cost is at least $O(np^2)$. A computationally more efficient solution is to use the fast iterative shrinkage-thresholding algorithm (FISTA) proposed by Beck and Teboulle (2009). In each iteration of the FISTA, the cost of computing the gradient is $O(np)$ if we use a store-and-compute method that is more efficient than evaluating $\widehat{\mathbf{\Sigma}}$ before the iterations start. Furthermore, Theorem 4.4 in Beck and Teboulle (2009) shows that the FISTA needs at least $O(n^{1/4})$ iterations to converge. Thus, the total computational cost for the FISTA to solve the ROAD problem is at least $O(n^{5/4}p)$. Therefore, our method is still faster than the FISTA, especially when K is small. One may also choose to use the covariance-based method (Friedman, Hastie and Tibshirani (2010)) to calculate the gradient. However, its efficiency depends on the choice of the tuning parameters and the initial value, so that the total computational cost is difficult to quantify, in general.

In conclusion, the greedy search algorithm keeps track of the index sets of selected variables $\widehat{S}_k$ and candidate variables $\widehat{C}_k$, and iteratively updates $\widehat{\mathbf{\Omega}}_k$ and $\widehat{\mathbf{\Sigma}}_{\widehat{S}_k\widehat{C}_k}$. It does not need to compute the whole $\widehat{\mathbf{\Sigma}}$ in advance; instead, it computes its elements as it goes based on selected and candidate variables. It relies solely on closed-form formulae to learn the increments of the Mahalanobis distance, without requiring a matrix inversion or solving an optimization problem. The algorithm terminates when no candidate variable produces an increment of the Mahalanobis distance of at least τ, which is a predefined stopping threshold that can be regarded as a tuning parameter that must be tuned using cross-validation. The greedy search algorithm is summarized in Algorithm 1.

Denote $\widehat{\mathcal{M}}$ and $\widehat{\boldsymbol{\Omega}}_{\widehat{\mathcal{M}}}$ as the last $\widehat{S}_k$ and $\widehat{\boldsymbol{\Omega}}_k$, respectively, when the greedy search algorithm terminates. We let $\widehat{\boldsymbol{\beta}}_{\widehat{\mathcal{M}}} = \widehat{\boldsymbol{\Omega}}_{\widehat{\mathcal{M}}}\widehat{\boldsymbol{\delta}}_{\widehat{\mathcal{M}}}$ and propose the following greedy search linear discriminant analysis (GS-LDA) rule:

$$D_{GS\text{-}LDA}(\boldsymbol{x}) = I\bigg(\widehat{\boldsymbol{\beta}}^T_{\widehat{\mathcal{M}}}(\boldsymbol{x}_{\widehat{\mathcal{M}}} - \widehat{\boldsymbol{\mu}}_{\widehat{\mathcal{M}}}) \le \log\left(\frac{\widehat{\pi}_1}{\widehat{\pi}_0}\right)\bigg),$$

where $\boldsymbol{x}_{\widehat{\mathcal{M}}}$ and $\widehat{\boldsymbol{\mu}}_{\widehat{\mathcal{M}}}$ are subvectors of $\boldsymbol{x}$ and $\widehat{\boldsymbol{\mu}}$, respectively, with indices in $\widehat{\mathcal{M}}$.

Algorithm 1 The greedy search algorithm.

Initialization: Compute and store $\widehat{\boldsymbol{\delta}}$ and the diagonal elements of $\widehat{\boldsymbol{\Sigma}}$ using (2.1) and (2.2).
Set $\widehat{s}_1 = \text{argmax}_{j\le p}\,\widehat{\delta}_j^2/\widehat{\sigma}_{jj}$, $\widehat{S}_1 = \{\widehat{s}_1\}$,
$\widehat{C}_1 = \{1,\ldots,p\}\backslash\{\widehat{s}_1\}$, $\widehat{\boldsymbol{\Omega}}_1 = \widehat{\sigma}^{-1}_{\widehat{s}_1\widehat{s}_1}$.
Compute and store $\widehat{\boldsymbol{\Sigma}}_{\widehat{S}_1\widehat{C}_1}$.

At the kth iteration:

Let $\widehat{\theta}_{\widehat{S}_{k-1}c} = (\widehat{\delta}_c - \widehat{\boldsymbol{\Sigma}}^T_{\widehat{S}_{k-1}c}\widehat{\boldsymbol{\Omega}}_{k-1}\widehat{\boldsymbol{\delta}}_{\widehat{S}_{k-1}})^2/(\widehat{\sigma}_{cc} - \widehat{\boldsymbol{\Sigma}}^T_{\widehat{S}_{k-1}c}\widehat{\boldsymbol{\Omega}}_{k-1}\widehat{\boldsymbol{\Sigma}}_{\widehat{S}_{k-1}c})$ for all $c \in \widehat{C}_{k-1}$.

Set $\widehat{s}_k = \text{argmax}_{c\in\widehat{C}_{k-1}}\,\widehat{\theta}_{\widehat{S}_{k-1}c}$, $\widehat{S}_k = \widehat{S}_{k-1}\cup\{\widehat{s}_k\}$, $\widehat{C}_k = \widehat{C}_{k-1}\backslash\{\widehat{s}_k\}$,

$\widehat{\boldsymbol{\rho}}_k = \widehat{\boldsymbol{\Omega}}_{k-1}\widehat{\boldsymbol{\Sigma}}_{\widehat{S}_{k-1}\widehat{s}_k}$, $\widehat{\alpha}_k = (\widehat{\sigma}_{\widehat{s}_k\widehat{s}_k} - \widehat{\boldsymbol{\Sigma}}^T_{\widehat{S}_{k-1}\widehat{s}_k}\widehat{\boldsymbol{\Omega}}_{k-1}\widehat{\boldsymbol{\Sigma}}_{\widehat{S}_{k-1}\widehat{s}_k})^{-1}$,

$\widehat{\boldsymbol{\Sigma}}_{\widehat{S}_k\widehat{C}_k} = (\widehat{\boldsymbol{\Sigma}}_{\widehat{C}_k\widehat{S}_{k-1}}\ \widehat{\boldsymbol{\Sigma}}_{\widehat{C}_k\widehat{s}_k})^T$

and $\widehat{\boldsymbol{\Omega}}_k = \begin{pmatrix} \widehat{\boldsymbol{\Omega}}_{k-1} + \widehat{\alpha}_k\widehat{\boldsymbol{\rho}}_k\widehat{\boldsymbol{\rho}}_k^T & -\widehat{\alpha}_k\widehat{\boldsymbol{\rho}}_k \\ -\widehat{\alpha}_k\widehat{\boldsymbol{\rho}}_k^T & \widehat{\alpha}_k \end{pmatrix}$.

Iterate until $\widehat{\theta}_{\widehat{S}_k c} < \tau$ for all $c \in \widehat{C}_k$, where τ is a predefined stopping threshold.

3. Theoretical Properties

Here, we give two theoretical results for the statistical properties of the GS-LDA rule. First, we show that if $\boldsymbol{\beta}$ is exactly sparse, the greedy search algorithm can correctly select its nonzero elements with high probability. Second, we show that the misclassification rate of the GS-LDA rule converges asymptotically to the Bayes error.

We begin by introducing some notation. For a matrix $\boldsymbol{A}$, a set S, and an index c, we denote $\boldsymbol{A}_{Sc}$ as the cth column of $\boldsymbol{A}$ with row indices in S. Denote $\boldsymbol{A}_{SS}$ as the submatrix of $\boldsymbol{A}$ with row and column indices in S. Denote $\lambda_{\min}(\boldsymbol{A})$ and $\lambda_{\max}(\boldsymbol{A})$ as the minimum and maximum eigenvalues, respectively, of $\boldsymbol{A}$. For two sequences a_n and b_n, we write $a_n \lesssim b_n$ if $a_n \le Cb_n$ for a generic positive constant C, write $a_n = o(b_n)$ if $a_n/b_n \to 0$, and $a_n \gg b_n$ if $b_n = o(a_n)$. To simplify the nonasymptotic statements, we assume throughout this paper that C_0 is an

arbitrarily large positive constant and C_1 is some generic positive constant, which may vary from line to line. In addition, we assume $n_k/n \to \ell \in (0,1)$ for $k = 0, 1$, and we assume normality in this section.

We first give a concentration result of the estimated increment $\widehat{\theta}_{Sc}$ for an arbitrary set S with s elements and an arbitrary $c \notin S$. We introduce the following regularity conditions.

Condition 1. $\max_{j \le p} |\delta_j| \le M < \infty$.

Condition 2. $0 < m \le \lambda_{\min}(\mathbf{\Sigma}) \le \lambda_{\max}(\mathbf{\Sigma}) \le M < \infty$.

Condition 1 requires that the elements of $\boldsymbol{\delta}$ be bounded, and condition 2 requires that the eigenvalues of $\mathbf{\Sigma}$ be bounded away from zero and ∞; these two conditions are also used in other works (Shao et al. (2011); Cai and Liu (2011)). These are mild boundedness conditions that simplify the nonasymptotic statement of the concentration results. Next, we give the concentration result of $\widehat{\theta}_{Sc} - \theta_{Sc}$.

Theorem 1. *Under Conditions* 1–2 *and if* $s^2\sqrt{(\log p)/n} = o(1)$, *it holds that*

$$P\left(|\widehat{\theta}_{Sc} - \theta_{Sc}| \lesssim s^2\sqrt{\frac{\log p}{n}} \max\left\{s^2\sqrt{\frac{\log p}{n}}, \sqrt{\theta_{Sc}}, \theta_{Sc}\right\}\right) \ge 1 - C_A p^{-C_B},$$

where C_A *is a generic positive constant and* C_B *is an arbitrary large positive constant.*

Theorem 1 shows that the concentration of $\widehat{\theta}_{Sc} - \theta_{Sc}$ depends on s and θ_{Sc}. It implies that when $\theta_{Sc} > 1$, $\widehat{\theta}_{Sc} - \theta_{Sc} = O_P(s^2\theta_{Sc}\sqrt{(\log p)/n})$, and when $s^4(\log p)/n < \theta_{Sc} < 1$, $\widehat{\theta}_{Sc} - \theta_{Sc} = O_P(s^2\sqrt{\theta_{Sc}(\log p)/n})$. When $\theta_{Sc} = 0$, it implies that $\widehat{\theta}_{Sc} - \theta_{Sc} = O_P(s^4(\log p)/n)$. These results show that it is more difficult to estimate a larger θ_{Sc}. In addition, when s gets larger, so does the estimation error, owing to the accumulation of estimation errors when estimating the unknown parameters, because when s gets larger, more parameters need to be estimated.

Theorem 1 is critical in studying the statistical properties of the GS-LDA rule. First, it indicates that as long as there is a large enough gap between θ_{Sc} and $\theta_{Sc'}$ for any $c' \neq c$, the indicator functions $I(\widehat{\theta}_{Sc} > \widehat{\theta}_{Sc'}) = I(\theta_{Sc} > \theta_{Sc'})$ hold with high probability. In other words, the order of $\widehat{\theta}_{Sc}$ and $\widehat{\theta}_{Sc'}$ reflects the true order of θ_{Sc} and $\theta_{Sc'}$. As shown below, this is the key to guaranteeing that the greedy search algorithm reaches variable selection consistency. Theorem 1 also gives guidance on how to choose the stopping threshold τ. When $\widehat{\theta}_{Sc}$ is small,

it indicates that θ_{Sc} is small or equal to zero. At that stage, adding additional variables does not improve the classification, and we should thus terminate the greedy search. More details on how to choose τ are given in Theorem 2. Finally, we give a corollary for the special case of $S = \emptyset$. This result is useful for proving the property of the initial iteration of our algorithm. Its proof follows directly from that of Theorem 1.

Corollary 1. *Under the conditions of Theorem* 1*, when* $S = \emptyset$*, it holds that*

$$P\left(|\widehat{\theta}_{Sc} - \theta_{Sc}| \lesssim \sqrt{\frac{\log p}{n}}\right) \geq 1 - C_A p^{-C_B},$$

where C_A *is a generic positive constant and* C_B *is an arbitrary large positive constant.*

Next, we prove that, if $\boldsymbol{\beta}$ is exactly sparse, in the sense that many of its elements are zero, the greedy search method can recover the support of $\boldsymbol{\beta}$ with high probability. Let $\mathcal{M} = \{j : \beta_j \neq 0\}$ be the support of $\boldsymbol{\beta}$, K be the number of elements in $\mathcal{M}$, and $\widehat{\mathcal{M}}$ be as defined in Section 2. We have the following variable selection consistency result.

Theorem 2. *Under Conditions* 1–2 *and*

Condition 3. $0 < m \leq \min_{S \subset \mathcal{M}} \max_{c \in \mathcal{M} \setminus S} \theta_{Sc} \leq \max_{S \subset \mathcal{M}} \max_{c \in \mathcal{M} \setminus S} \theta_{Sc} \leq M < \infty$*;*

Condition 4. $\min_{S \subset \mathcal{M}}(\max_{c \in \mathcal{M} \setminus S} \theta_{Sc} - \max_{c \notin \mathcal{M}} \theta_{Sc}) \gg K^2\sqrt{(\log p)/n}$*;*

if $K^2\sqrt{(\log p)/n} = o(1)$*, and we choose* $\tau \asymp K^4(\log p)/n$*, it holds that*

$$P\left(\widehat{\mathcal{M}} = \mathcal{M}\right) \geq 1 - C_A K p^{-C_B},$$

where C_A *is a generic positive constant and* C_B *is an arbitrary large positive constant.*

Condition 3 requires that for any $S \subset \mathcal{M}$, if we add another variable in $\mathcal{M}$ to those in S, the true increment θ_{Sc} should be bounded away from zero and infinity. The lower bound is mild, because, as shown in (2.3), θ_{Sc} is always nonnegative; because $\mathcal{M}$ contains all discriminative features, the lower bound requires only that at least one additional feature in $\mathcal{M}$ should produce a large enough increment of θ_{Sc} to pass the threshold. The upper bound is mainly introduced to simplify the expression. As shown in Theorem 1, the concentration of $\widehat{\theta}_{Sc}$ also depends on the magnitude of θ_{Sc}, requiring all θ_{Sc} to be bounded away from infinity.

This enables us to have a more succinct nonasymptotic statement in Theorem 2. Condition 4 requires that for any $S \subset \mathcal{M}$, the maximum increment produced by adding another feature in $\mathcal{M}$ should surpass the increment by adding a feature outside of $\mathcal{M}$ when the true θ_{Sc} is known. Such a condition naturally requires that adding a discriminative feature in $\mathcal{M}$ should bring more information than adding a non-discriminative one outside $\mathcal{M}$. As shown in Theorem 1, the component $K^2\sqrt{(\log p)/n}$ is the estimation error of $\widehat{\theta}_{Sc}$ to θ_{Sc}. Thus, once Condition 4 is assumed, with high probability, we have $\max_{c\in\mathcal{M}\setminus S}\widehat{\theta}_{Sc} > \max_{c\notin\mathcal{M}}\widehat{\theta}_{Sc}$. This is the key to ensuring that the greedy search algorithm chooses the informative features in $\mathcal{M}$. As reflected by the concentration result in Theorem 1, the choice of τ is essentially the order of $\widehat{\theta}_{Sc}$ when $\theta_{Sc} = 0$. This guarantees the exclusion of non-informative features. Finally, the assumption of $K^2\sqrt{(\log p)/n} = o(1)$ is a sparsity assumption on $\boldsymbol{\beta}$, which is similar to the condition needed for the LPD and ROAD methods; see Cai and Liu (2011) and Fan, Feng and Tong (2012).

Given the variable selection consistency, we next establish the error rate consistency of the GS-LDA rule. Without loss of generality, we assume that $\pi_0 = \pi_1 = 1/2$. By definition, the misclassification error of the GS-LDA rule is

$$\begin{aligned} R_{GS\text{-}LDA}(\boldsymbol{X}) &= \frac{1}{2}P\left(D_{GS\text{-}LDA}(\boldsymbol{x}) = 0 | \boldsymbol{x} \text{ comes from Class 1}\right) \\ &\quad + \frac{1}{2}P\left(D_{GS\text{-}LDA}(\boldsymbol{x}) = 1 | \boldsymbol{x} \text{ comes from Class 0}\right) \\ &= \frac{1}{2}\sum_{k=0}^{1}\Phi\left(\frac{(-1)^k\widehat{\boldsymbol{\beta}}_{\widehat{\mathcal{M}}}^T(\boldsymbol{\mu}_{k\widehat{\mathcal{M}}} - \bar{\boldsymbol{x}}_{k\widehat{\mathcal{M}}}) - \widehat{\boldsymbol{\delta}}_{\widehat{\mathcal{M}}}^T\widehat{\boldsymbol{\Omega}}_{\widehat{\mathcal{M}}}\widehat{\boldsymbol{\delta}}_{\widehat{\mathcal{M}}}/2}{\sqrt{\widehat{\boldsymbol{\delta}}_{\widehat{\mathcal{M}}}^T\widehat{\boldsymbol{\Omega}}_{\widehat{\mathcal{M}}}\boldsymbol{\Sigma}_{\widehat{\mathcal{M}}\widehat{\mathcal{M}}}\widehat{\boldsymbol{\Omega}}_{\widehat{\mathcal{M}}}\widehat{\boldsymbol{\delta}}_{\widehat{\mathcal{M}}}}}\right). \end{aligned} \tag{3.1}$$

The following theorem establishes the error rate consistency.

Theorem 3. *Under Conditions* 1–4, *if* $K^2\sqrt{(\log p)/n} = o(1)$ *and we choose* $\tau \asymp K^4(\log p)/n$, *it holds that*

(a) $R_{GS\text{-}LDA}(\boldsymbol{X}) = \Phi(-\sqrt{\Delta_p}/2\{1 + O_P(K\sqrt{(\log p)/n})\})$;

(b) $R_{GS\text{-}LDA}(\boldsymbol{X})/R_{Bayes} - 1 = O_P(\sqrt{(\log p)/n})$, *when* $\Delta_p < \infty$;

(c) $R_{GS\text{-}LDA}(\boldsymbol{X})/R_{Bayes} - 1 = O_P(\max\{\Delta_p^{-1}, K^2\sqrt{(\log p)/n}\})$ *when* $\Delta_p \to \infty$.

Theorem 3 proves that $R_{GS\text{-}LDA}(\boldsymbol{X})/R_{Bayes}$ converges to one in probability. Statement (a) shows that the convergence rate of $R_{GS\text{-}LDA}(\boldsymbol{X})$ to R_{Bayes} depends on K; for a larger K, the convergence is slower because more parameters need to be estimated. In statements (b) and (c), we show that the ratio

of $R_{GS\text{-}LDA}(\boldsymbol{X})/R_{Bayes}$ converges to one in probability. Because Δ_p itself can diverge, R_{Bayes} can converge to zero. Thus, statements (b) and (c) are stronger than showing $R_{GS\text{-}LDA}(\boldsymbol{X}) - R_{Bayes} \to 0$ in probability. Our result indicates that $R_{GS\text{-}LDA}(\boldsymbol{X})$ can converge to zero as fast as R_{Bayes} does, even when $R_{Bayes} \to 0$. Furthermore, we show that the convergence rates differ depending on whether Δ_p is bounded. Finally, similarly to the LPD and ROAD, our method relies on the sparsity assumption on $\boldsymbol{\beta}$ to reach the error rate consistency, as shown in Theorem 3. This assumption is needed to avoid the accumulation of errors when estimating $\boldsymbol{\beta}$ that could ruin the error rate consistency (Bickel and Levina ((2004)). In addition, Theorem 3 relies on the variable selection consistency established in Theorem 2.

4. Relaxation of the Normality Assumption

Although we have proved the theoretical results under the normality assumption, these results can still hold when the two classes follow more general distributions. As discussed in Shao et al. (2011), the Bayes rule remains the same, as long as there exists a unit vector $\boldsymbol{\gamma}$ such that, for any real number t and $k = 0, 1$, it holds that

$$P(\boldsymbol{\gamma}'\boldsymbol{\Sigma}^{-1/2}(\boldsymbol{x}_k - \boldsymbol{\mu}_k) \le t) = \Psi(t),$$

where $\Psi(t)$ is a cumulative distribution function with a density that is symmetric around zero and does not depend on $\boldsymbol{\gamma}$. In this case, the Bayes error is $\Psi(-\sqrt{\Delta_p}/2)$, which is still a decreasing function of the Mahalanobis distance Δ_p. Two key conditions for such a result are that the density function is symmetric around zero and the two classes have an equal covariance. Distributions satisfying this condition include the class of elliptical distributions with a density function of $c_p|\boldsymbol{\Sigma}|^{-1/2} f((\boldsymbol{x} - \boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}))$, where f is a monotone function in $[0, \infty)$ and c_p is a normalization constant. Examples of elliptical distributions include the multivariate normal, t, and double exponential distributions. Given such a general distributional assumption, the increment of Mahalanobis distance still quantifies how much a new variable can reduce the Bayes error. Interestingly, for all elliptical distributions, θ_{Sc} still has the form

$$\theta_{Sc} = \frac{C\{\mathrm{E}(z_c|\boldsymbol{z}_S = \boldsymbol{0})\}^2}{\mathrm{Var}(z_c|\boldsymbol{z}_S = \boldsymbol{0})},$$

where the positive constant C depends on the type of the distribution, and is equal to two if it is normal. The proof uses the conditional distribution of the

elliptical distributions, which is similar to what the multivariate normal distribution admits; see Theorem 2.18 of Fang, Kotz and Ng (2018). Thus, for all elliptical distributions, the contribution of a new variable to the classification depends on its effect size, conditional on other variables (i.e., the standardized $\mathrm{E}(z_c|\boldsymbol{z}_S = \mathbf{0})$).

However, under the more general distributional assumption, the convergence rates established in Theorems 1–3 may change. A closer look at the proofs reveals that the convergence rates depend on the tail probability, which is characterized by $\Psi(-x)$. The tail probability is the key to establishing the critical concentration results of $\widehat{\boldsymbol{\delta}}$ and $\widehat{\boldsymbol{\Sigma}}$, upon which the proofs are built; see Lemma 1 in the Supplementary Material. In general, we assume that

$$0 < \lim_{x\to\infty} \frac{x^{-\omega}\exp(-cx^{\varphi})}{\Psi(-x)} < \infty, \tag{4.1}$$

where $\varphi \in [0, 2]$, $c \in (0, \infty)$, and $\omega \in (0, \infty)$ are some constants. In particular, when Ψ is standard normal, (4.1) holds with $\omega = 1$, $c = 1/2$, and $\varphi = 2$. Then, if Ψ satisfies (4.1) with $\varphi = 2$, the same exponential-type concentration can be established so that all results in Theorem 1–3 remain the same. When Ψ satisfies (4.1) with $\varphi < 2$, the tail of the distribution is heavier. In that case, if we assume the moment condition that

$$\max_{k,j\le p} \mathrm{E}|x_{k,j}|^{2\nu} < \infty, \text{ for some } \nu > 0, \text{ and } k = 0, 1, \tag{4.2}$$

where $x_{k,j}$ is the jth element of $\boldsymbol{x}_k$, a polynomial-type concentration can be established so that Theorems 1–3 hold with all $(\log p)/n$ terms being replaced by $p^{4/\nu}/n$. In this case, p is only allowed to grow polynomially with n. These results are analogous to the discussions in Section 4 of Shao et al. (2011). To improve the convergence rates, one can replace $\widehat{\boldsymbol{\delta}}$ and $\widehat{\boldsymbol{\Sigma}}$ with robust estimators, such as the Huber estimator or the median-of-means estimator; see Avella et al. (2018). Correspondingly, the greedy search algorithm can be built upon these robust estimators. Once such robust estimators are used in the algorithm, even under the moment assumption in (4.2), Theorems 1–3 can still hold with the same exponential rate of convergence, using the concentration results established in Avella et al. (2018).

5. Simulation Studies

We investigate the numerical performance of our proposed method under four different scenarios. In the first two scenarios, we compare the classification

performance and the execution time of the proposed GS-LDA method with those of other LDA-based methods. These include the sparse discriminant analysis by Clemmensen et al. (2011) (SDA), the SLDA, LPD, and ROAD, and some other well-known classifiers in machine learning, such as the support vector machine (SVM) with a linear kernel and the logistic regression with an L_1-penalty (Logistic-L1). In the other two scenarios, we investigate the performance of the GS-LDA method for some ultrahigh-dimensional settings that involve tens of thousands of features. In these two scenarios, most existing methods cannot handle such high dimensions, and we only aim at testing the viability of GS-LDA. We implement the SDA using the **sparseLDA** package. We implement the SLDA using our own coding of the algorithm given in Shao et al. (2011). We implement the LPD using the **linprogPD** function from the **clime** package (`https://github.com/rluo/clime`). The implementation for the ROAD comes from the publicly available package developed by the authors. We implement the SVM using the **e1071** package, and implement the Logistic-L1 using the **glmnet** package. The optimal tuning parameters for each method are chosen using a grid search with five-fold cross-validation. The execution time is recorded as the time taken by each algorithm on a computing cluster with an Intel Xeon 3.4GHz CPU, with the tuning parameters fixed at their optimal values.

In the first two scenarios, we consider the following two choices of $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}$:

Scenario 1: $\boldsymbol{\mu}_0 = \mathbf{0}$ and $\boldsymbol{\mu_1} = (1, \ldots, 1, 0, \ldots, 0)^T$, where the first 10 elements are ones, and the rest are zeros. $\boldsymbol{\Sigma} = (\sigma_{ij})_{p\times p}$, where $\sigma_{ij} = 0.8^{|i-j|}$, for $1 \le i, j \le p$.

Scenario 2: $\boldsymbol{\mu}_0 = \mathbf{0}$ and $\boldsymbol{\mu_1} = \boldsymbol{\Sigma\beta}$, where $\boldsymbol{\beta} = (0.25, \ldots, 0.25, 0, \ldots, 0)^T$, with the first 10 elements of $\boldsymbol{\beta}$ equal to 0.25 and the rest zeros; $\boldsymbol{\Sigma} = (\sigma_{ij})_{p\times p}$, where $\sigma_{ij} = 0.8^{|i-j|}$, for $1 \le i, j \le p$.

For each scenario, we generate 200 training samples for each of the two classes from $N(\boldsymbol{\mu}_k, \boldsymbol{\Sigma})$; we let the dimension p vary from 500 to 2,000, with an increment of 500. We independently generate another 800 samples from each of these distributions as the test set. In Scenario 1, we set $\boldsymbol{\delta} = \boldsymbol{\mu}_0 - \boldsymbol{\mu}_1$ to be exactly sparse. This scenario is the same as Model 3 considered in Cai and Liu (2011). In Scenario 2, we set $\boldsymbol{\beta}$ to be exactly sparse because such a condition is imposed for the proposed GS-LDA method. We use five-fold cross-validation to choose the optimal tuning parameters in all seven methods. For each scenario, we run 100 replicates. We report the average misclassification rates and the execution time for each classifier in Figures 1 and 2. For Scenario 2, we also report the variable selection performance on $\boldsymbol{\beta}$ by the GS-LDA, ROAD, and Logistic-L1. We

measure the variable selection performance by sensitivity and specificity. Sensitivity is defined as the proportion of nonzero elements of $\boldsymbol{\beta}$ that are estimated as nonzero, and specificity is defined as the proportion of zero elements of $\boldsymbol{\beta}$ that are estimated as zero.

Figures 1 and 2 show that the GS-LDA has the best classification performance for all choices of p and under both scenarios. Its computational speed is also much faster than that of the other LDA-based classifiers, especially when the dimension p is high. It is also faster than the SVM and Logistic-L1, implemented by the **e1071** and **glmnet** packages, respectively, which are known to be computationally efficient. In both scenarios, when $p = 2000$, the average execution time is around 190 seconds for the SDA, 100 seconds for the ROAD, 20 seconds for the LPD, 3 seconds for the SVM, and 1 second for the SLDA, but only 0.05 second for the GS-LDA. The execution time for the Logistic-L1 depends on the Hessian matrix of the likelihood and the sparsity of $\boldsymbol{\beta}$. In Scenario 1, when $\boldsymbol{\beta}$ is weakly sparse, the GS-LDA is still faster than the Logistic-L1. In Scenario 2, when $\boldsymbol{\beta}$ is exactly sparse, their computational time is comparable. This is mainly because the **glmnet** package only updates nonzero components of $\boldsymbol{\beta}$ along its iterations. The proposed GS-LDA method thus offers a substantial boost in computational speed over most its competitors, while rendering an excellent classification rule. In terms of variable selection performance, in Scenario 2, the GS-LDA has similar specificity to that of the ROAD and Logistic-L1, and better sensitivity than the ROAD. However, owing to its lower sensitivity than the Logistic-L1, the GS-LDA has slightly higher misclassification error than that of the Logstic-L1 in this secnario. Finally, note that because the GS-LDA adds one variable at a time, the variation of its errors can be smaller than that of other optimization based methods, where the numbers of variables are determined by some tuning parameters, and small changes can result in multiple new variables being included in the classification rule. This can be seen from Figures 1 and 2.

To further investigate how many dimensions the GS-LDA method can efficiently handle, we simulate two additional scenarios, where we choose $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}$ as follows:

Scenario 3: $\boldsymbol{\mu}_0 = \mathbf{0}$ and $\boldsymbol{\mu}_\mathbf{1} = (1, \ldots, 1, 0, \ldots, 0)^T$, where the first 10 elements are ones and the rest are zeros; $\boldsymbol{\Sigma} = \boldsymbol{\Omega}^{-1}$, where $\boldsymbol{\Omega} = (\omega_{ij})_{p\times p}$ and $\omega_{ij} = \sqrt{ij}\{2I(i = j \neq p) + I(i = j = p) - I(|i - j| = 1)\}$.

Scenario 4: $\boldsymbol{\mu}_0 = \mathbf{0}$ and $\boldsymbol{\mu}_\mathbf{1} = \boldsymbol{\Sigma}\boldsymbol{\beta}$, where $\boldsymbol{\beta} = (1, \ldots, 1, 0, \ldots, 0)^T$, with the first 10 elements of $\boldsymbol{\beta}$ being ones and the rest being zeros; $\boldsymbol{\Sigma}$ is the same as in Scenario 3.

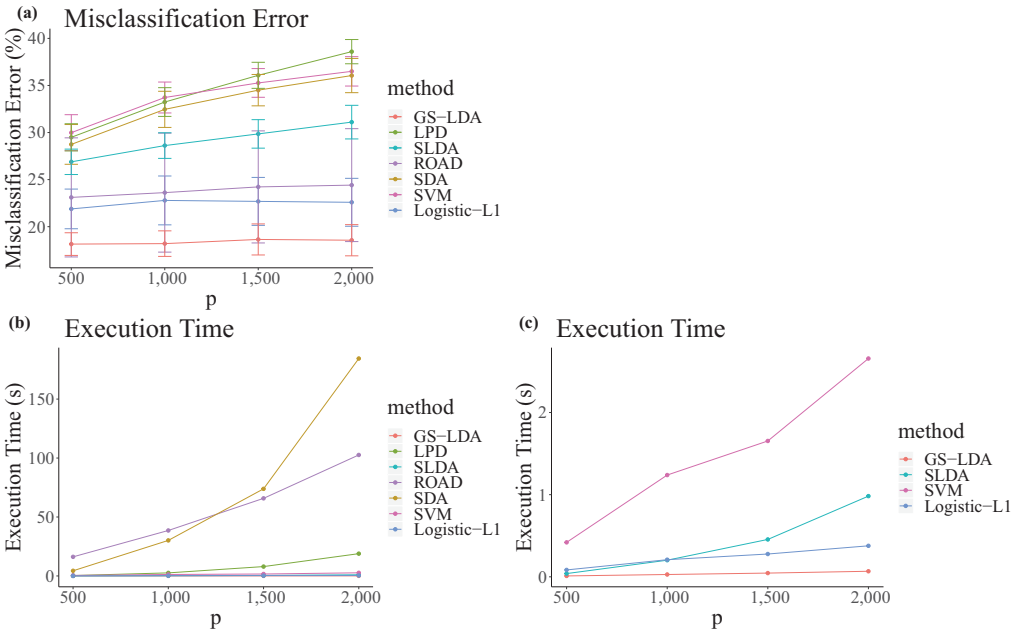

Figure 1. Numerical performance of the four classifiers in Scenario 1. Panel (c) is a zoomed plot of panel (b) for the GS-LDA , SLDA, SVM, and Logistic-L1.

For each of these two scenarios, we generate 100 training samples for each of the two classes from $N(\boldsymbol{\mu}_k, \boldsymbol{\Sigma})$ and set the dimension $p = 1 \times 10^4, 3 \times 10^4, 5 \times 10^4$, and 1×10^5. We independently generate another 400 samples for each of the two classes as the test set. Scenarios 3 and 4 are analogues to scenarios 1 and 2: $\boldsymbol{\delta}$ is exactly sparse in Scenario 3, and $\boldsymbol{\beta}$ is exactly sparse in Scenario 4. Again, we use five-fold cross-validation to choose the optimal stopping threshold for the GS-LDA. For each scenario, we run 100 replicates. We report the average misclassification rates and the execution time for the GS-LDA in Figures 3 and 4.

Figures 3 and 4 show that in both scenarios the GS-LDA method still performs well when the dimension is ultrahigh. When the dimension p grows, the misclassification error remains stable, and the execution time grows only moderately with p. Even when p is as big as 1×10^5, the execution time of the GS-LDA is only tens of seconds. In contrast, the SLDA, LPD, and ROAD cannot solve such a problem within tens of hours. As a result, they are excluded from the comparison in these two scenarios.

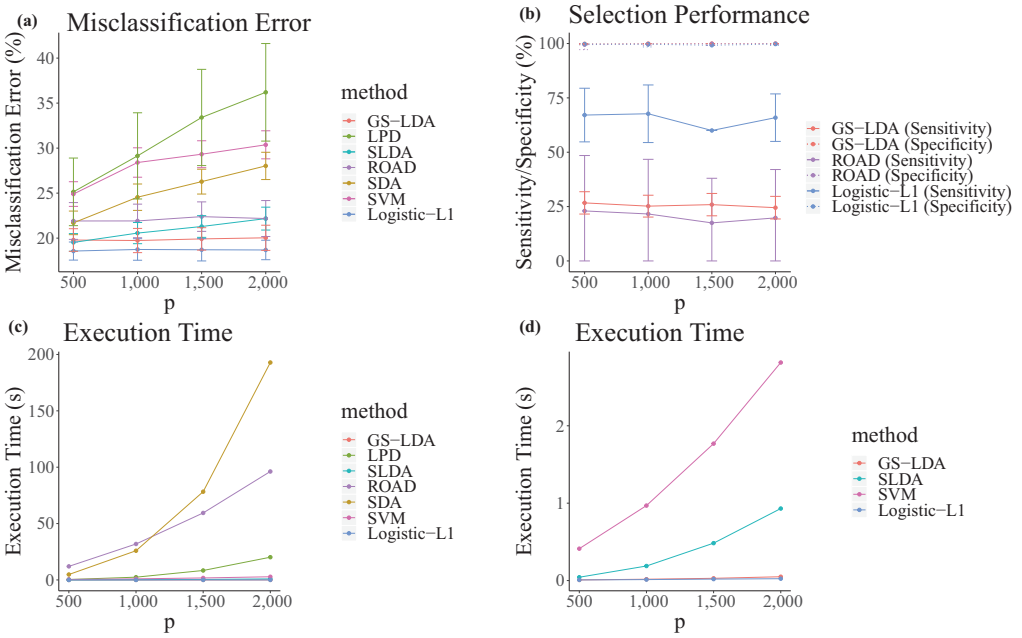

Figure 2. Numerical performance of the four classifiers in Scenario 2. Panel (d) is a zoomed plot of panel (c) for the GS-LDA , SLDA, SVM, and Logistic-L1.

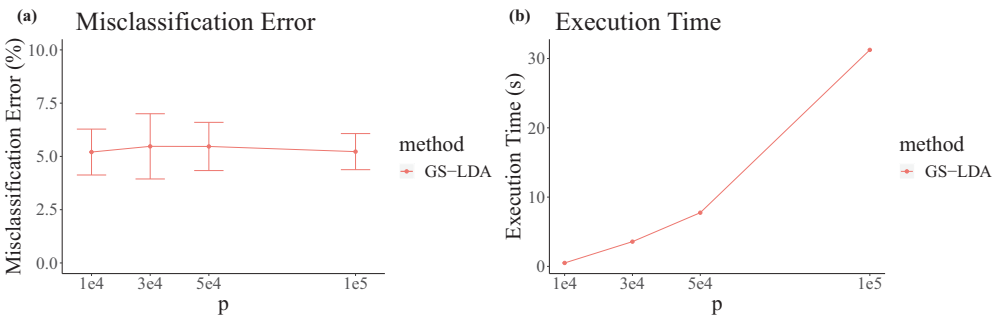

Figure 3. Numerical performance of the GS-LDA in Scenario 3.

6. An Application to Cancer Subtype Classification

To further illustrate the advantage of the GS-LDA method, we apply it to microarray data for classifying cancer subtypes. This data set contains 82 breast cancer subjects, with 41 ER-positive and 41 ER-negative. These subjects are sequenced using the Affymetrix Human Genome U133A Array, which measures the gene expression using 22283 probes. The raw data are available in the Gene Expression Omnibus database with the accession name GSE22093.

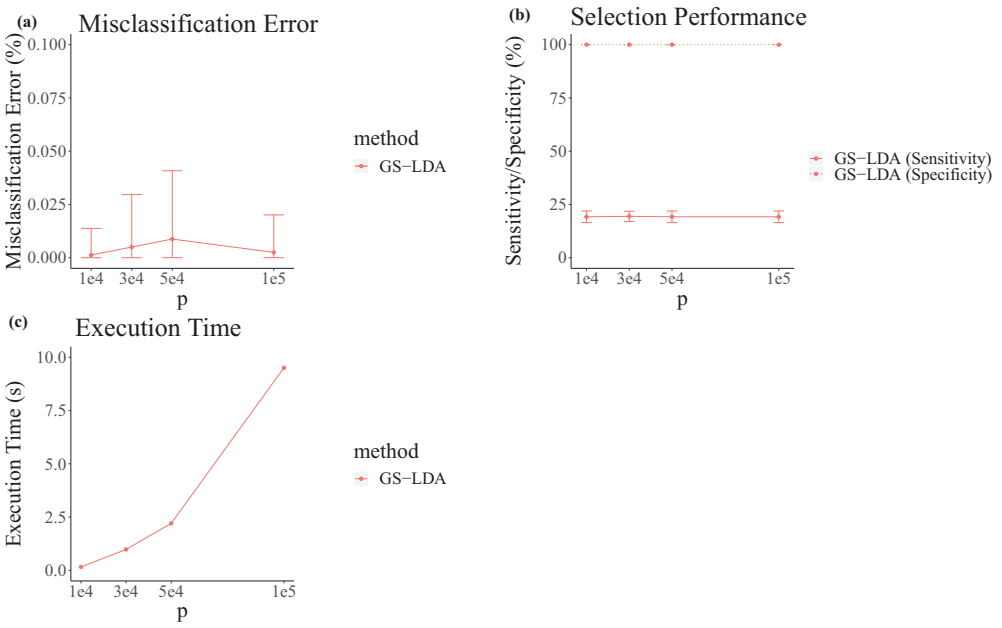

Figure 4. Numerical performance of the GS-LDA in Scenario 4.

We randomly split the data set into a training set of 60 samples and a test set of 22 samples, repeating the random split 100 times. Each time, we learn the GS-LDA, SLDA, ROAD, SVM, and Logistic-L1 from the training set, and obtain their misclassification errors by applying them to the test set. The LPD and SDA methods are excluded from this study because they cannot finish the training within 24 hours. The tuning parameters in these methods are chosen using five-fold cross-validation. The misclassification errors and the execution times of these methods are summarized in Table 1.

The GS-LDA method performs well for this data set, with a mean misclassification error of only 2.4%. On average, this is less than one error among the 22 samples in a testing set. This misclassification error is 47% better than that of the SVM, 56% better than that of the ROAD, 72% better than that of the SLDA, and equals to that of the Logistic-L1. In term of computational speed, the GS-LDA runs for only 0.3 seconds on average, which is over 5,000 times faster than the ROAD, over 1,000 times faster than the SLDA, over 10 times faster than the SVM, and close to that of the Logistic-L1 on this data set.

Interestingly, we found that in the 100 splits of the data set, the GS-LDA method frequently selected one particular variable: the expression of the ESR1 gene measured by probe "205225_at." This variable was selected 95 times by

Table 1. Numerical performance of the five classifiers in classifying cancer subtypes.

Methods	Misclassification Error (%)			Execution Time (s)
	Lower Quartile	Median	Upper Quartile	
GS-LDA	0	0	4.55	0.30
SLDA	4.55	9.09	13.64	355
ROAD	4.55	4.55	9.09	1576
SVM	0	4.55	5.68	3.08
Logistic-L1	0	0	4.55	0.10

the GS-LDA. It was selected first 84 times, and was the only variable selected 56 times. Iwamoto et al. (2010) defined the ER status by whether the subject's measured ESR1 expression using probe "205225_at" was higher than 10.18. In other words, the true decision rule is $D_{true}(\boldsymbol{x}) = I(x_{205225\_at} > 10.18)$. We compare the selection frequency of the GS-LDA, ROAD and Logistic-L1 over the 100 random splits; see Figure S1 in the Supplementary Material. Here, we find that the GS-LDA selects the true "205225_at" probe much more often with fewer false positives than the other two methods.

To further illustrate the merit of the GS-LDA method, we use all subjects, including both the positive and the negative groups, for training; the resulting GS-LDA rule is $D_{GS\text{-}LDA}(\boldsymbol{x}) = I(x_{205225\_at} > 10.50)$, which is very close to how the ER status is defined in the original study. When we train the ROAD, SLDA, and Logistic-L1 rules using the full data, we find that they also include probe "205225_at," but the ROAD includes another 15 probes, the Logistic-L1 includes another 28 probes, and the SLDA includes all probes. It is obvious that the GS-LDA gives a rule that is much closer to the truth.

7. Discussion

We have developed an efficient greedy search algorithm for performing an LDA with high-dimensional data. Motivated by the monotonicity property of the Mahalanobis distance, which characterizes the Bayes error of the LDA problem, our algorithm sequentially selects the features that produce the largest increments of Mahalanobis distance. In other words, it sequentially selects the most informative features until no additional feature can bring enough extra information to improve the classification. Our algorithm is computationally much more efficient than existing optimization-based or thresholding methods, because it does not need to solve an optimization problem or invert a large matrix. Indeed, it does not even need to compute the whole covariance matrix in advance; rather, it computes matrix elements as it goes in order to update the classification rule.

All calculations are based on some closed-form formulae. We proved that such an algorithm results in a GS-LDA rule that is both variable selection and error rate consistent, under a mild distributional assumption.

In practice, our method can also be modified to yield a nonlinear classification boundary by using nonlinear kernels or Gaussian copulas (Han, Zhao and Liu (2013)). Our method may also be extended to a multicategory discriminant analysis. Using similar ideas to those in Pan, Wang and Li (2016) and Mai, Yang and Zou (2019), we can translate a multicategory problem into multiple binary classification problems, to which our method is applicable. Finally, note that our method requires the key assumption that the two classes have the same covariance. If this is not the case, it becomes a quadratic discriminant analysis (QDA) problem. The Bayes error of such a problem has a much more completed form (Li and Shao (2015)). We leave developing an efficient algorithm to solve the high-dimensional QDA problem as a topic for future research.

Supplementary Material

The proofs of Theorems 1–3, equation (2.3), and the supporting lemmas are provided in the accompanying online Supplementary Material.

Acknowledgments

The authors gratefully acknowledge *NIH grants R01-AG073259, R01-HL149 683, and R01-HG009974.*

References

Anderson, T. W. (1962). *An Introduction to Multivariate Statistical Analysis*. Wiley, New York.

Avella, M., Battey, H., Fan, J. and Li, Q. (2018). Robust estimation of high dimensional covariance and precision matrices. *Biometrika* **105**, 271–284.

Beck, A. and Teboulle, M. (2009). A fast iterative shrinkage-thresholding algorithm for linear inverse problems *SIAM Journal on Imaging Sciences* **2**, 183–202.

Bickel, P. J. and Levina, E. (2004). Some theory for Fisher's linear discriminant function, naive Bayes, and some alternatives when there are many more variables than observations. *Bernoulli* **10**, 989–1010.

Bickel, P. J. and Levina, E. (2008a). Covariance regularization by thresholding. *The Annals of Statistics* **36**, 2577–2604.

Cai, T. and Liu, W. (2011). A direct estimation approach to sparse linear discriminant analysis. *Journal of the American Statistical Association* **106**, 1566–1577.

Candes, E. and Tao, T. (2007). The Dantzig selector: Statistical estimation when p is much larger than n. *The Annals of Statistics* **35**, 2313–2351.

Clemmensen, L., Hastie, T., Witten, D. and Ersb*o*ll, B. (2011). Sparse discriminant analysis. *Technometrics* **53**, 406–413.

Fan, J., Feng, Y. and Tong, X. (2012). A ROAD to classification in high dimensional space: The regularized optimal affine discriminant. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **74**, 745–771.

Fang, K. W., Kotz, S. and Ng, K. W. (2018). *Symmetric Multivariate and Related Distributions*. Chapman and Hall/CRC.

Friedman, J., Hastie, T. and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software* **33**, 1–22.

Han, F., Zhao, T. and Liu, H. (2013). CODA: High dimensional copula discriminant analysis. *Journal of Machine Learning Research* **14**, 629–671.

Iwamoto, T., Bianchini, G., Booser, D., Qi, Y., Coutant, C., Shiang, C. Y.-H. et al. (2010). Gene pathways associated with prognosis and chemotherapy sensitivity in molecular subtypes of breast cancer. *Journal of the National Cancer Institute* **103**, 264–272.

Li, Q. and Li, L. (2018). Integrative linear discriminant analysis with guaranteed error rate improvement. *Biometrika* **105**, 917–930.

Li, Q. and Shao, J. (2015). Sparse quadratic discriminant analysis for high dimensional data. *Statistica Sinica* **25**, 457–473.

Mai, Q., Yang, Y. and Zou, H. (2019). Multiclass sparse discriminant analysis. *Statistica Sinica* **29**, 97–111.

Mai, Q., Zou, H. and Yuan, M. (2012). A direct approach to sparse discriminant analysis in ultra-high dimensions. *Biometrika* **99**, 29–42.

Pan, R., Wang, H. and Li, R. (2016). Ultrahigh-dimensional multiclass linear discriminant analysis by pairwise sure independence screening. *Journal of the American Statistical Association* **111**, 169–179.

Shao, J., Wang, Y., Deng, X. and Wang, S. (2011). Sparse linear discriminant analysis by thresholding for high dimensional data. *The Annals of Statistics* **39**, 1241–1265.

Witten, D. M. and Tibshirani, R. (2011). Penalized classification using Fisher's linear discriminant. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **73**, 753–772.

Hannan Yang

Department of Biostatistics, University of North Carolina at Chapel Hill, Chapel Hill, NC 27599, USA.

E-mail: hnyang@live.unc.edu

Danyu Lin

Department of Biostatistics, University of North Carolina at Chapel Hill, Chapel Hill, NC 27599, USA.

E-mail: lin@bios.unc.edu

Quefeng Li

Department of Biostatistics, University of North Carolina at Chapel Hill, Chapel Hill, NC 27599, USA.

E-mail: quefeng@email.unc.edu

(Received January 2021; accepted December 2021)