Statistica Sinica Preprint No: SS-2021-0028	
Title	An Efficient Greedy Search Algorithm for High-Dimensional Linear Discriminant Analysis
Manuscript ID	SS-2021-0028
URL	http://www.stat.sinica.edu.tw/statistica/
DOI	10.5705/ss.202021.0028
Complete List of Authors	Hannan Yang, Danyu Lin and Quefeng Li
Corresponding Author	Quefeng Li
E-mail	quefeng@email.unc.edu
Notice: Accepted version subject to English editing.	

An Efficient Greedy Search Algorithm for High-dimensional Linear Discriminant Analysis

Hannan Yang, Danyu Lin and Quefeng Li

Department of Biostatistics, University of North Carolina, Chapel Hill

Abstract:

High-dimensional classification is an important statistical problem that has 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 to render classification rules—methods that are computationally prohibitive when the dimension is ultra-high. With the emergence of big data, it is increasingly important to develop more efficient algorithms to solve the high-dimensional LDA problem. 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 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 a LDA problem under some mild distributional assumptions. We demonstrate that this new algorithm drastically improves computational speed compared with 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. One simple classifier, the linear discriminant analysis (LDA), was widely used before the big data era (Anderson, 1962). But as Bickel and Levina (2004) have shown, when the number of features exceeds the sample size, the traditional LDA is no longer applicable due to the accumulation of errors in estimating the unknown parameters. To deal with the high-dimensional 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 et al., 2012; Mai et al., 2012; Han et al., 2013). Early works by Clemmensen et al. (2011) and Witten and Tibshirani (2011) proposed to solve the regularized Fisher’s discriminant problem with some sparsity-induced penalties. But

at that stage, there was little theory on the statistical properties of such classifiers. Later, when more regularized LDA classifiers (Shao et al., 2011; Cai and Liu, 2011; Fan et al., 2012; Mai et al., 2012; Han et al., 2013) were proposed to deal with the high-dimensional LDA problem, their statistical properties were thoroughly studied.

In general, these methods showed that as long as the unknown population parameters satisfy some sparsity assumptions, building a regularized LDA classifier accordingly 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 of population means and the covariance matrices are sparse, utilizing thresholding estimators (Bickel and Levina, 2008a) can still yield a consistent rule. On the other hand, Cai and Liu (2011), Fan et al. (2012) and Mai et al. (2012) separately developed distinct consistent rules while assuming the slope of the Bayes rule is sparse. Han et al. (2013) further relaxed the normality assumption on these rules and extended them to more general distributions by 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 all other aforementioned

rules must solve some large-scale optimization problems. Particularly when the dimension is ultra-high, it takes a long time to learn these rules. Therefore, we propose a computationally more efficient classifier that can be learned without the need 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 on the fly. Our method is therefore scalable for ultra-high-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 regarding 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 additional information a new feature brings 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 our method's misclassification error converges to the Bayes error. These results also hold under a general class of elliptical distributions. We numerically demonstrate that our method can achieve 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 provides details on how the efficient greedy search algorithm is devised. Section 3 describes the statistical properties of the proposed method in terms of its variable selection and error rate consistency. Section 4 further relaxes the normality assumption and shows that the statistical properties hold for a general class of distributions. Section 5 presents extensive numerical studies to 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 a 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 Materials.

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 $\mathbf{x}_k \in \mathbb{R}^p$ denotes a p -dimensional vector of features from the k th class that follows a normal distribution of $N(\boldsymbol{\mu}_k, \boldsymbol{\Sigma})$, where $\mathbf{x}_0$ and $\mathbf{x}_1$ are assumed to be independent. The Bayes rule of this classification problem is given by $D_{Bayes}(\mathbf{x}) = I(\boldsymbol{\delta}^T \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu}) \leq \log(\pi_1/\pi_0))$, where $\boldsymbol{\delta} = \boldsymbol{\mu}_0 - \boldsymbol{\mu}_1$, $\boldsymbol{\mu} = (1/2)(\boldsymbol{\mu}_0 + \boldsymbol{\mu}_1)$ and $\mathbf{x}$ 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 based on the training data $\mathbf{X} = \{\mathbf{x}_{ki}; k = 0, 1; i = 1, \dots, n_k\}$, where $\mathbf{x}_{ki}$'s are independent and identically distributed (i.i.d) samples from the k th class and n_k is the sample size of the k th class with $n = n_0 + n_1$. Then

the rule is applied to classify a new observation $\mathbf{x}$, 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 of

$$D_{LDA}(\mathbf{x}) = I(\hat{\boldsymbol{\delta}}^T \hat{\boldsymbol{\Sigma}}^{-1}(\mathbf{x} - \hat{\boldsymbol{\mu}}) \leq \log(\hat{\pi}_1/\hat{\pi}_0))$$

where

$$\hat{\pi}_k = n_k/n, \hat{\boldsymbol{\mu}}_k = \frac{1}{n_k} \sum_{i=1}^{n_k} \mathbf{x}_{ki}, \hat{\boldsymbol{\delta}} = \hat{\boldsymbol{\mu}}_0 - \hat{\boldsymbol{\mu}}_1, \quad (2.1)$$

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

But in the high-dimensional setting where $p > n$, the classical LDA method is no longer feasible, as $\hat{\boldsymbol{\Sigma}}$ is not invertible. Even if we replace $\hat{\boldsymbol{\Sigma}}^{-1}$ 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 in estimating those 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 et al., 2012; Mai et al., 2012; Han et al., 2013). In particular, Shao et al. (2011) proposed a sparse linear discriminant

analysis (SLDA) rule

$$D_{SLDA}(\mathbf{x}) = I(\tilde{\boldsymbol{\delta}}^T \tilde{\boldsymbol{\Sigma}}^{-1}(\mathbf{x} - \hat{\boldsymbol{\mu}}) \leq \log(\hat{\pi}_1/\hat{\pi}_0))$$

where $\tilde{\boldsymbol{\delta}}$ and $\tilde{\boldsymbol{\Sigma}}$ are the thresholding estimators that $\tilde{\boldsymbol{\delta}} = (\tilde{\delta}_j)$ with $\tilde{\delta}_j = \hat{\delta}_j I(|\hat{\delta}_j| > t_\delta)$ and $\hat{\delta}_j$ is the j th element of $\hat{\boldsymbol{\delta}}$, and $\tilde{\boldsymbol{\Sigma}} = (\tilde{\sigma}_{ij})$ with $\tilde{\sigma}_{ii} = \hat{\sigma}_{ii}$, $\tilde{\sigma}_{ij} = \hat{\sigma}_{ij} I(|\hat{\sigma}_{ij}| > t_\sigma)$ for $i \neq j$ and $\hat{\sigma}_{ij}$ is the (i, j) th element of $\hat{\boldsymbol{\Sigma}}$. 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

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

where λ is a tuning parameter and $(\hat{\boldsymbol{\delta}}, \hat{\boldsymbol{\Sigma}})$ is as defined in (2.1) and (2.2).

The regularized optimal affine discriminant (ROAD) method (Fan et al., 2012) estimates $\boldsymbol{\beta}$ by solving

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

where λ and γ are tuning parameters and $(\hat{\boldsymbol{\delta}}, \hat{\boldsymbol{\Sigma}})$ is as defined in (2.1)

and (2.2). Then, replacing $\tilde{\boldsymbol{\Sigma}}^{-1}\tilde{\boldsymbol{\delta}}$ in the SLDA rule with $\hat{\boldsymbol{\beta}}_{LPD}$ or $\hat{\boldsymbol{\beta}}_{ROAD}$

gives the corresponding LPD or ROAD rule. Both papers showed that the resulting rules' misclassification error asymptotically converges to the Bayes error, given some sparsity condition on β .

However, all of these methods 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 even need to evaluate the whole sample covariance matrix in advance but rather computes its elements on the fly, depending on which features enter the classification rule. We give more detailed discussion on comparing the computational complexity of these methods with ours later in this section. As will be seen in the numerical studies, our method has a much shorter learning time than these methods 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 a variable that brings the largest increment of the Mahalanobis distance. Such a variable can be regarded as the most informative one given the ones selected in previous steps. We terminate the iterations when the increment is smaller than a pre-defined 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 $\{1, \dots, p\}$, 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 only involving 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

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

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 $\mathbf{z} = \mathbf{x}_0 - \mathbf{x}_1$. Using the conditional distribution of multivariate normal distribution, we can easily show that

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

where z_c is the c th element of $\mathbf{z}$ and $\mathbf{z}_S$ is the subvector of $\mathbf{z}$ 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., $E(z_c)$), but rather its effect size conditional on other variables (i.e., the standardized $E(z_c|\mathbf{z}_S = \mathbf{0})$). In the extreme case that there is no difference in the c th variable between the two classes (i.e., $E(z_c) = 0$), adding such a variable to those in S can still reduce the Bayes error if $E(z_c|\mathbf{z}_S = \mathbf{0}) \neq 0$. This interpretation seems to be new in the LDA literature. In Section 4, we show that this interpretation not only holds for 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 $\hat{\boldsymbol{\delta}}$ and $\hat{\boldsymbol{\Sigma}}$ 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 $\hat{\boldsymbol{\Sigma}}$ and the inversion of its submatrices. We show that there is a more efficient method to compute $\hat{\theta}_{Sc}$ that only computes elements of $\hat{\boldsymbol{\Sigma}}$ on the fly, with no need to invert any 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 c th 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 $\widehat{C}_1 = \{1, \dots, p\} \setminus \{\widehat{s}_1\}$. To simplify calculations in the subsequent steps, we compute and store $\widehat{\Omega}_1 = \widehat{\sigma}_{\widehat{s}_1 \widehat{s}_1}^{-1}$ and the submatrix $\widehat{\boldsymbol{\Sigma}}_{\widehat{S}_1 \widehat{C}_1}$, i.e., the sample covariance of the selected and candidate variables. At this step, $\widehat{\Omega}_1$ and $\widehat{\boldsymbol{\Sigma}}_{\widehat{S}_1 \widehat{C}_1}$ are respectively a scalar and a vector of $p - 1$ elements. As will be seen below, the storage of such two matrices is the key to enabling fast computation. At the k th step, we compute

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

for all $c \in \widehat{C}_{k-1}$. As $\widehat{\delta}_c$, $\widehat{\sigma}_{cc}$, $\widehat{\boldsymbol{\Sigma}}_{\widehat{S}_{k-1} \widehat{C}_{k-1}}$ and $\widehat{\Omega}_{k-1}$ 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 $\widehat{C}_k = \widehat{C}_{k-1} \setminus \{\widehat{s}_k\}$. Next, we need to update $\widehat{\Omega}_k$, the estimated precision matrix of all selected variables in $\widehat{S}_k$. Using the Woodbury matrix identity, we have

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

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

On the other hand, both SLDA and LPD need to first compute $\hat{\Sigma}$, which requires $O(np^2)$ operations. For SLDA, it requires additional operations to obtain the regularized estimators $\tilde{\Sigma}$ and $\tilde{\delta}$. Besides that, inverting

$\tilde{\Sigma}$ costs another $O(p^{2+\epsilon})$ operations for some $\epsilon \in (0, 1]$, depending on the actual algorithm used to invert a matrix. Finally, computing the product $\tilde{\Sigma}^{-1}\tilde{\mathbf{d}}$ costs $O(p^2)$ operations. Thus the total computational cost of SLDA is at least $\max\{O(np^2), O(p^{2+\epsilon})\}$, which is much slower than ours when p is big. For LPD, the optimization problem can be solved by the primal-dual interior-point method (Candes and Tao, 2007). As shown by Candes and Tao (2007), in the scenario that $p \gg n$, 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 $\hat{\Sigma}$. 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 LPD is $\max\{O(Tnp^2), O(Tn^2)\}$, which is clearly slower than ours. For ROAD, if one chooses to evaluate $\hat{\Sigma}$ first and then solve the optimization problem, the computational cost is at least $O(np^2)$. A more computationally efficient solution is to use the fast iterative shrinkage-thresholding algorithm (FISTA) proposed by Beck and Teboulle (2009). In each iteration of FISTA, the cost to compute the gradient is $O(np)$ if we use a store-and-compute method that is more efficient than evaluating $\hat{\Sigma}$ before iterations start. Furthermore, it is shown by Theorem 4.4 in Beck and Teboulle (2009) that FISTA needs at least $O(n^{1/4})$ iterations to converge. Thus, the total computational cost

for FISTA to solve the ROAD problem is at least $O(n^{5/4}p)$. Therefore, our method is still faster than FISTA, especially when K is small. One may also choose to use the covariance-based method (Friedman et al., 2010) to calculate the gradient. However, its efficiency depends on the choice of tuning parameters and the initial value so that the total computational cost is hard to be quantified in general.

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

Denote $\hat{\mathcal{M}}$ and $\hat{\Omega}_{\hat{\mathcal{M}}}$ as the last $\hat{S}_k$ and $\hat{\Omega}_k$ when the greedy search algorithm terminates. We let $\hat{\beta}_{\hat{\mathcal{M}}} = \hat{\Omega}_{\hat{\mathcal{M}}}^{-1} \hat{\delta}_{\hat{\mathcal{M}}}$ and propose a greedy search

linear discriminant analysis (GS-LDA) rule as

$$D_{GS-LDA}(\mathbf{x}) = I(\hat{\boldsymbol{\beta}}_{\widehat{\mathcal{M}}}^T(\mathbf{x}_{\widehat{\mathcal{M}}} - \hat{\boldsymbol{\mu}}_{\widehat{\mathcal{M}}}) \leq \log(\hat{\pi}_1/\hat{\pi}_0))$$

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

3. Theoretical Properties

Here, we give two theoretical results regarding 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 non-zero elements with high probability. Second, we show that the misclassification rate of the GS-LDA rule asymptotically converges to the Bayes error.

We begin by introducing some notations. For a matrix $\mathbf{A}$, a set S , and an index c , we denote $\mathbf{A}_{S^c}$ as the c th column of $\mathbf{A}$ with row indices in S . Denote $\mathbf{A}_{SS}$ as the submatrix of $\mathbf{A}$ with row and column indices in S . Denote $\lambda_{\min}(\mathbf{A})$ and $\lambda_{\max}(\mathbf{A})$ as the minimum and maximum eigenvalues of $\mathbf{A}$. For two sequences a_n and b_n , we write $a_n \lesssim b_n$, if $a_n \leq Cb_n$ for a generic positive constant C ; write $a_n = o(b_n)$ if $a_n/b_n \rightarrow 0$, and $a_n \gg b_n$ if $b_n = o(a_n)$. To simplify the non-asymptotic statements, we assume throughout the paper that C_0 is an arbitrarily large positive constant and C_1 is some generic positive constants, which may vary from line to line. Also, we assume $n_k/n \rightarrow \ell \in (0, 1)$ for $k = 0, 1$, and we assume normality

Algorithm 1: The greedy search algorithm.

Initialization: Compute and store $\hat{\delta}$ and the diagonal elements of $\hat{\Sigma}$ using (2.1) and (2.2).

Set $\hat{s}_1 = \operatorname{argmax}_{j \leq p} \hat{\delta}_j^2 / \hat{\sigma}_{jj}$, $\hat{S}_1 = \{\hat{s}_1\}$,

$\hat{C}_1 = \{1, \dots, p\} \setminus \{\hat{s}_1\}$, $\hat{\Omega}_1 = \hat{\sigma}_{\hat{s}_1 \hat{s}_1}^{-1}$.

Compute and store $\hat{\Sigma}_{\hat{S}_1 \hat{C}_1}$.

At the k th iteration:

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

Set $\hat{s}_k = \operatorname{argmax}_{c \in \hat{C}_{k-1}} \hat{\theta}_{\hat{S}_{k-1}c}$, $\hat{S}_k = \hat{S}_{k-1} \cup \{\hat{s}_k\}$, $\hat{C}_k = \hat{C}_{k-1} \setminus \{\hat{s}_k\}$,

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

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

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

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

in this section.

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

the following regularity conditions.

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

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

Condition 1 requires elements of δ to be bounded, and condition 2 requires the eigenvalues of Σ to be bounded away from 0 and ∞ ; these two conditions are also used in other works (Shao et al., 2011; Cai and Liu, 2011). These boundedness conditions are mild conditions to simplify the non-asymptotic statement of concentration results. Next, we give the concentration result of $\hat{\theta}_{S_c} - \theta_{S_c}$.

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

$$P\left(|\hat{\theta}_{S_c} - \theta_{S_c}| \lesssim s^2 \sqrt{(\log p)/n} \max\{s^2 \sqrt{(\log p)/n}, \sqrt{\theta_{S_c}}, \theta_{S_c}\}\right) \geq 1 - C_A p^{-C_B},$$

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

Theorem 1 shows that the concentration of $\hat{\theta}_{S_c} - \theta_{S_c}$ depends on s and θ_{S_c} . It implies that when $\theta_{S_c} > 1$, $\hat{\theta}_{S_c} - \theta_{S_c} = O_P(s^2 \theta_{S_c} \sqrt{(\log p)/n})$, and when $s^4 (\log p)/n < \theta_{S_c} < 1$, $\hat{\theta}_{S_c} - \theta_{S_c} = O_P(s^2 \sqrt{\theta_{S_c} (\log p)/n})$. When $\theta_{S_c} = 0$, it implies that $\hat{\theta}_{S_c} - \theta_{S_c} = O_P(s^4 (\log p)/n)$. These results show that it is harder to estimate a larger θ_{S_c} . In addition, when s gets larger,

the estimation error also gets larger. This is due to the accumulation of estimation errors in estimating the unknown parameters, since 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 $I(\hat{\theta}_{Sc} > \hat{\theta}_{Sc'}) = I(\theta_{Sc} > \theta_{Sc'})$ hold with high probability. In other words, the order of $\hat{\theta}_{Sc}$ and $\hat{\theta}_{Sc'}$ reflect the true order of θ_{Sc} and $\theta_{Sc'}$. As will be seen below, this is the key to guarantee that the greedy search algorithm can reach variable selection consistency. Theorem 1 also gives guidance on how to choose the stopping threshold τ . When $\hat{\theta}_{Sc}$ is small, it indicates that θ_{Sc} is small or equals to zero. At that stage, adding additional variables does not improve the classification, and we should thus terminate the greedy search at that stage. More details on how to choose τ will be given in Theorem 2. Finally, we give a corollary for the special case of $S = \emptyset$. This result is useful in proving the property of the initial iteration of our algorithm. Its proof directly follows that of Theorem 1.

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

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

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

constant.

Next, we prove that, if β is exactly sparse in the sense that many of its elements are zero, the greedy search method can recover the support of β with high probability. Let $\mathcal{M} = \{j : \beta_j \neq 0\}$ be the support of β and 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

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

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 the ones in S , the true increment θ_{Sc} should be bounded away from zero and infinity. The lower bound is mild, for, as shown in (2.3), θ_{Sc} is always non-negative; as $\mathcal{M}$ contains all discriminative features, the lower bound only 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 writing. As shown in Theorem 1, the concentration of $\hat{\theta}_{Sc}$ 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 non-asymptotic 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 $\mathcal{M}$, 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 seen in Theorem 1, the component $K^2\sqrt{(\log p)/n}$ is the estimation error of $\hat{\theta}_{Sc}$ to θ_{Sc} . Thus, once Condition 4 is assumed, with high probability, we have $\max_{c \in \mathcal{M} \setminus S} \hat{\theta}_{Sc} > \max_{c \notin \mathcal{M}} \hat{\theta}_{Sc}$. This is key to ensure that the greedy search algorithm can choose the informative features in $\mathcal{M}$. As reflected by the concentration result in Theorem 1, the choice of τ is essentially the order of $\hat{\theta}_{Sc}$ 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 β , which is similar to the condition needed for LPD and ROAD methods; see Cai and Liu (2011) and Fan et al. (2012).

Given the variable selection consistency, next we 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

$$\begin{aligned} R_{GS-LDA}(\mathbf{X}) &= (1/2)P(D_{GS-LDA}(\mathbf{x}) = 0 | \mathbf{x} \text{ comes from Class 1}) \\ &\quad + (1/2)P(D_{GS-LDA}(\mathbf{x}) = 1 | \mathbf{x} \text{ comes from Class 0}) \\ &= \frac{1}{2} \sum_{k=0}^1 \Phi \left(\frac{(-1)^k \hat{\boldsymbol{\beta}}_{\widehat{\mathcal{M}}}^T (\boldsymbol{\mu}_{k,\widehat{\mathcal{M}}} - \bar{\mathbf{x}}_{k,\widehat{\mathcal{M}}}) - \hat{\boldsymbol{\delta}}_{\widehat{\mathcal{M}}}^T \hat{\boldsymbol{\Omega}}_{\widehat{\mathcal{M}}} \hat{\boldsymbol{\delta}}_{\widehat{\mathcal{M}}}/2}{\sqrt{\hat{\boldsymbol{\delta}}_{\widehat{\mathcal{M}}}^T \hat{\boldsymbol{\Omega}}_{\widehat{\mathcal{M}}} \boldsymbol{\Sigma}_{\widehat{\mathcal{M}}\widehat{\mathcal{M}}} \hat{\boldsymbol{\Omega}}_{\widehat{\mathcal{M}}} \hat{\boldsymbol{\delta}}_{\widehat{\mathcal{M}}}}} \right). \end{aligned} \quad (3.4)$$

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) $R_{GS-LDA}(\mathbf{X}) = \Phi(-\sqrt{\Delta_p}/2\{1 + O_P(K\sqrt{(\log p)/n})\})$.
- (b) $R_{GS-LDA}(\mathbf{X})/R_{Bayes} - 1 = O_P(\sqrt{(\log p)/n})$, when $\Delta_p < \infty$.
- (c) $R_{GS-LDA}(\mathbf{X})/R_{Bayes} - 1 = O_P(\max\{\Delta_p^{-1}, K^2 \sqrt{(\log p)/n}\})$,

when $\Delta_p \rightarrow \infty$.

Theorem 3 proves that the ratio of $R_{GS-LDA}(\mathbf{X})/R_{Bayes}$ converges to 1 in probability. Statement (a) shows that the convergence rate of $R_{GS-LDA}(\mathbf{X})$ 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 $R_{GS-LDA}(\mathbf{X})/R_{Bayes}$ converges to 1 in probability. Since

Δ_p itself can diverge, R_{Bayes} itself can converge to 0. Thus, statements (b) and (c) are stronger than showing $R_{GS-LDA}(\mathbf{X}) - R_{Bayes} \rightarrow 0$ in probability. Our result indicates that $R_{GS-LDA}(\mathbf{X})$ can converge to 0 as fast as R_{Bayes} does, even when $R_{Bayes} \rightarrow 0$. Also, we show that the convergence rates differ depending on whether Δ_p is bounded. Finally, we remark that similarly as LPD and ROAD, our method also relies on the sparsity assumption on β to reach error rate consistency as shown in Theorem 3. This assumption is needed to avoid the accumulation of errors in estimating β that could ruin the error rate consistency (Bickel and Levina, 2004). Besides, we point out that Theorem 3 also relies on the variable selection consistency we 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 γ such that for any real number t and $k = 0, 1$, it holds that

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

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

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

where the positive constant C depends on the type of the distribution, equaling 2 if it is normal. The proof is to use the conditional distribution of the elliptical distributions, which is similar to what the multivariate normal distribution admits; see Theorem 2.18 of Fang et al. (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 $E(z_c | \mathbf{z}_S = \mathbf{0})$.

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 $\hat{\boldsymbol{\delta}}$ and $\hat{\boldsymbol{\Sigma}}$ upon which the proofs are built; see Lemma 1 in the Supplementary Materials. In general, we assume that

$$0 < \lim_{x \rightarrow \infty} \frac{x^{-\omega} \exp(-cx^\varphi)}{\Psi(-x)} < \infty, \quad (4.5)$$

where $\varphi \in [0, 2]$, $c \in (0, \infty)$, and $\omega \in (0, \infty)$ are some constants. In particular, when Ψ is standard normal, (4.5) holds with $\omega = 1$, $c = 1/2$ and $\varphi = 2$. Then, if Ψ satisfies (4.5) 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 (4.5) with $\varphi < 2$, the tail of the distribution is heavier. In that case, if we assume the moment condition that

$$\max_{k,j \leq p} E|x_{k,j}|^{2\nu} < \infty, \text{ for some } \nu > 0, \text{ and } k = 0, 1 \quad (4.6)$$

where $x_{k,j}$ is the j th element of $\mathbf{x}_k$, the 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 $\hat{\delta}$ and $\hat{\Sigma}$ 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 (4.6), 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 execution time of the proposed GS-LDA method with some other LDA-based methods, such as the sparse discriminant analysis by Clemmensen et al. (2011) (SDA), SLDA, LPD, and ROAD, along with 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 ultra-high-dimensional settings that

involve tens of thousands of features. In these two scenarios, most of the existing methods could not handle such high dimensions and we only aim at testing the viability of GS-LDA. We implement SDA using the **sparseLDA** package. We implement SLDA based on our own coding of the algorithm given in Shao et al. (2011). We implement LPD using the **linprogPD** function from the **clime** package (<https://github.com/rluo/clime>). The implementation for ROAD comes from the publicly available package developed by the authors. We implement SVM using the **e1071** package and implement the Logistic-L1 using the **glmnet** package. The optimal tuning parameters for each method is chosen by a grid search using five-fold cross-validation. The executing time is recorded as the time that each algorithm runs on a computing cluster with Intel Xeon 3.4GHz CPU, while the tuning parameters being fixed at their optimal values.

In the first two scenarios, we consider the following two choices of μ_k and Σ .

Scenario 1: $\mu_0 = \mathbf{0}$ and $\mu_1 = (1, \dots, 1, 0, \dots, 0)^T$, where the first 10 elements are ones and the rest are zeros. $\Sigma = (\sigma_{ij})_{p \times p}$, where $\sigma_{ij} = 0.8^{|i-j|}$ for $1 \leq i, j \leq p$.

Scenario 2: $\mu_0 = \mathbf{0}$ and $\mu_1 = \Sigma\beta$, where $\beta = (0.25, \dots, 0.25, 0, \dots, 0)^T$, where the first 10 elements of β are 0.25's and the rest are zeros. $\Sigma = (\sigma_{ij})_{p \times p}$,

where $\sigma_{ij} = 0.8^{|i-j|}$ for $1 \leq i, j \leq p$.

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 2000 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 as such a sparsity 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 GS-LDA, ROAD and Logistic-L1. We measure 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 1 and 2. Its computational speed is also much faster than the other LDA-based classifiers,

especially when the dimension p is high. It is also faster than SVM and Logistic-L1, implemented by the **e1071** and **glmnet** packages, which are known to be computationally efficient. In both scenarios, when $p = 2000$, the average execution time is around 190 seconds for SDA, 100 seconds for ROAD, 20 seconds for LPD, 3 seconds for SVM, and 1 second for SLDA, but only 0.05 second for GS-LDA. The execution time for the Logistic-L1 depends on the Hessian matrix of the likelihood and the sparsity of β . In Scenario 1, when β is weakly sparse, GS-LDA is still faster than Logistic-L1. In Scenario 2, when β is exactly sparse, their computational time is comparable. This is mainly due to the fact that the **glmnet** package only updates nonzero components of β along its iterations. The proposed GS-LDA method thus offers a substantial boost in computational speed over most other competitors, while rendering an excellent classification rule. In terms of variable selection performance, in Scenario 2, GS-LDA has similar specificity to ROAD and Logistic-L1, and better sensitivity than ROAD. But due to its lower sensitivity than Logistic-L1, GS-LDA has slightly higher misclassification error than Logistic-L1 in this scenario. Finally, we remark that since GS-LDA adds one variable at a time, its errors' variation can be smaller than other optimization based methods, where the number of involved variables are determined by some tuning parameters and small

changes of them can involve multiple new variables into the classification rule. This can be seen from Figures 1 and 2.

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

To further investigate how many dimensions the GS-LDA method can efficiently handle, we simulate two additional scenarios, where we choose μ_k and Σ as follows.

Scenario 3: $\mu_0 = \mathbf{0}$ and $\mu_1 = (1, \dots, 1, 0, \dots, 0)^T$, where the first 10 elements are ones and the rest are zeros. $\Sigma = \Omega^{-1}$, where $\Omega = (\omega_{ij})_{p \times p}$, and $\omega_{ij} = \sqrt{ij}\{2I(i = j \neq p) + I(i = j = p) - I(|i - j| = 1)\}$.

Scenario 4: $\mu_0 = \mathbf{0}$ and $\mu_1 = \Sigma\beta$, where $\beta = (1, \dots, 1, 0, \dots, 0)^T$, where the first 10 elements of β are ones and the rest are zeros. Σ is the same as in Scenario 3.

For each of these two scenarios, we generate 100 training samples for each

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

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, while $\boldsymbol{\beta}$ is exactly sparse in Scenario 4. Again, we use five-fold cross-validation to choose the optimal stopping threshold in GS-LDA. For each scenario, we run 100 replicates; we report the average misclassification rates and the execution time for GS-LDA in Figures 3 and 4.

Figures 3 and 4 show that in both scenarios the GS-LDA method can still perform well when the dimension is ultra-high. As the figures show, when the dimension p grows, the misclassification error remains stable, and the execution time only moderately grows with p . Even when p is as big as

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

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

1×10^5 , the GS-LDA's execution time is only tens of seconds. In contrast, SLDA, LPD and ROAD cannot solve such a problem within tens of hours. For that reason, they were excluded from comparison in these two scenarios.

6. An application to cancer subtype classification

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

We randomly split the dataset 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 as it cannot finish the training within 24 hours. The tuning parameters in these methods are chosen by five-fold cross-validation. The misclassification errors and the execution time of these methods are summarized in Figure 1.

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

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

The GS-LDA method performs well for this dataset, with a mean misclassification error of only 2.4%. On average, this is less than 1 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 ties with that of Logistic-L1 . In term of computational speed, the GS-LDA runs for only 0.3 second on average, which is over 5000 times faster than ROAD, over 1000 times faster than SLDA, over ten times faster than SVM and close to Logistic-L1 on this dataset.

Interestingly, we found that in the 100 splits of the dataset, 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 GS-LDA; it was the first-selected variable 84 times; and it was the only variable selected 56 times. Further reading of the original study (Iwamoto et al., 2010) reveals that the authors defined the ER status by whether the subject’s measured ESR1 expression through probe “205225_at” was higher than 10.18. In other words, the true decision rule is $D_{true}(\mathbf{x}) = I(x_{205225_at} > 10.18)$. We compare the selection frequency of GS-LDA, ROAD and Logistic-L1 over the 100 random splits; see Figure S1 in the supplement. It is seen that GS-LDA selects the true “205225_at” probe much more often with less 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 $D_{GS-LDA}(\mathbf{x}) = I(x_{205225_at} > 10.50)$, which is very close to how the ER status is defined in the original study. When we train ROAD, SLDA, and Logistic-L1 rules using full data, we find that they also include probe “205225_at”, but ROAD includes another 15 probes, 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

In this paper, we developed an efficient greedy search algorithm for performing linear discriminant analysis 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 the existing optimization-based or thresholding methods, as 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 on the fly 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 non-linear classification boundary by using non-linear kernels or Gaussian copulas (Han et al., 2013). Our method may also be extended to multiclass discriminant analysis. Using similar ideas as in Pan et al. (2016) and Mai et al.

REFERENCES

(2019), we can translate a multicategory problem into multiple binary classification problems, to which our method is applicable. Finally, we note that our method requires a key assumption: that the two classes have the same covariance. If that 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 the question of how to develop an efficient algorithm to solve the high-dimensional QDA problem as a future research topic.

Supplementary Materials

Proofs: The proofs of Theorems 1–3, equation (2.3) and the supporting lemmas are provided in the accompanying online supplementary material. (SS-supplement.pdf)

Acknowledgements

The authors gratefully acknowledge *NIH grants R01-AG073259, R01-HL149683 and R01-HG009974.*

References

Anderson, T. W. (1962). *An introduction to multivariate statistical analysis*. Wiley, New York.

REFERENCES

- 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.
- Bickel, P. J. and Levina, E. (2008b). Regularized estimation of large covariance matrices. *The Annals of Statistics* **36**, 199–227.
- Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press
- 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 Ersbøll, 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:

REFERENCES

- 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., Ya-Hui Shiang, C., Santarpia, L., Matsuoka, J., Hortobagyi, G. N., Symmans, W. F. (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

REFERENCES

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.

Department of Biostatistics, University of North Carolina, Chapel Hill

E-mail: hnyang@live.unc.edu

Department of Biostatistics, University of North Carolina, Chapel Hill

E-mail: lin@bios.unc.edu

Department of Biostatistics, University of North Carolina, Chapel Hill

E-mail: quefeng@email.unc.edu