Hindawi
Journal of Probability and Statistics
Volume 2022, Article ID 1041752, 14 pages
https://doi.org/10.1155/2022/1041752

Research Article

NetDA: An R Package for Network-Based Discriminant Analysis Subject to Multilabel Classes

Li-Pang Chen

Department of Statistics, National Chengchi University, No. 64, Section 2, Zhinan Road, Wenshan District, Taipei 116, Taiwan

Correspondence should be addressed to Li-Pang Chen; lchen723@nccu.edu.tw

Received 18 March 2022; Revised 1 June 2022; Accepted 18 July 2022; Published 27 September 2022

Academic Editor: Hyungjun Cho

In this paper, we introduce the R package NetDA, which aims to deal with multiclassification with network structures in predictors accommodated. To address the natural feature of network structures, we apply Gaussian graphical models to characterize dependence structures of the predictors and directly estimate the precision matrix. After that, the estimated precision matrix is employed to linear discriminant functions and quadratic discriminant functions. The R package NetDA is now available on CRAN, and the demonstration of functions is summarized as a vignette in the online documentation.

1. Introduction

Multiclassification, known as a classification problem that the number of classes is greater than two, is a great challenge in data science. In supervised learning, discriminant analysis has been a useful method to do classification. In the conventional method (e.g., Hastie et al. [1]; Section 4.3), linear discriminant functions, which are formulated in terms of mean vectors and the inverse of covariance matrices of the predictors, are used to classify subjects. In addition, some advanced methods have been proposed to address complex settings in the past literature. For example, Guo et al. [2] discussed the LDA method and its application in microarray data analysis. Safo and Ahn [3] studied generalized sparse linear discriminant analysis for multilabel responses. In the presence of high-dimensional predictors, several advanced approaches have also been explored (e.g., Clemmensen et al. [4]; Witten and Tibshirani [5]).

However, network structures of predictors, which reflect (pairwise) dependence among predictors, are ubiquitous in data analysis [6]. In the recent developments, Chen et al. [7] proposed a graphical-based logistic regression model. He et al. [8] proposed surrogate variables that were transformed from network structures and implemented them to the support vector machine. Regarding the framework of discriminant analysis, Cai et al. [9] and Liu et al. [10] developed graph-based linear discriminant analysis, but their approaches are restricted to binary responses. Moreover, for general data analysts, it is important for them to directly implement existing software and do data analysis. However, rare software related to classification with network structure accommodated has been available. While some R packages related to discriminant analysis exist, such as MASS, sparseLDA, and penalizedLDA, they are not able to handle network structures in predictors.

Motivated by these concerns and to address these challenges, we follow the strategy proposed by Chen [11] and develop an R package, which is called NetDA. Under the normality assumption for predictors, we apply the graphical lasso method to estimate precision matrices and the corresponding network structures for the predictors. Since precision matrices are the inverse of covariance matrices, it motivates us to directly implement them to linear/quadratic discriminant functions. This strategy is different from the conventional linear discriminant analysis that simply employs empirical estimates of covariance matrices. Moreover, the other issue is prediction. Based on fitted models and predicted values, we also develop a function that contains several commonly used criteria to assess the performance of classification and prediction.

The article is organized as follows. Section 2 introduces the data structure and outlines the methodology in our package. Section 3 describes the usage of the package NetDA. Section 4 illustrates the package by a real dataset. We finally conclude the article in Section 5.

2. Overview of Methodology

In this section, we primarily overview the data structure and network-based discriminant analysis proposed by Chen [11].

2.1. Data Structure. Suppose that the data contain n subjects that come from I classes, where I is a fixed integer greater than 2 and the classes are nominal. Let n_i be the size in class $i$with $i = 1, \ldots, I$, and hence, $n = \sum_{i=1}^{I} n_i$. Let $\mathbf{Y}$ denote the n-dimensional vector of responses with the jth component being $Y_j = i$, which reflects the class membership that the jth subject is in the ith class for $i = 1, \ldots, I$ and $j = 1, \ldots, n$.

Let $p > 1$ denote the dimension of predictors for each subject. Define $\mathbf{X} = [X_{j,k}]$ as the $n \times p$ matrix of predictors for $j = 1, \ldots, n$ and $k = 1, \ldots, p$, where the component $X_{j,k}$ represents the kth predictor for the jth subject. Furthermore, let $\mathbf{X}_{\cdot k} = (X_{1,k}, \ldots, X_{n,k})^\top$ represent the n-dimensional vector of the kth predictor in the kth column of $\mathbf{X}$, and let $\mathbf{X}_{j\cdot} = (X_{j,1}, \ldots, X_{j,p})^\top$ denote the p-dimensional predictor vector for the jth subject in the jth row of $\mathbf{X}$. Let $\{\{\mathbf{X}_{j\cdot}, Y_j\}: j = 1, \ldots, n\}$ denote an independent and identically distributed (i.i.d.) sample. We let lower case letters represent realized values for the corresponding random variables. For example, $\mathbf{x}_{j\cdot}$ stands for a realized value of $\mathbf{X}_{j\cdot}$.

2.2. Gaussian Graphical Models. For $i = 1, \ldots, I$ and $j = 1, \ldots, n$, let $f_{j|i}(\mathbf{x}_{j\cdot})$ denote the conditional probability density function of the predictor $\mathbf{X}_{j\cdot}$ taking a value $\mathbf{x}_{j\cdot}$ given that subject j comes from the i th class. We particularly consider the case where the conditional distribution $\mathbf{X}_{j\cdot}$ given $Y_j = i$ is assumed to be a multivariate normal distribution with a mean vector $\boldsymbol{\mu}_\mathbf{i}$ and a positive-definite covariance matrix $\boldsymbol{\Sigma}_\mathbf{i}$. Then, the conditional probability density function is given by

$$f_{j|i}\left(\mathbf{x}_{j\cdot}\right) = \frac{1}{(2\pi)^{p/2}\left|\boldsymbol{\Sigma}_i\right|^{1/2}} \exp\left\{-\frac{1}{2}\left(\mathbf{x}_{j\cdot} - \boldsymbol{\mu}_i\right)^\top \boldsymbol{\Sigma}_i^{-1}\left(\mathbf{x}_{j\cdot} - \boldsymbol{\mu}_i\right)\right\}. \tag{1}$$

Moreover, by the suitable reparametrization (e.g., Hastie et al. [12]; p. 246), we can transfer (1) to the Gaussian graphical model (GGM) based on class i. The exact formulation is given by

$$f\left(\mathbf{x}_{j\cdot}; \boldsymbol{\beta}_i, \boldsymbol{\Theta}_i\right) = \exp\left\{\sum_{r\in V} \beta_{ir} x_{j,r} - \frac{1}{2}\sum_{(s,t)\in E} \theta_{i,st} x_{j,s} x_{j,t} - \frac{1}{2}\log\det\left(\frac{\boldsymbol{\Theta}_i}{2\pi}\right)\right\}, \tag{2}$$

where $V = \{1, \ldots, p\}$ includes all the indices and $E \subset V \times V$ contains all pairs with unequal coordinates, which yields a graph $G \triangleq (V, E)$ (e.g., Hastie et al. [1]; Chapter 17); $\Theta_i = [\theta_{i,st}]$ is the $p \times p$ symmetric precision matrix with $\Theta_i \triangleq \Sigma_i^{-1}$; and $\beta_i = (\beta_{i1}, \ldots, \beta_{ip})^\top$ is the p-dimensional vector of parameters with $\mu_i \triangleq -\Theta_i^{-1}\beta_i$. Our main interest is to estimate Θ_i, since, as we will see later, the main concern in discriminant analysis is to estimate the inverse of the covariance matrix. On the other hand, from the perspective of graphical models, nonzero $\theta_{i,st}$ implies that $X_{j,s}$ and $X_{j,t}$ are conditionally dependent given other variables in class i, while zero value of $\theta_{i,st}$ gives conditional independence of $X_{j,s}$ and $X_{j,t}$ given other variables. Thus, the precision matrix Θ_i reflects the network structure of the predictors.

In the past literature, graphical LASSO (GLASSO) [13] is a common method to estimate Θ_i. The key idea of GLASSO is based on the likelihood function. To see this, we follow the similar discussion in page 247 of Hastie et al. [12] and write the log-likelihood function of Θ_i based on (2) with $\beta_i = 0$:

$$\mathscr{L}\left(\boldsymbol{\Theta}_\mathbf{i}\right) = \frac{1}{n_i}\sum_{j=1}^{n_i} \log\left\{f\left(\mathbf{x}_{j\cdot}; \boldsymbol{\Theta}_\mathbf{i}\right)\right\} = \log\det\left(\boldsymbol{\Theta}_\mathbf{i}\right) - \operatorname{trace}\left(\mathbf{S}_\mathbf{i}\boldsymbol{\Theta}_\mathbf{i}\right), \tag{3}$$

where $\mathbf{S}_i = (1/n_i)\sum_{j=1}^{n_i} \mathbf{x}_{j\cdot}\mathbf{x}_{j\cdot}^\top$, and $\operatorname{trace}(\cdot)$ is the sum of diagonal entries for a square matrix, and

$$\operatorname{logdet}\left(\boldsymbol{\Theta}_\mathbf{i}\right) = \begin{cases} \displaystyle\sum_{k=1}^{p} \log\{\lambda\left(\boldsymbol{\Theta}_\mathbf{i}\right)\}, & \text{if } \boldsymbol{\Theta}_\mathbf{i} \text{ is positive definite,} \\ -\infty, & \text{otherwise,} \end{cases} \tag{4}$$

with $\lambda(\Theta_i)$ being the jth eigenvalue of Θ_i. Assume that the precision matrix Θ_i is sparse. To estimate $\theta_{i,st}$ and identify network structures by retaining dependent pairs of vertices and removing independent ones, we apply the L_1-norm as a constraint to achieve the desired result. In other words, the estimator of Θ_i can be obtained by the following optimization:

$$\widehat{\boldsymbol{\Theta}}_\mathbf{i} = \arg\max_{\boldsymbol{\Theta}_\mathbf{i}}\left\{\operatorname{logdet}\left(\boldsymbol{\Theta}_\mathbf{i}\right) - \operatorname{trace}\left(\mathbf{S}_\mathbf{i}\boldsymbol{\Theta}_\mathbf{i}\right) - \zeta\rho\left(\boldsymbol{\Theta}_\mathbf{i}\right)\right\}, \tag{5}$$

where $\rho(\Theta_i) \triangleq \sum_{s\neq t}|\theta_{i,st}|$ is the penalty function and ζ is a tuning parameter. The optimization problem (5) is called GLASSO [13]. The detailed algorithm can be found in page 248 of Hastie et al. [12], and the estimator of ζ can be determined by Bayesian information criterion (BIC). By the similar discussion in Yuan and Lin [14], the estimated network structure determined by (5) is equal to the true graph with probability approaching one under suitable conditions. For the computation, the R package glasso can be implemented to derive the estimate $\widehat{\Theta}_i$.

2.3. Discriminant Analysis. The idea of discriminant analysis is to model the distribution of the predictors $\mathbf{X}_{j\cdot}$ separately for each of the response classes Y_j, and then to use the Bayes theorem to describe the conditional probabilities $P(Y_j = i|\mathbf{X}_{j\cdot} = \mathbf{x}_{j\cdot})$ (e.g., James et al. [15]).

Specifically, let $\pi_i \triangleq P(Y_j = i)$ denote the probability that the j th subject is randomly selected from class i so that $\sum_{i=1}^{I} \pi_i = 1$. Moreover, applying the Bayes theorem to the conditional density function $f_{j|i}(\mathbf{x}_{j\cdot})$ and π gives the posterior probability as

$$P\left(Y_j = i|\mathbf{X}_{j\cdot} = \mathbf{x}_{j\cdot}\right) = \frac{f_{j|i}\left(\mathbf{x}_{j\cdot}\right)\pi_i}{\sum_{l=1}^{I} f_{j|l}\left(\mathbf{x}_{j\cdot}\right)\pi_l}, \tag{6}$$

for $i = 1, \ldots, I$ and $j = 1, \ldots, n$.

To compare two classes i and l with $i \neq l$, we calculate the log-ratio of (6), given by

$$\log\left\{\frac{P\left(Y_j = i|\mathbf{X}_{j\cdot} = \mathbf{x}_{j\cdot}\right)}{P\left(Y_j = l|\mathbf{X}_{j\cdot} = \mathbf{x}_{j\cdot}\right)}\right\} = \log\left(\frac{f_{j|i}\left(\mathbf{x}_{j\cdot}\right)}{f_{j|l}\left(\mathbf{x}_{j\cdot}\right)}\right) + \log\left(\frac{\pi_i}{\pi_l}\right). \tag{7}$$

With a distribution assumption (1) imposed, we now have two scenarios for the first term in the right-hand side of (7), say $\log\left(f_{j|i}\left(\mathbf{x}_{j\cdot}\right)/f_{j|l}\left(\mathbf{x}_{j\cdot}\right)\right)$.

Scenario 1. If the covariance matrices Σ_i in (1) are assumed to be common, that is, $\Sigma_i = \Sigma$ for every i with Σ being a positive definite matrix, (7) becomes

$$\log\left(\frac{\pi_i}{\pi_l}\right) - \frac{1}{2}\left(\boldsymbol{\mu}_i + \boldsymbol{\mu}_l\right)^{\top}\boldsymbol{\Sigma}^{-1}\left(\boldsymbol{\mu}_i + \boldsymbol{\mu}_l\right) + \mathbf{x}_{j\cdot}^{\top}\boldsymbol{\Sigma}^{-1}\left(\boldsymbol{\mu}_i + \boldsymbol{\mu}_l\right). \tag{8}$$

If equation (8) >0, then $P\left(Y_j = i|\mathbf{X}_{j\cdot} = \mathbf{x}_{j\cdot}\right) > P\left(Y_j = l|\mathbf{X}_{j\cdot} = \mathbf{x}_{j\cdot}\right)$, showing that subject j with predictors $\mathbf{X}_{j\cdot} = \mathbf{x}_{j\cdot}$ is more likely selected from class i than from class l. Consequently, (8) defines a boundary between classes i and l in the sense that there is a linear function in $\mathbf{x}_{j\cdot}$ separating classes i and l.

Motivated by the form of (8), we consider a linear function in $\mathbf{x}$ as

$$\delta_i\left(\mathbf{x}\right) \triangleq \log\left(\pi_i\right) - \frac{1}{2}\boldsymbol{\mu}_i^{\top}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_i + \mathbf{x}^{\top}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_i. \tag{9}$$

Moreover, μ_i and π_i can be empirically estimated, respectively, as

$$\widehat{\boldsymbol{\mu}}_i = \frac{1}{n_i}\sum_{Y_j = i}\mathbf{x}_{j\cdot}, \qquad \widehat{\pi}_i = \frac{n_i}{n}. \tag{10}$$

For the estimation of Σ^{-1}, or equivalently Θ, we adopt (5) by pooling all subjects in the dataset and denote $\widetilde{\Theta}$ as the estimator of Θ. Therefore, (9) can be estimated as

$$\widetilde{\delta}_i\left(\mathbf{x}\right) = \log\left(\widehat{\pi}_i\right) - \frac{1}{2}\widehat{\boldsymbol{\mu}}_i^{\top}\widetilde{\boldsymbol{\Theta}}\widehat{\boldsymbol{\mu}}_i + \mathbf{x}^{\top}\widetilde{\boldsymbol{\Theta}}\widehat{\boldsymbol{\mu}}_i, \tag{11}$$

and we call (11) the network-based linear discriminant function (NetLDA) and it is used to determine the class label for a new observation. For the prediction of a new subject with the predictor $\widetilde{\mathbf{x}}$, we first calculate $\widetilde{\delta}_i\left(\widetilde{\mathbf{x}}\right)$ using (11) for $i = 1, \ldots, I$. Next, we find i^* that is defined as

$$i^* = \arg\max_{i=1,\ldots,I}\widetilde{\delta}_i\left(\widetilde{\mathbf{x}}\right); \tag{12}$$

and the class label for this subject is then predicted as i^*.

Scenario 2. We allow $\Sigma_i \neq \Sigma_l$, or equivalently, $\Theta_i \neq \Theta_l$, for any $i \neq l$ and $i, l = 1, \ldots, I$. Then, under a distribution assumption (1), we have

$$\begin{aligned}\log\left\{\frac{f_{j|i}\left(\mathbf{x}_{j\cdot}\right)}{f_{j|l}\left(\mathbf{x}_{j\cdot}\right)}\right\} &= \log\left[\frac{\left(1/\left(2\pi\right)^{p/2}\left|\boldsymbol{\Sigma}_i\right|^{1/2}\right)\exp\left\{-(1/2)\left(\mathbf{x}_{j\cdot} - \boldsymbol{\mu}_i\right)^{\top}\boldsymbol{\Theta}_i\left(\mathbf{x}_{j\cdot} - \boldsymbol{\mu}_i\right)\right\}}{\left(1/\left(2\pi\right)^{p/2}\left|\boldsymbol{\Sigma}_l\right|^{1/2}\right)\exp\left\{-(1/2)\left(\mathbf{x}_{j\cdot} - \boldsymbol{\mu}_l\right)^{\top}\boldsymbol{\Theta}_l\left(\mathbf{x}_{j\cdot} - \boldsymbol{\mu}_l\right)\right\}}\right] \\ &= \log\left(\frac{\left|\boldsymbol{\Theta}_l\right|^{-(1/2)}}{\left|\boldsymbol{\Theta}_i\right|^{-(1/2)}}\right) + \frac{1}{2}\left\{\left(\mathbf{x}_{j\cdot} - \boldsymbol{\mu}_l\right)^{\top}\boldsymbol{\Theta}_l\left(\mathbf{x}_{j\cdot} - \boldsymbol{\mu}_l\right) - \left(\mathbf{x}_{j\cdot} - \boldsymbol{\mu}_i\right)^{\top}\boldsymbol{\Theta}_i\left(\mathbf{x}_{j\cdot} - \boldsymbol{\mu}_i\right)\right\}.\end{aligned} \tag{13}$$

Replacing the first term in the right-hand side of (7) by (13) yields

$$\log\left(\frac{\pi_i}{\pi_l}\right) + \log\left(\frac{\left|\boldsymbol{\Theta}_l\right|^{-(1/2)}}{\left|\boldsymbol{\Theta}_i\right|^{-(1/2)}}\right) + \frac{1}{2}\left\{\left(\mathbf{x}_{j\cdot} - \boldsymbol{\mu}_l\right)^{\top}\boldsymbol{\Theta}_l\left(\mathbf{x}_{j\cdot} - \boldsymbol{\mu}_l\right) - \left(\mathbf{x}_{j\cdot} - \boldsymbol{\mu}_i\right)^{\top}\boldsymbol{\Theta}_i\left(\mathbf{x}_{j\cdot} - \boldsymbol{\mu}_i\right)\right\}. \tag{14}$$

Therefore, based on (14), we further define a quadratic function of $\mathbf{x}$ based on the class i:

$$\varphi_i\left(\mathbf{x}\right) \triangleq \log\left(\pi_i\right) + \frac{1}{2}\log\left|\boldsymbol{\Theta}_i\right| - \frac{1}{2}\left(\mathbf{x} - \boldsymbol{\mu}_i\right)^{\top}\boldsymbol{\Theta}_i\left(\mathbf{x} - \boldsymbol{\mu}_i\right). \tag{15}$$

For $i = 1, \ldots, I$, the estimator of the precision matrix Θ_i, denoted as $\widehat{\Theta}_i$, is obtained by (5) based on the predictor information in class i; μ_i and π_i can be estimated by (10). Therefore, (15) can be estimated by

$$\widehat{\varphi}_i(\mathbf{x}) = \log(\widehat{\pi}_i) + \frac{1}{2}\log\left|\widehat{\boldsymbol{\Theta}}_i\right| - \frac{1}{2}(\mathbf{x} - \widehat{\boldsymbol{\mu}}_i)^\top \widehat{\boldsymbol{\Theta}}_i(\mathbf{x} - \widehat{\boldsymbol{\mu}}_i). \quad (16)$$

The function (16) is called the network-based quadratic discriminant function (NetQDA) and is used to determine the class label for a new observation. For the prediction of a new subject with the predictor $\widetilde{\mathbf{x}}$, we first calculate $\widehat{\varphi}_i(\widetilde{\mathbf{x}})$ using (16) for $i = 1, \ldots, I$. Next, we find i^* that is defined as

$$i^* = \arg\max_{i=1,\ldots,I} \widehat{\varphi}_i(\widetilde{\mathbf{x}}); \quad (17)$$

and the class label for this subject is then predicted as i^*.

2.4. Assessment of Classification and Prediction. We first introduce micro-averaged metrics (e.g., Chen et al. [7]). Let $\mathscr{V}$ and $\mathscr{T}$ represent the classes of the subject indexes for validation and training datasets, respectively. Let $n = |\mathscr{T}|$ and $m = |\mathscr{V}|$ denote the sizes of the training and validation data, respectively. We use the training data $\mathscr{T}$ to fit models and then apply fitted models to compute predicted values $\widehat{Y}_j$ for $j \in \mathscr{V}$. After that, for $i = 1, \ldots, I$, we calculate the number of the true positives (TP), the number of the false positives (FP), and the number of the false negatives (FN) under the validation data $\mathscr{V}$, respectively:

$$\begin{aligned} \mathrm{TP}_i &= \sum_{j\in\mathscr{V}} \mathbb{I}\left(Y_j = i, \widehat{Y}_j = i\right), \\ \mathrm{FP}_i &= \sum_{j\in\mathscr{V}} \mathbb{I}\left(Y_j \neq i, \widehat{Y}_j = i\right), \\ \mathrm{FN}_i &= \sum_{j\in\mathscr{V}} \mathbb{I}\left(Y_j = i, \widehat{Y}_j \neq i\right), \end{aligned} \quad (18)$$

where $\mathbb{I}(\cdot)$ is the indicator function.

We define precision (PRE) and recall (REC) under the validation data $\mathscr{V}$, respectively, as

$$\begin{aligned} \mathrm{PRE} &= \frac{\sum_{i=1}^{I} \mathrm{TP}_i}{\sum_{i=1}^{I} \mathrm{TP}_i + \sum_{i=1}^{I} \mathrm{FP}_i}, \\ \mathrm{REC} &= \frac{\sum_{i=1}^{I} \mathrm{TP}_i}{\sum_{i=1}^{I} \mathrm{TP}_i + \sum_{i=1}^{I} \mathrm{FN}_i}. \end{aligned} \quad (19)$$

Then, micro-F-score is defined as

$$F = 2 \times \frac{\mathrm{PRE} \times \mathrm{REC}}{\mathrm{PRE} + \mathrm{REC}}. \quad (20)$$

According to definitions in (19), when all subjects are correctly classified, FP and FN are equal to zero, yielding that PRE and REC are equal to one; if all subjects are falsely classified, then TP is equal to zero, and thus, PRE and REC are equal to zero. Therefore, values of PRE and REC are between zero and one. Moreover, under the range $[0, 1]$, the F-score falls in $[0, 1]$ as well by treating 0/0 as zero. In principle, the higher values of PRE, REC, and F-score reflect the better performance and the more accurate classification (e.g., Chen et al. [7]).

In addition to criteria above, the other commonly used criterion is the adjusted Rand index (ARI). For $i, i' = 1, \ldots, I$ and under the validation data $\mathscr{V}$, we define

$$n_{ii'} = \sum_{j\in\mathscr{V}} \mathbb{I}\left(Y_j = i, \widehat{Y}_j = i'\right). \quad (21)$$

Moreover, we define $a_i = \sum_{i'=1}^{I} n_{ii'}$ for $i = 1, \ldots, I$ and $b_{i'} = \sum_{i=1}^{I} n_{ii'}$ for $i' = 1, \ldots, I$. Then, ARI under the validation data $\mathscr{V}$ is defined as Hubert and Arabie [16].

$$\mathrm{ARI} = \frac{\sum_{i,i'} \binom{n_{ii'}}{2} - \left(\left\{\sum_i \binom{a_i}{2} \sum_{i'} \binom{b_{i'}}{2}\right\} \Big/ \binom{n}{2}\right)}{\left(\left\{\sum_i \binom{a_i}{2} + \sum_{i'} \binom{b_{i'}}{2}\right\} /2\right) - \left(\left\{\sum_i \binom{a_i}{2} \sum_{i'} \binom{b_{i'}}{2}\right\} \Big/ \binom{n}{2}\right)}. \quad (22)$$

As mentioned in Hubert and Arabie [16], ARI is bounded above by one, and the higher value of ARI indicates the more accurate classification.

2.5. Benchmark of NetDA. The conventional linear discriminant methods (e.g., MASS, sparseLDA, and penalizedLDA) aim to adopt (9) with Θ estimated by the inverse of the empirical estimator of Σ. However, this approach may encounter cumbersome computation or possible singularity when calculating inverse matrices. In addition, if Θ is sparse, all entries in empirical estimators of Θ are nonzero, and it indicates that some unconnected pairs of predictors may be falsely included. As a result, imprecise estimator of Θ may implicitly affect the performance of classification.

Unlike existing methods, the first contribution of NetDA is to estimate Θ directly. The graphical lasso method is a tool to identify zero entries in Θ and estimate nonzero ones. This approach enables us to retain connected pairs of predictors and exclude unconnected ones. Moreover, our implementation can avoid computing inverse of Σ.

The second contribution of NetDA is to handle heterogeneous network structure that stratifies the predictor information by class when characterizing the predictor network structures, and is able to deal with multiclassification by adopting network structure in each class.

3. Details of NetDA

In this section, we first overview the technique that we need in the existing package. After that, we describe functions in our package.

3.1. Library Overview. In our package, we use the package glasso in R software. Specifically, the package glasso follows from the graphical lasso method proposed by Friedman et al. [13]. The purpose of glasso is to detect network structures of random vectors that follow multivariate normal distributions. In particular, under multivariate normal distributions and operations in glasso, the detection of network structures is equivalent to the estimation of precision matrix.

3.2. WineData. In our package, we take the wine dataset as an example, which is available in https://archive.ics.uci.edu/ml/datasets/wine. These data were collected based on a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. In this dataset, there are three types of wines and 13 constituents, including alcohol (Alcohol), malic acid (Malic acid), ash (Ash), alkalinity of ash (Alkalinity), magnesium (Magnesium), total phenols (phenols), flavanoids (Flavanoids), nonflavanoid phenols (Nonflavanoid), proanthocyanins (Proanthocyanins), color intensity (Color), hue (Hue), OD280/OD315 of diluted wines (OD280), and proline (Proline).

In the following analysis, the response is types of wines that are labeled as 1, 2, and 3; constituents are treated as predictors that are continuous. The goal is to adopt the information of constituents to construct predictive models and then use them to classify type of wines for a given subject.

3.3. NetDA. NetDA contains two methods. The first method is called NetLDA, which aims to estimate the precision matrix by pooling all individuals in the data, and the corresponding discriminant function is given by (11). The second approach is called NetQDA, whose strategy is to estimate precision matrices based on individuals in different classes, and then use class-dependent estimated precision matrices to define quadratic discriminant functions in (16). Unlike NetLDA, NetQDA takes possibly class-dependent network structures of the predicted variables into account and uses network structures in different classes to determine which classes individuals belong to. When either linear discriminant functions or quadratic discriminant functions are obtained, they can be used to determine the class for a new subject.

To implement the NetLDA and NetQDA methods, we use the following command:

NetDA (X, Y, method, X_test) where the meaning of each argument is described as follows:

(i) X: this is an $n \times p$ matrix of the predictors from the training data

(ii) Y: this is an n-dimensional vector of the response from the training data, whose elements are positive integers and reflect class-labels

(iii) Method: it is a scalar to determine the classification method: method = 1 represents NetLDA in (11), and method = 2 represents NetQDA in (16)

(iv) X_test: this is an $m \times p$ matrix of the predictors from the validation data

The purpose of NetDA is to apply the training data "X" and "Y" to determine a fitted model that is specified by the argument "method." After that, we use "X_test and a fitted model to determine the predicted class for subjects in the validation data. Therefore, the function NetDA returns a list of components:

(i) yhat: it is a vector of predicted responses obtained by NetLDA or NetQDA based on the predictors in the validation data (X_test).

(ii) Network: this is the estimators of precision matrices. If "method = 1" is chosen, then there is one precision matrix; if "method = 2" is given, then there are I precision matrices.

3.4. Metrics. The function Metrics is utilized to assess the performance of classification and prediction based on some commonly used criteria that are introduced in the Section 2.4. Specifically, given responses from the validation data and predicted values obtained by NetDA, we first derive a confusion matrix to see the classification result. To further assess the performance of prediction, we evaluate precision, recall, F-score, and ARI defined in (19), (20), and (22), respectively.

To obtain the desired results, we use the following command:

Metrics (yhat, Y_test) where the meaning of each argument is described as follows:

(i) yhat: this is an m-dimensional vector of the predicted responses determined by NetDA or other methods

(ii) Y_test: this is an m-dimensional vector of the response from the validation data

The function metrics returns a list of components:

(i) Confusion matrix: a confusion matrix based on predicted values (yhat) and responses from the validation data (Y_test)

(ii) (PRE, REC, F-score): values of precision, recall, and F-score defined in (19) and (20), respectively

(iii) ARI: values of the ARI defined in (22)

4. Demonstration of NetDA

In this section, we demonstrate standard analysis of classification and prediction based on two functions in the

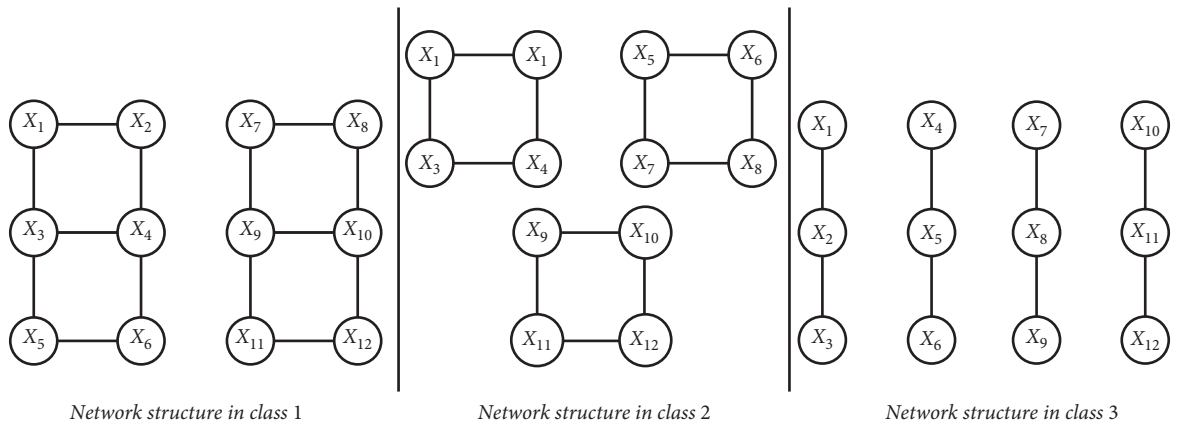

Figure 1: Predictor network structures for four different classes.

package. To show the advantage of NetDA, we compare with existing packages MASS, sparseLDA, and penalizedLDA.

4.1. Simulation. Let $I = 3$ denote the number of classes, and let $p = 12$ denote the dimension of predictors. We specify the sample size $n = 600$, in which the size of the ith class is given by $n_i = (n/I)$ for $i = 1, \ldots, I$. For each class, we consider different network structures of predictors. Specifically, for class $i = 1, \ldots, I$, let $\mathbf{\Omega}_{o,i}$ denote a $p \times p$ matrix whose diagonal entries are zero and off-diagonal entries are specified as either one or zero to reflect edges of the corresponding two nodes in Figure 1. That is, for $s \neq t$, entry (s, t) in $\mathbf{\Omega}_{o,i}$ is 1 if the edge exists between $\mathbf{X}_{\cdot s}$ and $\mathbf{X}_{\cdot t}$ and 0 otherwise. In addition, we further define a $p \times p$ diagonal matrix $\mathbf{\Omega}_{d,i}$ whose nonzero entries are taken as the common value $0.1 + |\Lambda_{\min}(\mathbf{\Omega}_{o,i})|$, where $\Lambda_{\min}(\mathbf{\Omega}_{o,i})$ represents the smallest eigenvalue of $\mathbf{\Omega}_{o,i}$. Finally, we define the precision matrix as $\mathbf{\Theta}_i \triangleq \Omega_{o,i} + \mathbf{\Omega}_{d,i}$ that is invertible. Therefore, based on Gaussian graphical models, the p-dimensional vector of predictors in class i is generated from a multivariate normal distribution with mean zero and the covariance matrix $\Sigma_i = \mathbf{\Theta}_i^{-1}$ for $i = 1, \ldots, I$.

Let Theta1, Theta2, and Theta3 denote $p \times p$ matrices that reflect network structures in left, middle, and right panels of Figure 1, respectively, and one can specify those three matrices as follows:

>Theta1

	[1,]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]	[,11]	[,12]
[1,]	0	1	1	0	0	0	0	0	0	0	0	0
[2,]	1	0	0	1	0	0	0	0	0	0	0	0
[3,]	1	0	0	1	1	0	0	0	0	0	0	0
[4,]	0	1	1	0	0	1	0	0	0	0	0	0
[5,]	0	0	1	0	0	1	0	0	0	0	0	0
[6,]	0	0	0	1	1	0	0	0	0	0	0	0
[7,]	0	0	0	0	0	0	0	1	1	0	0	0
[8,]	0	0	0	0	0	0	1	0	0	1	0	0
[9,]	0	0	0	0	0	0	1	0	0	1	1	0
[10,]	0	0	0	0	0	0	0	1	1	0	0	1
[11,]	0	0	0	0	0	0	0	0	1	0	0	1
[12,]	0	0	0	0	0	0	0	0	0	1	1	0

. (23)

> Theta2

	[, 1]	[, 2]	[, 3]	[, 4]	[, 5]	[, 6]	[, 7]	[, 8]	[, 9]	[, 10]	[, 11]	[, 12]
[1,]	0	1	1	0	0	0	0	0	0	0	0	0
[2,]	1	0	0	1	0	0	0	0	0	0	0	0
[3,]	1	0	0	1	0	0	0	0	0	0	0	0
[4,]	0	1	1	0	0	0	0	0	0	0	0	0
[5,]	0	0	0	0	0	1	1	0	0	0	0	0
[6,]	0	0	0	0	1	0	0	1	0	0	0	0
[7,]	0	0	0	0	1	0	0	1	0	0	0	0
[8,]	0	0	0	0	0	1	1	0	0	0	0	0
[9,]	0	0	0	0	0	0	0	0	0	1	1	0
[10,]	0	0	0	0	0	0	0	0	1	0	0	1
[11,]	0	0	0	0	0	0	0	0	1	0	0	1
[12,]	0	0	0	0	0	0	0	0	0	1	1	0

. (24)

> Theta3

	[, 1]	[, 2]	[, 3]	[, 4]	[, 5]	[, 6]	[, 7]	[, 8]	[, 9]	[, 10]	[, 11]	[, 12]
[1,]	0	1	0	0	0	0	0	0	0	0	0	0
[2,]	1	0	1	0	0	0	0	0	0	0	0	0
[3,]	0	1	0	0	0	0	0	0	0	0	0	0
[4,]	0	0	0	0	1	0	0	0	0	0	0	0
[5,]	0	0	0	1	0	1	0	0	0	0	0	0
[6,]	0	0	0	0	1	0	0	0	0	0	0	0
[7,]	0	0	0	0	0	0	0	1	0	0	0	0
[8,]	0	0	0	0	0	0	1	0	1	0	0	0
[9,]	0	0	0	0	0	0	0	1	0	0	0	0
[10,]	0	0	0	0	0	0	0	0	0	0	1	0
[11,]	0	0	0	0	0	0	0	0	0	1	0	1
[12,]	0	0	0	0	0	0	0	0	0	0	1	0

. (25)

Following the description above, we generate the $n \times p$ dimensional matrix for predictors and then determine the simulated data:

>Theta1 = Theta1 + diag (0.1 + abs (min (eigen (Theta1) $value)), p)

>Sigma1 = cov2cor (solve (Theta1))

Table 1: Simulation results.

	MASS	sparseLDA	penalizedLDA	NetDA (NetLDA)	NetDA (NetQDA)
PRE	0.627	0.686	0.669	0.729	0.891
REC	0.627	0.686	0.669	0.729	0.891
F-score	0.627	0.686	0.669	0.729	0.891
ARI	0.521	0.543	0.530	0.622	0.707

```
>X1 = mvrnorm (n = (n/I), rep (0, p), Sigma1, tol = 1e − 6, empirical = FALSE)
>
>Theta2 = Theta2 + diag (0.1 + abs (min (eigen (Theta2) $value)), p)
>Sigma2 = cov2cor (solve (Theta2))
>X2 = mvrnorm (n = (n/I), rep (0, p), Sigma2, tol = 1e − 6, empirical = FALSE)
>
>Theta3 = Theta3 + diag (0.1 + abs (min (eigen (Theta3) $value)), p)
>Sigma3 = cov2cor (solve (Theta3))
>X3 = mvrnorm (n = (n/I), rep (0, p), Sigma3, tol = 1e − 6, empirical = FALSE)
>data = cbind (c (rep (1, n/I), rep (2, n/I), rep (3, n/I)), rbind (X1, X2, X3))
```

To perform classification, we implement NetDA function with two different scenarios described in Section 2.3. In addition, we demonstrate three existing methods labeled as lda (MASS), sda (sparseLDA), and pda (penalizedLDA), respectively. Detailed descriptions are given below:

```
>Y = data [, 1]
>X = data [, 2 : 13]
>#Demonstration of MASS
>lda = lda (Y·X, prior = c (length (which (Y = = 1)), length (which (Y = = 2)), ++length (which (Y = = 3)))/ length (Y))
>yhat_lda = predict (lda, data.frame (X)) $class
>
>#Demonstration of sparseLDA
>y = matrix (0, n, I)
>y [1 : 200, 1] = 1
>y [201 : 400, 2] = 1
>y [401 : 600, 3] = 1
>colnames (y) < −c ("1," "2," "3")
>sda = sda (data.frame (X), y, lambda = 1e − 6, stop = −1, maxIte = 25, +trace = TRUE)
>yhat_sda = as.numeric (unlist (predict (sda, data.-frame (X)) $class))
>#Demonstration of penalizedLDA
>pda = PenalizedLDA (X, Y, lambda = 0.14, K = 2)
>yhat_pda = as.numeric (unlist (predict (pda, data.-frame (X)))) [1 : n]
>#Demonstration of NetDA
>yhat_netlda = NetDA (X, Y, method = 1, X) $yhat
>yhat_netqda = NetDA (X, Y, method = 2, X) $yhat
```

After that, to assess the performance of classification, we adopt the function Metrics to compute values of criteria (19) and (20) as shown by [2] and (22) indicated by [3].

```
>F_lda = Metrics (yhat_lda, Y) [2]
>F_sda = Metrics (yhat_sda, Y) [2]
>F_pda = Metrics (yhat_pda, Y) [2]
>F_netlda = Metrics (yhat_netlda, Y) [2]
>F_netqda = Metrics (yhat_netqda, Y) [2]
>ARI_lda = Metrics (yhat_lda, Y) [3]
>ARI_sda = Metrics (yhat_sda, Y) [3]
>ARI_pda = Metrics (yhat_pda, Y) [3]
>ARI_netlda = Metrics (yhat_netlda, Y) [3]
>ARI_netqda = Metrics (yhat_netqda, Y) [3]
```

We repeat above simulations 500 times and summarize numerical results in Table 1. We observe that the package NetDA provides higher values of PRE, REC, F-score, and adn ARI, showing that the classification obtained by the NetDA method is more accurate than that determined by other methods. Specifically, compare with MASS and NetLDA, we can see that the latter outperforms the former method, which is due to the incorporation of network structure with irrelevant pairs of predictors removed from $\boldsymbol{\Theta}$. On the other hand, for the comparison between NetLDA and NetQDA, we can see that the NetQDA is much better than the NetLDA method, because the NetQDA method successfully detects network structures from each class, and those detected network structures are valid to do classification. Those numerical findings verify the discussion in Subsection "Benchmark of NetDA."

4.2. Real Data Analysis. In this study, we take the wine dataset as an example, which is introduced in Section 3, to demonstrate the package NetDA. To demonstrate the functions and perform classification and prediction, we first split the full data into the training data and the validation data. In our example, we take the first 45 samples in each class to obtain the training data and use the remaining samples in each class to form the validation data.

```
>data (WineData)
>Y = WineData [, 1] #the response
>X = WineData [, 2 : 14] #the predictors
>D1 = WineData [which (Y = = 1), ]
>D2 = WineData [which (Y = = 2), ]
>D3 = WineData [which (Y = = 3), ]
>#An example of user-specific training data and validation data
>#"Train" represents the training data and "Test" represents validation data in our example.
>Train = rbind (D1 [1 : 45, ], D2 [1 : 45, ], D3 [1 : 45, ])
>Test = rbind (D1 [46 : dim (D1) [1], ], D2 [46 : dim (D2) [1], ], D3 [46 : dim (D3) [1], ])
>#The response (Y) and predictors (X) in the training data
>X = Train [, 2 : 14] > Y = Train [, 1]
>#The response (Y_test) and predictors (X_test) in the validation data
>X_test = Test [, 2 : 14] > Y_test = Test [, 1]
```

When the training data and the validation data are determined, we employ the function NetDA to perform classification. We insert "*X*," "*Y*," and "*X*_test" to the function NetDA, and we denote "NetLDA" and "NetQDA" as the argument method = 1 and method = 2, respectively. The resulting vectors of predicted classes and estimated precision matrices are given by "$yhat" and "$Network," respectively.

```
>NetDA (X, Y, method = 1, X_test) -> NetLDA
>yhat_lda = NetLDA$yhat
>Net_lda = NetLDA$Network
>yhat_lda
[1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2
[39] 2 2 2 2 3 3 3 3
>round (Net_lda, 3)
```

	[, 1]	[, 2]	[, 3]	[, 4]	[, 5]	[, 6]	[, 7]	[, 8]	[, 9]	[, 10]	[, 11]	[, 12]	[, 13]
[1,]	3.739	−0.415	−0.665	0.107	0.004	0.000	−0.444	0.000	0.000	−0.508	0.000	−0.291	−0.004
[2,]	−0.415	1.247	−0.511	−0.035	0.000	0.109	0.324	−0.225	0.065	0.000	1.413	0.017	0.000
[3,]	−0.665	−0.511	19.463	−0.936	−0.059	−0.055	−0.619	0.000	0.000	−0.087	0.000	0.000	−0.005
[4,]	0.107	−0.035	−0.936	0.167	0.001	−0.048	0.152	−0.160	−0.038	−0.023	−0.035	−0.013	0.000
[5,]	0.004	0.000	−0.059	0.001	0.006	0.000	0.007	0.087	−0.036	0.003	0.007	0.013	0.000
[6,]	0.000	0.109	−0.055	−0.048	0.000	8.774	−3.847	0.000	−0.364	−0.249	0.000	−1.274	−0.001
[7,]	−0.444	0.324	−0.619	0.152	0.007	−3.847	6.127	2.104	−1.800	0.016	−2.845	−1.877	−0.002
[8,]	0.000	−0.225	0.000	−0.160	0.087	0.000	2.104	46.879	0.000	−0.084	0.000	0.762	−0.002
[9,]	0.000	0.065	0.000	−0.038	−0.036	−0.365	−1.800	0.000	5.582	−0.216	0.000	−0.391	0.001
[10,]	−0.508	0.000	−0.087	−0.023	0.003	−0.249	0.016	−0.084	−0.216	0.467	1.663	0.606	0.000
[11,]	0.000	1.413	0.000	−0.035	0.007	0.000	−2.845	0.000	0.000	1.662	32.850	0.000	−0.002
[12,]	−0.291	0.017	0.000	−0.013	0.013	−1.274	−1.877	0.762	−0.391	0.606	0.000	5.667	−0.001
[13,]	−0.004	0.000	−0.005	0.000	0.000	−0.001	−0.002	−0.002	0.001	0.000	−0.002	−0.001	0.000

(26)

```
#############
>NetDA (X, Y, method = 2, X_test) -> NetQDA
>yhat_qda = NetQDA$yhat
>Net_qda = NetQDA$Network
>yhat_qda
[1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
[42] 2 3 3 3 3
>round (Net_qda [[1]], 3)
```

	[, 1]	[, 2]	[, 3]	[, 4]	[, 5]	[, 6]	[, 7]	[, 8]	[, 9]	[, 10]	[, 11]	[, 12]	[, 13]
[1,]	7.176	0.011	0.000	0.470	−0.073	0.000	−0.404	0.000	−1.754	−0.175	0.000	−0.159	−0.005
[2,]	0.012	2.721	0.000	0.005	−0.007	0.000	0.000	0.051	−0.171	0.000	2.039	0.000	0.003
[3,]	0.000	0.000	23.194	−0.959	−0.161	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
[4,]	0.470	0.005	−0.959	0.236	−0.011	0.138	0.076	−0.397	−0.101	0.046	−0.244	0.170	0.000
[5,]	−0.073	−0.007	−0.161	−0.011	0.013	−0.097	0.008	−0.085	0.049	−0.042	−0.029	−0.023	0.000
[6,]	0.000	0.000	0.000	0.138	−0.097	22.439	−7.472	0.000	−0.680	−2.326	0.000	−0.330	0.002
[7,]	−0.405	0.000	0.000	0.076	0.008	−7.471	16.520	0.000	−3.594	−2.131	0.000	0.000	0.001
[8,]	0.000	0.051	0.000	−0.397	−0.085	0.000	0.000	69.356	0.000	0.056	0.000	0.159	0.000
[9,]	−1.754	−0.171	0.000	−0.101	0.049	−0.680	−3.594	0.000	8.449	−0.246	0.000	0.000	0.003
[10,]	−0.175	0.000	0.000	0.046	−0.042	−2.326	−2.131	0.056	−0.246	2.509	0.566	0.000	−0.006
[11,]	0.000	2.039	0.000	−0.244	−0.029	0.000	0.000	0.000	0.000	0.563	48.139	0.000	−0.008
[12,]	−0.158	0.000	0.000	0.170	−0.023	0.328	0.000	0.159	0.000	0.000	0.000	7.213	0.004
[13,]	−0.005	0.003	0.000	0.000	0.000	0.002	0.001	0.000	0.003	−0.006	−0.008	0.004	0.000

(27)

```
>round (Net_qda [[2]], 3)
```

	[, 1]	[, 2]	[, 3]	[, 4]	[, 5]	[, 6]	[, 7]	[, 8]	[, 9]	[, 10]	[, 11]	[, 12]	[, 13]
[1,]	4.060	0.019	0.710	−0.128	−0.005	0.000	−0.527	0.000	0.432	−0.783	0.000	0.169	−0.001
[2,]	0.021	2.357	0.000	−0.173	−0.017	0.018	−0.015	−1.439	−0.359	0.138	0.912	0.122	0.004
[3,]	0.708	0.000	18.423	−1.213	−0.044	0.000	−0.006	0.000	0.000	0.000	0.000	0.826	0.004
[4,]	−0.128	−0.173	−1.213	0.208	0.007	0.082	−0.043	−0.214	0.059	0.056	−0.160	−0.393	−0.001
[5,]	−0.005	−0.017	−0.044	0.007	0.005	−0.033	0.063	0.090	−0.066	0.000	−0.049	0.014	0.000
[6,]	0.000	0.017	0.000	0.082	−0.033	7.713	−5.999	0.866	1.732	0.443	0.000	−0.822	−0.002
[7,]	−0.527	−0.015	−0.006	−0.043	0.063	−5.999	10.017	0.181	−3.932	−1.565	−1.637	−1.902	0.002
[8,]	0.000	1.440	0.000	−0.214	0.090	0.869	0.173	46.367	0.000	0.000	0.000	5.077	−0.003
[9,]	0.432	−0.359	0.000	0.059	−0.066	1.733	−3.932	0.000	5.282	0.506	0.000	−0.648	−0.001
[10,]	−0.783	0.137	0.000	0.056	0.000	0.443	−1.565	0.000	0.506	1.733	0.542	0.763	−0.001
[11,]	0.000	0.912	0.000	−0.160	−0.049	0.000	−1.634	0.000	0.000	0.542	29.109	0.000	0.008
[12,]	0.170	0.121	0.826	−0.393	0.014	−0.822	−1.902	5.077	−0.648	0.763	0.000	7.293	0.002
[13,]	−0.001	0.004	0.004	−0.001	0.000	−0.002	0.002	−0.003	−0.001	−0.001	0.008	0.002	0.000

(28)

```
>round (Net_qda [[3]], 3)
```

	[, 1]	[, 2]	[, 3]	[, 4]	[, 5]	[, 6]	[, 7]	[, 8]	[, 9]	[, 10]	[, 11]	[, 12]	[, 13]
[1,]	4.261	−0.254	0.000	−0.072	0.020	0.000	0.000	0.000	−1.200	−0.202	−0.098	−0.832	0.003
[2,]	−0.254	0.951	0.000	−0.096	0.017	0.350	0.336	0.000	0.302	0.050	0.000	0.146	0.000
[3,]	0.000	0.000	39.923	−2.076	−0.119	−1.713	0.000	0.000	0.000	0.000	0.000	−1.640	0.006
[4,]	−0.072	−0.096	−2.076	0.359	−0.010	−0.326	−0.073	−0.193	−0.115	−0.010	−0.178	−0.055	0.000
[5,]	0.020	0.017	−0.119	−0.010	0.022	0.060	−0.431	0.371	−0.011	0.021	−0.034	0.040	−0.001
[6,]	0.000	0.350	−1.713	−0.326	0.060	10.790	−0.498	−0.740	−4.461	0.055	0.000	−1.173	0.000
[7,]	0.000	0.336	0.000	−0.073	−0.431	−0.498	24.098	0.000	−1.503	−0.930	0.000	2.400	0.018
[8,]	0.000	0.000	0.000	−0.193	0.371	−0.740	0.000	50.622	0.000	0.000	0.000	0.000	−0.014
[9,]	−1.200	0.302	0.000	−0.115	−0.011	−4.461	−1.503	0.000	12.612	−1.032	0.000	0.846	−0.008
[10,]	−0.202	0.050	0.000	−0.010	0.021	0.055	−0.930	0.000	−1.032	0.410	1.353	0.070	−0.001
[11,]	−0.098	0.000	0.000	−0.178	−0.034	0.000	0.000	0.000	0.000	1.353	50.699	0.000	−0.005
[12,]	−0.832	0.146	−1.640	−0.055	0.040	−1.173	2.400	0.000	0.846	0.070	0.000	13.312	−0.008
[13,]	0.003	0.000	0.006	0.000	−0.001	0.000	0.018	−0.014	−0.008	−0.001	−0.005	−0.008	0.000

(29)

Moreover, for the visualization of estimated network structures, we further apply the packages network, GGally and sna, to draw the network structure based on the estimated precision matrices. The following commands are implemented to draw network structures, and the corresponding figures determined by NetDA with the arguments method = 1 and method = 2 are displayed in Figures 2 and 3, respectively.

```
>library (network)
>library (GGally)
>library (sna)
>material_name = c ("Alcohol," "Malic acid," "Ash," "Alkalinity," "Magnesium," "phenols," + "Flavanoids," "Nonflavanoid," "Proanthocyanins," "Color," "Hue," "OD280," "Proline")
>ggnet2 (Net_lda, mode = "circle," size = 8, label = material_name, label.size = 5)
>ggnet2 (Net_qda [[1]], mode = "circle," size = 8, label = material_name, label.size = 5)
>ggnet2 (Net_qda [[2]], mode = "circle," size = 8, label = material_name, label.size = 5)
>ggnet2 (Net_qda [[3]], mode = "circle," size = 8, label = material_name, label.size = 5)
```

From Figures 2 and 3, we can observe that precision matrices provide complex network structures in predictors. In particular, in Figure 3, we can see that the estimated class-dependent network structures are different from each other, and the network structure in class 2 looks more complex than others. To assess the performance of prediction, we input predicted values (yhat_lda or yhat_qda) and responses in the validation data (Y_test) to the function Metrics, and the resulting values are displayed below.

```
>Metrics (yhat_lda, Y_test)
$"Confusion matrix"
     [, 1] [, 2] [, 3]
[1, ]   15    0    0
[2, ]    0   26    0
[3, ]    0    1    4
$"(PRE, REC, F-score)"
[1] 0.9782609 0.9782609 0.9782609
$ARI
[1] 0.9410827
#############
>Metrics (yhat_qda, Y_test)
$"Confusion matrix"
     [, 1] [, 2] [, 3]
[1, ]   15    0    0
[2, ]    0   27    0
[3, ]    0    0    4
$"(PRE, REC, F-score)"
[1] 1 1 1
$ARI
[1] 1
```

Finally, we further adopt the function lda in the packages MASS, sparseLDA, and penalizedLDA to perform the conventional discriminant methods and compare them with our NetDA. Detailed implementations and numerical results are given below:

```
>Wine = data.frame (cbind (Y, X))
>##Demonstration of MASS
>lda = lda (Y., Wine, prior = c (length (which (Y = = 1)), length (which (Y = = 2)), +length (which (Y = = 3)))/ length (Y))
>predict (lda, X_test) $class -> lda_pred
```

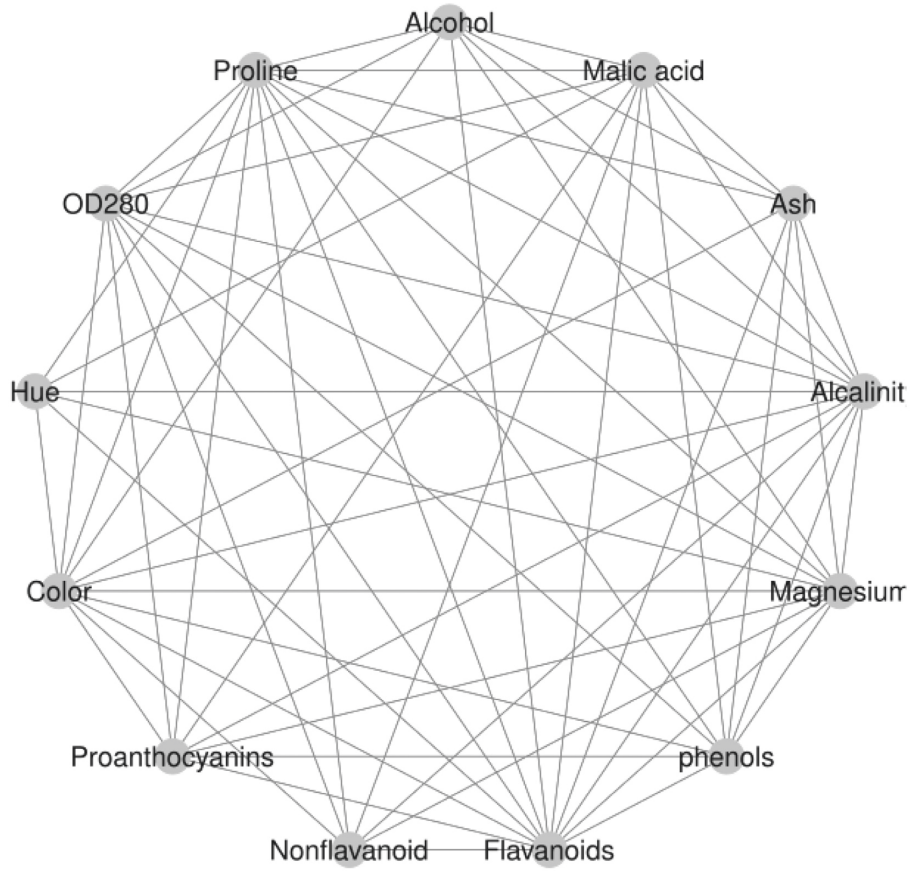

Figure 2: Pooled subject-based network structures obtained by NetDA under the argument method = 1.

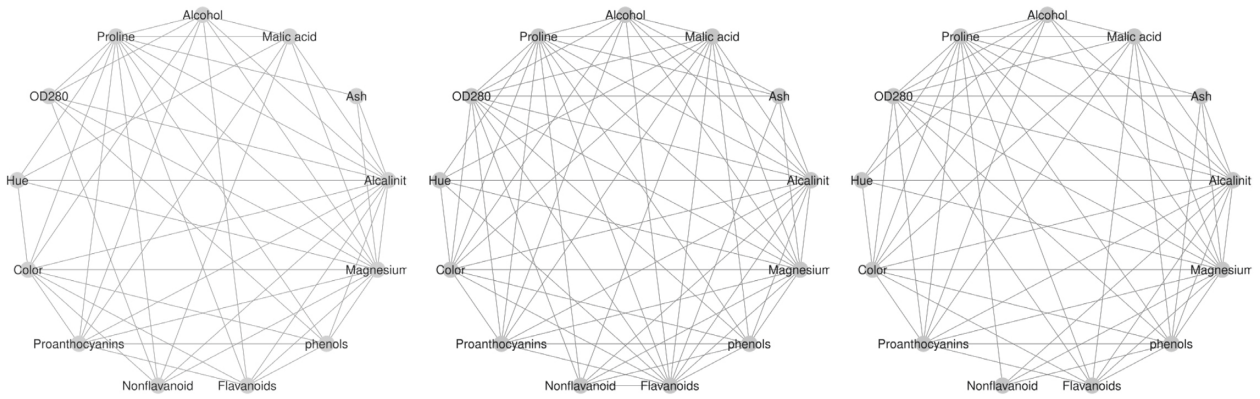

Figure 3: Class-dependent network structures obtained by NetDA under the argument method = 2. Left, middle, and right panels are based on predictors in classes 1, 2, and 3, respectively.

```
>
>Metrics (lda_pred, Y_test)
$"Confusion matrix"
[, 1] [, 2] [, 3]
[1, ] 15 1 0
[2, ] 0 26 0
[3, ] 0 0 4
$"(PRE, REC, F-score)"
[1] 0.9782609 0.9782609 0.9782609
$ARI
[1] 0.9196658
>#Demonstration of sparseLDA
>n = length (Y)
>I = max (Y)
>y = matrix (0, n, I)
>y [1 : 45, 1] = 1
```

```
>y [46 : 90, 2] = 1
>y [91 : 135, 3] = 1
>colnames (y) <- c ("1," "2," "3")
>sda = sda (data.frame (X), y, lambda = 1e − 6, stop = −1, maxIte = 25, +trace = TRUE)
ite: 1 ridge cost: 101.1419 |b|_1: 0.001636884
ite: 2 ridge cost: 47.61608 |b|_1: 0.002647311
ite: 3 ridge cost: 47.61608 |b|_1: 0.002647311
ite: 1 ridge cost: 129.1129 |b|_1: 0.01364973
ite: 2 ridge cost: 129.1129 |b|_1: 0.01364973
final update, total ridge cost: 176.729 |b|_1: 0.01629704
>sda_pred = as.numeric (unlist (predict (sda, data.-frame (X_test)) $class))
>Metrics (sda_pred, Y_test)
$"Confusion matrix"
[, 1] [, 2] [, 3]
[1, ] 14 26 2
[2, ] 0 0 0
[3, ] 0 0 1
$"(PRE, REC, F-score)"
[1] 0.3488372 0.3488372 0.3488372
$ARI
[1] 0.07272024
>#Demonstration of penalizedLDA
>pda = PenalizedLDA (X, Y, lambda = 0.14, K = 2)
>pda_pred = as.numeric (unlist (predict (pda, data.-frame (X_test)))) [1 : 46]
>Metrics (pda_pred, Y_test)
$"Confusion matrix"
[, 1] [, 2] [, 3]
[1, ] 14 4 0
[2, ] 0 21 0
[3, ] 0 1 3
$"(PRE, REC, F-score)"
[1] 0.8837209 0.8837209 0.8837209
$ARI
[1] 0.6229476
```

In general, we can see that NetLDA and NetQDA have the satisfactory performance in prediction. For the NetLDA method, there is one misclassification as shown in the confusion matrix, while the predicted classes determined by NetQDA are all equal to the responses in the validation data. From the comparison to NetDA, we observe from a confusion matrix determined by the conventional linear discriminant analysis (lda) is comparable to that obtained by the NetLDA method, but it is interesting to see that the value of ARI determined by NetLDA is slightly larger than that based on lda. In addition, it is clear to see that the NetQDA method is better than lda. On the contrary, it is surprising to see that two penalized methods sparseLDA and penalizedLDA do not have satisfactory performance of classification and prediction, especially that sparseLDA has the most unexpected result. In summary, the numerical results in this data analysis show (a) the importance of incorporating predictor network structures in the classification procedure, and (b) the advantage of adopting class-dependent network structures.

5. Summary

Classification and prediction have been important topics in supervised learning, and discriminant analysis is a useful method in statistical learning. While many methods have been developed, little method has been available to handle potential network structures in predictors when building predictive models. In addition, rare relevant software has been developed for statistical analysts whose interest is to incorporate network structures and obtain precise classification.

To address this concern, we develop an R package NetDA for public use. Our package provides two functions. The function NetDA aims to incorporate the information of network structures in predictors to do linear or quadratic discriminant functions. The other function Metrics summarizes some useful and informative criteria to assess the performance of classification and prediction. A detailed documentation and concrete examples illustrate the validity of the methods in this package. Finally, some further developments can be explored based on the current package, including the alternative approaches of detection of network structures (e.g., Hastie et al. [12]; Section 9.4), nonparametric discriminant analysis with network structure accommodated (e.g., Chen [17]), and analysis of noisy data, such as measurement error models (e.g., Chen and Yi [18]; Chen and Yi [19]).

Data Availability

Data used to support this study are available at https://archive.ics.uci.edu/ml/datasets/wine.

Conflicts of Interest

The author declares that there are no conflicts of interest.

Acknowledgments

The author also specifically thanks Ms. Lingyu Cai for kind assistance in preparing a vignette. This research was supported by the Ministry of Science and Technology with grant ID 110-2118-M-004-006-MY2.

References

[1] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer, New York, NY, USA, 2008.

[2] Y. Guo, T. Hastie, and R. Tibshirani, "Regularized linear discriminant analysis and its application in microarrays," *Biostatistics*, vol. 8, no. 1, pp. 86–100, 2007.
[3] S. E. Safo and J. Ahn, "General sparse multi-class linear discriminant analysis," *Computational Statistics & Data Analysis*, vol. 99, pp. 81–90, 2016.
[4] L. Clemmensen, T. Hastie, D. Witten, and B. Ersbøll, "Sparse discriminant analysis," *Technometrics*, vol. 53, no. 4, pp. 406–413, 2011.
[5] D. M. Witten and R. Tibshirani, "Penalized classification using Fisher's linear discriminant," *Journal of the Royal Statistical Society: Series B*, vol. 73, no. 5, pp. 753–772, 2011.
[6] L.-P. Chen, "Multiclassification to gene expression data with some complex features," *Biostatistics and Biometrics Open Access Journal*, vol. 99, Article ID 555751, 2018.
[7] L.-P. Chen, G. Y. Yi, Q. Zhang, and W. He, "Multiclass analysis and prediction with network structured covariates," *Journal of Statistical Distributions and Applications*, vol. 6, no. 1, 2019.
[8] W. He, G. Y. Yi, and L.-P. Chen, "Support vector machine with graphical network structures in features," in *Proceedings of the 15th International Conference on Machine Learning and Data Mining, MLDM 2019*, vol. 2, pp. 557–570, 2019, https://easychair.org/publications/preprint/g6d1.
[9] W. Cai, G. Guan, R. Pan, X. Zhu, and H. Wang, "Network linear discriminant analysis," *Computational Statistics & Data Analysis*, vol. 117, pp. 32–44, 2018.
[10] J. Liu, G. Yu, and Y. Liu, "Graph-based sparse linear discriminant analysis for high-dimensional classification," *Journal of Multivariate Analysis*, vol. 171, pp. 250–269, 2019.
[11] L.-P. Chen, "Network-based discriminant analysis for multiclassification," *Journal of Classification*, 2022.
[12] T. Hastie, R. Tibshirani, and M. Wainwright, *Statistical Learning with Sparsity: The Lasso and Generalizations*, CRC Press, New York, NY, USA, 2015.
[13] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, 2008.
[14] M. Yuan and Y. Lin, "Model selection and estimation in the Gaussian graphical model," *Biometrika*, vol. 94, no. 1, pp. 19–35, 2007.
[15] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning: With Applications in R*, Springer, New York, NY, USA, 2017.
[16] L. Hubert and P. Arabie, "Comparing partitions," *Journal of Classification*, vol. 2, no. 1, pp. 193–218, 1985.
[17] L.-P. Chen, "Nonparametric discriminant analysis with network structures in predictor," *Journal of Statistical Computation and Simulation*, 2022.
[18] L.-P. Chen and G. Y. Yi, "Analysis of noisy survival data with graphical proportional hazards measurement error models," *Biometrics*, vol. 77, no. 3, pp. 956–969, 2021.
[19] L.-P. Chen and G. Y. Yi, "De-noising analysis of noisy data with graphical models," *Electronic Journal of Statistics*, vol. 16, pp. 3861–3909, 2022.