Discriminative Principal Component Analysis for High Dimensional Classification with Applications in NIR Spectroscopy

Xiaoke Liu

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

of

University College London.

Department of Statistical Science

University College London

August 13, 2020

I, Xiaoke Liu, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

Abstract

Principal component analysis (PCA) has been widely applied in various fields such as bioscience, chemistry, computer science and social science as a signal processing, dimension reduction or feature extraction tool. Regardless of its popularity, when PCA is used as a preliminary dimension reduction step in developing classification rules with high-dimensional data it has a drawback that as an unsupervised method PCA fails to use the class labels when constructing the components. As a result, its maximization of the variance of the projected patterns is not necessarily in favour of discrimination among classes.

To address this problem, in this thesis we propose five methods from three perspectives: 1) We propose two methods, reweighted PCA and between PCA, which combine supervised information in the feature generation step of PCA, so that more discriminating features are constructed within the classic PCA framework. 2) We propose two feature filtering methods, reordered PCA and stepwise-reordered PCA. In these methods, principal components are generated with the classic PCA framework, but re-ranked and selected according to their discriminating power with quadratic discriminant analysis (QDA). 3) We propose a penalised QDA based supervised feature extraction method to replace PCA, which can use the label information to generate more discriminating features.

We use two near infrared (NIR) spectroscopic data sets, a wheat data set and a paddy rice data set to evaluate our methods in both binary and multi-class classification. We compare our methods with the classic principal component discrim-

inant analysis (PCDA) and partial least squares discriminant analysis (PLS-DA). Enhancements in classification accuracy are witnessed for all our modified methods in all examples compared with the classic PCDA. Four simulations have been constructed to help understanding the mechanism and evaluating the performance of our penalised QDA based feature extraction method in Chapter 3.

Impact Statement

Principal component analysis is an indispensable signal processing, dimension reduction and feature extraction tool which has been extensively applied in various fields such as bioscience, chemistry, computer science and social science. Regardless of its popularity, when PCA is used as a preliminary dimension reduction step in high dimensional classification it has a drawback that as an unsupervised method PCA fails to use the class labels when constructing the components. As a result, its maximization of the variance of the projected patterns is not necessarily in favour of discrimination among classes.

In this thesis, we propose five variants of PCA to resolve the abovementioned problem and improve the performance of PCA in binary and multi-class classification. We apply our methods in two NIR spectroscopic data sets to classify wheat and paddy rice samples. The results show that our methods can considerably improve the classification accuracy in both binary and multi-class classification.

The impact of our work is owing to the extensive applications of PCA, and the effectiveness of NIR spectroscopic technique.

PCA is a fundamental tool in analysing high-dimensional data such as gene expression data, facial image data and medical image data. Hence our methods have potential in improving the classification in other impactful fields, such as face recognition, tumor diagnosis, etc.

Besides, over the past decades NIR has increasingly been adopted as an efficient analytical tool in various fields, such as petrochemical, pharmaceutical, envi-

ronmental, agricultural, food and biomedical sectors. For example in food science, NIR spectroscopy is widely used in the determination of the origin of food, quality of food, and in adulteration detection. Our methods have been shown to perform well in NIR spectral classification. Accordingly, our work will have huge potential in attracting funding and collaboration from industries such as food, agriculture and pharmacy for real-world applications, and also boost innovation in the development of NIR spectral analysis technique in these fields.

Acknowledgements

First and foremost, I would like to express my deepest and sincere gratitude to my supervisor, Professor Thomas Fearn, for his highly professional guidance throughout my PhD and his patience to read my reports and drafts. I have learnt a tremendous amount from him, not only from his immense knowledge and expertise in statistical machine learning and chemometrics, but also from his shining character, responsible attitude to research and the pleasant way he treats others. He is the most wise, gentle, responsible and supportive person I have ever met. Besides, I am very grateful to my second supervisor, Dr. Jinghao Xue, for his kind suggestions on my upgrade report and my PhD research. I would like to thank my PhD viva examiners Dr. Ioanna Manolopoulou and Dr. Claire Gormley, for the constructive suggestions they gave to me and the inspiring discussion we had on my viva.

A special thank goes to family, especially my dearest parents, Luo Yang and Liu Dequn, for all their love, understanding, support and encouragement throughout my life. Without them, I would not have been able to pursue my dream.

I would like to thank my boyfriend, for all his support and encouragement during the last two years, professional, mental and emotional. I would like to thank my best friend, Liyang Hao for the joy and laughter she brought to me and all the great time we spent together since we met in 10 years ago. Last, I would like to thank all my friends especially Rui Zhu, Wen Zhang and Xiaoou Lu, who accompanied me and encouraged me to strive towards my goal. I would not have been able to complete this thesis without all the help, kindness and love around me.

Contents

1 Introductory Material **15**

1.1 Near Infrared Spectral Data and its Classification 15

1.1.1 Introduction to Near Infrared Spectroscopy 15

1.1.2 NIR Spectral Classification 19

1.2 Discriminant Analysis 21

1.2.1 Fisher's Linear Discriminant 21

1.2.2 Relationship between Fisher Linear Discriminant and Linear Regression in Binary Classification 23

1.2.3 Linear Discriminant Analysis and Quadratic Discriminant Analysis 27

1.2.4 LDA as a Feature Extraction Technique 31

1.2.5 Comparison of LDA and QDA in Classification 34

1.3 Principal Component Analysis 35

1.3.1 Algorithm 36

1.3.2 Further Developments 37

1.3.3 Discriminative PCA 40

1.3.4 Scaling in PCA 42

1.4 Partial least squares discriminant analysis 43

1.4.1 Introduction to Partial Least Squares Regression and Partial Least Squares Discriminant Analysis 43

1.4.2 The optimisation idea behind the NIPALS algorithm 47
1.4.3 PLS as a Discriminative Dimension Reduction Technique . 48
1.4.4 Nonlinear PLS 52

2 Reweighted PCA and Reordered PCA 54

2.1 Introduction 54
2.2 Methodologies 59
2.2.1 Decomposition of the total covariance matrix 59
2.2.2 Reweighting algorithms 60
2.2.3 Reordering algorithms 67
2.3 Examples 69
2.3.1 Binary classification with wheat data set 69
2.3.2 Binary classification with paddy rice data set 80
2.3.3 Three-class classification with wheat data set 87
2.3.4 Three-class classification with the paddy rice data set 88
2.4 Conclusion 90

3 A Penalised QDA-based Feature Extraction Method 93

3.1 Introduction 93
3.2 Methodologies 98
3.2.1 Feature Extraction Criterion 98
3.2.2 Algorithms 105
3.3 Simulations 112
3.3.1 Scenario 1 113
3.3.2 Scenario 2 117
3.3.3 Scenario 3 120
3.3.4 Scenario 4 123
3.4 Examples 125
3.4.1 Conclusion 129

4 General Conclusions **132**

Appendices **138**

A Decomposition of total covariance **138**

B Comparison of classification error rates of the classic PCA-QDA and our reweighted PCA-QDA (rPCA-QDA) on the wheat data set **141**

Bibliography **142**

List of Figures

1.1.1 NIR spectra plot of avian and fish particles 16

2.1.1 An illustrative example where PCA fails to extract discriminative features . 55

2.1.2 An illustrative example showing the discriminatory power of within-group variation 57

2.3.1 Spectra of wheat samples from class 1 (variety 3) and class 2 (variety 9) . 71

2.3.2 Second derivative of spectra of wheat samples 71

2.3.3 LOOCV classification error rate of the wheat data with classic PCA-QDA, Reweighted PCA-QDA, Between PCA-QDA, Reordered PCA-QDA, Stepwise-reordered PCA-QDA and PLS-QDA. 72

2.3.4 A comparison of the discrimination power of the first six PCs from the two groups. 75

2.3.5 Discrimination power of classic PCs 77

2.3.6 Double CV error rates of the six methods in the binary wheat example. 80

2.3.7 Spectra of paddy rice samples from class 1 and class 2 82

2.3.8 The second derivative of the spectra of paddy rice samples 82

2.3.9 CV error rates of the above methods in binary paddy rice example. . 83

2.3.10 Discrimination power of classic PCs in binary paddy rice example . 84

2.3.11 Classification performance of the six methods in double CV in the binary paddy rice example, with subfigure (a): error rates in the test sets, subfigure (b): the number of components needed in each method to accomplish the corresponding classification error rate. . . 86

2.3.12 Classification error rates of the abovementioned six methods in the three-class example via LOOCV and double CV. 87

2.3.13 Classification error rates of the above six methods in the paddy rice three-class example via LOOCV and double CV. 89

3.3.1 An illustrative example of the variance heterogeneity. In the $\boldsymbol{x}_1$ direction samples from class 1 follow $\boldsymbol{N}(0.5, 0.8^2)$ while samples from class 2 follow $\boldsymbol{N}(-0.5, 5^2)$. 113

3.3.2 Illustrative scatter plot of the two classes in the direction of $\boldsymbol{x}_1$ or $\boldsymbol{x}_2$. Class 1 and class 2 are set to have same mean in direction $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ but class 1 is set to have higher variance than class 2. This heterogeneity in variance can be used to discriminate the two classes. 115

3.3.3 Illustrative scatter plot of the two classes in direction $\boldsymbol{x}_3$. Class 1 and class 2 are set to have same mean in direction $\boldsymbol{x}_3$ but class 2 is set to have greater variance than class 1. This heterogeneity in variance can be used to discriminate the two classes. 115

3.3.4 Illustrative scatter plot of the two classes in the other directions, $\boldsymbol{x}_4$, $\boldsymbol{x}_5$, ..., $\boldsymbol{x}_{10}$. The two classes are set to follow identical distribution on the remaining 7 directions. They are the noise directions. 115

3.3.5 Classification performance of PCA-QDA, PCA-LDA, PLS-QDA, PLS-DA and our method in the first scenario. The triangular symbol in each box represents the average error rate over 10 simulations. . . 116

3.3.6 Classification performance of PCA-QDA, PCA-LDA, PLS-QDA, PLS-DA and our QDA-based method under the second scenario . . 119

3.3.7 Classification performance of PCA-QDA, PCA-LDA, PLS-QDA, PLS-DA and our QDA-based method under the third scenario 122

3.3.8 Classification performance of PC-DA (PCA-QDA, PCA-LDA), PLS-QDA, PLS-DA and our QDA-based method under the fourth scenario 124

3.4.1 Spectra plot of class 1 (variety 3) and class 2 (variety 8) 126

3.4.2 Classification performance of PCA-QDA, PCA-LDA, PLS-QDA, PLS-DA and our QDA-based method in the training and test set. The triangular symbol in each box represents the average error rate over 10 simulations. 128

List of Tables

2.3.1 Composition of the wheat data set 69

2.3.2 Composition of the two target classes 70

2.3.3 Cosines of angles between the first four PCs obtained from Group 1, Group 2, classic PCA and reweighted PCA 74

2.3.4 Cosines of angles between the first two PC directions and dominating directions of the two groups, and cosines between the first two PCs and the mean difference direction. 78

2.3.5 Composition of the paddy rice data 81

2.3.6 Cosines of angles between the classic PCs and the main directions of the two groups 84

B1 Comparison of classification error rates of the classic PCA-QDA and our reweighted PCA-QDA (denoted as rPCA-QDA on the table) on 6 varieties of wheat 141

Chapter 1

Introductory Material

1.1 Near Infrared Spectral Data and its Classification

1.1.1 Introduction to Near Infrared Spectroscopy

Over the past three decades, Near-infrared (NIR) spectroscopy has increasingly been adopted as an efficient analytical tool in various fields, such as the petrochemical (Murugesan et al., 2009; Meher et al., 2006), pharmaceutical (Gendrin et al., 2008; Roggo et al., 2007), environmental (Nyström and Dahlquist, 2004; Shepherd and Walsh, 2007), clinical (Erickson and Godavarty, 2009; Caplan et al., 2006; Sakudo et al., 2006), agricultural (Shepherd and Walsh, 2007; Moreda et al., 2009; Williams et al., 1987), food (Karoui and De Baerdemaeker, 2007; Prieto et al., 2009a) and biomedical sectors (Landau et al., 2006). The most salient advantage of NIR spectroscopy over other analytical techniques is its ability to record spectra for solid and liquid samples without complex pretreatment of the samples. This characteristic makes it especially attractive for straightforward, speedy characterization of natural and synthetic products (Xiaobo et al., 2010).

The near infrared region of the electromagnetic spectrum is from 780 nm to 2500 nm. The transmittance or reflectance in this region is measured and recorded

as the NIR spectral data. This region comprises broad bands associated with the combinations of vibration modes (O-H, N-H, and C-H) and overtones of molecular vibrations. As a result NIR spectroscopy can be used in predicting the content of chemical components, detecting adulteration or distinguishing different species of samples (Prieto et al., 2009a; Wang et al., 2017). Besides, by constantly analysing the spectra of samples from different batches quality changes can be monitored and thus NIR spectroscopy can be used in fault diagnosis and quality control in chemical process (Cho et al., 2005; Lee et al., 2004; Wang et al., 2017).

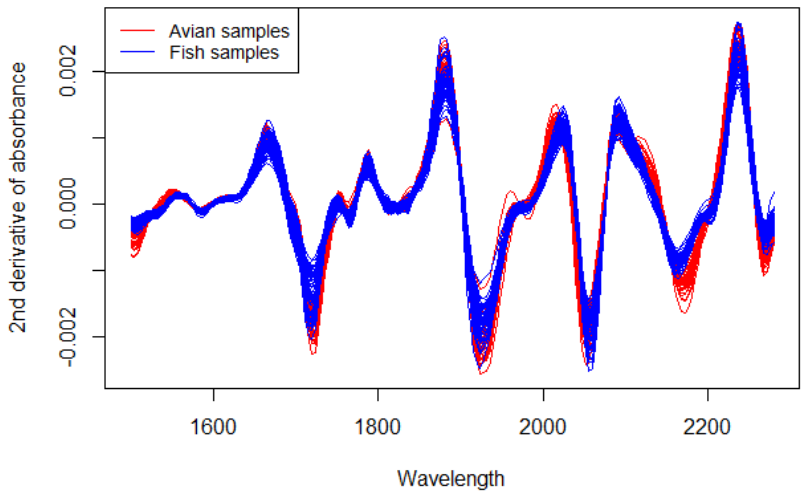


Figure 1.1.1. NIR spectra plot of avian and fish particles

Figure 1.1.1 shows an example of NIR spectroscopic data. These are spectra of avian and fish particles collected by NIR microscopy (Pérez-Marín et al., 2009). They have been collected as the training data for further classification on the mixtures of animal protein by-products particles. Here wavelengths from 1500nm to 2400nm have been selected and the absorbances at these wavelengths have been recorded. Derivatives of the absorbance are often taken to remove the additive baseline shift of spectra. In this example, the second derivative of the absorbance is used. In this figure, each curve depicts the spectrum of a particle. The blue ones correspond to spectra of the avian samples while the red ones correspond to that of the

fish samples. Calibration models can be built based on the training spectra of the two classes and then used in predicting the species of test samples. This is a simple example of how NIR spectroscopy can contribute to distinguishing sample species.

Despite the widespread application of NIR spectroscopy, it also has some limitations. One crucial problem is due to the high dimension of the spectral data. The NIR region includes wavelengths from 780 nm to 2500 nm and the absorbances at these wavelengths are recorded as predictor variables. The measurement step length of absorbance depends on the spectral resolution and it is typically set to be 2nm. Thus the number of variables can be hundreds or even more than a thousand. However, the reference data needed for calibration is often relatively expensive to obtain (Blanco and Villarroya, 2002) and the typical sample size of a NIR data set is a couple of hundred or even less (Pérez-Marín et al., 2009). In other words the number of variables p is usually enormous compared with the number of observations n in the NIR spectral analysis. Due to this reason, classic regression methods such as multiple linear regression (MLR) and classification methods such as linear discriminant analysis (LDA) fail to work. Besides, absorbances at adjacent wavelengths can contain very similar information and thus there exists strong collinearity between NIR absorbances. Apart from these problems, absorbances in some ranges may be noninformative or even harmful to further analysis (Xiaobo et al., 2010). Due to the above reasons, suitable dimension reduction or variable selection techniques are commonly applied to construct or select informative features from a large number of highly correlated variables in the NIR spectral analysis.

The most widely used dimension reduction techniques in chemometrics include principal component analysis (PCA) and partial least squares (PLS) regression. Both methods construct latent variables comprised of combinations of the original features and accordingly project data from the original high-dimensional feature space to a low-dimensional subspace. PCA aims at extracting a small number of latent variables to maintain the highest variability among the predictor

variables, while PLS generates latent variables which have the largest covariance with the predicted variables. With the use of PCA and PLS, data are projected to a low-dimensional subspace and further analysis can be carried out on this low-dimensional subspace, such as quantitative prediction, classification and so on. By doing this, most information is maintained with less features while noise is reduced. It has been shown in many literatures that calibration models with dimension reduction techniques usually perform well in chemometrics owing to their ability to reduce collinearity, band overlaps, and interactions (Berrueta et al., 2007; Xiaobo et al., 2010). Despite their benefits, dimension reduction methods usually suffer from the fact that the latent variables are hard to interpret in terms of original features. Another problem of dimension reduction with PCA is that the feature contributing most to maintaining variability among predictor variables does not necessarily contribute to further tasks such as prediction or classification, a problem that is a focus of this thesis.

Another way to alleviate the high dimensionality of NIR data is variable selection. Variable selection techniques are based on the principle of choosing a subset of variables from the original large number of features that produce the smallest possible errors when used to perform operations such as making quantitative determinations or discriminating between dissimilar samples. It is commonly acknowledged that the predictive ability will be increased and the complexity of the model will be reduced by a judicious selection of wavelengths. Classically, this selection is made from the basic knowledge about the spectroscopic properties of the sample. Nowadays, it is more often to select variables based on some statistical or machine learning methods, such as interactive variable selection, uninformative variable elimination, interval PLS, significance tests of model parameters, and genetic algorithms (Xiaobo et al., 2010). Classic statistical methods such as LASSO can also be employed to select variables for NIR data. However, since NIR data usually have hundreds or even more than one thousand variables, using LASSO in

such high dimension can be time demanding. Accordingly, dimension reduction technique is more suitable in this case and in this thesis, we focus on dimension reduction techniques for NIR spectral data.

Note that the common way of analysing NIR data is to regard each spectrum as a vector of p dimension where p is the number of wavelengths. In the meanwhile, another way of analysing NIR data is to regard each spectrum as a function of wavelength and then employ functional data analysis (FDA) on the smoothed spectral data (Aguilera et al., 2013). In FDA the smoothing of data usually requires extra tuning parameters (such as the number and location of knots) and additional assumptions of data (such as the degree of smoothness). Typical FDA methods that can be applied to NIR spectral analysis include functional linear regression (Saeys et al., 2008), functional PCA (Huang et al., 2008) and functional PLS (Preda et al., 2007), etc. As summarized by Aguilera (Aguilera et al., 2013), the FDA of NIR data yields similar results on prediction to its discrete counterpart, while usually requires more complex models.

1.1.2 NIR Spectral Classification

As discussed in the previous section, NIR spectroscopy is widely used in the quantitative prediction of the content of chemical compositions, adulteration detection, classification and chemical process control. Among these, classification is an important task and there has been a long history of using NIR spectroscopy in the classification of food, for example, dairy products (Wang et al., 2017; Rodriguez-Otero et al., 1997), meat products (Prieto et al., 2009a,b), oil (Wang et al., 2017), grains (Delwiche and Norris, 1993) and the geographical classification of alcoholic beverages (Liu et al., 2006; Wang et al., 2017) and honey (Tewari and Irudayaraj, 2005), etc. NIR spectroscopy is also widely used in the classification of pharmaceuticals (Gendrin et al., 2008; Roggo et al., 2007; Wang and Yu, 2015), wood (Tsuchikawa and Kobori, 2015) and paper (Tsuchikawa, 2007), etc.

The most widely used classification models in NIR spectral classification can be roughly divided into two categories, depending on whether they can work in the situation of large p small n. If the method implicitly includes a dimension reduction procedure and it can work well when the number of variables exceeds the number of samples, there is no need to apply a dimension reduction or variable selection technique prior to it. This kind of method typically includes partial least squares discriminant analysis (PLS-DA) and soft independent modelling of class analogy (SIMCA), and their variants (Rosipal et al., 2003; Rosipal and Trejo, 2001). With the principle of PLS, PLS-DA can extract features owning the highest covariance with the label variables and then use the extracted features in the classification. As one of the most widely used methods in chemometrics, PLS-DA has been extensively applied to the geographical classification of red wine (Wang et al., 2017; Liu et al., 2006), honey (Tewari and Irudayaraj, 2005), Chinese herbs (Wang and Yu, 2015), the discrimination of meat (Prieto et al., 2009a) and the identification of wood species (Tsuchikawa and Kobori, 2015), etc. SIMCA constructs an independent PC subspace for each class with relatively small number of PCs, projects the training data to the corresponding PC subspace, and uses the residual distances of the training samples to determine a critical distance for classification, with F distribution. A new observation is assigned to the class when its residual distance to the PC subspace is below the statistical limit for this class. The main difference between SIMCA and PCA is that it builds a PC model for each class while PCA builds one model based on all samples. SIMCA implicitly contains a dimension reduction step with PCA. This characteristic makes SIMCA applicable to high-dimensional data. SIMCA has also been widely used in the discrimination of food (Prieto et al., 2009a), medicine (Roggo et al., 2007), and wood (Tsuchikawa, 2007), etc.

On the contrary, some other classification methods fail to work in the large p small n case and a dimension reduction step is necessary before the classifier. These kinds of methods typically include linear discriminant analysis (LDA) and quadratic

discriminant analysis (QDA). LDA and QDA have been widely applied to the geographical classification of the red wine (Liu et al., 2006), Chinese rice wine (Wang et al., 2017), medicine (Roggo et al., 2007), as well as the identification of meat of different quality grade (Prieto et al., 2009a), demolition waste of wood, plastics, and stone (Tsuchikawa, 2007), etc. In the above applications PCA is usually applied as a dimension reduction and pre-processing procedure for LDA and QDA.

Apart from the above methods, k-nearest neighbours and support vector machine do not explicitly require a dimension reduction step, but they can also benefit from a dimensionality reduction procedure prior to them (Berrueta et al., 2007).

In this thesis we mainly focus on LDA, QDA and PLS-DA. These algorithms will be discussed in detail in the next three sections.

1.2 Discriminant Analysis

1.2.1 Fisher's Linear Discriminant

The classic method of linear discrimination was described by Fisher (Fisher, 1936) for two classes and extended to more by Rao (Rao, 1948). This method focuses on binary classification and it aims to find a linear combination of variables that well separates the two classes.

To be specific, Fisher's linear discriminant is implemented by finding an orientation vector $\boldsymbol{w}$ that maximises the following objective $\boldsymbol{J}(\boldsymbol{w})$ in binary classification:

$$\boldsymbol{J}(\boldsymbol{w}) = \frac{\boldsymbol{w}^T \boldsymbol{S}_B \boldsymbol{w}}{\boldsymbol{w}^T \boldsymbol{S}_W \boldsymbol{w}}, \tag{1.2.1}$$

where $\boldsymbol{S}_B$ is the between-group sum-of-squares and products (SSP) matrix:

$$\boldsymbol{S}_B = \sum_{i=1}^{2} n_i (\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})^T = \frac{n_1 n_2}{n_1 + n_2} (\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)^T, \tag{1.2.2}$$

and $\boldsymbol{S}_W$ is the within-group sum-of-squares and products matrix,

$$\boldsymbol{S}_W = \sum_{i=1}^{2} \sum_{j=1}^{n_i} (\boldsymbol{x}_{ij} - \bar{\boldsymbol{x}}_i)(\boldsymbol{x}_{ij} - \bar{\boldsymbol{x}}_i)^T. \tag{1.2.3}$$

In formula (1.2.2) and (1.2.3), n_i is the number of data examples in group i (where i =1, 2), $\bar{\boldsymbol{x}}_i$ is the column mean of group i and $\bar{\boldsymbol{x}}$ is the column mean of all samples.

Differentiating (1.2.1) with respect to $\boldsymbol{w}$ (Bishop, 2006), we find that $\boldsymbol{J}(\boldsymbol{w})$ is maximised when:

$$(\boldsymbol{w}^T \boldsymbol{S}_B \boldsymbol{w}) \boldsymbol{S}_W \boldsymbol{w} = (\boldsymbol{w}^T \boldsymbol{S}_W \boldsymbol{w}) \boldsymbol{S}_B \boldsymbol{w}. \tag{1.2.4}$$

As both $(\boldsymbol{w}^T \boldsymbol{S}_B \boldsymbol{w})$ and $(\boldsymbol{w}^T \boldsymbol{S}_W \boldsymbol{w})$ are scalars we can divide the above equation by $(\boldsymbol{w}^T \boldsymbol{S}_W \boldsymbol{w})$. Let $\lambda = \frac{(\boldsymbol{w}^T \boldsymbol{S}_B \boldsymbol{w})}{(\boldsymbol{w}^T \boldsymbol{S}_W \boldsymbol{w})}$, then equation (1.2.4) becomes:

$$\lambda \boldsymbol{S}_W \boldsymbol{w} = \boldsymbol{S}_B \boldsymbol{w}, \quad \Rightarrow \quad \boldsymbol{S}_W^{-1} \boldsymbol{S}_B \boldsymbol{w} = \lambda \boldsymbol{w}, \tag{1.2.5}$$

if $\boldsymbol{S}_W$ is invertible. Then we can find $\boldsymbol{w}$ by solving this eigen-decomposition problem and $\boldsymbol{w}$ is the eigenvector of $\boldsymbol{S}_W^{-1} \boldsymbol{S}_B$.

In particular, note that

$$\begin{aligned} \boldsymbol{S}_B \boldsymbol{w} &= \frac{n_1 n_2}{n_1 + n_2} (\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)^T \boldsymbol{w} \\ &= \frac{n_1 n_2}{n_1 + n_2} (\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)^T \boldsymbol{w} (\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2), \end{aligned} \tag{1.2.6}$$

Here $\frac{n_1 n_2}{n_1 + n_2} (\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)^T \boldsymbol{w}$ is a scalar, thus $\boldsymbol{S}_B \boldsymbol{w}$ is in the same direction as $(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)$.

According to equation (1.2.5),

$$\boldsymbol{S}_W \boldsymbol{w} = \frac{1}{\lambda} \boldsymbol{S}_B \boldsymbol{w}, \tag{1.2.7}$$

so that $\boldsymbol{S}_W\boldsymbol{w}$ must be in the same direction as $(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)$ as well, namely

$$\boldsymbol{S}_W\boldsymbol{w} \propto (\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2), \tag{1.2.8}$$

and then

$$\boldsymbol{w} \propto \boldsymbol{S}_W^{-1}(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2), \tag{1.2.9}$$

when $\boldsymbol{S}_W$ is invertible. Symbol $\propto$ represents proportional to and here it means in the same direction. In other words, formula (1.2.9) means that $\boldsymbol{w}$ is in the same direction as $\boldsymbol{S}_W^{-1}(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)$. Therefore the orientation vector $\boldsymbol{w}$ is found as the unit vector in the direction of $\boldsymbol{S}_W^{-1}(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)$ and it also can be obtained by implementing eigen-decomposition on $\boldsymbol{S}_W^{-1}\boldsymbol{S}_B$, as long as $\boldsymbol{S}_W$ is invertible.

1.2.2 Relationship between Fisher Linear Discriminant and Linear Regression in Binary Classification

As is shown by Zhang et al. (2005) and Ye (2007), when linear regression (LR) is used as the classifier in a binary classification the estimate of the linear regression coefficient vector is proportional to the orientation vector $\boldsymbol{w}$ obtained from LDA. Moreover, LR and LDA are found to achieve equivalent classification boundaries as long as the proportions of the two classes in the training set are equal (Ripley, 2007). This relationship provides the possibility to transform linear regression problems with label outputs into LDA problems, or vice versa. In this section, we will show the relationship between the LR coefficient and the LDA orientation vector in detail.

For simplicity we assume the two classes are of the same sample size and $y_i = 1$ if the sample i belongs to class 1 and $y_i = -1$ if the sample belongs to class 2. Assume the total sample size is n then the sample size of the two groups $n_1 = n_2 = \frac{n}{2}$.

For this purpose, we fit a linear regression model with coefficients $\boldsymbol{w}$ and w_0

where w_0 is the intercept, i.e.

$$\hat{\boldsymbol{y}} = \boldsymbol{X}\boldsymbol{w} + \boldsymbol{h}w_0, \tag{1.2.10}$$

where $\hat{\boldsymbol{y}} \in \mathbb{R}^{n\times 1}$ is the predicted value, $\boldsymbol{X} \in \mathbb{R}^{n\times p}$ is the data matrix, $\boldsymbol{h} \in \mathbb{R}^{n\times 1}$ is a vector of 1s where n is the total number of samples. Then the sum of squared errors is:

$$SSE = ||\boldsymbol{y} - \hat{\boldsymbol{y}}||^2 = ||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w} - \boldsymbol{h}w_0||^2 = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w} - \boldsymbol{h}w_0)^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w} - \boldsymbol{h}w_0) \tag{1.2.11}$$

Taking the first derivative of SSE with respect to $\boldsymbol{w}$ and w_0 and then setting them to zero, we get

$$\frac{\partial SSE}{\partial \boldsymbol{w}} = -\boldsymbol{X}^T\boldsymbol{y} + nw_0\bar{\boldsymbol{x}} + \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w} = 0, \tag{1.2.12}$$

$$\frac{\partial SSE}{\partial w_0} = w_0 + \boldsymbol{w}^T\bar{\boldsymbol{x}} = 0. \tag{1.2.13}$$

Here $\bar{\boldsymbol{x}}$ is the column vector containing the column mean of $\boldsymbol{X}$. From equation (1.2.13):

$$w_0 = -\boldsymbol{w}^T\bar{\boldsymbol{x}}. \tag{1.2.14}$$

Substitute (1.2.14) into (1.2.12) we obtain:

$$-n\boldsymbol{w}^T\bar{\boldsymbol{x}}\bar{\boldsymbol{x}} + \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w} = \boldsymbol{X}^T\boldsymbol{y}, \tag{1.2.15}$$

which leads to

$$(\boldsymbol{X}^T\boldsymbol{X} - n\bar{\boldsymbol{x}}\bar{\boldsymbol{x}}^T)\boldsymbol{w} = \boldsymbol{X}^T\boldsymbol{y}. \tag{1.2.16}$$

Note that when the two classes are of equal size, $\boldsymbol{X}^T\boldsymbol{y} = \frac{n}{2}(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)$. Then (1.2.16)

becomes:

$$(\boldsymbol{X}^T\boldsymbol{X} - n\bar{\boldsymbol{x}}\bar{\boldsymbol{x}}^T)\boldsymbol{w} = \frac{n}{2}(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2). \tag{1.2.17}$$

Since

$$\boldsymbol{S}_T = \sum_{i=1}^{n}(\boldsymbol{x}_i - \bar{\boldsymbol{x}})(\boldsymbol{x}_i - \bar{\boldsymbol{x}})^T = \sum_{i=1}^{n}\boldsymbol{x}_i\boldsymbol{x}_i^T - \sum_{i=1}^{n}\bar{\boldsymbol{x}}\bar{\boldsymbol{x}}^T = \boldsymbol{X}^T\boldsymbol{X} - n\bar{\boldsymbol{x}}\bar{\boldsymbol{x}}^T, \tag{1.2.18}$$

is the total SSP matrix, we have

$$\boldsymbol{S}_T\boldsymbol{w} = \frac{n}{2}(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2). \tag{1.2.19}$$

Similarly we have,

$$\boldsymbol{S}_W = \boldsymbol{S}_1 + \boldsymbol{S}_2 = (\boldsymbol{X}_1^T\boldsymbol{X}_1 - n_1\bar{\boldsymbol{x}}_1\bar{\boldsymbol{x}}_1^T) + (\boldsymbol{X}_2^T\boldsymbol{X}_2 - n_2\bar{\boldsymbol{x}}_2\bar{\boldsymbol{x}}_2^T) = \boldsymbol{X}^T\boldsymbol{X} - \frac{n}{2}(\bar{\boldsymbol{x}}_1\bar{\boldsymbol{x}}_1^T + \bar{\boldsymbol{x}}_2\bar{\boldsymbol{x}}_2^T), \tag{1.2.20}$$

where $\boldsymbol{S}_1$ and $\boldsymbol{S}_2$ are the SSP matrices of the two groups, $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ are the data matrices of the two groups. From formula (1.2.2) we have,

$$\boldsymbol{S}_B = \frac{n_1 n_2}{n_1 + n_2}(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)^T = \frac{n}{4}(\bar{\boldsymbol{x}}_1\bar{\boldsymbol{x}}_1^T + \bar{\boldsymbol{x}}_2\bar{\boldsymbol{x}}_2^T) - \frac{n}{2}\bar{\boldsymbol{x}}_1\bar{\boldsymbol{x}}_2^T. \tag{1.2.21}$$

Then

$$\begin{aligned}\boldsymbol{S}_W + \boldsymbol{S}_B &= \boldsymbol{X}^T\boldsymbol{X} - \frac{n}{4}(\bar{\boldsymbol{x}}_1\bar{\boldsymbol{x}}_1^T + \bar{\boldsymbol{x}}_2\bar{\boldsymbol{x}}_2^T) - \frac{n}{2}\bar{\boldsymbol{x}}_1\bar{\boldsymbol{x}}_2^T \\ &= \boldsymbol{X}^T\boldsymbol{X} - \frac{n}{4}(\bar{\boldsymbol{x}}_1 + \bar{\boldsymbol{x}}_2)(\bar{\boldsymbol{x}}_1 + \bar{\boldsymbol{x}}_2)^T \\ &= \boldsymbol{X}^T\boldsymbol{X} - n\bar{\boldsymbol{x}}\bar{\boldsymbol{x}}^T = \boldsymbol{S}_T,\end{aligned} \tag{1.2.22}$$

or namely,

$$\boldsymbol{S}_T = \boldsymbol{S}_W + \boldsymbol{S}_B. \tag{1.2.23}$$

Then from equation (1.2.19) we have,

$$(\boldsymbol{S}_W + \boldsymbol{S}_B)\boldsymbol{w} = \frac{n}{2}(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2). \tag{1.2.24}$$

From (1.2.6) we know that no matter the direction of $\boldsymbol{w}$, $\boldsymbol{S}_B\boldsymbol{w}$ is always in the direction of $(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)$, then according to equation (1.2.24), $\boldsymbol{S}_W\boldsymbol{w}$ is also in the direction of $(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)$, namely

$$\boldsymbol{S}_W\boldsymbol{w} \propto (\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2), \tag{1.2.25}$$

and

$$\boldsymbol{w} \propto \boldsymbol{S}_W^{-1}(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2), \tag{1.2.26}$$

if $\boldsymbol{S}_W$ is invertible.

Compare formula (1.2.26) with formula (1.2.9), we shall see the direction of the coefficient vector of linear regression is the same as the projection vector of Fisher linear discriminant.

So far we have shown that the coefficient vector of linear regression is in the same direction as the projection vector of Fisher linear discriminant, when the sample size is equal for the two groups. When the sample sizes are not equal, according to Bishop (2006), same conclusion can be made by setting $y_i = \frac{n}{n_1}$ for group 1 and $y_i = -\frac{n}{n_2}$ for group 2, where n_1 and n_2 are the sample sizes of the two groups and n is the total sample size.

The above derivation shows that Fisher linear discriminant analysis with equal sample size finds the same orientation direction as LR. In other words using LDA as the classifier is analogous to using LR as the classifier. Note that PLS-DA uses the principle of PLS to reduce dimension while using LR as its classifier, then PLS-DA can be regarded as an analogue of PLS-LDA (as DA in this case is analogous to LDA). Here by PLS-LDA we mean, using PLS in dimension reduction and latent variable extraction and then using LDA to classify samples with the new

variables. This conclusion makes it easier to compare PLS-DA and PCA-LDA in high-dimensional classification. PCA-LDA uses PCA to reduce dimension and then classify the samples with LDA on the PC subspace. As PLS-DA is an analogue of PLS-LDA, the difference of PLS-DA and PCA-LDA is regarded as mostly due to the dimension reduction step with PCA or PLS. Then we can easily compare the performance of PCA and PLS in high dimensional classification.

1.2.3 Linear Discriminant Analysis and Quadratic Discriminant Analysis

The well-known linear discriminant analysis and quadratic discriminant analysis are developed based on the classic Fisher's linear discriminant analysis. Though they are derived in a probabilistic way, it can be shown that in binary classification with balanced groups the projection vector in LDA is identical to the orientation vector in Fisher linear discriminant, which is also the reason why these two algorithms are regarded closely related to each other. The algorithm of the well-known LDA and QDA in binary classification are as follows.

In general binary classification, we assume label variable y = 1 for samples from the first class and label variable y= -1 for samples from the second class.

Assume $\boldsymbol{x}\,|\,y=1 \sim \boldsymbol{N}(\boldsymbol{\mu}_1,\boldsymbol{\Sigma}_1)$ and $\boldsymbol{x}\,|\,y=-1 \sim \boldsymbol{N}(\boldsymbol{\mu}_2,\boldsymbol{\Sigma}_2)$. Here LDA assumes $\boldsymbol{\Sigma}_1$ and $\boldsymbol{\Sigma}_2$ to be identical, while QDA is more general and does not make this assumption. To simplify the notation, we write $\mathbb{P}(\boldsymbol{x},y=1) = f(\boldsymbol{x}|y=1)\mathbb{P}(Y=1)$ where $f(\boldsymbol{x}|y=1)$ donotes the conditional density of $\boldsymbol{x}$ given $y=1$. $\mathbb{P}(\boldsymbol{x},y=-1)$ will be defined in a similar way.

Then for data belonging to class 1,

$$\mathbb{P}(\boldsymbol{x},y=1) = \frac{1}{\sqrt{(2\pi)^p|\boldsymbol{\Sigma}_1|}}\exp\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_1)^T\boldsymbol{\Sigma}_1^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_1)\}\cdot\mathbb{P}(y=1), \quad (1.2.27)$$

where p is the dimension of $\boldsymbol{x}$ and $\mathbb{P}(y=1)$ is the prior probability of class 1.

Similarly the joint probability of class 2 can be obtained,

$$\mathbb{P}(\boldsymbol{x}, y=-1) = \frac{1}{\sqrt{(2\pi)^p |\boldsymbol{\Sigma}_2|}} \exp\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_2)^T \boldsymbol{\Sigma}_2^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_2)\} \cdot \mathbb{P}(y=-1), \tag{1.2.28}$$

where $\mathbb{P}(y=-1)$ is the prior probability of class 2.

For simplicity we take the logarithm for these two joint probability formulas:

$$g_1(\boldsymbol{x}) = -\frac{p}{2} ln(2\pi) - \frac{1}{2} ln(|\boldsymbol{\Sigma}_1|) - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_1)^T \boldsymbol{\Sigma}_1^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_1) + ln(\mathbb{P}(y=1)), \tag{1.2.29}$$

and

$$g_2(\boldsymbol{x}) = -\frac{p}{2} ln(2\pi) - \frac{1}{2} ln(|\boldsymbol{\Sigma}_2|) - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_2)^T \boldsymbol{\Sigma}_2^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_2) + ln(\mathbb{P}(y=-1)). \tag{1.2.30}$$

Then sample $\boldsymbol{x}$ is classified as class 1 (y = 1) if $g_1(\boldsymbol{x}) - g_2(\boldsymbol{x}) > 0$, namely if

$$\frac{1}{2} ln\left(\frac{|\boldsymbol{\Sigma}_2|}{|\boldsymbol{\Sigma}_1|}\right) + \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_2)^T \boldsymbol{\Sigma}_2^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_2) - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_1)^T \boldsymbol{\Sigma}_1^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_1) + ln\frac{\mathbb{P}(y=1)}{\mathbb{P}(y=-1)} > 0, \tag{1.2.31}$$

i.e.

$$(\boldsymbol{x}-\boldsymbol{\mu}_2)^T \boldsymbol{\Sigma}_2^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_2) - (\boldsymbol{x}-\boldsymbol{\mu}_1)^T \boldsymbol{\Sigma}_1^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_1) + ln|\boldsymbol{\Sigma}_2| - ln|\boldsymbol{\Sigma}_1| + 2ln\frac{\mathbb{P}(y=1)}{\mathbb{P}(y=-1)} > 0. \tag{1.2.32}$$

Otherwise, an input $\boldsymbol{x}$ is classified as class 2 (y = -1).

Here if $\boldsymbol{\Sigma}_1$ and $\boldsymbol{\Sigma}_2$ are identical, the problem reduces to the LDA case. Denote $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}$, then the classification criterion can be simplified as

$$(\boldsymbol{x}-\boldsymbol{\mu}_2)^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_2) - (\boldsymbol{x}-\boldsymbol{\mu}_1)^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_1) + 2ln\frac{\mathbb{P}(y=1)}{\mathbb{P}(y=-1)} > 0, \tag{1.2.33}$$

which reduces to

$$(\boldsymbol{\mu}_1^T - \boldsymbol{\mu}_2^T)\boldsymbol{\Sigma}^{-1}\boldsymbol{x} > ln\frac{\mathbb{P}(y=-1)}{\mathbb{P}(y=1)} + \frac{1}{2}\boldsymbol{\mu}_1^T\boldsymbol{\Sigma}\boldsymbol{\mu}_1 - \frac{1}{2}\boldsymbol{\mu}_2^T\boldsymbol{\Sigma}\boldsymbol{\mu}_2. \tag{1.2.34}$$

If we define:

$$\boldsymbol{w} = (\boldsymbol{\mu}_1^T - \boldsymbol{\mu}_2^T)\boldsymbol{\Sigma}^{-1}, \tag{1.2.35}$$

and

$$T = ln\frac{\mathbb{P}(y=-1)}{\mathbb{P}(y=1)} + \frac{1}{2}\boldsymbol{\mu}_1^T\boldsymbol{\Sigma}\boldsymbol{\mu}_1 - \frac{1}{2}\boldsymbol{\mu}_2^T\boldsymbol{\Sigma}\boldsymbol{\mu}_2, \tag{1.2.36}$$

then (1.2.33) can be rewritten as:

$$\boldsymbol{w}\cdot\boldsymbol{x} > T. \tag{1.2.37}$$

This criterion is a linear inequality of $\boldsymbol{x}$ and that is why this algorithm is named as linear discriminant analysis.

Formula (1.2.35) can be rewritten as:

$$\boldsymbol{w}^T = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2). \tag{1.2.38}$$

In practice, the common covariance of the two groups $\boldsymbol{\Sigma}$ can be estimated by $\frac{1}{n-2}\boldsymbol{S}_W$ where $\boldsymbol{S}_W$ is the pooled within-group SSP matrix and n is the total sample size. $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ can be estimated by the sample mean of the two classes, $\bar{\boldsymbol{x}}_1$ and $\bar{\boldsymbol{x}}_2$ respectively. Then the LDA projection vector $\boldsymbol{w}$ in (1.2.38) becomes:

$$\boldsymbol{w}^T = \frac{1}{n-2}\boldsymbol{S}_W^{-1}(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2). \tag{1.2.39}$$

Note that compared with the scale, the direction of the projection vector is usually

of higher importance and the projection vector $\boldsymbol{w}$ is usually set to be of norm 1, then

$$\boldsymbol{w}^T \propto \boldsymbol{S}_W^{-1}(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2), \tag{1.2.40}$$

is the unit vector in the direction of $\boldsymbol{S}_W^{-1}(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)$, which is the same as the orientation vector in the Fisher linear discriminant (1.2.9). Then the projection vector $\boldsymbol{w}$ in LDA and the orientation vector in Fisher linear discriminant are transpose to each other. In Fisher linear discriminant the orientation vector is defined as a column vector and in LDA the projection vector is defined as a row vector, but actually they are unit vectors in the same direction. This suggests that although Fisher linear discriminant makes no distributional assumptions, the orientation vector in Fisher linear discriminant is optimal in the case of two multivariate normals with equal variances.

On the other hand, if $\boldsymbol{\Sigma}_1$ and $\boldsymbol{\Sigma}_2$ cannot be regarded as identical, the simplification in (1.2.33) cannot be made. In other words the classification criterion would be:

$$\boldsymbol{x}^T(\boldsymbol{\Sigma}_2^{-1} - \boldsymbol{\Sigma}_1^{-1})\boldsymbol{x} + 2(\boldsymbol{\mu}_1^T\boldsymbol{\Sigma}_1^{-1} - \boldsymbol{\mu}_2^T\boldsymbol{\Sigma}_2^{-1})\boldsymbol{x} > 2ln\frac{\mathbb{P}(y=-1)}{\mathbb{P}(y=1)} + ln|\boldsymbol{\Sigma}_1| - ln|\boldsymbol{\Sigma}_2| + \boldsymbol{\mu}_1^T\boldsymbol{\Sigma}_1^{-1}\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2^T\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\mu}_2. \tag{1.2.41}$$

Let

$$\boldsymbol{W}_1 = (\boldsymbol{\Sigma}_2^{-1} - \boldsymbol{\Sigma}_1^{-1}), \tag{1.2.42}$$

$$\boldsymbol{w}_2 = 2(\boldsymbol{\mu}_1^T\boldsymbol{\Sigma}_1^{-1} - \boldsymbol{\mu}_2^T\boldsymbol{\Sigma}_2^{-1}), \tag{1.2.43}$$

$$T = 2ln\frac{\mathbb{P}(y=-1)}{\mathbb{P}(y=1)} + ln|\boldsymbol{\Sigma}_1| - ln|\boldsymbol{\Sigma}_2| + \boldsymbol{\mu}_1^T\boldsymbol{\Sigma}_1^{-1}\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2^T\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\mu}_2, \tag{1.2.44}$$

criterion (1.2.41) can be rewritten as:

$$\boldsymbol{x}^T\boldsymbol{W}_1\boldsymbol{x} + \boldsymbol{w}_2\boldsymbol{x} > T. \tag{1.2.45}$$

(1.2.45) is a quadratic function of the input $\boldsymbol{x}$ and that is why this algorithm is called

quadratic discriminant analysis.

1.2.4 LDA as a Feature Extraction Technique

So far we have shown the orientation vector of Fisher linear discriminant and the probabilistic version of LDA in the binary case. Actually, LDA can not only be used as a classifier in the binary classification, but also in multiclass classification. Apart from these, the idea of LDA, especially the Fisher criterion, is widely used in high-dimensional feature extraction. In this section, we will show some variants of LDA, which make it a fundamental feature extraction technique in multiclass high-dimensional classification.

Multiclass LDA is implemented by searching for the orientation vectors $\boldsymbol{W}$ such that $\frac{|\boldsymbol{W}^T \boldsymbol{S}_B \boldsymbol{W}|}{|\boldsymbol{W}^T \boldsymbol{S}_W \boldsymbol{W}|}$ is maximised, where $\boldsymbol{S}_W = \sum_{i=1}^{c} \sum_{j=1}^{n_i} (\boldsymbol{x}_{ij} - \bar{\boldsymbol{x}}_i)(\boldsymbol{x}_{ij} - \bar{\boldsymbol{x}}_i)^T$ is the within-class SSP matrix, $\boldsymbol{S}_B = \sum_{i=1}^{c} n_i (\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})^T$ is the between-class SSP matrix, c is the number of classes and $|\cdot|$ is the determinant. $\boldsymbol{W}$ is found to contain the eigenvectors of $\boldsymbol{S}_W^{-1} \boldsymbol{S}_B$. However, when the dimensionality of the data exceeds the number of samples, $\boldsymbol{S}_W$ becomes singular and its inversion becomes impossible, accordingly the evaluation of eigenvalues and eigenvectors of $\boldsymbol{S}_W^{-1} \boldsymbol{S}_B$ becomes impossible. This is called the small sample size (SSS) problem of LDA (Sharma and Paliwal, 2015b). Variants of LDA have been developed during the last two decades to resolve this SSS problem and make it a feasible feature extraction technique in high-dimensional classification.

Four spaces are mainly employed to reduce dimensionality and resolve the SSS problem of LDA, the null space of $\boldsymbol{S}_W$ ($\boldsymbol{S}_w^{null}$), the range space of $\boldsymbol{S}_W$ ($\boldsymbol{S}_w^{range}$), the range space of $\boldsymbol{S}_B$ ($\boldsymbol{S}_b^{range}$) and the null space of $\boldsymbol{S}_B$ ($\boldsymbol{S}_b^{null}$). Here the null space of a matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ consists of all the vectors $\boldsymbol{w} \in \mathbb{R}^{n \times 1}$ such that $\boldsymbol{A}\boldsymbol{w} = \boldsymbol{0}$, while the range space of a matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ is the space spanned by the column vectors of $\boldsymbol{A}$, i.e. the range space of $\boldsymbol{A}$ consists of vectors $\boldsymbol{w} \in \mathbb{R}^{m \times 1}$ such that $\boldsymbol{w} = \boldsymbol{A}\boldsymbol{x}$ where $\boldsymbol{x} \in \mathbb{R}^{n \times 1}$. Since the goal of LDA is to maximise the ratio of $|\boldsymbol{w}^T \boldsymbol{S}_B \boldsymbol{w}|$ to $|\boldsymbol{w}^T \boldsymbol{S}_W \boldsymbol{w}|$,

the null space of $\boldsymbol{S}_W$ is generally regarded as of high discriminative power, for on this subspace $|\boldsymbol{w}^T \boldsymbol{S}_W \boldsymbol{w}| = |\boldsymbol{w}^T \cdot \boldsymbol{0}| = 0$ for all $\boldsymbol{w}$ (Sharma and Paliwal, 2015b). $\boldsymbol{S}_b^{range}$ contains the between class information and it is regarded as discriminative as well. On the contrary, $\boldsymbol{S}_b^{null}$ is the space on which $|\boldsymbol{w}^T \boldsymbol{S}_B \boldsymbol{w}| = |\boldsymbol{w}^T \cdot \boldsymbol{0}| = 0$ for all $\boldsymbol{w}$ and it contributes very little to the separation of the classes. Accordingly, $\boldsymbol{S}_b^{null}$ is regarded as of the lowest discriminating power among the above four. Dimensionality reduction is then carried out by jointly employing the above four spaces and the resolutions to this problem are categorised into 4 types by the different combination of spaces used in the method.

The first category of resolutions to SSS is to combine the usage of $\boldsymbol{S}_b^{range}$ and $\boldsymbol{S}_w^{range}$. The typical methods include direct LDA (Yu and Yang, 2001) and its variants (Lu et al., 2003; Song et al., 2007; Paliwal and Sharma, 2010). In direct LDA, firstly the data are projected to the range space of $\boldsymbol{S}_B$ while discarding $\boldsymbol{S}_b^{null}$. After the first projection the dimension of the data is reduced from p to $(c-1)$, where c is the number of classes, and the singular problem is solved. Afterwards the data are further projected to the range space of $\boldsymbol{S}_W$ with small eigenvalues. However, this type of methods fails to use the discriminative $\boldsymbol{S}_w^{null}$ and thus can be enhanced by including $\boldsymbol{S}_w^{null}$ into consideration.

The second category of resolutions to SSS is to combine the usage of the most discriminative two spaces $\boldsymbol{S}_w^{null}$ and $\boldsymbol{S}_b^{range}$. This type of methods include Null LDA (NLDA) (Chen et al., 2000), orthogonal LDA (OLDA) (Ye, 2005) and their variants (Chu and Thye, 2010; Sharma and Paliwal, 2012a). The null LDA technique finds the orientation $\boldsymbol{w}$ in two stages. In the first stage, it computes $\boldsymbol{w}$ such that $\boldsymbol{w}^T \boldsymbol{S}_W \boldsymbol{w}$ = 0, i.e., data are projected onto the null space of $\boldsymbol{S}_W$. In the second stage it finds $\boldsymbol{w}$ that satisfies $\boldsymbol{w}^T \boldsymbol{S}_B \boldsymbol{w} \neq 0$.

The third category of resolutions combines the usage of $\boldsymbol{S}_w^{null}$, $\boldsymbol{S}_b^{range}$ and $\boldsymbol{S}_w^{range}$. The most well known methods in this category include regularised LDA (Friedman, 1989; Dai and Yuen, 2007) and improved regularised LDA (Sharma et al., 2014).

To overcome the singularity problem, in the regularized LDA a small perturbation matrix has been added to $\boldsymbol{S}_W$. This makes the matrix non-singular and invertible. Then the original Fisher criterion will be generalised as:

$$\boldsymbol{J}(\boldsymbol{w},\delta) = \frac{\boldsymbol{w}^T \boldsymbol{S}_B \boldsymbol{w}}{\boldsymbol{w}^T (\boldsymbol{S}_W + \delta \boldsymbol{I}) \boldsymbol{w}}, \tag{1.2.46}$$

and a generalised eigen-decomposition will be implemented to find the vector $\boldsymbol{w}$:

$$(\boldsymbol{S}_W + \delta \boldsymbol{I})^{-1} \boldsymbol{S}_B \boldsymbol{w} = \lambda \boldsymbol{w}. \tag{1.2.47}$$

Here the transformation vector $\boldsymbol{w}$ is found to be the eigenvector of $(\boldsymbol{S}_W + \delta \boldsymbol{I})^{-1} \boldsymbol{S}_B$. The additional δ in the regularised method helps in incorporating both the null space and range space of $\boldsymbol{S}_W$ and $(\boldsymbol{S}_W + \delta \boldsymbol{I})$ is invertible now. The performance of this method is highly dependent on the choice of δ (Sharma and Paliwal, 2015b). This δ can be chosen either by cross-validation, or in a deterministic approach (Sharma and Paliwal, 2015a). In the improved RDA, the above criterion (1.2.46) is converted into a constrained maximisation problem and the value of δ can be determined using method of Lagrange multipliers (Sharma et al., 2014).

Methods from the final category use all of the four spaces. For example in the two stage LDA (Sharma and Paliwal, 2012b), the eigenvectors of $\boldsymbol{S}_W^{'-1} \boldsymbol{S}_B$ and $\boldsymbol{S}_B^{'-1} \boldsymbol{S}_W$ are concatenated, so as to include the information of $\boldsymbol{S}_b^{null}$ and $\boldsymbol{S}_w^{range}$. Here $\boldsymbol{S}_W^{'}$ is a regularised version of $\boldsymbol{S}_W$ while $\boldsymbol{S}_B^{'}$ is a regularised version of $\boldsymbol{S}_B$.

In this section we introduce four typical categories of methods to resolve the SSS problem of LDA so as to make it a feature extraction technique in high-dimensional classification. However the performance of a given category of methods can vary a lot, depending on how effectively the corresponding spaces are utilised. And the performance of a given method also varies from case to case, depending on the discriminating power of the spaces it uses in a specific data set

(Sharma and Paliwal, 2015b). It has been shown in real data experiments that using variants of LDA such as OLDA and NLDA to directly extract features led to inferior performance compared with applying the conventional LDA after dimension reduction with PCA (Prasad et al., 2010).

Although none of the abovementioned LDA-based methods are explicitly used in this thesis, the idea of using discriminant analysis as a feature extraction method enlightens us to propose our penalised QDA feature extraction method (Chapter 3).

1.2.5 Comparison of LDA and QDA in Classification

QDA and LDA make different assumptions on the covariance structure of the data and thus obtain different type of decision boundaries. As shown in (1.2.35) to (1.2.37) LDA assumes the covariances to be homogeneous and thus leads to a linear boundary, while according to (1.2.42) to (1.2.45) QDA assumes the covariance of different classes to be inhomogeneous and this gives a quadratic boundary.

There is a long-lasting debate on comparison of the performance of LDA and QDA. In some literature (Hong et al., 2017; Siqueira et al., 2017; Naghibi et al., 2018; Costa et al., 2017) QDA was found to be more powerful than LDA, while in some other literature (Wu et al., 1996, 2003; Vaid et al., 2001) QDA was found less powerful than LDA. Meanwhile, in many other literatures (Balabin et al., 2010; Kim et al., 2011; Higdon et al., 2004) these two methods were found to be of comparable discrimination power.

The performance of LDA and QDA highly depends on the data set we implement the two algorithms on. If the homogeneous covariance assumption is clearly inappropriate (Wu et al., 1996; Yan and Dai, 2011) or when the linear boundary is no longer adequate to separate classes (Friedman et al., 2001a), QDA outperforms LDA. While when the true decision boundary is linear on the predictor variables, or when the sample size is too small to afford a quadratic decision boundary or too small for the number of parameters to be estimated, LDA out performs QDA (Wu

et al., 2003, 1996; Vaid et al., 2001).

QDA was found to work well after dimension reduction or variable selection techniques, as higher dimension means a dramatic increase in the number of parameters to be estimated in QDA (Yan and Dai, 2011). To be specific, if we assume p to be the dimensionality of feature space and c is the number of classes to separate, in LDA there are $(c-1)\times(p+1)$ parameters to estimate while in QDA there are $(c-1)\times\left(\frac{p(p+3)}{2}+1\right)$ parameters (Friedman et al., 2001a). Therefore, it is important to control the number of parameters, and thus the number of features if QDA is employed as the classifier. For this reason, dimension reduction is often seen as a necessary step before QDA.

In this thesis we use QDA as our classifier after dimension reduction with some novel variants of PCA. PCA-based methods help reducing noise while maintaining the most influential features, which lightens the burden of the parameter estimation in QDA. In this case QDA can utilise the second order structure of the variables to build a nonlinear classification. The combination of PCA and QDA is shown to outperform the combination of PCA and LDA in all of our examples.

1.3 Principal Component Analysis

Principal component analysis is a well known dimension reduction technique that has been extensively applied in signal processing, pattern recognition and information retrieval (Yu et al., 2006; Duda et al., 2001). In PCA, an orthogonal transformation is employed to convert the original possibly correlated variables into a set of linearly uncorrelated variables called principal components (PCs) with the variability captured decreasing PC by PC (Jolliffe, 1986). Each principal component is a linear combination of the original variables and can be obtained as the solution to an eigendecomposition problem of the covariance matrix or, alternatively, from the singular value decomposition (SVD) of the centred data matrix (Jolliffe and Cadima, 2016).

One of the novel methods proposed in this thesis is based on the covariance matrix decomposition version of PCA. Hence we introduce PCA below in this way.

1.3.1 Algorithm

Consider a data matrix $\boldsymbol{X} \in \mathbb{R}^{n\times p}$ in which there are n instances $(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)^T$ of a p-dimensional vector $\boldsymbol{x}_i \in \mathbb{R}^{p\times 1}$.

The classic PCA algorithm starts with data centering. The data matrix $\boldsymbol{X}$ is centred by subtracting the column means from it:

$$\boldsymbol{X}^c = \boldsymbol{X} - \boldsymbol{h}\boldsymbol{u}^T, \tag{1.3.1}$$

where $\boldsymbol{h} \in \mathbb{R}^{n\times 1}$ is a column vector of 1s, $\boldsymbol{u} \in \mathbb{R}^{p\times 1}$ is a column vector containing the mean of each column of $\boldsymbol{X}$. Then the principal components can be obtained by implementing eigendecomposition on the sample covariance matrix.

In the classic PCA the sample covariance matrix $\boldsymbol{S}$ is denoted as:

$$\boldsymbol{S} = \frac{1}{n-1}(\boldsymbol{X}^c)^T\boldsymbol{X}^c. \tag{1.3.2}$$

In eigendecomposition this symmetric sample covariance $\boldsymbol{S}$ is decomposed as:

$$\boldsymbol{S} = \boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{V}^T, \tag{1.3.3}$$

where $\boldsymbol{V} \in \mathbb{R}^{p\times p}$ contains the normalised eigenvectors of $\boldsymbol{S}$ as its columns and $\boldsymbol{\Sigma}$ is a diagonal matrix of eigenvalues $\sigma_1 \geqslant \sigma_2 \geqslant \ldots \geqslant \sigma_p \geqslant 0$.

Dimension reduction is implemented by selecting the first k columns ($k \leqslant p$) from $\boldsymbol{V}$ and projecting data to this subspace of k dimensions. In dimension reduction, some criteria can be used to choose the threshold k, for example requiring that at least 80% of the total variation should be contained in the first k components (Jolliffe, 1986). If used as a dimension reduction method before discrimination, the

number of components k kept in the model can be tuned with the target of minimising cross-validated classification error rate.

Then the projected data matrix of lower dimension can be calculated by:

$$\boldsymbol{Z} = \boldsymbol{X}^c \boldsymbol{V}_k, \tag{1.3.4}$$

where $\boldsymbol{X}^c \in \mathbb{R}^{n\times p}$ is the centred data matrix and $\boldsymbol{V}_k \in \mathbb{R}^{p\times k}$ consists of the first k PC loadings. Here $\boldsymbol{Z} \in \mathbb{R}^{n\times k}$ are the first k principal component scores.

1.3.2 Further Developments

PCA was invented in 1901 by Karl Pearson (Pearson, 1901) as an analogue of the principal axis theorem in mechanics and was later independently developed and named by Harold Hotelling in the 1930s (Hotelling, 1933). However, it was computationally infeasible to use PCA on large data set until electronic computers became widely available (Jolliffe and Cadima, 2016). Since then its utility has burgeoned and can be discovered in many disciplines including signal and image processing in engineering (Algazi et al., 1993; Sirovich and Kirby, 1987; Kirby and Sirovich, 1990; Turk and Pentland, 1991), gene expression analysis in biology (Yeung and Ruzzo, 2001; Sturn et al., 2002), and even in finance and other social science fields (Zou et al., 2006; Olawale and Garwe, 2010; Ince and Trafalis, 2007).

In analytical chemistry, PCA was firstly introduced by Malinowski in 1960s under the name principal factor analysis (Wold et al., 1987) and since the 1970s a large number of applications in analytical chemistry have been published. Especially, PCA can contribute to quantitative prediction of the concentration of chemical elements (Chiang et al., 2000; Moghimi et al., 2010; Geladi, 2003), chemical image analysis (Geladi et al., 1989; Gowen et al., 2008), fault detection in chemical processes (Chiang et al., 2000; Kresta et al., 1991; Wise et al., 1990) and classification (Kallithraka et al., 2001; Christy et al., 2004; Tewari and Irudayaraj, 2005).

In high-dimensional classification, PCA is usually applied as a dimension reduction technique before the classifier, such as LDA, QDA or support vector machine (SVM) (Berrueta et al., 2007; Christy et al., 2004; Tewari and Irudayaraj, 2005). Regardless of its prevalence, the classic PCA approach has many limitations and this has led to the development of various adaptations and extensions of PCA in the last two decades.

Firstly, the classical PCA approach is based on the calculation of sample mean and sample covariance and as a result it is very sensitive to the presence of outliers (Candès et al., 2011; Hubert and Engelen, 2004). This has led to attempts to define robust variants of PCA. There are two kinds of robust approaches of PCA. One type is based on PCA on a robust covariance matrix. Among these multivariate trimming (MVT), minimum volume ellipsoid (MVE) and minimum covariance determinant (MCD) (Egan and Morgan, 1998; Devlin et al., 1981; Hubert et al., 2005) are the most prevailing robust estimation methods. The other group of approaches are based on the idea of searching the projections of the data most exposing outliers (Daszykowski et al., 2007). Based on this idea, Li and Chen proposed their projection-pursuit method (Li and Chen, 1985). This method sought low-dimensional projections that maximise a robust measure of spread and it was further developed by Xie et al. (Xie et al., 1993) and Hubert et al. (Hubert et al., 2002) in chemometrics.

Another prevailing adaptation of PCA is sparse PCA. In the classic algorithm, each PC is a linear combination of all variables and the loadings are typically nonzero, which makes PCs lack interpretability with respect to the original features (Zou et al., 2006). It has been an interesting topic for years how to achieve dimension reduction while simultaneously reducing the number of explicitly used variables. This has led to the development of sparse PCA. The widely acknowledged sparse PCA was proposed in 2006 by Zou et.al (Zou et al., 2006) After that, Guan et al. developed a probabilistic version of robust PCA in 2009 (Guan and Dy,

2009), Toczydlowska (2020).

Due to its nature as a linear projection method, another limitation of PCA is its inability to discover nonlinear relationships in the data. This led to the development of nonlinear PCA. A well-known nonlinear form of PCA, kernel PCA was proposed in 1997 (Schölkopf et al., 1997) via the use of integral operator kernel functions. Since the computational complexity of kernel PCA does not grow with the dimensionality of the feature space, it was shown to be more efficient than the classic algorithm in both theory, simulation and empirical studies when the number of variables is far beyond the number of observations (Kim et al., 2002). Kernel PCA was widely used in fault detection (Chiang et al., 2000) and chemical process monitoring (Cho et al., 2005; Lee et al., 2004) in chemistry.

Another limitation of PCA is that it is neither a probabilistic model nor a generative model (Kim and Lee, 2003), thus it is difficult to combine PCA with other probabilistic models (Tipping and Bishop, 1999). To address this limitation of PCA, probabilistic PCA (PPCA) was built by Roweis et al. in 1998 (Roweis, 1998) and by Tipping et al. in 1999 from the perspective of a Gaussian latent variable model (Tipping and Bishop, 1999). In probabilistic PCA, the observed variables $\boldsymbol{X}$ of p dimensions are regarded as generated from latent variables $\boldsymbol{Z}$ of d dimensions, where $d < p$. The latent variables $\boldsymbol{Z}$ follow a multivariate Gaussian distribution $\boldsymbol{N}(\boldsymbol{0},\boldsymbol{I})$ while the observed variables $\boldsymbol{X}$ are the results of linear transformation of the latent variables $\boldsymbol{Z}$, plus a location parameter $\boldsymbol{\mu}$ and a random noise $\boldsymbol{\varepsilon} \sim \boldsymbol{N}(\boldsymbol{0},\sigma^2\boldsymbol{I})$, namely $\boldsymbol{X} = \boldsymbol{W}\boldsymbol{Z} + \boldsymbol{\mu} + \boldsymbol{\varepsilon}$ and $\boldsymbol{X} \sim \boldsymbol{N}(\boldsymbol{\mu},\boldsymbol{W}\boldsymbol{W}^T + \sigma^2\boldsymbol{I})$. Then the maximum likelihood estimates of $\boldsymbol{W}$ and $\boldsymbol{\sigma}$ can be obtained either in a closed form (Tipping and Bishop, 1999) or by the Expectation maximisation (EM) algorithm (Roweis, 1998). The EM version of PPCA has the advantage of reducing the computational complexity from $\boldsymbol{O}(np^2)$ to $\boldsymbol{O}(ndp)$, and is able to process missing data (Nyamundanda et al., 2010). Afterwards, a robust version of PPCA was developed. The Gaussian distribution in the regular PPCA was replaced by the student's t-distribution to achieve a

more robust model (Archambeau et al., 2006; Chen et al., 2009) Yu et al. proposed a supervised PPCA in 2006 (Yu et al., 2006) in which the response variables are considered as a linear transformation of the latent variables $\boldsymbol{Z}$ as well, and the estimate of parameters can be obtained by EM algorithm. In 2010, a novel extension of PPCA, called probabilistic principal component and covariates analysis (Nyamundanda et al., 2010) was introduced to extend the usage of PPCA to metabolomic data. This method provides a flexible approach to jointly model metabolomic data and additional covariate information. Also, PPCA has become a common technique in detecting abnormal events of chemical processes after Kim et al. introduced it to analytical chemistry in 2003 (Kim and Lee, 2003).

1.3.3 Discriminative PCA

Apart from the above adaptions, one more limitation of PCA has drawn considerable attention from researchers, and is a focus of this thesis. As an unsupervised method, PCA fails to use the class labels of the observations (Chen and Sun, 2005; Huang et al., 2015). As a result, when it is utilised as a dimension reduction technique in a supervised classification, its maximisation of variance of the projected patterns might not necessarily be in favour of discrimination among classes. This problem has been identified in many literatures (Chen and Sun, 2005; Huang et al., 2015; Fan et al., 2014; Qiu et al., 2012).

Solutions to this problem can be generally classified into three categories. The first category of approaches add a subspace-based pre-processing step before PCA. Data are firstly projected to a subspace which is more discriminative than the original feature space and then the classic PCA can be implemented on this discriminative subspace. This type of methods include discriminative common vectors PCA (Cevikalp et al., 2005), dimension reduction by orthogonal projection for discrimination (DROP-D) (Hadoux et al., 2015) and discriminative PCA (Qiao, 2019). In discriminative common vectors PCA, data are firstly projected to the null space

of $\boldsymbol{S}_W$ and the classic PCA is implemented. In DROP-D, data are projected to the null space of $\boldsymbol{S}_B$, then the within SSP matrix $\tilde{\boldsymbol{S}}_W$ of the projected data is calculated and the dominating directions in $\tilde{\boldsymbol{S}}_W$ are discarded from the original feature space by orthogonal projection. Then the classic PCA is implemented on the new subspace orthogonal to the dominating directions in $\tilde{\boldsymbol{S}}_W$. Here we use the tilde symbol to distinguish the within SSP matrix $\tilde{\boldsymbol{S}}_W$ of the projected data (projected on the null space of $\boldsymbol{S}_B$), with the within SSP matrix $\boldsymbol{S}_W$ of the raw data. While in the discriminative PCA proposed by Qiao in 2019 (Qiao, 2019) data are projected to the intersection of two subspaces, the range space of $\boldsymbol{S}_B$ and the small eigenvector subspace of $\boldsymbol{S}_W$. Here by small eigenvector subspace of $\boldsymbol{S}_W$ we mean the subspace spanned by the eigenvectors of $\boldsymbol{S}_W$ with small eigenvalues, and then the classic PCA is implemented on this intersection subspace. All these approaches are based on subspace method and orthogonal projection.

The second category of approaches enhances the discriminating power of PCA by combining supervised information into the feature generation procedure. The most straightforward way to do it is to include the label variable into the data matrix when implementing PCA (Chen and Sun, 2005). In robust discriminative PCA (Xu et al., 2018), the idea of ridge regression is incorporated in the feature generation of PCA. In this algorithm, features are generated with the target of simultaneously minimising the prediction error of ridge regression and the reconstruction error of PCA. Although this algorithm is not specialised for classification, it combines supervised information within the classic PCA framework. Eigenboosting by Grabner et al. (Grabner et al., 2007) tries to balance the discrimination power and generalisation power of features as well. Here the generalisation power is guaranteed by PCA while the discrimination power is measured by a perceptron-based model. Features providing a good balance between these two are generated.

The third category of approaches add a filter after the PC generation and re-rank PCs according to their discriminating power. In other words, in this type of

methods PCs are generated as usual, but the most discriminating ones are selected and included in the calibration model while the non-discriminating ones are discarded from the model. The discriminating power of a PC could be measured by the similarity between this PC and the mean difference direction in the PC subspace (Zhu and Martinez, 2006), or the similarity between this PC and the hyperplane separating vector of LDA and SVM in the PC subspace (Thomaz and Giraldi, 2010). For example, the hyperplane separating vector in LDA is the transformation vector $\boldsymbol{w}$ that maximises $\boldsymbol{w}^T \boldsymbol{S}'_B \boldsymbol{w}$ over $\boldsymbol{w}^T \boldsymbol{S}'_W \boldsymbol{w}$ where $\boldsymbol{S}'_B$ and $\boldsymbol{S}'_W$ are the between-group SSP matrix and within-group SSP matrix in the PC subspace (Thomaz and Giraldi, 2010). The prime symbol is used here to distinguish the above scatter matrices from the scatter matrices in the original feature space. Also, the discriminating power can be measured by the Fisher's discriminant ratio of different PCs (Huang et al., 2015) or more naively the group mean difference of the corresponding PC scores (Grabner et al., 2007).

1.3.4 Scaling in PCA

Scaling is another important topic in PCA. Rescaling all features for equal times will magnify or reduce the total covariance matrix by the same times. This will impact the magnitude of the covariance matrix and PCs, but not the direction of PCs. Rescaling a subset of features will influence not only the magnitude of the covariance matrix, but also the composition of it and will lead to totally different PCs. In the new PCs the magnified features will own higher weights, the reduced features will become own less weights.

Whether to conduct a rescaling before PCA depends on the data. If all variables in the data are measured on the same scale having the same unit, it may be a good idea not to rescale the variables. This is also the case of NIR. NIR spectroscopy measures the transmittance or reflectance against NIR wavelengths and all variables have the same scale. Thus in this thesis we do not do scaling before PCA.

On the other hand, if you have different types of variables with different units, it is probably wise to scale the data first. If some variables have scales very different from the others, the variables with significantly high numerical values will have more weights in the PCA because of their greater numerical values. In this case it is wise to do rescaling before implementing PCA. For example, log transformation is generally used to decrease the impact of dominant values.

1.4 Partial least squares discriminant analysis

1.4.1 Introduction to Partial Least Squares Regression and Partial Least Squares Discriminant Analysis

Partial least squares regression (PLSR) is a statistical tool in modelling the relationship between one or more response variables and multiple explanatory variables. It is particularly suited when the matrix $\boldsymbol{X}$ of predictors has more variables than observations, or when there is multicollinearity among the explanatory variables. Unlike PCA which constructs components that contain the highest variability of $\boldsymbol{X}$, it tries to find latent variables that describe as much the variation of the input variables as possible and simultaneously have maximal correlation with the target value in $\boldsymbol{Y}$ (Berrueta et al., 2007). PLS achieves this by maximising the covariance of the constructed components and the response variables. Partial least squares discriminant analysis (PLS-DA) is a widely applicable high-dimensional classification method based on PLSR, using response variables which are categorical instead of numerical. When dealing with c-class classification ($c > 2$), PLS-DA employs c dummy variables with entries 0 or 1 to represent labels of c classes. Specifically, if the i-th sample belongs to the j-th class, $y_{ij} = 1$ while all $y_{ik} = 0$ for all $1 \leqslant k \leqslant c$ and $k \neq j$. In the implementation of multigroup PLS-DA an arbitrary column of $\boldsymbol{Y}$ is used as the starting vector $\boldsymbol{u}$ to calculate the weight vector (see the NIPALS algorithm below for more details).

PLSR can be regarded as a generalisation of multiple linear regression (MLR). Traditionally, this modeling of $\boldsymbol{Y}$ by means of $\boldsymbol{X}$ is done using MLR, which works well as long as the $\boldsymbol{X}$-variables are few and uncorrelated (Wold et al., 2001). However, when the number of variables exceeds the number of observations and when there exists collinearity between variables, $\boldsymbol{X}^T\boldsymbol{X}$ is noninvertible and the collinearity will lead to inaccurate estimates of model parameters and accordingly poor prediction performance. In this case, PLSR is more used than MLR, as it can extract a few predictive factors from strongly correlated, noisy and numerous $\boldsymbol{X}$ variables and also simultaneously model the response variables (Wold et al., 2001).

There are many ways to implement PLS, including the nonlinear iterative partial least squares algorithm (NIPALS) by Wold, et al. (Wold et al., 1984), the non-orthogonalized scores algorithm by Martens, et al. (Martens and Naes, 1992), SIMPLS by De Jong (De Jong, 1993), etc. Here we only introduce the most widely applicable algorithm, NIPALS (Wold et al., 1984).

We consider the general case in which the response variable has more than one column. Assume the predictor variables $\boldsymbol{X}$ and the response variables $\boldsymbol{Y}$ have been transformed to have zero column means. Then the nonlinear iterative partial least squares algorithm (NIPALS) is as follows:

(A) Get a starting vector $\boldsymbol{u}$, usually one of the $\boldsymbol{Y}$ columns (with a single $\boldsymbol{y}$, $\boldsymbol{u} = \boldsymbol{y}$).

(B) The $\boldsymbol{X}$-weights, $\boldsymbol{w}$ can be calculated as:

$$\boldsymbol{w}_{old} = \frac{\boldsymbol{X}^T\boldsymbol{u}}{\boldsymbol{u}^T\boldsymbol{u}}, \tag{1.4.1}$$

and if $\|\cdot\|$ represents the L2-norm of a vector,

$$\boldsymbol{w} = \frac{\boldsymbol{w}_{old}}{||\boldsymbol{w}_{old}||}. \tag{1.4.2}$$

(C) Then the $\boldsymbol{X}$-scores, $\boldsymbol{t}$:

$$\boldsymbol{t} = \boldsymbol{X}\boldsymbol{w}. \tag{1.4.3}$$

(D) The $\boldsymbol{Y}$-weights, $\boldsymbol{q}$ can be calculated as:

$$\boldsymbol{q}_{old} = \frac{\boldsymbol{Y}^T\boldsymbol{t}}{\boldsymbol{t}^T\boldsymbol{t}}, \tag{1.4.4}$$

and then normalised as:

$$\boldsymbol{q} = \frac{\boldsymbol{q}_{old}}{||\boldsymbol{q}_{old}||}. \tag{1.4.5}$$

(E) An updated set of $\boldsymbol{Y}$-scores, $\boldsymbol{u}$ is calculated as:

$$\boldsymbol{u} = \frac{\boldsymbol{Y}\boldsymbol{q}}{\boldsymbol{q}^T\boldsymbol{q}}. \tag{1.4.6}$$

(F) Iterate from step B) to step E) until the component $\boldsymbol{t}$ in the next two adjacent iterations does not change significantly, i.e. $\frac{||\boldsymbol{t}_{old}-\boldsymbol{t}_{new}||}{||\boldsymbol{t}_{new}||} < \eta$, where $\boldsymbol{t}_{old}$ and $\boldsymbol{t}_{new}$ are the same components obtained from two adjacent iterations and η is a pre-defined small number. If this condition is satisfied, continue with step G).

(G) The loading vector $\boldsymbol{p}$ of $\boldsymbol{X}$ can be calculated as:

$$\boldsymbol{p}_{old} = \frac{\boldsymbol{X}^T\boldsymbol{t}}{\boldsymbol{t}^T\boldsymbol{t}}, \tag{1.4.7}$$

and normalised:

$$\boldsymbol{p} = \frac{\boldsymbol{p}_{old}}{||\boldsymbol{p}_{old}||}. \tag{1.4.8}$$

(H) Deflate the present component from $\boldsymbol{X}$ and $\boldsymbol{Y}$ and replace $\boldsymbol{X}$ and $\boldsymbol{Y}$ by the

deflated matrices $\boldsymbol{E}$ and $\boldsymbol{F}$ in the generation of the next PLS component, i.e.

$$\boldsymbol{E} = \boldsymbol{X} - \boldsymbol{t}\boldsymbol{p}^T, \tag{1.4.9}$$

$$\boldsymbol{F} = \boldsymbol{Y} - \boldsymbol{t}\boldsymbol{q}^T. \tag{1.4.10}$$

(I) Continue with the next component (back to step A) until the maximium number of components is reached, or cross-validation shows that more factors will not contribute to improving the prediction performance.

The matrices of $\boldsymbol{W}$, $\boldsymbol{T}$, $\boldsymbol{P}$ and $\boldsymbol{Q}$ can be obtained by combining the corresponding vectors in the algorithm. Then an explicit relationship between $\boldsymbol{X}$ and $\boldsymbol{Y}$ can be denoted as:

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{B} + \varepsilon, \tag{1.4.11}$$

where

$$\boldsymbol{B} = \boldsymbol{W}(\boldsymbol{P}^T\boldsymbol{W})^{-1}\boldsymbol{Q}^T, \tag{1.4.12}$$

and here ε is the error term.

Then for a new sample $\boldsymbol{x}$, the corresponding predictive value would be:

$$\hat{\boldsymbol{y}} = \boldsymbol{x}\boldsymbol{B}. \tag{1.4.13}$$

If used in binary classification, i.e. PLS-DA, then $\hat{\boldsymbol{y}}$ = 1 if $\boldsymbol{x}\boldsymbol{B} > 0$ and $\hat{\boldsymbol{y}}$ = -1 if $\boldsymbol{x}\boldsymbol{B} < 0$.

In summary, as a multivariate regression method PLSR has the advantage of coping with highly collinear and numerous predictor variables. Here the best number of components can be decided via cross-validation, with the target of minimising prediction error. Similarly, the discriminant analysis PLS-DA based on PLSR is also free from the constraint that the number of predictor variables should not exceed the number of observations. For PLS-DA the best number of component can also be

decided by cross-validation, with the target of maximising classification accuracy. Compared with the unsupervised dimension reduction method PCA, PLS has the advantage of taking the label information into consideration (Andrade-Garda et al., 2009).

1.4.2 The optimisation idea behind the NIPALS algorithm

NIPALS is the most extensively applied PLS algorithm which was proposed in 1984 by Wold et al. (Wold et al., 1984). It finds PLS factors by iterations and deflations. However, what NIPALS tries to optimise is not clear at first when it was proposed. In this section we will show a brief derivation, from which we can clearly see what optimisation problem NIPALS actually solves and how the NIPALS factors are alike (Frank and Friedman, 1993).

From step B) in the NIPALS algorithm,

$$\boldsymbol{w}_{old} = \frac{\boldsymbol{X}^T\boldsymbol{u}}{\boldsymbol{u}^T\boldsymbol{u}} \quad \text{and} \quad \boldsymbol{w} = \frac{\boldsymbol{w}_{old}}{||\boldsymbol{w}_{old}||}, \tag{1.4.14}$$

$$\text{then } ||\boldsymbol{w}_{old}|| = \frac{||\boldsymbol{X}^T\boldsymbol{u}||}{\boldsymbol{u}^T\boldsymbol{u}} \quad \text{and} \quad \boldsymbol{w} = \frac{\boldsymbol{w}_{old}}{||\boldsymbol{w}_{old}||} = \frac{\boldsymbol{X}^T\boldsymbol{u}}{\boldsymbol{u}^T\boldsymbol{u}} \cdot \frac{\boldsymbol{u}^T\boldsymbol{u}}{||\boldsymbol{X}^T\boldsymbol{u}||} = \frac{\boldsymbol{X}^T\boldsymbol{u}}{||\boldsymbol{w}^T\boldsymbol{u}||}. \tag{1.4.15}$$

From step C) we know,

$$\boldsymbol{t} = \boldsymbol{X}\boldsymbol{w}. \tag{1.4.16}$$

From step D),

$$\boldsymbol{q}_{old} = \frac{\boldsymbol{Y}^T\boldsymbol{t}}{\boldsymbol{t}^T\boldsymbol{t}} \quad \text{and} \quad \boldsymbol{q} = \frac{\boldsymbol{q}_{old}}{||\boldsymbol{q}_{old}||}, \tag{1.4.17}$$

$$\text{then } ||\boldsymbol{q}_{old}|| = \frac{||\boldsymbol{Y}^T\boldsymbol{t}||}{\boldsymbol{t}^T\boldsymbol{t}} \quad \text{and} \quad \boldsymbol{q} = \frac{\boldsymbol{q}_{old}}{||\boldsymbol{q}_{old}||} = \frac{\boldsymbol{Y}^T\boldsymbol{t}}{\boldsymbol{t}^T\boldsymbol{t}} \cdot \frac{\boldsymbol{t}^T\boldsymbol{t}}{||\boldsymbol{Y}^T\boldsymbol{t}||} = \frac{\boldsymbol{Y}^T\boldsymbol{t}}{||\boldsymbol{Y}^T\boldsymbol{t}||}. \tag{1.4.18}$$

From step E) we know,

$$\boldsymbol{u} = \frac{\boldsymbol{Y}\boldsymbol{q}}{\boldsymbol{q}^T\boldsymbol{q}} = \boldsymbol{Y}\boldsymbol{q}, \tag{1.4.19}$$

as $\boldsymbol{q}$ is of norm 1. When the convergence is obtained, $\boldsymbol{w}$, $\boldsymbol{u}$, $\boldsymbol{t}$ are $\boldsymbol{q}$ are stable. Then

we can substitute (1.4.16), (1.4.18) and (1.4.19) into (1.4.15),

$$\boldsymbol{w} = \frac{\boldsymbol{X}^T\boldsymbol{u}}{||\boldsymbol{w}^T\boldsymbol{u}||} = \frac{\boldsymbol{X}^T\boldsymbol{Y}\boldsymbol{q}}{||\boldsymbol{w}^T\boldsymbol{Y}\boldsymbol{q}||} = \frac{\boldsymbol{X}^T\boldsymbol{Y}\cdot\frac{\boldsymbol{Y}^T\boldsymbol{t}}{||\boldsymbol{Y}^T\boldsymbol{t}||}}{||\boldsymbol{w}^T\boldsymbol{Y}\cdot\frac{\boldsymbol{Y}^T\boldsymbol{t}}{||\boldsymbol{Y}^T\boldsymbol{t}||}||} = \frac{\boldsymbol{X}^T\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{t}}{||\boldsymbol{X}^T\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{t}||} = \frac{\boldsymbol{X}^T\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{X}\boldsymbol{w}}{||\boldsymbol{X}^T\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{X}\boldsymbol{w}||} \tag{1.4.20}$$

Assume $\lambda = ||\boldsymbol{X}^T\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{X}\boldsymbol{w}||$, then

$$\boldsymbol{w} = \frac{1}{\lambda}\boldsymbol{X}^T\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{X}\boldsymbol{w} \quad \text{or} \quad \lambda\boldsymbol{w} = \boldsymbol{X}^T\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{X}\boldsymbol{w}. \tag{1.4.21}$$

Here $\boldsymbol{w}$ is the eigenvector of $\boldsymbol{X}^T\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{X}$. In fact, (1.4.21) is exactly the update rule in the Power method used for computing the largest eigenvalue-eigenvector pair for the symmetric matrix $\boldsymbol{X}^T\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{X}$ (Watkins, 2004). In other words, the loading vector $\boldsymbol{w}$ found by NIPALS is the eigenvector corresponding to the largest eigenvalue of $\boldsymbol{X}^T\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{X}$.

Note that the eigenvector of $\boldsymbol{X}^T\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{X}$ corresponding to its largest eigenvalue is the solution to this optimisation problem:

$$\begin{aligned}
&\arg\max_{||\boldsymbol{w}||=1} \boldsymbol{w}^T\boldsymbol{X}^T\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{X}\boldsymbol{w} \\
=&\arg\max_{||\boldsymbol{w}||=1} (\boldsymbol{Y}^T\boldsymbol{X}\boldsymbol{w})^T(\boldsymbol{Y}^T\boldsymbol{X}\boldsymbol{w}) \\
=&\arg\max_{||\boldsymbol{w}||=1} (\mathrm{Cov}(\boldsymbol{Y},\boldsymbol{X}\boldsymbol{w}))^T(\mathrm{Cov}(\boldsymbol{Y},\boldsymbol{X}\boldsymbol{w})).
\end{aligned} \tag{1.4.22}$$

Hence, NIPALS finds the loading vector $\boldsymbol{w}$ that maximises the covariance between the PLS factor $\boldsymbol{t} = \boldsymbol{X}\boldsymbol{w}$ and $\boldsymbol{Y}$.

1.4.3 PLS as a Discriminative Dimension Reduction Technique

Though both aim at extracting informative features from high-dimensional data, PLS is generally regarded as a more applicable dimension reduction technique in classification task compared with PCA (Berrueta et al., 2007). One obvious reason is that PCA and PLS have different goals in the feature generation. PLS aims at

maximising the covariance between $\boldsymbol{y}$ and the generated feature while PCA aims at maximising the variance of the feature itself. In other words, PLS makes better use of the label information than PCA and thus can generate features with higher correlation with the response variable. According to Friedman et al. (2001b), this can be easily shown:

In PCA, the m-th principal direction $\boldsymbol{v}_m$ solves:

$$\begin{aligned} &\max_{\boldsymbol{v}_m} \mathrm{Var}(\boldsymbol{X}\boldsymbol{v}_m), \\ &\text{subject to } ||\boldsymbol{v}_m|| = 1 \text{ and } \boldsymbol{v}_m^T \boldsymbol{S} \boldsymbol{v}_l = 0, \text{ for } l = 1, 2, \ldots, m-1. \end{aligned} \tag{1.4.23}$$

Here $\boldsymbol{S}$ is the sample covariance matrix of data. The condition $\boldsymbol{v}_m^T \boldsymbol{S} \boldsymbol{v}_l = 0$ ensures that the m-th feature $\boldsymbol{t}_m = \boldsymbol{X}\boldsymbol{v}_m$ is uncorrelated with all the previous features $\boldsymbol{t}_l = \boldsymbol{X}\boldsymbol{v}_l$, for $l = 1, 2, \ldots, m-1$, note that:

$$\begin{aligned} (\boldsymbol{t}_m - \bar{\boldsymbol{t}}_m)^T (\boldsymbol{t}_l - \bar{\boldsymbol{t}}_l) &= (\boldsymbol{t}_m - \bar{\boldsymbol{x}}^T \boldsymbol{v}_m)^T (\boldsymbol{t}_l - \bar{\boldsymbol{x}}^T \boldsymbol{v}_l) = (\boldsymbol{t}_m - 0)^T (\boldsymbol{t}_l - 0) = \boldsymbol{t}_m^T \boldsymbol{t}_l \\ &= (\boldsymbol{X}\boldsymbol{v}_m)^T (\boldsymbol{X}\boldsymbol{v}_l) = \boldsymbol{v}_m^T \boldsymbol{X}^T \boldsymbol{X} \boldsymbol{v}_l = (n-1) \boldsymbol{v}_m^T \boldsymbol{S} \boldsymbol{v}_l = 0, \end{aligned} \tag{1.4.24}$$

where $\bar{\boldsymbol{x}}$ is the column mean of $\boldsymbol{X}$. Since in PCA $\boldsymbol{X}$ is mean-centred, $\bar{\boldsymbol{x}} = 0$.

While in PLS with univariate $\boldsymbol{y}$, the m-th PLS direction $\boldsymbol{w}_m$ solves:

$$\begin{aligned} &\max_{\boldsymbol{w}_m} \ \mathrm{Cov}(\boldsymbol{y}, \boldsymbol{X}\boldsymbol{w}_m), \\ \Leftrightarrow &\max_{\boldsymbol{w}_m} \sqrt{\mathrm{Var}(\boldsymbol{X}\boldsymbol{w}_m)} \mathrm{Corr}(\boldsymbol{X}\boldsymbol{w}_m, \boldsymbol{y}) \sqrt{\mathrm{Var}(\boldsymbol{y})}, \\ \Leftrightarrow &\max_{\boldsymbol{w}_m} \sqrt{\mathrm{Var}(\boldsymbol{X}\boldsymbol{w}_m)} \mathrm{Corr}(\boldsymbol{X}\boldsymbol{w}_m, \boldsymbol{y}), \\ \Leftrightarrow &\max_{\boldsymbol{w}_m} \mathrm{Var}(\boldsymbol{X}\boldsymbol{w}_m) \mathrm{Corr}^2(\boldsymbol{y}, \boldsymbol{X}\boldsymbol{w}_m). \\ &\text{subject to } ||\boldsymbol{w}_m|| = 1 \text{ and } \boldsymbol{w}_m^T \boldsymbol{S} \boldsymbol{w}_l = 0, \text{ for } l = 1, \ldots, m-1. \end{aligned} \tag{1.4.25}$$

Here the constraint $\boldsymbol{w}_m^T \boldsymbol{S} \boldsymbol{w}_l = 0$ guarantees that the m-th PLS score $\boldsymbol{t}_m$ is orthogonal to the previous scores as:

$$\boldsymbol{t}_m^T \boldsymbol{t}_l = (\boldsymbol{X}\boldsymbol{w}_m)^T \boldsymbol{X}\boldsymbol{w}_l = \boldsymbol{w}_m^T (\boldsymbol{X}^T \boldsymbol{X}) \boldsymbol{w}_l = (n-1) \boldsymbol{w}_m^T \boldsymbol{S} \boldsymbol{w}_l = 0, \tag{1.4.26}$$

for $l = 1, \ldots, m-1$. However unlike PCA, here the orthogonality between $\boldsymbol{w}_m$ and $\boldsymbol{w}_l$ cannot be guaranteed. As mentioned in the last section, there are other versions of PLS with different constraints. In some other algorithms, for example, in the non-orthogonalised scores PLS algorithm (Martens and Naes, 1992), the loading vectors $\boldsymbol{w}_m$ and $\boldsymbol{w}_l$ are orthogonal, but the scores $\boldsymbol{t}_m$ and $\boldsymbol{t}_l$ are non-orthogonal. Unlike PCA, PLS cannot have both orthogonal loadings and uncorrelated (or orthogonal) scores. It needs to choose one from orthogonal scores and orthogonal loadings.

Comparing formula (1.4.14) with (1.4.25), the feature generation criterion of PLS penalises the criterion of PCA with a correlation term $\text{Corr}^2(\boldsymbol{y}, \boldsymbol{X}\boldsymbol{w}_m)$. Thus the generated PLS features will have stronger power in predicting the response variable, i.e. the class label in classification task. Then the PLS factors are in general regarded as higher discriminating power than PCs.

The discriminating ability of PLS can be more clearly seen by analysing the direction of the first PLS component in classification. Here for simplicity we assume a binary classification task with balanced sample size, $n_1 = n_2 = \frac{1}{2}n$ and $y_i = 1$ if sample i belongs to class 1 and $y_i = -1$ if sample i belongs to class 2.

Firstly, the data can be centred by:

$$\boldsymbol{X} = \boldsymbol{X}_0 - \boldsymbol{h}\bar{\boldsymbol{x}}_0, \tag{1.4.27}$$

$$\boldsymbol{y} = \boldsymbol{y}_0 - \boldsymbol{h}\bar{\boldsymbol{y}}_0. \tag{1.4.28}$$

Here $\boldsymbol{X}_0$ and $\boldsymbol{y}_0$ are the original uncentred data, $\bar{\boldsymbol{x}}_0$ contains the column mean of $\boldsymbol{X}_0$ while $\bar{\boldsymbol{y}}_0$ is the mean of $\boldsymbol{y}$, $\boldsymbol{h} \in \mathbb{R}^{n \times 1}$ is a column vector of 1s. Note that under our

setting $\boldsymbol{y}_0$ already has zero mean. Nevertheless, to follow the classic PLS framework and without loss of generality, here we still implement the centralisation of $\boldsymbol{y}_0$.

The first loading vector $\boldsymbol{w}_1$ maximises the sample covariance of the first feature $\boldsymbol{t}_1 = \boldsymbol{X}\boldsymbol{w}_1$ and the centred $\boldsymbol{y}$. The sample covariance of a mean-centred $n \times m$ matrix $\tilde{\boldsymbol{X}}$ and a mean centred $n \times p$ matrix $\tilde{\boldsymbol{y}}$ is defined as:

$$\mathrm{Cov}(\tilde{\boldsymbol{X}}, \tilde{\boldsymbol{y}}) = \frac{1}{n-1}\tilde{\boldsymbol{X}}^T\tilde{\boldsymbol{y}}. \tag{1.4.29}$$

Here $\boldsymbol{X}\boldsymbol{w}_1$ is a mean-centred $n \times 1$ vector and $\boldsymbol{y}$ is a mean centred $n \times 1$ vector, then:

$$\begin{aligned}
&\max_{\boldsymbol{w}_1} \mathrm{Cov}(\boldsymbol{X}\boldsymbol{w}_1, \boldsymbol{y}), \\
\Leftrightarrow &\max_{\boldsymbol{w}_1} \frac{1}{n-1}(\boldsymbol{X}\boldsymbol{w}_1)^T\boldsymbol{y}, \\
\Leftrightarrow &\max_{\boldsymbol{w}_1} \boldsymbol{w}_1^T\boldsymbol{X}^T\boldsymbol{y}, \\
\Leftrightarrow &\max_{\boldsymbol{w}_1} \boldsymbol{w}_1^T(n_1\bar{\boldsymbol{x}}_1 - n_2\bar{\boldsymbol{x}}_2), \\
\Leftrightarrow &\max_{\boldsymbol{w}_1} \boldsymbol{w}_1^T(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2), \\
\Leftrightarrow &\max_{\boldsymbol{w}_1} (\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)^T\boldsymbol{w}_1.
\end{aligned} \tag{1.4.30}$$

Here $\bar{\boldsymbol{x}}_1$ and $\bar{\boldsymbol{x}}_2$ are the column vectors containing the column mean of the two classes. Aiming at maximising $\mathrm{Cov}(\boldsymbol{y}^T, \boldsymbol{X}\boldsymbol{w}_1)$, $\boldsymbol{w}_1$ has to be in the same direction as $(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)$ with norm 1. Namely,

$$\boldsymbol{w}_1 = \frac{(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)}{(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)^T(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)}. \tag{1.4.31}$$

Then the first PLS factor,

$$\boldsymbol{t}_1 = \boldsymbol{X}\boldsymbol{w}_1. \tag{1.4.32}$$

By simple algebra we can find, to maximise covariance between $\boldsymbol{t}$ and $\boldsymbol{y}$, the first PLS projection direction $\boldsymbol{w}_1$ needs to be in the mean difference direction. This

mean difference direction contains information about different group means and it often can contribute to the separation of the two classes. Thus PLS is generally regarded as of higher discriminating power than PCA. Nevertheless, this is not always true. It was shown in many real examples that PCA-based models gave more accurate classification than PLS-based models, (Khan et al., 2018; Sampson et al., 2011) and it was shown in many other literatures that if the label information can be combined in the PCA algorithm, even better classification can be obtained (Yu et al., 2006; Perez and Narasimhan, 2018). This is the reason why we want to develop discriminative PCA algorithms.

1.4.4 Nonlinear PLS

PLSR is an extension of MLR to cope with high dimensional and highly collinear data. It builds a linear relationship for the feature $\boldsymbol{t}$ generated from $\boldsymbol{X}$ and the feature $\boldsymbol{u}$ from $\boldsymbol{Y}$. An important extension to the classic PLS algorithm is not to simply consider a linear relationship, but also take into account the nonlinear relationship among variables.

One of the most straightforward way to build nonlinear PLS is to expand the data matrices by nonlinear terms (such as quadratic terms, cubic terms, logarithms, etc.) and then apply the classic linear PLS framework in the nonlinear data frame. Many nonlinear PLS algorithms have been developed based on this idea in the past three decades (Wold et al., 1989; Berglund and Wold, 1997; Baffi et al., 1999; Verdun et al., 2012). Several quadratic PLS algorithms have been built by integrating quadratic features within the linear framework (Baffi et al., 1999; Mejdell and Skogestad, 1991). For example, the input matrix can be extended by including second order terms of the original variables (square terms and interactions) and then a linear PLS can be performed on the extended input and output matrices (Wold et al., 1989). This approach can be generalised by applying quadratic transformations to both the predictor and the predicted variables (Mejdell and Skogestad, 1991).

However, the number of nonlinear terms increases excessively with the dimension of variables and the results become difficult to compute and interpret (Baffi et al., 1999). Afterwards, this type of methods has been further developed by employing the kernel trick. The original data space can be expanded to an even higher dimensional feature space by various kernel functions and the classic linear PLS algorithm is implemented on this high-dimensional feature space (Rosipal et al., 2003; Rosipal and Trejo, 2001).

Another way to develop nonlinear PLS models is to introduce a nonlinear function between the predictor and the response latent variables without modifying the input and output matrices. Wold et al. (Wold et al., 1989) proposed a polynomial PLS algorithm which modified the relationship between the output scores $\boldsymbol{u}$ and the input scores $\boldsymbol{t}$ to be polynomial. Wold then went on to propose a SPLINE-PLS algorithm where a smooth bivariate spline function (quadratic or cubic) was used to fit the non-linear mapping between each pair of latent variables (Wold, 1992).

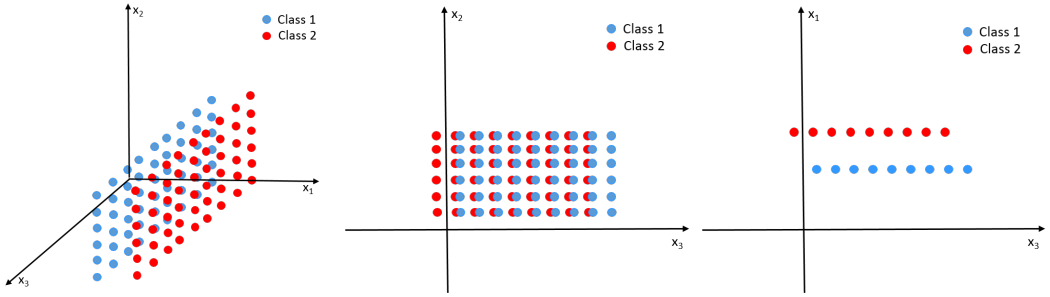
The idea behind these nonlinear PLS algorithms is that, a linear function is sometimes not sufficient to model the relationship between the predictors and the response variables. This idea, especially the idea of quadratic PLS, has given rise to our penalised QDA algorithm, which will be discussed in detail in Chapter 3.

Chapter 2

Reweighted PCA and Reordered PCA

2.1 Introduction

As discussed in the introductory chapter the classic PCA has many limitations, such as its sensitivity to the outliers and deficiency in constructing nonlinear features. Apart from these, one limitation of PCA has drawn considerable attention from researchers. As an unsupervised method, PCA fails to use the class labels of the observations (Chen and Sun, 2005; Huang et al., 2015). As a result, when it is utilised as a dimension reduction technique in a classification, its maximization of variance of the projected patterns might not necessarily be in favour of discrimination among classes. If the total variation is mainly caused by the difference between different classes, the generated PCs will work well for classification. However, if the total variability is mainly caused by the variance within classes, then these features may not be useful in classification (Fan et al., 2014). This problem has been identified in many literatures (Chen and Sun, 2005; Huang et al., 2015; Qiu et al., 2012; Fan et al., 2014) and can be easily illustrated by the following example. Figure 2.1.1 is an illustrative example showing that large variability does not always relate to good discriminability. Assume the blue spots and the red spots are the two classes in the



(a) Data in 3 dimensions **(b)** Projections onto x_2 and x_3 **(c)** Projections onto x_1 and x_3

Figure 2.1.1. An illustrative example where PCA fails to extract discriminative features

3-dimensional space (see Figure 2.1.1 (a)), and our target is to extract features to efficiently separate the two classes. Most of the variability is in the x_2 & x_3 plane, and the first two PCs will be dominated by these variables. However the direction needed for discrimination is x_1.

There are various kinds of approaches to avoid this problem. One straightforward idea would be instead of using PCA to extract features, supervised feature extraction methods can be employed. This kind of approach is widely used in pattern recognition, especially, face recognition (Ye and Xiong, 2006; Bhele and Mankar, 2012). As discussed in the last chapter, the most well-known supervised feature extraction method is LDA. However, LDA fails to work when the number of variables exceeds the sample size. Therefore, extensions of LDA to deal with small sample size (SSS) issues have been investigated in many literatures (Sharma and Paliwal, 2015b; Yu and Yang, 2001; Chen et al., 2000; Ye, 2005). Among them RDA, OLDA and NLDA are the best-known ones (Ye and Xiong, 2006). They can be used as feature extraction methods in the SSS case. However, it was shown in some real data experiments that using variants of LDA to directly extract features has led to inferior performance compared with applying the conventional LDA after dimension reduction with PCA (Prasad et al., 2010). Although dimension reduction with PCA might fail to use the supervised information to extract discriminative features, it is

conducive to noise reduction and is still a commonly used step in high-dimensional classification.

The second kind of solutions are implemented by combining the supervised information in the feature extraction step of PCA, so as to generate more appropriate features for discrimination. As PCA is based on eigendecompositon of the sample covariance matrix, Chen and Sun suggested to put the class label at the end of sample matrix as the additional dimension and conducted eigendecomposition on the new covariance matrix (Chen and Sun, 2005). Similar to NLDA, Hadoux et al. identified a subspace of within-group variation that was orthogonal to the between-group variation, and projected data orthogonal to this subspace before implementing PCA (Hadoux et al., 2015). They claimed the between-group variation was always important in classification while the within-group variation did not contain much discriminative information. By projecting data orthogonal to the within-group variation subspace, they excluded some dimensions in the within-group variation that were unhelpful in discrimination. This method is most appropriate for LDA, which takes no account of the difference of within-group variation of different groups. However, it can be argued that the within-group variation can contain discriminative information as well. Totally excluding it from the model may have the risk of losing discriminative information. This can be illustrated in the following example.

Figure 2.1.2 is the scatter plot of two classes in a 3-dimensional space. The blue spots and the red spots represent samples from the two classes respectively. As shown, the blue class mainly varies in x_1 direction while the red class mainly varies in x_2 direction. The two classes show a slight mean difference in x_3 direction. In binary classification, the between-class variation is the variance of two group means and it is significant in classification, while x_1 and x_2 only contain within-group variation and are orthogonal to the between-class variation. Hadoux et al. suggested that these two dimensions should be discarded from the feature space. However, in this illustrative example the directions x_1 and x_2 expose very different

within-class variation structures of the two classes. Class 1 varies in the x_1 direction while class 2 varies in the x_2 direction. Most of the samples can be categorised into the correct group using information provided by x_1 and x_2 dimensions. In other words, although x_1 and x_2 mainly describe within-group variability and are orthogonal to the between-group variation, they have considerable discriminative power as well. If there is noise in the x_3 direction this additional within-group information can be very valuable. Accordingly, discarding the whole within-group variation may lose some discriminant information.

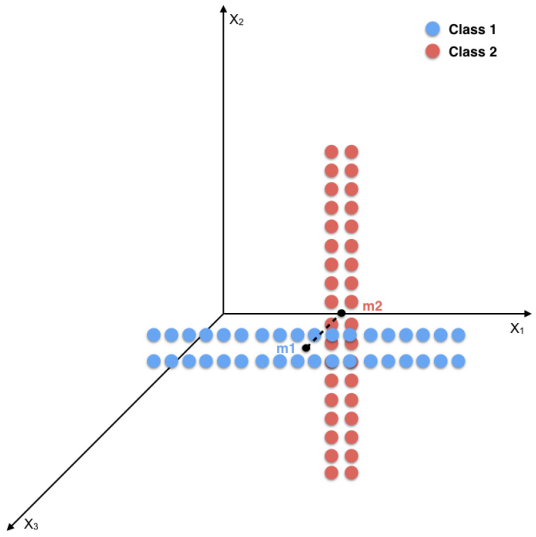


Figure 2.1.2. An illustrative example showing the discriminatory power of within-group variation

In the c-class classification, the total covariance can be decomposed as the weighted sum of a between-group covariance matrix and c within-group covariance matrices (Krzanowski, 2000). The between-group covariance is the variance of the group means. It reflects location difference of the groups and is usually discriminative. In terms of dimensions in the within-group covariance, they are non-discriminative if they correspond to the common variation of different groups and are discriminative if they correspond to the distinct characteristic of an individual class (Fan et al., 2014). This has led to the development of our reweighting PCA algorithms. As shown in the first illustrative example, the total covariance matrix

can be dominated by the within-group covariance, especially when the data are of high dimension. Accordingly, the information contained in the between-class covariance may not appear in early PCs. Based on this idea, we consider highlighting the discriminative between covariance over the within covariance by giving it higher weight. Similarly, the weights of the within-group covariance matrices can be modified as well. They are naturally weighted by the sample size of each group. However, these may not be the best weights for discrimination. The within covariance that can describe distinct variation structure would ideally be highlighted over those that mainly describe common variations. Based on the above idea, we propose two reweighting algorithms, one called reweighted PCA, in which the c+1 covariance matrices are all reweighted by introducing c weight parameters. The second algorithm is called between-PCA, in which dimensions from between-group covariance are extracted first, and then the c within-group covariances are reweighted. In both cases the best weight can be obtained by cross-validation. These two algorithms will be introduced in detail in the next section.

Apart from replacing PCA with a supervised feature extraction method or modifying the feature extraction step of PCA, the third kind of approach is to generate PCs as the classic algorithm does, but select the discriminative ones from the generated ones. This approach is implemented by adding a feature filtering step to the classic PCA and it is very easy to apply. The key step in this kind of approach is how to measure the discriminative power of PCs. Huang et al. (2015) proposed to re-rank PCs according to the Fisher criterion. In other words, they used the idea of LDA to evaluate the discriminatory power of PCs. However as we said, LDA takes no account of the difference of within-group variation and accordingly neglects the discriminative information contained in the within covariances. Compared with LDA, QDA can better recognise the difference of within variation and identify the corresponding discriminative features. In this thesis, we propose two QDA based reordered PCA algorithms, in which the generated PCs are ranked

by the classification performance of QDA using the corresponding PCs as the new predictors. PCs corresponding to high classification accuracy are considered more discriminative and then selected into the model. Compared with the LDA-based reordered PCA algorithm that Huang, et al proposed in 2015, our algorithms have three improvements: 1) We use QDA to select PCs instead of LDA, which can make better use of the discriminative information in the within covariance. 2) A cut-off number q is set to avoid the influence of noise and only the first q PCs are taken into reordering instead of using all the features. The most discriminative k PCs out of q are selected. Here both q and k can be tuned via cross-validation. 3) We allow the PCs to be selected individually, or jointly in a stepwise manner.

In summary, in this chapter we propose four PCA-based discriminative dimension reduction methods, reweighted PCA, between PCA, reordered PCA and stepwise-reordered PCA, to remedy the deficiency of the classic PCA algorithm in high-dimensional classification problems. The full algorithms will be discussed in detail in the next section.

2.2 Methodologies

2.2.1 Decomposition of the total covariance matrix

As shown in, for example, Krzanowski (2000) if data can be categorised into multiple groups the total sample covariance matrix can be decomposed as the sum of within-group covariances and between-group covariances of different group means.

Suppose our data contain c groups and n samples in total, n_i is the number of data examples $\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}, \ldots, \boldsymbol{x}_{in_i}$ in group i (where $i \leqslant c$ and $n = \sum_{i=1}^{c} n_i$), and $\bar{\boldsymbol{x}}_i$ is a column vector containing the mean of group i. Then the within-group covariance $\boldsymbol{S}_i$ of class i can be denoted as:

$$\boldsymbol{S}_i = \frac{1}{n_i - 1} \sum_{j=1}^{n_i} (\boldsymbol{x}_{ij} - \bar{\boldsymbol{x}}_i)(\boldsymbol{x}_{ij} - \bar{\boldsymbol{x}}_i)^T, \tag{2.2.1}$$

where the superscript T denotes transpose of the vector.

Assume $\bar{\boldsymbol{x}}$ to be the mean vector of all data examples, then the between-class sum-of-squares and products matrix,

$$\boldsymbol{S}_B = \sum_{i=1}^{c} n_i(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})^T. \tag{2.2.2}$$

As shown in Krzanowski (2000), the total sample covariance $\boldsymbol{S}_T$ can be decomposed as follows:

$$\boldsymbol{S}_T = \frac{1}{n-1}(\sum_{i=1}^{c}(n_i - 1)\boldsymbol{S}_i + \boldsymbol{S}_B). \tag{2.2.3}$$

The full derivation is shown in the Appendix A.

Assume $\boldsymbol{S}_i^{'} = \frac{n_i-1}{n-1}\boldsymbol{S}_i$ and $\boldsymbol{S}_B^{'} = \frac{1}{n-1}\boldsymbol{S}_B$, then:

$$\boldsymbol{S}_T = \sum_{i=1}^{c} \boldsymbol{S}_i^{'} + \boldsymbol{S}_B^{'}. \tag{2.2.4}$$

Note that as shown in the first chapter, PCA can then be implemented via eigen-decomposition of this total covariance matrix.

2.2.2 Reweighting algorithms

2.2.2.1 Reweighted principal component analysis (Reweighted PCA)

As shown in section (1.2.1), in binary classification the between-group covariance $\boldsymbol{S}_B^{'} = \frac{1}{n-1}\boldsymbol{S}_B = \frac{n_1 n_2}{n(n-1)}(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)^T$ contains information about different group means. Here $\boldsymbol{S}_B^{'}$ is a matrix with rank 1 and the only nonzero eigenvector of $\boldsymbol{S}_B^{'}$ is the unit vector in the mean difference direction $(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)$. As we discussed, the mean difference direction is usually conducive to distinguishing the two classes. In other words, in binary classification the between-group covariance $\boldsymbol{S}_B^{'}$ contains important information discriminating the two classes and should be attached more emphasis.

While in multi-class classification the between-class sum-of-squares and products matrix:

$$\begin{aligned}
\boldsymbol{S}_B &= \sum_{i=1}^{c} n_i(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})^T \\
&= \sum_{i=1}^{c} n_i(\bar{\boldsymbol{x}}_i\bar{\boldsymbol{x}}_i^T - \bar{\boldsymbol{x}}\bar{\boldsymbol{x}}_i^T - \bar{\boldsymbol{x}}_i\bar{\boldsymbol{x}}^T + \bar{\boldsymbol{x}}\bar{\boldsymbol{x}}^T) \\
&= \sum_{i=1}^{c} n_i\bar{\boldsymbol{x}}_i\bar{\boldsymbol{x}}_i^T - \bar{\boldsymbol{x}}\sum_{i=1}^{c} n_i\bar{\boldsymbol{x}}_i^T - (\sum_{i=1}^{c} n_i\bar{\boldsymbol{x}}_i)\bar{\boldsymbol{x}}^T + \sum_{i=1}^{c} n_i\bar{\boldsymbol{x}}\bar{\boldsymbol{x}}^T \\
&= \sum_{i=1}^{c} n_i\bar{\boldsymbol{x}}_i\bar{\boldsymbol{x}}_i^T - n\bar{\boldsymbol{x}}\bar{\boldsymbol{x}}^T \\
&= \sum_{i=1}^{c} n_i\bar{\boldsymbol{x}}_i\bar{\boldsymbol{x}}_i^T - \frac{1}{n}\big(\sum_{i=1}^{c} n_i\bar{\boldsymbol{x}}_i\big)\cdot\big(\sum_{i=1}^{c} n_i\bar{\boldsymbol{x}}_i^T\big) \\
&= \frac{1}{n}\Big(\sum_{i=1}^{c-1}\sum_{j=(i+1)}^{c} n_i n_j(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}}_j)(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}}_j)^T\Big),
\end{aligned} \tag{2.2.5}$$

also contains the discriminative mean difference information and the between-group covariance $\boldsymbol{S}_B^{'} = \frac{1}{n-1}\boldsymbol{S}_B$ should also be emphasized. In other words higher weight is expected to attach to this between-group covariance over the within-covariance. Similarly, to emphasize the distinct variation pattern of an individual class, it is useful to attach different weights to different within covariance matrices.

Accordingly, we introduce c parameters to (2.2.4) to modify the relative weight of these $c+1$ covariance matrices. Here α is used to modify the weight of between covariance relative to the pooled within covariance while β_i ($i \leqslant c-1$) is used to change the relative weights of the within covariance matrices.

Then the reweighted total sample covariance matrix $\boldsymbol{S}_T^{'}$ can be denoted as:

$$\boldsymbol{S}_T^{'} = (1-\alpha)\boldsymbol{S}_W^{'} + \alpha\boldsymbol{S}_B^{'}, \tag{2.2.6}$$

where

$$\boldsymbol{S}_W^{'} = \beta_1\boldsymbol{S}_1^{'} + \Sigma_{l=2}^{c-1}\Pi_{s=1}^{l-1}(1-\beta_s)\beta_l\boldsymbol{S}_l^{'} + \Pi_{i=1}^{c-1}(1-\beta_i)\boldsymbol{S}_c^{'}, \tag{2.2.7}$$

is the reweighted pooled within covariance.

Here all the weight parameters α and β's can take values from 0 to 1. The larger α is, the more importance is attached to the between covariance $\boldsymbol{S}_B^{'}$. The larger β_i is, the more highlighted is $\boldsymbol{S}_i^{'}$ over the other $\boldsymbol{S}_j^{'}$'s for all $j \neq i$. Note that in the c-class case, the natural weighting is α=$\frac{1}{c+1}$ and β_i = $\frac{1}{c+1-i}$. With this weight, the reweighted total sample covariance $\boldsymbol{S}_T^{'}$ is equivalent to the original sample covariance matrix $\boldsymbol{S}_T$ up to a scalar. This can be easily shown as follows. When α=$\frac{1}{c+1}$ and β_i = $\frac{1}{c+1-i}$ $(i = 1,2,\ldots,c-1)$, the reweighted total covariance

$$\begin{aligned}
\boldsymbol{S}_T^{'} =& (1-\frac{1}{c+1})\boldsymbol{S}_W^{'} + \frac{1}{c+1}\boldsymbol{S}_B^{'} \\
=& \frac{c}{c+1}\left(\frac{1}{c}\boldsymbol{S}_1^{'} + \Sigma_{l=2}^{c-1}\left(\Pi_{s=1}^{l-1}\frac{c-s}{c+1-s}\right)\frac{1}{c+1-l}\boldsymbol{S}_l^{'}\right. \\
&\left. + \left(\Pi_{i=1}^{c-1}\frac{c-i}{c+1-i}\right)\boldsymbol{S}_c^{'}\right) + \frac{1}{c+1}\boldsymbol{S}_B^{'} \\
=& \frac{1}{c+1}\sum_{i=1}^{c}\boldsymbol{S}_i^{'} + \frac{1}{c+1}\boldsymbol{S}_B^{'} \\
\propto& \sum_{i=1}^{c}\boldsymbol{S}_i^{'} + \boldsymbol{S}_B^{'} = \boldsymbol{S}_T.
\end{aligned} \tag{2.2.8}$$

As shown, if α=$\frac{1}{c+1}$ and β_i = $\frac{1}{c+1-i}$ $(i = 1,2,\ldots,c-1)$ the reweighted total covariance $\boldsymbol{S}_T^{'}$ is proportional to the origianl covariance $\boldsymbol{S}_T$. As eigendecomposition will not be affected by the magnitude of proportional matrices, identical eigenvectors will be extracted for $\boldsymbol{S}_T^{'}$ and $\boldsymbol{S}_T$. Namely identical PC directions will be generated before and after reweighting. Accordingly, these weights can be regarded as a baseline to compare with. If running as a dimension reduction before classification, the best weights can be found via cross-validation with the target of minimising classification error rate in the cross-validation.

Here as all the weight parameters are bounded in [0,1], one naive idea could be finding the best weights by grid search. However grid search is extremely time-consuming in practice. Instead we use the Nelder-Mead simplex optimisation

method (Nelder and Mead, 1965) to find the optimal value for α and β_i's. However, as Nelder-Mead is an unconstrained method, a large loss is added to the objective function when any α or β goes beyond the range [0,1].

With each combination of α and β's we could reconstruct a total covariance matrix according to equation (2.2.6) and (2.2.7), then the reweighted PCA can proceed by carrying out eigendecomposition on the reweighted total sample covariance matrix $\boldsymbol{S}_T^{'}$ and then projecting data to the subspace with lower dimension as the classic PCA does. Note that the reweighted sample covariance matrix can be decomposed as:

$$\boldsymbol{S}_T^{'} = \boldsymbol{V}^{'}\boldsymbol{\Sigma}^{'}(\boldsymbol{V}^{'})^T, \tag{2.2.9}$$

where $\boldsymbol{V}^{'} \in \mathbb{R}^{p\times p}$ contains the eigenvectors of the reweighted total covariance $\boldsymbol{S}_T^{'}$ as its columns and $\boldsymbol{\Sigma}^{'}$ contains eigenvalues of $\boldsymbol{S}_T^{'}$, $\sigma_1^{'} \geqslant \sigma_2^{'} \geqslant \ldots \geqslant \sigma_p^{'}$. The first k PC loadings can be selected as usual and data can be projected to the subspace of lower dimension:

$$\boldsymbol{Z}_k^{'} = \boldsymbol{X}^c\boldsymbol{V}_k^{'}. \tag{2.2.10}$$

Here $\boldsymbol{X}^c$ is the centred data matrix and $\boldsymbol{Z}_k^{'}$ are the first k PCs of this reweighted PCA algorithm.

Afterwards, classification with QDA can be carried out on the projected data of lower-dimension. The number of components k kept in the model can be chosen by the classification performance. For each possible k, we implement the reweighting algorithm and find the best α and β_i ($i = 1,2,\ldots,c-1$). The specific k that provides the most precise classification in the cross-validation with the corresponding weights α and β_i is selected as the optimal k.

So far we have shown the reweighted PCA algorithm in the general c-class case. In this thesis we focus on the two-class and three-class classification problems. In the binary case, equation (2.2.6) can be simplified as:

$$\boldsymbol{S}_T^{'} = (1-\alpha)(\beta \boldsymbol{S}_1^{'} + (1-\beta)\boldsymbol{S}_2^{'}) + \alpha \boldsymbol{S}_B^{'}, \tag{2.2.11}$$

and only two weight parameters are needed.

2.2.2.2 Between principal component analysis (Between PCA)

As discussed, the between covariance usually contains information useful for group separation and thus should be highlighted. Combining this idea with reweighting, it is reasonable to consider extracting PCs from the between covariance first and then reweighting the within-group covariance matrices before extracting further PCs from their combination.

Note that $\boldsymbol{S}_B^{'} = \frac{1}{n-1}\boldsymbol{S}_B$ and together with equation (2.2.2), the between-class covariance in the c-class case:

$$\boldsymbol{S}_B^{'} = \frac{1}{n-1}\sum_{i=1}^{c} n_i(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})^T. \tag{2.2.12}$$

Here $\boldsymbol{S}_B^{'}$ is a matrix of rank $c-1$ and the PCs of $\boldsymbol{S}_B^{'}$ can be obtained by eigen-decomposition,

$$\boldsymbol{S}_B^{'} = \boldsymbol{V}_B \boldsymbol{\Sigma}_B \boldsymbol{V}_B^T, \tag{2.2.13}$$

where $\boldsymbol{V}_B$ contains the orthonormal eigenvectors of $\boldsymbol{S}_B^{'}$ as its columns and $\boldsymbol{\Sigma}_B$ contains the corresponding eigenvalues. And then

$$\boldsymbol{Z}_B = \boldsymbol{X}^c \boldsymbol{V}_B, \tag{2.2.14}$$

become the first $(c-1)$ PCs of this between PCA algorithm.

To keep the orthogonality of PCs data are projected to the $p-(c-1)$ dimensional subspace orthogonal to the subspace spanned by $\boldsymbol{V}_B$, i.e.,

$$\tilde{\boldsymbol{X}}^c = \boldsymbol{X}^c(\boldsymbol{I} - \boldsymbol{V}_B \boldsymbol{V}_B^T). \tag{2.2.15}$$

In the $p-(c-1)$ dimensional subspace, the within-group covariance of each class can be denoted as usual:

$$\tilde{\boldsymbol{S}}_i = \frac{1}{n_i - 1} \sum_{j=1}^{n_1} (\tilde{\boldsymbol{x}}_{ij} - \bar{\tilde{\boldsymbol{x}}}_i)(\tilde{\boldsymbol{x}}_{ij} - \bar{\tilde{\boldsymbol{x}}}_i)^T, \tag{2.2.16}$$

for all $i \leqslant c$. Here the within-group covariance is calculated with the projected data instead of the raw data and the tilde symbol is used to distinguish the projected data from the raw one.

Then the pooled within covariance:

$$\tilde{\boldsymbol{S}}_W = \sum_{i=1}^{c} \frac{n_i - 1}{(\sum_{i=1}^{c} n_i) - c} \tilde{\boldsymbol{S}}_i. \tag{2.2.17}$$

As before, $c-1$ parameters can be applied into the above formula, so as to modify the relative weights of these c within covariance matrices, and we have:

$$\tilde{\boldsymbol{S}}_W^{'} = \beta_1 \tilde{\boldsymbol{S}}_1^{'} + \Sigma_{l=2}^{c-1} \Pi_{s=1}^{l-1} (1-\beta_s) \beta_l \tilde{\boldsymbol{S}}_l^{'} + \Pi_{i=1}^{c-1} (1-\beta_i) \tilde{\boldsymbol{S}}_c^{'}, \tag{2.2.18}$$

where $\tilde{\boldsymbol{S}}_i^{'} = \frac{n_i - 1}{(\sum_{i=1}^{c} n_i) - c} \tilde{\boldsymbol{S}}_i$ for $1 \leqslant i \leqslant c$. Here all the weight parameters are from $[0,1]$. The larger β_i is, the more highlighted is $\tilde{\boldsymbol{S}}_i^{'}$ over the other $\tilde{\boldsymbol{S}}_j^{'}$ for all $j \neq i$. Similar to the reweighted PCA, the optimal values of the weight parameters can be obtained by Nelder-Mead simplex optimisation, aiming at minimising the cross-validation classification error rate with QDA. With each specific weight series of β we can reconstruct a pooled within covariance matrix according to equation (2.2.18). Then the within PCs can be obtained by eigendecomposition of the reweighted pooled within covariance, i.e. we have:

$$\tilde{\boldsymbol{S}}_W^{'} = \boldsymbol{V}_W \tilde{\boldsymbol{\Sigma}}_W \boldsymbol{V}_W^T, \tag{2.2.19}$$

where $\boldsymbol{V}_W$ contains the eigenvectors of $\tilde{\boldsymbol{S}}_W^{'}$ as its columns, then:

$$\boldsymbol{Z}_W = \tilde{\boldsymbol{X}}^c \boldsymbol{V}_W, \tag{2.2.20}$$

are the within PC scores of this between PCA algorithm. Then combined with the between PCs $\boldsymbol{Z}_B$ in equation (2.2.14), the full PCs $\boldsymbol{Z} = (\boldsymbol{Z}_B, \boldsymbol{Z}_W)$ can be obtained. Here the number of PCs kept in the model can be chosen as usual, by cross-validation. As the between covariance is usually discriminative, we normally take all PCs from $\boldsymbol{Z}_B$. In this case, the cross-validation is mostly used to determine the number of PCs retained from $\boldsymbol{Z}_W$.

So far we have shown the between PCA algorithm in the general c-class case. In this thesis we focus on the two-class and three-class classification problems. In binary classification this between PCA algorithm can be simplified, for in this case the only nonzero eigenvector of $\boldsymbol{S}_B^{'}$ is:

$$\boldsymbol{v}_B = \frac{(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)}{\sqrt{(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)^T (\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)}}. \tag{2.2.21}$$

Therefore we only need to project data orthogonal to this mean difference direction $\boldsymbol{v}_B$, and in this case equation (2.2.18) can be simplified as:

$$\tilde{\boldsymbol{S}}_w^{'} = \beta \tilde{\boldsymbol{S}}_1^{'} + (1-\beta)\tilde{\boldsymbol{S}}_2^{'}. \tag{2.2.22}$$

Since here we have only one parameter to tune, the one-dimensional bounded optimisation method (Brent, 2013) can be applied to search for the optimal weight β.

2.2.3 Reordering algorithms

2.2.3.1 Reordered principal component analysis (Reordered PCA)

Similar to the reweighting algorithms, the reordering algorithms also target making PCA a more appropriate dimension reduction technique in classification. The reweighting algorithms accomplish this by modifying the covariance matrix and then generating more discriminative PCs, while the reordering algorithms accomplish this by ordering the classic PCs by their discrimination power and selecting the discriminative ones from them.

As in the classic algorithm, data are centred first and the covariance matrix is calculated as usual. However in the reordered PCA, PCs are no longer ranked by their associated eigenvalues, they are sorted by their discrimination power in a classifier. In the proposed method we use QDA as the classifier. Compared with LDA, QDA can better utilise different within-group covariance structure to identify the discriminative information in within-group covariance. Accordingly, the discrimination power of a PC can be defined as the classification accuracy when data are projected to this specific PC direction and the corresponding projections are used as the single predictor of a QDA model. This predictive power can be assessed by cross-validation using a single PC in QDA.

We set a cut-off point in the reordering algorithm. We only take the first q PCs into the reordering scheme and the most discriminative k PCs out of q PCs are selected for discrimination. Here both k and q can be tuned by cross-validation. The reason for setting up a cut-off point is threefold. First, in the classic PCA PCs with very small eigenvalues are more likely to contain noise instead of capturing any valid information about the data and thus taking these PCs into consideration will not contribute to classification. Second, with the decrease in eigenvalues the estimation of PCs gets more and more difficult. The estimate of small PCs can be unstable. For example, the 20th PC in each fold can be much different in direction

than the first a few PCs. Thus invoking PCs at the tail by their index may not be a good idea. Third, a cut-off point can help saving computing time. Nevertheless, it is worth further investigating the impact of the cut-off point on the reordered algorithm. This can be regarded as an influential future work.

In this reordered PCA algorithm, the constraint of the first q PCs guarantees the generalisation power of the features while the reordering scheme enhances the discriminative ability of the features. Combining both robustness and discriminability, this reordered PCA algorithm is expected to work better with the subsequent classification with QDA, especially when the total covariance matrix is dominated by some common but not discriminative features.

2.2.3.2 Stepwise-reordered principal component analysis (Stepwise-reordered PCA)

Stepwise-reordered PCA is a similar idea to the reordered PCA. Instead of ranking PCs by their contribution to the total variability we rank PCs by their discrimination power. However as the name implies, in the stepwise-reordered PCA, PCs are no longer ranked and selected individually, they are ranked sequentially to build a classifier.

Specifically, in the stepwise-reordered PCA the first $PC_{(1)}$ is the one that provides the highest classification accuracy with the univariate QDA. Namely, the first stepwise-reordered PC is the same as the first PC in the reordered PCA algorithm. In the stepwise algorithm, the second PC is the one that provides the most accurate two components QDA with the pre-selected predecessor. The rest of the PCs can be selected in the same manner. As usual, both the number of PCs considered q and the number in the final classifier k can be tuned by LOOCV in the training set.

2.3 Examples

Two near infrared spectral data sets, a wheat data set and a paddy rice data set, have been explored to illustrate the superiority of our discriminative PCA algorithms to the classic algorithm in high-dimensional classification.

2.3.1 Binary classification with wheat data set

The wheat data consist of NIR transmission spectra on 292 samples of unground wheat. The spectra were measured using a Tecator Infratec Grain Analyzer which measures transmittance through the wheat sample of radiation at 100 wavelengths from 850 to 1048 nm in steps of 2 nm (Fearn et al., 1999). The wheat samples were classified into nine varieties, on the basis of known provenance, and the sample size for each variety can be found in Table 2.3.1. Here one binary classification example and one three-class classification example will be discussed with the wheat data. Improvement in classification accuracy can be witnessed in both binary and multi-class cases by replacing the classic PCA with our discriminative dimension reduction algorithms.

Variety	1	2	3	4	5	6	7	8	9	Total
Number of samples	52	14	36	29	68	13	16	37	27	292

Table 2.3.1

Composition of the wheat data set

From Table 2.3.1, variety 2, variety 6 and variety 7 have very limited samples and thus these three varieties have been excluded from our classification example. To do binary classification, we need choose a pair of groups from the other six groups. There are 15 possible pairs considering 6 possible varieties. Apart from the pair of variety 1 and variety 5 and the pair of variety 1 and variety 9, which can be well separated by classic PCA-QDA with less than 8% classification error rate,

decrease in classification error rate can be witnessed from all of the other 13 pairs (see Appendix B) with our reweighted PCA algorithm. Here we choose two groups with relative comparable sample size to prevent from the impact of data unbalance on the example. Here we use variety 3 and variety 9 as our example to illustrate the idea of our reweighting and reordering algorithms. The composition of our target data can be found in Table 2.3.2.

Class	Number of samples
Class 1 (Variety 3)	36
Class 2 (Variety 9)	27
Total	63

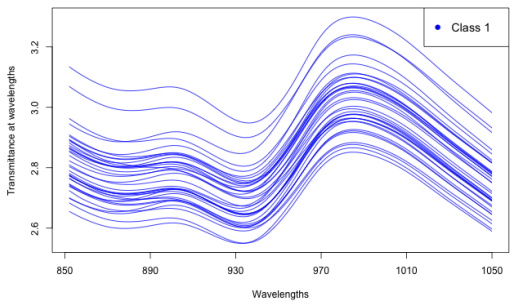
Table 2.3.2

Composition of the two target classes

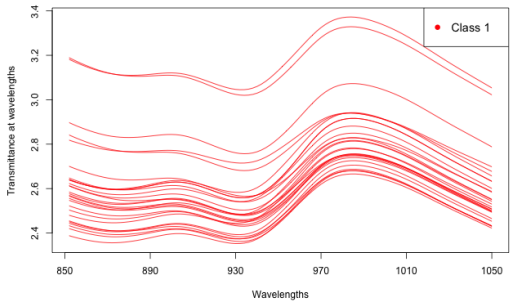
A plot of the spectra of the two classes can be found in Figure 2.3.1. In the figure, each curve represents the spectrum of a sample. The blue curves in figure (a) represent the spectra of samples from class 1 while the red curves in figure (b) represent those of class 2. Here the number of wavelenghs is 100, while the number of observations is 63. The number of predictors exceeds the number of observations. A small number of discriminative features need to be extracted from the original high-dimensional data and then classification will be implemented on the new features.

Here we take the second derivative of the spectra as our data. Since the second derivative of the spectra removes the additive baseline shift and the multiplicative effect of the spectra, it is much likely to lead to higher classification result than the raw spectra. Here we choose to use the second derivative as it provides the most accurate classification with the methods. In general, the selection of the raw data, first derivative or second derivative mostly depends on the corresponding classification performance. We choose the one that gives the most accurate classification results with the pre-selecting methods. The second derivative of the spectra is shown in

Figure 2.3.2.

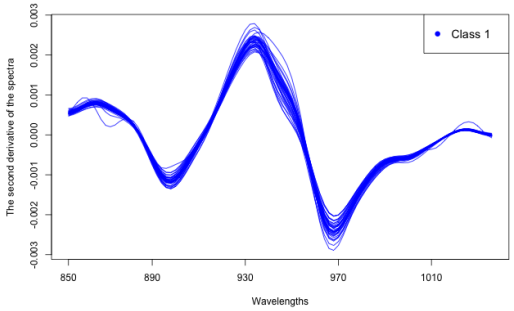


(a) Spectra of samples from class 1

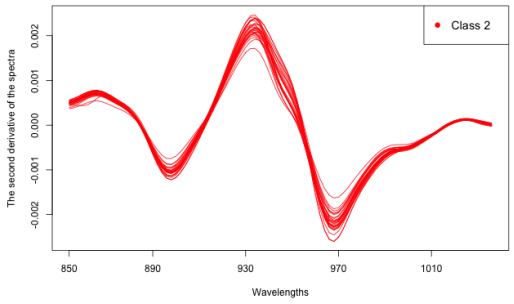


(b) Spectra of samples from class 2

Figure 2.3.1. Spectra of wheat samples from class 1 (variety 3) and class 2 (variety 9)



(a) Second derivative of spectra from class 1



(b) Second derivative of spectra from class 2

Figure 2.3.2. Second derivative of spectra of wheat samples

Our reweighting algorithms (reweighted PCA and between PCA) and reordering algorithms (reordered PCA and stepwise-reordered PCA) are applied to extract discriminative features from high-dimensional spectral data. Their performances with QDA classifier are compared with that of PCA and PLS. Here PLS-QDA proceeds by generating factors according to the principle of PLS, maximising the covariance of the generated factors and the categorical response variables, and then using the generated factors in the QDA classifier. We use QDA as the classifier instead of LDA based on two reasons: 1) As we discussed before, compared with LDA, QDA can better identify the discriminative information in within-group covariance. 2) Empirically, QDA can provide lower classification error rate in this

example than LDA.

Here performance of the above six methods in the leave-one-out cross-validation (LOOCV) will be discussed first, so as to illustrate the idea of the six methods in this specific example. Afterwards, results of a double cross-validation will be presented as a verification of performance. To begin with, classification performances of the above methods in the LOOCV are shown in the following figure.

Figure 2.3.3 shows the classification performance of the above six high-dimensional classification methods via LOOCV. In the figure, the black line and the green line represent classification error rates of classic PCA-QDA and PLS-QDA. The other four coloured lines show the classification error rates of our modified algorithms.

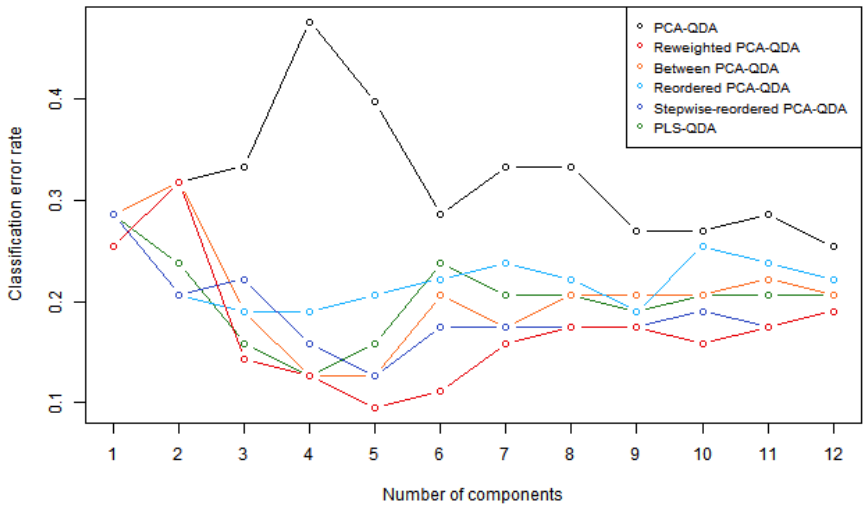


Figure 2.3.3. LOOCV classification error rate of the wheat data with classic PCA-QDA, Reweighted PCA-QDA, Between PCA-QDA, Reordered PCA-QDA, Stepwise-reordered PCA-QDA and PLS-QDA.

Considering the limited sample size, here we only explore models with no more than 12 components. Around 95% of the total variation can be explained by these 12 components. It was argued in many literatures that in PCA capturing 85% or 90% of the total variation would be sufficient while including more features into

the model has the risk of bringing in noise and damaging further analysis (Ferré, 1995; Peres-Neto et al., 2005; Jolliffe, 1986). Here if all PCs are used, all methods will eventually converge to the same point as identical subspace is employed for each method. Nevertheless, this convergence point does not necessarily lead to lowest classification error rate or optimal classification performance.

Above all, a significant decrease in classification error rate can be witnessed from all of our modified algorithms when compared with classic PCA-QDA. The lowest error rate among all the methods is obtained by reweighted PCA-QDA at 9.5% with 5 components while the lowest error rate the classic PCA-QDA can achieve is 25.4% with 10 components. The classification error rate decreases by 15.9% and the number of components needed for discrimination reduces from 10 to 5.

Specifically, reweighted PCA-QDA achieves its lowest error rate 9.5% with between weight $\alpha = 0.525$ and within weight $\beta = 0.025$. Compared with the natural weights $\alpha = \frac{1}{3}$ and $\beta = \frac{1}{2}$, between covariance is given higher weight and the variance of group 1 is given much less weight than that of group 2. In binary classification the between-covariance contains the mean difference direction which reveals the location difference of the two classes in the space. This mean difference direction is usually discriminative and highlighting it would benefit the classification. This is why reweighting towards the between covariance helps. The within weight β controls the relative weight of the two within-covariance matrices. When the weight of group 1, β is 0.025, the relative weight of group 2 is $(1-\beta) = 0.975$. That is to say, the covariance of group 2 is strongly highlighted over the covariance of group 1. The reason why this asymmetrical weight helps in classification can be found in the following analysis.

Table 2.3.3 lists the cosines of the angles between the first four PCs of group 1, group 2, classic PCA and reweighted PCA. If the cosine value is close to 1 (or -1), these two PCs are collinear while if the cosine value is close to 0, the corre-

sponding PCs are almost orthogonal. Here classic PCA is implemented on the two groups separately to extract dominant directions from them. Then we can analyse the relationship between the classic PCs, reweighted PCs and the main directions of the two groups.

Table 2.3.3

Cosines of angles between the first four PCs obtained from Group 1, Group 2, classic PCA and reweighted PCA

	Group1 PC1	Group2 PC1	Group1 PC2	Group2 PC2
Classic PC1	0.9810	-0.9893	0.1207	0.0804
Reweighted PC1	0.9438	-0.9904	0.1950	-0.0115

	Group1 PC2	Group2 PC2	Group1 PC3	Group2 PC3
Classic PC2	0.9805	-0.9886	-0.1159	-0.0151
Reweighted PC2	-0.9434	0.9999	0.1636	0.0027

	Group1 PC3	Group2 PC3	Group1 PC4	Group2 PC4
Classic PC3	-0.9503	0.0450	-0.0116	0.4765
Reweighted PC3	-0.0562	-0.6654	0.0624	0.3604

	Group1 PC3	Group2 PC3	Group1 PC4	Group2 PC4
Classic PC4	0.0130	-0.2648	0.9863	-0.1353
Reweighted PC4	-0.2209	0.7356	-0.1617	0.4293

As shown in Table 2.3.3, the first two classic and reweighted PCs have similar directions. They are almost collinear with the first two PCs of the two groups, having nearly 1 (or -1) cosine values. As shown in Figure 2.3.2, the classification performance of the first two reweighted PCs and the first two classic PCs are comparable as well. In other words, the improvement in classification accuracy is not because of the first two PCs.

Nevertheless, the performance of the third and the fourth PCs of the two algorithms are very different. As can be seen from Figure 2.3.3, the third and the fourth classic PCs are not only unhelpful, but even harmful to classification. The error rate rises considerably after including the third and the fourth classic PC. Meanwhile,

the third and the fourth reweighted PCs successfully decrease the classification error rate by 19.1%.

In Table 2.3.3, the third and the fourth classic PCs are dominated by the first group and are almost collinear with the third and the fourth PCs of group 1. Meanwhile, the third and the fourth reweighted PCs are closer to the corresponding PCs of group 2. They describe some variation of the second group. If we can show that the third and the fourth PC of group 2 can contribute more to classification than that of group 1, we can understand why the third and the fourth reweighted PCs outperform the classic ones. This is shown in Figure 2.3.4.

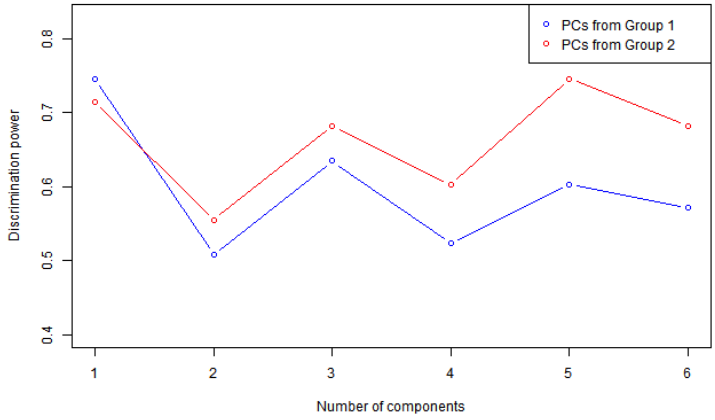


Figure 2.3.4. A comparison of the discrimination power of the first six PCs from the two groups.

Figure 2.3.4 shows the discrimination power of the first six PCs of the two groups. As defined in the methodology section, the discrimination power of a PC can be defined as the classification accuracy when data are projected to this specific PC direction and the corresponding projections are used as the single predictor in a QDA classifier. In the figure the blue line and the red line show the discrimination power of the two groups respectively. Except for the first PC, the other PCs of group 2 are significantly more discriminative than that of group 1, while actually the first PC of the two groups do not differ much in either direction or discrimination

power. The cosine value between these two PCs is -0.9536 which means they are almost collinear while the discrimination power only differs by 3.2%. So in this case reweighting towards the second group is almost harmless in respect of the first component, and beneficial to classification if considering the following PCs.

So far we have shown how reweighting helps in binary classification. Reweighting towards the between-covariance attaches more importance to the mean difference direction while reweighting towards the more discriminative group contributes to generating features with higher discrimination power.

As for between PCA, as a one-way reweighting algorithm, it extracts the mean difference direction first and then reweights the two within covariance matrices, just as the reweighted PCA does. In this example, when the relative weight of group 1 equals 0.144 and the weight of group 2 equals 0.856, the lowest error rate of 12.7% is achieved. Compared with the lowest error rate 25.4% of classic PCA-QDA, the error rate decreases to about a half by implementing this one-way reweighting algorithm. Similar to the two-way reweighting algorithm, here we emphasize group 2 over group 1 as it contains more discriminative features.

So far we have explained how the reweighted PCA and the between PCA help in this high-dimensional classification issue and next we will investigate the reordered and the stepwise-reordered algorithm in this example.

In the reordered algorithm, instead of ranking PCs according to their associated eigenvalues, we rank PCs by their discrimination power and select PCs with high discrimination power. In this algorithm, only the first q PCs are taken into the reordering scheme and ranked by their discrimination power in descending order. The first k PCs among these q PCs which provide the highest overall accuracy will be included in the discrimination model.

In this specific example, the reordered PCA-QDA obtains its lowest error rate 19.1% with 3 components and cut-off number $q = 11$, while the stepwise-reordered PCA-QDA achieves its lowest error rate 12.7% with 5 components and cut-off num-

ber $q = 11$. Note that the lowest error rate accomplished by the classic PCA-QDA is 25.4% with 12 components. By introducing this simple feature filtering scheme, the classification error rate can be decreased by 12.7% with the stepwise-reordered PCA-QDA and by 6.5% with the reordered PCA-QDA. The number of components used in discrimination can be reduced from 12 to 5 with the stepwise-reordered algorithm and from 12 to 3 with the reordered algorithm.

Here we can witness improvements in two aspects. The first improvement is that the number of components needed for discrimination is reduced. This is a direct outcome of selecting PCs. The second improvement is the decrease in classification error rate. The error rate decreases to about half of its previous value by applying the stepwise algorithm. This is much likely to happen if the total variation is dominated by some common variation of the two groups. If this is the case, the first few PCs will correspond to the common variation of groups. Although these PCs contain high variability and the directions they correspond to are the most influential directions in the data, if used in classification they may not be favourable. This idea can be further verified in Figure 2.3.5 and table 2.3.4.

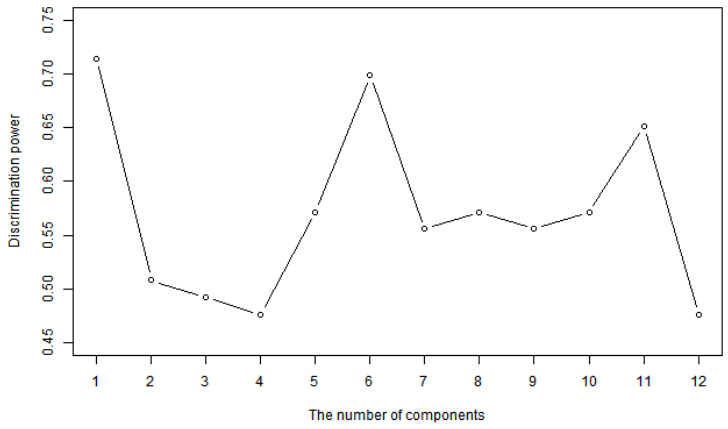


Figure 2.3.5. Discrimination power of classic PCs

Figure 2.3.5 shows the discrimination power of the first 12 PCs in the classic PCA algorithm. As we can see, the first, the sixth and the eleventh PCs have

high discriminative power, while the second, the third and the fourth classic PCs only have around 50% discrimination power, which means if the corresponding PC directions are used for discrimination only half of the samples can be correctly classified. This is only as good as random guess. The low discriminating power of leading PCs are mainly due to the fact that they mostly correspond to the common variation of the two classes while revealing little distinct variation. These features are good at capturing variability but are not beneficial to classification. As a result, they should be excluded from the model.

Here we only show the discriminative power of the first 12 PCs. The reason is twofold. First, 12 PCs contain about 95% of the total variability. It was argued in many literatures that in PCA capturing 85% or 90% of the total variation would be sufficient while including more features has the risk of bringing in noise. Second, experiment result shows that the most accurate classification is achieved with 3 components and cutting off point 11. The largest cutting off point we consider is 20. In other words, next couple of PCs (from the 12nd PC to the 20th PC) will not contribute to the classification anyway. So we omit them.

Table 2.3.4

Cosines of angles between the first two PC directions and dominating directions of the two groups, and cosines between the first two PCs and the mean difference direction.

	Group1 PC1	Group2 PC1	Mean difference
PC1	0.981	-0.989	0.972

	Group1 PC2	Group2 PC2	Mean difference
PC2	0.944	-0.990	0.100

Table 2.3.4 lists cosines of angles between the first two classic PCs and the dominating directions of the two classes (the first two PC directions of each group), as well as the cosines between the first two PCs and the mean difference direction. As shown in the table, the first two PC directions are almost collinear with the

first two PCs of the two groups. In other words the first two PCs describe the common variation of the two groups. As we discussed, we cannot expect features describing the common pattern of the two groups to be good at distinguishing one from the other. As shown in Figure 2.3.5, the second PC fails to assign half of the samples to the correct group. It is not a good discrimination feature. In terms of the first PC, although it also describes common variation of the two groups it owns high discrimination power. In Figure 2.3.5 it successfully classifies more than 70% of the samples. This is because, as shown in Table 2.3.4, the first classic PC happens to be collinear with the mean difference direction. As we discussed before, the mean difference direction is usually discriminative as it demonstrates position difference of the two groups. Accordingly, the first PC is a discriminative feature. However, except the first one, all of the next 5 PCs are of limited use in classification. They misclassify around 50% of the samples. That is why reordering PCs helps in enhancing classification accuracy.

So far we have explained how reweighting algorithms (reweighted PCA and between PCA) and reordering algorithms (reordered PCA and stepwise-reordered PCA) help in enhancing classification in the wheat example. Figure 2.3.6 shows the classification error rates of the six methods via double cross-validation. Here as the sample size is limited, we cannot have a separate training and test set. Therefore double cross-validation is used to verify the results.

In double cross-validation, we randomly split the data 10 times. Each time, 10 samples are selected as the validation samples while the remaining 53 samples are taken as the calibration samples. Here we control the sample proportion of the two classes in the calibration and the validation set to be as close as possible. In each run, the best number of components used in discrimination, the optimal weights in the reweighting algorithms are decided via LOOCV in the calibration set. The calibrated model is used to predict labels for the validation samples. The average error rate over 10 repetitions of validation is regarded as a measurement of

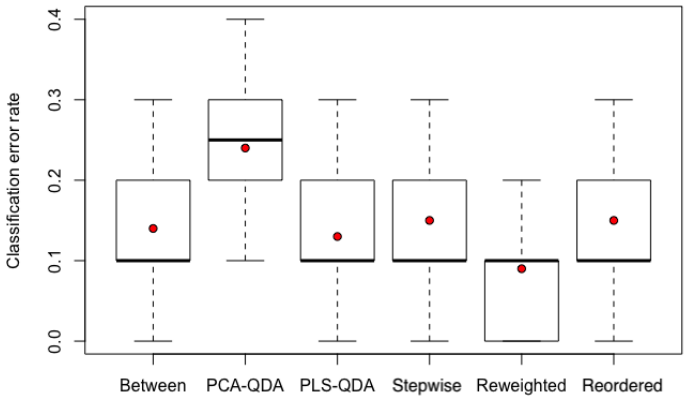


Figure 2.3.6. Double CV error rates of the six methods in the binary wheat example.

the model performance.

In Figure 2.3.6, each box represents the error rates of a method. The red spot represents the average error rate over the 10 validation sets and the black line corresponds to the median. As we can see, classic PCA-QDA owns the highest average error rate 24% among the six, while reweighted PCA-QDA achieves the lowest error rate 9%, i.e., the highest classification accuracy among all. The average error rate of between PCA-QDA, reordered PCA-QDA and stepwise-reordered PCA-QDA are 14%, 15% and 15% respectively, while that of PLS-QDA is 13%. In this example, all of our four discriminative modifications outperform PCA but only reweighted PCA-QDA outperforms PLS-QDA. We will see one example later in which the reordered PCA manages to extract discriminative features more efficiently than all the others.

2.3.2 Binary classification with paddy rice data set

The paddy rice data consist of NIR spectra of 100 paddy rice samples that were cultivated from 2014 to 2017. The samples were naturally classified into four groups according to their year of cultivation. Since the illumination and moisture condition differs from year to year, the nutriment content and quality grade varies for paddy rice cultivated in different years. This leads to different selling price in

the market. Some vendors try to cheat the grain cultivated in good natural conditions with the bad one. Besides, the content of nutriment decreases gradually with the time of storage, which also causes different selling price for grain cultivated in different year. Accordingly, it is an important topic to distinguish the cultivation year of grain. NIR spectroscopy can be of great use in this circumstance. Paddy rice cultivated in different years varies in nutrient and thus has different absorptivity against the NIR spectroscopy. In this example, we use NIR spectroscopy to distinguish paddy rice grown in different years.

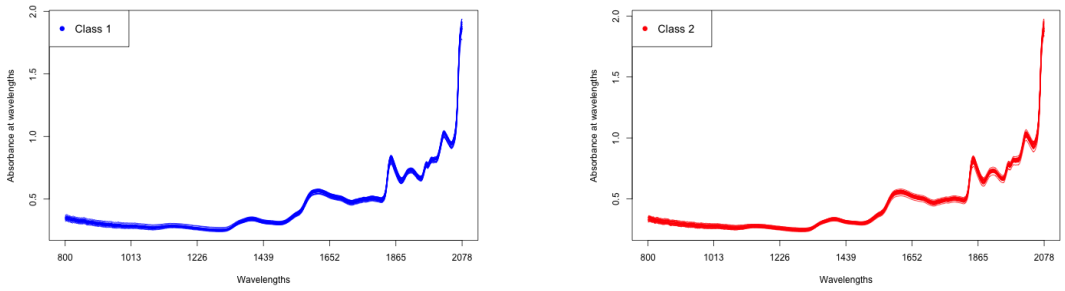
Year of cultivation	Number of samples
2014	24
2015	29
2016	25
2017	22
Total	100

Table 2.3.5

Composition of the paddy rice data

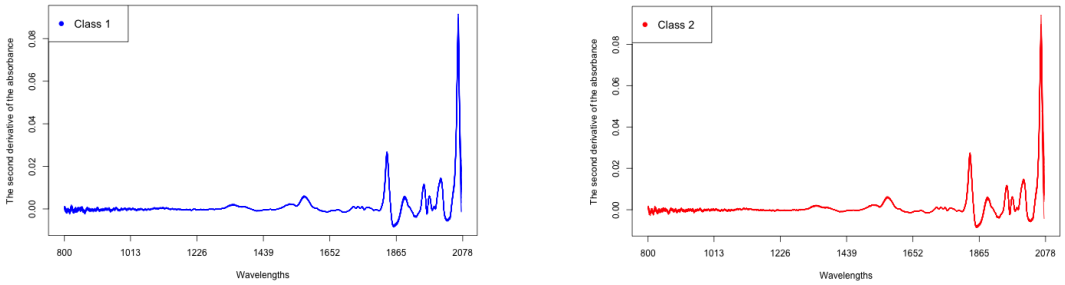
The year of cultivation and the corresponding sample size can be found in Table 2.3.5. For each sample, the absorbance at 1154 wavelengths from 800 to 2782 nm was recorded and used as spectral information to predict labels for unknown samples. As the number of predictors is far beyond the number of observations, dimension reduction is necessary before classification. Here a binary example and a three-class example will be discussed with the paddy rice data. Samples from year 2017 can be easily separated from the others with less than 5% error rate, while samples from the other three groups are difficult to discriminate with the classic PCA-QDA algorithm. Here we use samples from year 2014 and 2016 to illustrate our dimension reduction methods in binary classification. The spectra of the two classes are shown in Figure 2.3.7. In the figure each curve represents the spectrum of a sample. The blue curves in the left subfigure represent the spectra of samples from class 1 while the red curves in right subfigure represent those of class 2. Here

the number of predictors exceeds the number of observations. A small number of discriminative features needs to be extracted from the original high-dimensional data and then classification will be implemented using the new features. As usual, to remove the additive baseline shift of spectra and obtain better classification, we take the second derivative of the spectra and the corresponding spectra are shown in Figure 2.3.8.



(a) Spectra of samples from class 1 **(b)** Spectra of samples from class 2

Figure 2.3.7. Spectra of paddy rice samples from class 1 and class 2



(a) 2nd derivative of spectra of class 1 **(b)** 2nd derivative of spectra of class 2

Figure 2.3.8. The second derivative of the spectra of paddy rice samples

As before, we firstly use the performance in the LOOCV to understand the mechanism of our modified methods in this example, and then present the results of double CV for verification. The corresponding LOOCV classification error rate can be found in Figure 2.3.9.

As in the wheat example, due to the limited sample size we take no more than 10 components into our model. Overall, the lowest error rate 2.0% is achieved by

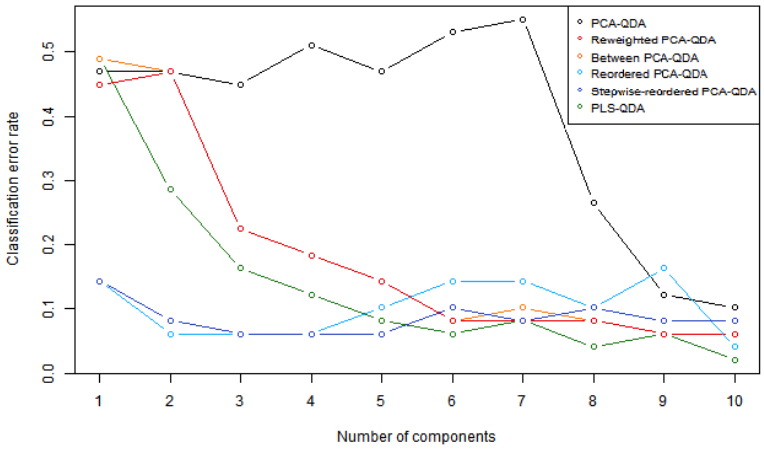


Figure 2.3.9. CV error rates of the above methods in binary paddy rice example.

the PLS-QDA algorithm with 10 components. In this case PLS-QDA manages to categorise 48 samples into the correct group and ends up with only 1 misclassification. Meanwhile, reordered PCA-QDA misclassifies 1 more sample than the PLS-QDA, the reweighted algorithms misclassify 2 more samples than PLS, while the best result the classic PCA-QDA can achieve is 44 correct classifications and 5 errors. In other words, all of our methods succeed in outperforming PCA-QDA but are inferior to PLS-QDA in accuracy. Here reweighted PCA-QDA achieves the lowest error rate with $\alpha = 0.912$, $\beta = 0.275$ and the number of components $k = 9$, between PCA-QDA achieves its optimum with $\beta = 0.2358$ and $k = 9$, reordered PCA-QDA performs best when k= 10 and $q = 11$ while stepwise-reordered PCA-QDA performs best when $k = 3$ and $q = 9$.

However, we may notice that the reordering methods achieve a comparable accuracy with only 2 or 3 components. With two components the reordered method successfully classifies 46 samples into the correct group and ends up with only 3 errors and the stepwise-reordered method makes only 4 errors. Note that the best result the classic PCA-QDA can achieve is 5 errors with 10 components. The reordered and the stepwise-reordered PCA-QDA with 2 components outperform the classic algorithm with 10 components. Furthermore, with the same number of

components the PLS algorithm can only achieve around 70% accuracy. In this case the reordered methods succeed in extracting discriminative features more efficiently than the others. The reason why reordering has such a significant impact in this example can be found in the following analysis.

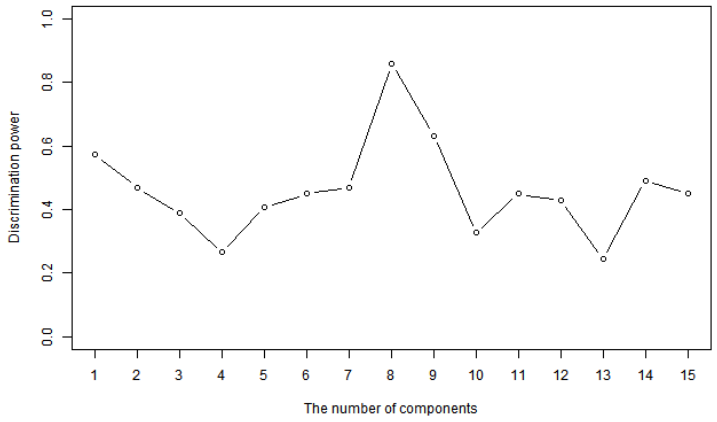


Figure 2.3.10. Discrimination power of classic PCs in binary paddy rice example

Figure 2.3.10 shows the discrimination power of the classic PCs while Table 2.3.6 demonstrates the directions of them. In Figure 2.3.10 the first seven PCs except the first one own poor discrimination power. Less than half of the samples can be correctly categorised if data are projected to each single PC direction. Consequently, including these PCs in the model will not only be unhelpful but also detrimental to the discrimination.

Table 2.3.6

Cosines of angles between the classic PCs and the main directions of the two groups

	Group1 PC1	Group2 PC1
PC1	0.9118	-0.9768

	Group1 PC2	Group2 PC2
PC2	0.9230	0.9763

	Group1 PC3	Group2 PC3
PC3	0.9686	0.9732

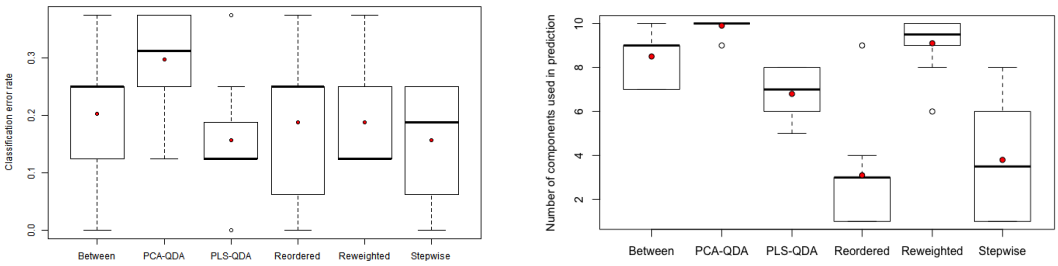
Table 2.3.6 illustrates the relationship of the first three PCs and the dominant directions of the two groups. From the table, the first three classic PCs are almost collinear with the common dominant directions of the two groups. Namely, they mainly describe the common variation of the two groups. We cannot expect features that describe the common variation of the two groups but display no position difference to be good at distinguishing one from the other. Unsurprisingly, as shown in Figure 2.3.9 these PCs are not discriminative.

From Figure 2.3.10 there exist some discriminative features. The 8th PC manages to correctly classify 85.7% of the samples while the 9th PC manages to classify 63.3% of the samples. They are more discriminative than the first seven PCs. However, since the total variation is dominated by the common variation of the two groups, these discriminative features cannot get exposed easily, while by including a simple feature reordering and filtering scheme PCs with high discrimination power can be extracted with few components.

Last but not least, even if the discriminative components (the 8th and the 9th PC) are included in the classic model, the classic PCA-QDA still cannot accomplish the same precision as the reordered methods. With 9 components the classic PCA-QDA misclassifies 5 samples while with 2 components only the reordered method misclassifies 3 samples. This indicates the reordering scheme benefits the classification not only by extracting good features quicker and reducing the number of components, but also by avoiding the detrimental features such as the 4th PC in this example and then enhancing the absolute classification accuracy.

So far we have explained how reordering contributes to the binary classification. In terms of the reweighting algorithm, it highlights the between-covariance and the more discriminative group as usual. The highest precision 93.9% can be obtained when the between weight $\alpha = 0.912$ and within-ratio $\beta = 0.275$.

Figure 2.3.11 shows the classification performance of the six methods via double cross-validation. As usual we randomly split the data 10 times. Each time, 4



(a) Classification error rates of each model

(b) Number of PCs needed in each model

Figure 2.3.11. Classification performance of the six methods in double CV in the binary paddy rice example, with subfigure (a): error rates in the test sets, subfigure (b): the number of components needed in each method to accomplish the corresponding classification error rate.

samples from each class are selected as the validation samples while the remaining 41 samples are taken as the calibration samples. The average error rate over the 10 validation sets is regarded as a measurement of the model performance.

In Figure 2.3.11, subfigure (a) displays the classification error rates of the above six methods while the right subfigure shows the corresponding number of components each method uses. As we can see the classic PCA-QDA owns the highest error rate 30%, among the six. When comparing the median error rate, PLS-QDA and the reweighted PCA-QDA own the lowest median error rate. When comparing the mean, PLS-QDA and the stepwise-reordered PCA-QDA achieve the lowest average error rate, 15%. When comparing the number of components each method uses, the reordered and the stepwise-reordered PCA-QDA use significantly fewer PCs than the other methods. On average, the reordered PCA-QDA needs only 3.1 PCs and the stepwise-reordered PCA-QDA only needs 3.8 PCs to outperform the classic PCA-QDA with 10 components. When considering the average classification accuracy, PLS-QDA and stepwise reordered PCA-QDA obtain identically high classification accuracy, however stepwise reordered algorithm achieves this with 3.8 components only, while PLS-QDA needs 6.5 components to achieve

the same accuracy. Hence, the stepwise reordered approach is considered more efficient in this example due to the smaller number of required components. This paddy rice example shows that the reordering scheme can play an indispensable role in reducing the number of components while enhancing the performance of binary classification.

2.3.3 Three-class classification with wheat data set

So far we have shown how reweighting and reordering contribute to binary classification. Now we will apply our models to multi-class case. A three-class classification example will be explored using the wheat data. Here we take three varieties among the nine as our target data and implement the above six methods as before. We choose variety 5, variety 8 and variety 9 as our target groups and as usual, the second derivative of the spectra is used. The corresponding classification results are shown in the following figure.

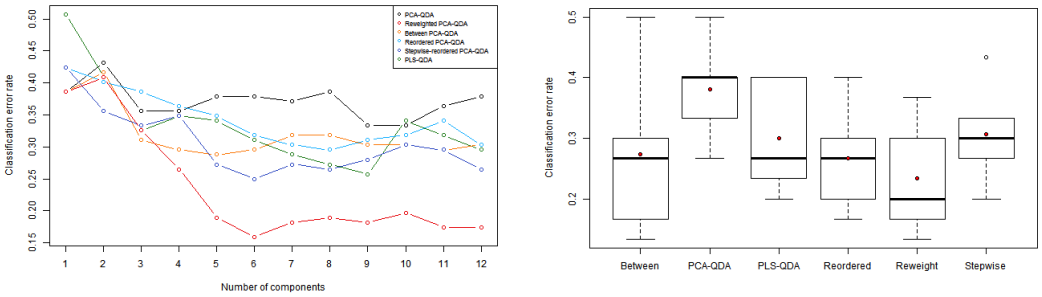


Figure 2.3.12. Classification error rates of the abovementioned six methods in the three-class example via LOOCV and double CV.

Figure 2.3.12 shows the three-class classification performances of the above six methods via LOOCV and double CV. The left subfigure represents the classification performance in LOOCV, the right subfigure represents that in the double CV. In the LOOCV, all of our four algorithms outperform classic PCA-QDA while the reweighted PCA-QDA provides the most precise classification among the six. The

LOOCV error rate decreases from 33.3% to 15.9% after reweighting. In double CV the error rates witnesses a remarkable decrease for all of our PCA modifications. The Reweighted PCA-QDA performs the best, with 23% average error rate, followed by Reordered PCA with 26% error rate and Between PCA with 27% error rate. Meanwhile, Stepwise-reordered PCA-QDA and PLS-QDA achieve 30% and 31% error rate respectively. The classic algorithm only has around 62% average accuracy in this three-class classification example. We may notice here the reweighted PCA performs much better than the between PCA. The between group covariance can contain noise as well. If this is the case, PCs from the between covariance are less discriminative. Highly reweighting towards the between covariance matrix or using its PCs directly as features can lead to unsatisfactory result. On the contrary, reweighted PCA can avoid this by giving less weight to the between group covariance and higher weight to the individual group containing more discriminative information.

In this multi-class example the reordered method is significantly better than the classic PCA-QDA and even slightly outperforms PLS-QDA. PCA generates factors merely according to the contribution to variability. With more classes, the generated PCs are more likely to be a blend of within-group variations. The PCs do not target at discrimination and they are not representative of a single class either. Accordingly, classification based on these features can be even less satisfactory than in the binary case. In other words, in multi-class classification the role of reordering and reweighting becomes even more important.

2.3.4 Three-class classification with the paddy rice data set

In this section we discuss the performance of the abovementioned six methods in three-class classification with the paddy rice data set. The paddy rice data consist of NIR spectra of 100 paddy rice samples that were cultivated from 2014 to 2017.

Paddy rice samples cultivated in year 2017 can be easily separated from the others, thus samples of year 2014, 2015 and 2016 are selected as the three classes. As usual, we take the second derivative of the spectra, to remove the addictive baseline shift and the multiplicative effect.

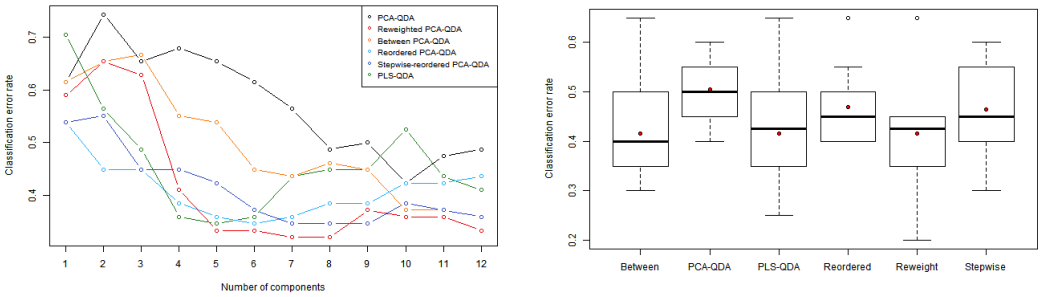


Figure 2.3.13. Classification error rates of the above six methods in the paddy rice three-class example via LOOCV and double CV.

Figure 2.3.13 shows the performance of the six methods in three-class classification via LOOCV and double CV. Improvement in accuracy can be witnessed from all of our methods in LOOCV as well as double CV, when compared with the classic PCA-QDA. In double CV, data are split into a training set with 58 samples and a test set with 20 samples 10 times. We control the sample proportion of the three classes in the training and the test set to be as close to their natural proportion as possible. Namely, every time we randomly select 6 samples from year 2014, 7 samples from year 2015 and 6 samples from year 2016 to be the test set, and the remaining samples become the training set. The corresponding error rates in 10 test sets can be found in the right subfigure of Figure 2.3.13. The lowest average error rate 42% is obtained by the between, the reweighted as well as the PLS-QDA algorithms. The reordered method and the stepwise method decrease the median error rate from 50% to 45% with the average number of components decreasing from 9.2 to 6.4 and 6.9 respectively. We may notice that after our enhancements the average error rate in the test set is still above 40% in each three-class paddy rice example. In

other words, our methods need to be further enhanced in multi-class classification tasks and this could be investigated in future work.

2.4 Conclusion

In this chapter, we proposed four PCA-based discriminative dimension reduction methods, the reweighted PCA, the between PCA, the reordered PCA and the stepwise-reordered PCA. All of them aim at incorporating supervised information with PCA and making PCA a more appropriate dimension reduction method in high dimensional classification. In reweighted PCA and between PCA, the between-group-covariance is given more importance, to expose the discriminative feature contained in it more efficiently, and the within-group-covariance of an individual class which contains more discriminative information is emphasized over the one mainly describing common variation. By doing that, the reconstructed pooled covariance matrix becomes more discriminative than the original one. In reordered PCA and stepwise-reordered PCA, PCs are generated as usual but re-ordered by the discrimination power with QDA. By applying this simple feature filtering scheme to classic PCA, the discriminating power of selected features can be enhanced remarkably. The results for two NIR spectral data sets, the wheat data and the paddy rice data, have verified the effectiveness of our algorithms. Improvements in classification accuracy can be witnessed in both binary classification and multi-class classification by replacing the classic PCA with our discriminative algorithms.

Reweighted PCA-QDA obtains the highest accuracy in three classification examples out of four, compared with between, reordered, stepwise reordered, classic PCA-QDA and PLS-QDA. Namely, generally speaking, reweighted PCA-QDA is the most accurate algorithm among the four proposed methods. Moreover, in both three-class classification examples the highest accuracy is obtained by the reweighted algorithm. Reweighted PCA gives different weights to different groups, which makes it more adaptive to multi-class classification. Accordingly, the re-

weighted algorithm has great potential to be used in multi-class classification. However, one potential concern about the reweighted algorithm is on its computation time. In c class classification a c-dimensional optimisation problem in a bounded area is implicitly contained in the procedure of the reweighted algorithm. As a result, the computation time of the reweighted algorithm increases significantly with the increase of the number of classes.

On the contrary, the computation complexity of the reordered algorithm and the stepwise reordered method does not grow with the number of classes. The reordered algorithms only include a filter step under the classic PCA framework, which does not require complex computation or optimisation technique. This provides the reordered algorithms high potential to replace the reweighted algorithm in multi-class classification with a large number of classes. Moreover, in the binary classification of the paddy rice data, the stepwise reordered PCA-QDA achieved the highest accuracy with only 3.8 PCs on average. This further verifies the computation efficiency of the stepwise reordered method. In other words, when the computation efficiency is the main concern of the users or when we have multi-class classification problems with a large number of classes, the stepwise reordered method is more likely to be the appropriate algorithm to apply.

In terms of interpretability, all of our proposed methods have high interpretability. As we discussed, the reweighting algorithms usually attach higher weights to the between covariance to uncover the difference of group means, as well as the group with distinct variation information, to help generating more discriminative PCs. Meanwhile, the reordering algorithms extract PCs with high discriminative power first. The mechanism of all proposed methods are clear and easy to understand.

In this chapter, we primarily apply our methods to balanced data set. The reason is twofold. First, unlike medical data the inherent unbalance of NIR data is usually not severe. The costs of misclassifying different samples are usually

comparable as well. Second, we want to avoid the impact of data unbalance on the experiment results and simplify the analysis. On the binary classification task of the paddy rice data the two classes contain 24 and 25 samples respectively while on the three-class classification task of paddy rice data, the three classes contain 24, 25 and 29 samples. Accordingly, the highly asymmetric weights obtained in the experiments and the varied rank of PCs are mainly due to the difference in discriminating power instead of the impact of data imbalance.

Nevertheless, our methods can be applied in unbalanced classification. In the binary classification of the wheat data there is slight data imbalance. The sample size of one group is 1.33 times of that of the other. In the three-class classification of the wheat data, one group is about 2.5 times of the sample size of another group. Our proposed methods work well and can significantly enhance the classification performance in both cases. Namely, our methods are applicable to imbalanced classification. Moreover, owing to the asymmetric weights given to different classes, the reweighted algorithm has inherent capability in handling unbalanced data set. Intuitively, when one group has significant fewer samples than the other groups, higher weight can be attached to this minority group, so as to give higher importance to its samples and improve the classification accuracy of this minority group. Nevertheless, when the data are extremely unbalanced and when one class has too limited samples to be estimated accurately, the weights given to the classes should consider both the discriminating power and generalisation power of the generated PC. The performance of the proposed methods under extremely data unbalance can be regarded as a rewarding future work.

Chapter 3

A Penalised QDA-based Feature Extraction Method

3.1 Introduction

As discussed in Chapter 2, as an unsupervised method, PCA fails to use the label information of the observations (Chen and Sun, 2005; Huang et al., 2015). It gives high weights to features with higher variabilities irrespective of whether they contribute to classification. This may give rise to the situation where the chosen principal component corresponds to the attribute with the highest variability but without any discriminating power (Pechenizkiy et al., 2006).

In chapter 2 we proposed four methods to overcome this deficiency of PCA. Reweighted PCA and between PCA enhance PCA by reweighting the total covariance matrix. Reordered PCA and stepwise reordered PCA enhance PCA by ranking the classic PCs by their discriminating power with QDA and only retaining the discriminative ones in the model. The idea of selecting features based on the performance in a classifier can be further developed as generating features specialised for a classifier. This enlightens the penalised QDA based feature extraction method that we propose in this chapter. As this method is from a supervised point of view and it is more based on QDA than PCA, we use a separate chapter to illustrate it.

The most well-known supervised feature extraction method is Fisher LDA. In binary classification it aims at finding a normalised vector $\boldsymbol{w}$ that maximises $\frac{\boldsymbol{w}^T \boldsymbol{S}_B \boldsymbol{w}}{\boldsymbol{w}^T \boldsymbol{S}_W \boldsymbol{w}}$, where $\boldsymbol{S}_B$ is the between-class SSP matrix and $\boldsymbol{S}_W$ is the within-class SSP matrix. When $\boldsymbol{S}_W$ is non-singular $\boldsymbol{w}$ is found to be the eigenvector of $\boldsymbol{S}_W^{-1}\boldsymbol{S}_B$. However, when the number of variables exceeds the number of observations, $\boldsymbol{S}_W$ does not have full rank and the inverse of $\boldsymbol{S}_W$ is inaccessible. Consequently, the classic Fisher LDA fails to work in the small sample size case (Huang et al., 2002).

As introduced in the first chapter, variants of LDA have been developed in the past two decades to solve this problem, such as orthogonal LDA (Ye and Xiong, 2006), RDA (Friedman, 1989) and Null Space LDA (Chen et al., 2000). However, it was shown in some real data experiments that using Orthogonal LDA, Null Space LDA or RDA to directly extract features has led to inferior performance compared with applying the conventional LDA after dimension reduction with PCA, as PCA reduces noise through its feature generation (Prasad et al., 2010).

De Jong and Kiers (1992) claimed that when generating features two criteria should be taken into account and also balanced: (1) the features should account for much of the variance of the data, as this stabilises the estimation, and (2) the features should correlate well with the predicted variables, as this entails a good fit. PCA attaches full importance to the first criterion while LDA-based feature extraction methods devote most of the attention to the second criterion.

As we know, PLS extracts features that maximise $\mathrm{Cov}(\boldsymbol{t},\boldsymbol{y})$, i.e.:

$$\begin{aligned}
&\max_{\boldsymbol{t}} \ \mathrm{Cov}(\boldsymbol{t},\boldsymbol{y}),\\
\Leftrightarrow &\max_{\boldsymbol{t}} \sqrt{\mathrm{Var}(\boldsymbol{t})}\mathrm{Corr}(\boldsymbol{t},\boldsymbol{y})\sqrt{\mathrm{Var}(\boldsymbol{y})},\\
\Leftrightarrow &\max_{\boldsymbol{t}} \sqrt{\mathrm{Var}(\boldsymbol{t})}\mathrm{Corr}(\boldsymbol{t},\boldsymbol{y}),\\
\Leftrightarrow &\max_{\boldsymbol{t}} \mathrm{Var}(\boldsymbol{t})\mathrm{Corr}^2(\boldsymbol{t},\boldsymbol{y}),
\end{aligned} \tag{3.1.1}$$

where $\boldsymbol{t}$ is the generated PLS component. In the above criterion (3.1.1), maximising $\mathrm{Var}(\boldsymbol{t})$ guarantees the first criterion of high variability, while maximising $\mathrm{Corr}^2(\boldsymbol{t},\boldsymbol{y})$ ensures the second criterion of a good fit. PLS manages to balance between stability and goodness-of-fit while PCA neglects the latter. This is also the reason why PLS components are generally regarded of higher predictive power than PCs (Berrueta et al., 2007).

Similarly, De Jong and Kiers (1992) proposed a PLS-like feature extraction method in 1992, which is called principal covariates regression (PCovR) (De Jong and Kiers, 1992). In this method the criterion of feature generation is to maximise:

$$\max_{\boldsymbol{t}} \left(\alpha R^2_{\boldsymbol{X}\boldsymbol{t}} + (1-\alpha) R^2_{\boldsymbol{y}\boldsymbol{t}} \right), \tag{3.1.2}$$

where $R^2_{\boldsymbol{X}\boldsymbol{t}}$ is the percentage of variance in the independent variables $\boldsymbol{X}$ explained by the feature $\boldsymbol{t}$ while $R^2_{\boldsymbol{y}\boldsymbol{t}}$ is the percentage of variance in $\boldsymbol{y}$ explained by $\boldsymbol{t}$. α is a weight parameter to balance between stability and goodness-of-fit.

Note that PCA generates features $\boldsymbol{t}$ such that $\mathrm{Var}(\boldsymbol{t})$ is maximised, the first criterion of stability is already satisfied. Inspired by PLS and PCovR, the performance of PCA in classification can be enhanced by including a criterion corresponding to the discriminating power. Further enlightened by PCovR, we can use a weight parameter β to balance stability and discriminability, i.e. the feature generation criterion can be:

$$\max_{\boldsymbol{t}} \left(\beta \, \mathrm{Var}(\boldsymbol{t}) + (1-\beta) \boldsymbol{I}_D(\boldsymbol{t}) \right), \tag{3.1.3}$$

where $\boldsymbol{I}_D$ is a measure of classification performance, e.g. classification accuarcy, log-likehood, etc., and it also depends on the feature $\boldsymbol{t}$. Here we employ the loglikelihood of QDA as the indicator of discriminative power and correspondingly use the logarithm of the sample variance $\log(\mathrm{Var}(\boldsymbol{t}))$ as the indicator of generalisation

ability, instead of $\mathrm{Var}(\boldsymbol{t})$. Now we define the loglikelihood of QDA.

We assume in binary classification:

$$y_i = \begin{cases} 0, \text{ if the } i\text{-th sample belongs to class 1} \\ 1, \text{ if the } i\text{-th sample belongs to class 2} \end{cases} \tag{3.1.4}$$

We denote the probability of classifying the i-th sample to class 2 ($y_i = 1$) in QDA by p_i and correspondingly the probability of classifying it into class 1 is $(1-p_i)$. Then the likelihood of observing this sample as it is:

$$p_i^{y_i}(1-p_i)^{1-y_i}. \tag{3.1.5}$$

Accordingly, the likelihood of observing all samples is:

$$\Pi_{i=1}^{n} p_i^{y_i}(1-p_i)^{1-y_i}. \tag{3.1.6}$$

Formula (3.1.6) is the probability of observing the current data, taking logarithm of the above likelihood we can get the loglikelihood:

$$\log\left(\Pi_{i=1}^{n} p_i^{y_i}(1-p_i)^{1-y_i}\right). \tag{3.1.7}$$

This loglikelihood can be regarded as an indicator of classification performance. The higher value obtained in (3.1.7), the more precise the classifier is. Then combining it with $\log \mathrm{Var}(\boldsymbol{t})$, we get the following feature generation criterion:

$$\max_{\boldsymbol{t}} \left(\beta \log\left(\Pi_{i=1}^{n} p_i^{y_i}(1-p_i)^{1-y_i}\right)|_{\boldsymbol{t}} + (1-\beta)\log \mathrm{Var}(\boldsymbol{t})\right). \tag{3.1.8}$$

Here the loglikelihood $\log\left(\Pi_{i=1}^{n} p_i^{y_i}(1-p_i)^{1-y_i}\right)$ depends on the generated feature $\boldsymbol{t}$, and β is a weight parameter to balance generalisation ability and discrimination

power. For simplicity we divide (3.1.8) by β and get:

$$\max_{\boldsymbol{t}} \Big(\log(\Pi_{i=1}^{n} p_i^{y_i}(1-p_i)^{1-y_i})|_{\boldsymbol{t}} + \alpha \log \mathrm{Var}(\boldsymbol{t}) \Big), \tag{3.1.9}$$

where $\alpha = \frac{1-\beta}{\beta}$.

The reason why we use the loglikelihood of QDA as an indicator of discrimination is, as we discussed in the last chapter QDA has some advantages over LDA in classification. In spite of its higher requirement of sample size to obtain robust estimation of the parameters, QDA can utilise the heterogeneity in variation of the groups to better classify them, while LDA fails to use this information. The classic PCA generates features merely according to the contribution to variability and fails to utilise the heterogeneity in variation as well. By employing the QDA loglikelihood into the feature generation criterion, not only can we improve the discrimination power of the features but also maintain some important second order and non-linear structure of the data.

The importance of the second order information has been recognised in the literatures (Baffi et al., 1999; Wold et al., 1989). Various quadratic PLS algorithms were developed during the last a few decades (Wold et al., 1989; Berglund and Wold, 1997; Wold, 1992). The original PLS algorithm generates features which have high covariance with the response variables and can fit the response variables well in a linear regression. Quadratic PLS generates features having high covariance with the response variables and fitting the response variables well in a quadratic regression. The motivation of our algorithm is to find features specialised for QDA. Though all considering quadratic relationship, quadratic PLS optimises for quadratic regression, while our algorithm optimises for QDA. This is the difference between our method and other quadratic variants of PLS.

So far we have discussed the idea of building a penalised QDA-based feature extraction method. The detailed derivation of the feature generation criterion in bi-

nary classification and the complete algorithm can be found in the next section. The idea of this penalised QDA-based feature extraction method can be easily extended to multi-class case as QDA is also a widely used multi-class classifier.

3.2 Methodologies

3.2.1 Feature Extraction Criterion

Assume we have a data matrix $\boldsymbol{X} \in \mathbb{R}^{n\times p}$ in which there are n instances $(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)^T$ of a p-dimensional vector $\boldsymbol{x}_i \in \mathbb{R}^{p\times 1}$. Features $\boldsymbol{t}_1$, $\boldsymbol{t}_2$, $\cdots$, $\boldsymbol{t}_m$, $\cdots$, $\boldsymbol{t}_M$ ($M \leqslant p$) are designed to be generated sequentially. The m-th feature $\boldsymbol{t}_m$ is generated based on its predecessors $\boldsymbol{t}_1$, $\boldsymbol{t}_2$, $\cdots$, $\boldsymbol{t}_{m-1}$ in an iterative fashion and thus during the generation of $\boldsymbol{t}_m$, all the previous $(m-1)$ features can be regarded as known.

Each time when a new feature $\boldsymbol{t}_m$ is generated, the original data are projected to the subspace orthogonal to $\boldsymbol{t}_m$ and the subsequent features are generated from this subspace orthogonal to $\boldsymbol{t}_m$ and to all the earlier features. By doing this, the orthogonality of features can be guaranteed and the collinearity problem of features can be avoided. The detailed orthogonalisation procedure will be discussed in section (3.2.2.3). Without loss of generality, we introduce the generation procedure of the m-th feature $\boldsymbol{t}_m$.

Let

$$\boldsymbol{t}_m = \boldsymbol{X}_m \boldsymbol{w}_m, \tag{3.2.1}$$

where $\boldsymbol{X}_m$ is the projected data matrix onto the subspace orthogonal to the previous $(m-1)$ features and $\boldsymbol{w}_m \in \mathbb{R}^{p\times 1}$ is the new loading vector used to generate $\boldsymbol{t}_m$. The m-th component $\boldsymbol{t}_m$ has the form $\boldsymbol{t}_m = \begin{pmatrix} t_{1,m} \\ t_{2,m} \\ \cdots \\ t_{n,m} \end{pmatrix}$, where $t_{i,m}$ is the m-th component

score of the i-th sample. We denote the series of all m components as $\boldsymbol{T}_m = (\boldsymbol{t}_1, \boldsymbol{t}_2, \cdots, \boldsymbol{t}_m) = (\boldsymbol{T}_{m-1}, \boldsymbol{t}_m)$, where $\boldsymbol{T}_{m-1} = (\boldsymbol{t}_1, \boldsymbol{t}_2, \cdots, \boldsymbol{t}_{m-1})$ are the $(m-1)$ components that have already been generated. Since during the generation of the m-th component $\boldsymbol{t}_m$, the previous $m-1$ components are regarded as known. $\boldsymbol{t}_m$ is the only unknown part in $\boldsymbol{T}_m$.

Similarly, we can use a column vector $\boldsymbol{T}_{i,m} = \begin{pmatrix} t_{i,1} \\ t_{i,2} \\ \cdots \\ t_{i,m} \end{pmatrix}$ to denote the first m component scores of the i-th sample. Note that $\boldsymbol{T}_{i,m} = \begin{pmatrix} \boldsymbol{T}_{i,m-1} \\ t_{i,m} \end{pmatrix}$, where $\boldsymbol{T}_{i,m-1}$ denotes the previous $(m-1)$ scores of the i-th sample. Similarly, $t_{i,m}$ is the only unknown part in $\boldsymbol{T}_{i,m}$.

In m components QDA, if the i-th sample belongs to class 1, we have

$$\mathbb{P}(\boldsymbol{T}_{i,m}|y_i = 0) = \frac{1}{(2\pi)^{\frac{m}{2}}|\boldsymbol{\Sigma}_{0m}|^{\frac{1}{2}}} \exp(-\frac{1}{2}(\boldsymbol{T}_{i,m} - \mathbf{U}_{0m})^T \boldsymbol{\Sigma}_{0m}^{-1}(\boldsymbol{T}_{i,m} - \mathbf{U}_{0m})), \quad (3.2.2)$$

where $\boldsymbol{\Sigma}_{0m}$ is the covariance matrix of the first m components of class 1 and $\mathbf{U}_{0m}$ is the mean vector of class 1. Similarly, if the i-th sample belongs to class 2, we have

$$\mathbb{P}(\boldsymbol{T}_{i,m}|y_i = 1) = \frac{1}{(2\pi)^{\frac{m}{2}}|\boldsymbol{\Sigma}_{1m}|^{\frac{1}{2}}} \exp(-\frac{1}{2}(\boldsymbol{T}_{i,m} - \mathbf{U}_{1m})^T \boldsymbol{\Sigma}_{1m}^{-1}(\boldsymbol{T}_{i,m} - \mathbf{U}_{1m})), \quad (3.2.3)$$

where $\boldsymbol{\Sigma}_{1m}$ is the covariance matrix of the first m components of class 2 and $\mathbf{U}_{1m}$ is the mean of class 2. In practice, the covariance matrices $\boldsymbol{\Sigma}_{0m}$, $\boldsymbol{\Sigma}_{1m}$ and the means $\mathbf{U}_{0m}$ and $\mathbf{U}_{1m}$ are regarded as known and estimated by the sample covariance matrices and sample means of the two groups.

Taking logarithms of (3.2.2) and (3.2.3) we get:

$$\log \mathbb{P}(\boldsymbol{T}_{i,m}|y_i = k) = -\frac{m}{2}\log(2\pi) - \frac{1}{2}\log|\boldsymbol{\Sigma}_{km}| - \frac{1}{2}(\boldsymbol{T}_{i,m} - \mathbf{U}_{km})^T \boldsymbol{\Sigma}_{km}^{-1}(\boldsymbol{T}_{i,m} - \mathbf{U}_{km}), \tag{3.2.4}$$

for $k = 0$ and 1.

Assume p_i to be the conditional probability of classifying the i-th sample in class 2 ($y_i = 1$) given m component scores $\boldsymbol{T}_{i,m}$:

$$p_i = \mathbb{P}(y_i = 1|\boldsymbol{T}_{i,m}) = \frac{\mathbb{P}(\boldsymbol{T}_{i,m}|y_i = 1)\mathbb{P}(y_i = 1)}{\sum_{k=0}^{1}\mathbb{P}(\boldsymbol{T}_{i,m}|y_i = k)\mathbb{P}(y_i = k)}, \tag{3.2.5}$$

then the corresponding conditional probability of classifying this sample in class 1 ($y_i = 0$) is:

$$1 - p_i = \mathbb{P}(y_i = 0|\boldsymbol{T}_{i,m}) = \frac{\mathbb{P}(\boldsymbol{T}_{i,m}|y_i = 0)\mathbb{P}(y_i = 0)}{\sum_{k=0}^{1}\mathbb{P}(\boldsymbol{T}_{i,m}|y_i = k)\mathbb{P}(y_i = k)}. \tag{3.2.6}$$

Taking logarithms of (3.2.5) and (3.2.6), it follows that

$$\log p_i = \log \mathbb{P}(\boldsymbol{T}_{i,m}|y_i = 1) + \log \mathbb{P}(y_i = 1) - \log \sum_{k=0}^{1}\mathbb{P}(\boldsymbol{T}_{i,m}|y_i = k)\mathbb{P}(y_i = k), \tag{3.2.7}$$

$$\log(1 - p_i) = \log \mathbb{P}(\boldsymbol{T}_{i,m}|y_i = 0) + \log \mathbb{P}(y_i = 0) - \log \sum_{k=0}^{1}\mathbb{P}(\boldsymbol{T}_{i,m}|y_i = k)\mathbb{P}(y_i = k) \tag{3.2.8}$$

For the i-th sample, the likelihood of observing it as it is:

$$p_i^{y_i}(1 - p_i)^{1-y_i}, \tag{3.2.9}$$

then the likelihood of observing the current data:

$$\Pi_{i=1}^{n} p_i^{y_i}(1 - p_i)^{1-y_i} \tag{3.2.10}$$

To enhance the classification performance of QDA and generate features specialised for QDA, we generate components which maximise (3.2.10) of QDA. Namely, our first goal is to:

$$\begin{aligned}
&\max_{p_i, 1\leqslant i\leqslant n} \Pi_{i=1}^{n} p_i^{y_i}(1-p_i)^{1-y_i} \\
\Rightarrow &\max_{p_i, 1\leqslant i\leqslant n} \log\Pi_{i=1}^{n} p_i^{y_i}(1-p_i)^{1-y_i} \\
\Rightarrow &\max_{p_i, 1\leqslant i\leqslant n} \left(\sum_{i=1}^{n} y_i \log p_i + (1-y_i)\log(1-p_i)\right) \qquad (3.2.11)
\end{aligned}$$

Substituting (3.2.4), (3.2.7) and (3.2.8) into formula (3.2.11), we obtain

$$\begin{aligned}
&\log\Pi_{i=1}^{n} p_i^{y_i}(1-p_i)^{1-y_i} \\
=&\left(\sum_{i=1}^{n} y_i \log p_i + (1-y_i)\log(1-p_i)\right) \\
=&\sum_{i=1}^{n}\left\{ y_i\left[\log\mathbb{P}(\boldsymbol{T}_{i,m}|y_i=1) + \log\mathbb{P}(y_i=1) - \log\sum_{k=0}^{1}\mathbb{P}(\boldsymbol{T}_{i,m}|y_i=k)\mathbb{P}(y_i=k)\right]\right. \\
&\left. +(1-y_i)\left[\log\mathbb{P}(\boldsymbol{T}_{i,m}|y_i=0) + \log\mathbb{P}(y_i=0) - \log\sum_{k=0}^{1}\mathbb{P}(\boldsymbol{T}_{i,m}|y_i=k)\mathbb{P}(y_i=k)\right]\right\} \\
=&\left\{ \frac{1}{2}\sum_{i=1}^{n}(y_i-1)(\boldsymbol{T}_{i,m}-\boldsymbol{U}_{0m})\Sigma_0^{-1}(\boldsymbol{T}_{i,m}-\boldsymbol{U}_{0m})^T - \frac{1}{2}\sum_{i=1}^{n} y_i(\boldsymbol{T}_{i,m}-\boldsymbol{U}_{1m})\Sigma_1^{-1}(\boldsymbol{T}_{i,m}-\boldsymbol{U}_{1m})^T\right. \\
&+\frac{1}{2}\sum_{i=1}^{n}(y_i-1)\log|\boldsymbol{\Sigma}_{0m}| - \frac{1}{2}\sum_{i=1}^{n} y_i\log|\boldsymbol{\Sigma}_{1m}| + \sum_{i=1}^{n} y_i\log\mathbb{P}(y_i=1) + \sum_{i=1}^{n}(1-y_i)\log\mathbb{P}(y_i=0) \\
&\left. -\sum_{i=1}^{n}\log\left(\sum_{k=0}^{1}\frac{1}{|\boldsymbol{\Sigma}_{km}|^{\frac{1}{2}}}\exp(-\frac{1}{2}(\boldsymbol{T}_{i,m}-\mathbf{U}_{km})\boldsymbol{\Sigma}_{km}^{-1}(\boldsymbol{T}_{i,m}-\mathbf{U}_{km})^T)\mathbb{P}(y_i=k)\right)\right\}. \qquad (3.2.12)
\end{aligned}$$

We can clearly see from formula (3.2.12) that the explicit variable here is $\boldsymbol{T}_{i,m}$. Note that $\boldsymbol{T}_{i,m}$ can be denoted as $\boldsymbol{T}_{i,m} = \begin{pmatrix} \boldsymbol{T}_{i,m-1} \\ t_{i,m} \end{pmatrix}$ and $\boldsymbol{T}_{i,m-1}$ contains the previous $(m-1)$ scores of the i-th sample. Note that features are generated sequentially and during the generation of the m-th feature, all the previous $(m-1)$ features can be regarded as known. Namely, during the generation of the m-th feature, $\boldsymbol{T}_{i,m-1}$ is

regarded as known and $t_{i,m}$ is the only unknown part in $\boldsymbol{T}_{i,m}$. In other words, the QDA criterion (3.2.11) can be rewritten as:

$$\max_{t_{i,m},1\leqslant i\leqslant n} \log\Big(\Pi_{i=1}^{n}\big(p_i(t_{i,m})\big)^{y_i}\big(1-p_i(t_{i,m})\big)^{1-y_i}\Big). \tag{3.2.13}$$

Here p_i is regarded as a function of $t_{i,m}$.

The second goal of feature generation is to maintain as much information of the data as possible. To achieve it, we want the new feature $\boldsymbol{t}_m$ to contain as much variability as possible, i.e., we want

$$\max_{t_{i,m},1\leqslant i\leqslant n} \frac{1}{n-1}\sum_{i=1}^{n}(t_{i,m}-\bar{t}_m)^2, \tag{3.2.14}$$

or namely,

$$\max_{t_{i,m},1\leqslant i\leqslant n} \Big(\log\sum_{i=1}^{n}(t_{i,m}-\bar{t}_m)^2\Big), \tag{3.2.15}$$

Here $\bar{t}_m$ is the mean of all $t_{i,m}$'s ($1 \leqslant i \leqslant n$) and (3.2.14) is the sample variance of feature $\boldsymbol{t}_m$.

Combining the QDA criterion in (3.2.13) and the variability criterion inspired by PCA in (3.2.15), the feature generation criterion becomes:

$$\max_{t_{i,m},1\leqslant i\leqslant n} \mathbb{F},$$
$$\text{where } \mathbb{F} = \log\Big(\Pi_{i=1}^{n}\big(p_i(t_{i,m})\big)^{y_i}\big(1-p_i(t_{i,m})\big)^{1-y_i}\Big) + \alpha\Big(\log\sum_{i=1}^{n}(t_{i,m}-\bar{t}_m)^2\Big). \tag{3.2.16}$$

Here α is a weight parameter to balance QDA and PCA and it can be chosen by cross-validation.

The above criterion controls the relative importance of PCA and QDA in the

feature generation. The first half of the criterion guarantees the classification performance with QDA while the second half of the criterion indicates the variability criterion of PCA and secures the generalisability of this method.

Note that $t_{i,m} = \boldsymbol{x}_{i,m}^T \boldsymbol{w}_m$ where $\boldsymbol{x}_{i,m}$ is the projected data of sample i onto the subspace orthogonal to the preceding features, and $\boldsymbol{w}_m$ is the true implicit variable in the above criterion. Therefore, to generate a feature which can contribute to QDA classification while maintaining high variability, we need to find a loading vector $\boldsymbol{w}_m$ that maximises (3.2.16).

Namely, the above feature generation criterion can be re-formulated as a function with respect to $\boldsymbol{w}_m$, i.e.:

$$\begin{gathered}\max_{\mathbf{w}_m} \mathbb{G}, \\ \text{where } \mathbb{G} = \log\Big(\Pi_{i=1}^{n}(p_i(t_{i,m}))^{y_i}(1-p_i(t_{i,m}))^{1-y_i}\Big) + \alpha\Big(\log\sum_{i=1}^{n}(t_{i,m}-\bar{t}_m)^2\Big), \\ t_{i,m} = \mathbf{x}_{i,m}^T \mathbf{w}_m.\end{gathered} \tag{3.2.17}$$

This maximisation problem can be further converted to a minimisation problem by setting $\mathbb{J}(\boldsymbol{w}_m) = -\mathbb{G}(\boldsymbol{w}_m)$, namely the optimisation problem turns into:

$$\begin{gathered}\min_{\boldsymbol{w}_m} \mathbb{J}, \\ \text{where } \mathbb{J} = -\log\Big(\Pi_{i=1}^{n}(p_i(t_{i,m}))^{y_i}(1-p_i(t_{i,m}))^{1-y_i}\Big) - \alpha\Big(\log\sum_{i=1}^{n}(t_{i,m}-\bar{t}_m)^2\Big), \\ t_{i,m} = \boldsymbol{x}_{i,m}^T \boldsymbol{w}_m.\end{gathered} \tag{3.2.18}$$

Here the weight parameter α is employed to balance between the QDA criterion and the variability criterion. Since α can be any non-negative value, first of all a proper

range should be decided for α and then we can find the optimal $\boldsymbol{w}_m$'s ($1 \leqslant m \leqslant M$) for possible values of α inside this range. In practice, the range of α can be found this way: the scale of the QDA part can be found by taking a generous number of PCs, fitting (3.2.1) with $\alpha = 0$ and implementing the following optimisation algorithm, then in this case the value of $\mathbb{J}$ is all about the QDA term and can be regarded as an indicator of the scale of the QDA part. The scale of the variability part can be found by implementing classic PCA on the data. We take the log sample variance of the first PC as the rough scale of the variability part. Then α varies in the range where the contribution of the QDA part is no more than 10 times of the contribution of the variability part and no less than $\frac{1}{10}$ of that as well. Once we get the range, possible values of α can be refined by grid search. For a specific value of α, features are generated according to the above criterion (3.2.18). We select the combination of $\boldsymbol{w}_m$'s and α that provides the lowest average classification error rate in the cross-validation.

So far we have obtained the feature generation criterion of the m-th feature. This minimisation problem can be solved using a gradient descent based method. However our problem is nonconvex and under nonconvexity there is no guarantee of global optimum (Ruder, 2016a). Gradient descent tends to stop at a local optimum or even a saddle point without further searching of the global optimum under nonconvexity while repeated randomised initialisation is commonly used in non-convex optimisation tasks such as parameter tuning in deep learning (Sutskever et al., 2013). Here we borrow the idea of repeated randomised initialisation to find a better value for $\boldsymbol{w}_m$. In practice, we start the optimisation of $\boldsymbol{w}_m$ from $2p$ random initial values and obtain $2p$ different stable points of $\boldsymbol{w}_m$, then we select best $\boldsymbol{w}_m$ from these $2p$ candidates which gives the lowest objective value $\mathbb{J}(\boldsymbol{w}_m)$. Here p is the dimension of $\boldsymbol{w}_m$, and the definition of stable point will be explained in detail in section 3.2.2.3. Though by doing this we still have no guarantee of global optimum, we have better chance to obtain a good $\boldsymbol{w}_m$ which yields a small loss. Also, results of

real data experiments and simulations show that repeated randomised initialisations help in refining the value of $\boldsymbol{w}_m$.

3.2.2 Algorithms

The aim of this method is to generate features specialised for QDA classification while conserving as much variability as possible. The full algorithm is divided into a training phase algorithm and a test phase algorithm and will be introduced successively. Afterwards, a few details in the algorithms will be discussed, such as the definition of convergence and stable points in the algorithm, the gradient descent technique used in this method, data deflation and orthogonalisation, etc.

3.2.2.1 Notation

Firstly, let us clarify the notation used in the following algorithms. In the following algorithms:

p	The number of variables in the spectral data;
M	The maximum number of components we consider;
n_{train}	The number of training samples;
n_{test}	The number of test samples;
$\boldsymbol{X}_{train} \in \mathbb{R}^{n_{train} \times p}$	Training data matrix;
$\boldsymbol{X}_{test} \in \mathbb{R}^{n_{test} \times p}$	Test data matrix;
$\boldsymbol{\mu}_{train} \in \mathbb{R}^{p \times 1}$	A column vector containing the column means of $\boldsymbol{X}_{train}$;

In the cross-validation phase, the training data $\boldsymbol{X}_{train}$ is further split into F folds. Each time we randomly select 1 fold to be the validated data $\boldsymbol{X}_{cv}$ in the cross-validation, and use the remaining $(F-1)$ folds as the calibration data $\boldsymbol{X}_{cal}$, i.e. we have:

n_{cal}	The number of calibration samples in each CV split;
n_{cv}	The number of test samples in each CV split;
$\boldsymbol{X}_{cal} \in \mathbb{R}^{n_{cal} \times p}$	Calibration data in each CV split (different for each split);

$\boldsymbol{X}_{cv} \in \mathbb{R}^{n_{cv} \times p}$ Validation data in each CV split (different for each split);

$\boldsymbol{\mu}_{cal} \in \mathbb{R}^{p \times 1}$ Column mean of $\boldsymbol{X}_{cal}$;

$\boldsymbol{w}_m \in \mathbb{R}^{p \times 1}$ The loading vector of the m-th component;

$\tilde{\boldsymbol{w}}_m \in \mathbb{R}^{p \times 1}$ The optimal loading vector of the m-th component;

q The number of stable points of $\boldsymbol{w}_m$ used to select $\tilde{\boldsymbol{w}}_m$;

$\tilde{\boldsymbol{w}}_{ml} \in \mathbb{R}^{p \times 1}$ The l-th stable point found of $\boldsymbol{w}_m$ ($1 \leqslant l \leqslant q$);

k_m Index of the iterations in the gradient descent of $\boldsymbol{w}_m$;

K The max iteration times allowed in the gradient descent of $\boldsymbol{w}_m$;

$\boldsymbol{w}_m^{k_m} \in \mathbb{R}^{p \times 1}$ The value of $\boldsymbol{w}_m$ in the k_m-th iteration of gradient descent;

$\mathbb{J}(\boldsymbol{w}_m)$ Objective function to be optimised shown in formula (3.2.18);

$\boldsymbol{I}_p \in \mathbb{R}^{p \times p}$ Identity matrix of dimension p.

$\boldsymbol{g}_m^{k_m} = r_{k_m} \bigtriangledown \mathbb{J}(\boldsymbol{w}_m | \mathbf{w}_m = \mathbf{w}_m^{k_m})$ The scaled gradient of $\mathbb{J}(\boldsymbol{w}_m)$ in the k_m-th iteration;

$\mathbf{1}_{cal} \in \mathbb{R}^{n_{cal} \times 1}$, $\mathbf{1}_{cv} \in \mathbb{R}^{n_{cv} \times 1}$, $\mathbf{1}_{train} \in \mathbb{R}^{n_{train} \times 1}$, $\mathbf{1}_{test} \in \mathbb{R}^{n_{test} \times 1}$ are vectors of $\mathbf{1}$s.

3.2.2.2 Algorithm

As discussed, we split data into a training set and a test set. We run the training phase algorithm on the training data and use the model obtained from the training phase to predict labels for the test data, according to the test phase algorithm. We use the following Algorithm 1 at page 108 for the training phase and Algorithm 2 at page 109 for test phase.

Note that the whole training algorithm is implemented using a fixed α. As we discussed in section (3.2.1), in theory α can take any non-negative value. Here α is set to vary in a range where the magnitude of the QDA loglikelihood is no more than 10 times of the magnitude of the log variability, and no less than $\frac{1}{10}$ of that as well. Possible candidates for α can be decided by grid search. For any specific value of α, a series of $\tilde{\boldsymbol{w}}_m$ ($1 \leqslant m \leqslant M$) can be found following Algorithm 1. We select the combination of α and the corresponding $\tilde{\boldsymbol{w}}_m$'s which gives the lowest average classification error rate in the cross-validation.

Algorithm 1 Algorithm in the training phase (with a fixed weight α)

1: Split the training data into F folds for F-fold cross-validation;
Denote the calibration data in each cross-validation split as $\boldsymbol{X}_{cal}$;
Denote the validation data in each cross-validation split as $\boldsymbol{X}_{cv}$;
2: **for** each fold f, $1 \leqslant f \leqslant F$ **do**
3: Centralise the data:
$\boldsymbol{X}_{cal} \leftarrow \boldsymbol{X}_{cal} - \mathbf{1}_{cal}\boldsymbol{\mu}_{cal}^T$;
$\boldsymbol{X}_{cv} \leftarrow \boldsymbol{X}_{cv} - \mathbf{1}_{cv}\boldsymbol{\mu}_{cal}^T$;
4: **for** the number of components m, $1 \leqslant m \leqslant M$ **do**
5: Draw an initial random value $\boldsymbol{w}_m^0$ for $\boldsymbol{w}_m$;
6: Update $\boldsymbol{w}_m$ using constrained mini-batch stochastic gradient descent (see section (3.2.2.3)): $\boldsymbol{w}_m^{k_m+1} = \frac{\boldsymbol{w}_m^{k_m} - \boldsymbol{g}_m^{k_m}}{\sqrt{1-2(\boldsymbol{w}_m^{k_m})^T\boldsymbol{g}_m^{k_m} + ||\boldsymbol{g}_m^{k_m}||^2}}$, for $0 \leqslant k_m \leqslant K-1$;
7: Stop updating when the convergence (stable point) is obtained or when the max number of iterations K is reached;
8: Repeat steps 5-7 until q optimal $\tilde{\boldsymbol{w}}_{ml}$ are found, where $1 \leqslant l \leqslant q$;
Set $\tilde{\boldsymbol{w}}_m = \underset{\tilde{\boldsymbol{w}}_{ml}}{\operatorname{argmin}} \{\mathbb{J}(\tilde{\boldsymbol{w}}_{ml}) \text{ for } 1 \leqslant l \leqslant q\}$;
9: Compute the m-th generated factor:
$\boldsymbol{t}_{cal,m} = \boldsymbol{X}_{cal}\tilde{\boldsymbol{w}}_m$;
$\boldsymbol{t}_{cv,m} = \boldsymbol{X}_{cv}\tilde{\boldsymbol{w}}_m$;
10: Build QDA classifier with the generated factors $\boldsymbol{t}_{cal,1}, \boldsymbol{t}_{cal,2}, \ldots, \boldsymbol{t}_{cal,m}$;
Use the model to predict CV data with $\boldsymbol{t}_{cv,1}, \boldsymbol{t}_{cv,2}, \ldots, \boldsymbol{t}_{cv,m}$ and record the number of errors $n_{f,m}$ in the f-th fold;
11: Deflate and orthogonalise the data by:
$\boldsymbol{E}_{cal} = \boldsymbol{X}_{cal}(\boldsymbol{I}_p - \frac{\tilde{\boldsymbol{w}}_m\tilde{\boldsymbol{w}}_m^T\boldsymbol{X}_{cal}^T\boldsymbol{X}_{cal}}{||\tilde{\boldsymbol{w}}_m^T\boldsymbol{X}_{cal}^T\boldsymbol{X}_{cal}\tilde{\boldsymbol{w}}_m||})$;
$\boldsymbol{E}_{cv} = \boldsymbol{X}_{cv}(\boldsymbol{I}_p - \frac{\tilde{\boldsymbol{w}}_m\tilde{\boldsymbol{w}}_m^T\boldsymbol{X}_{cal}^T\boldsymbol{X}_{cal}}{||\tilde{\boldsymbol{w}}_m^T\boldsymbol{X}_{cal}^T\boldsymbol{X}_{cal}\tilde{\boldsymbol{w}}_m||})$;
12: Replace $\boldsymbol{X}_{cal}$ with the deflated data matrix $\boldsymbol{E}_{cal}$;
Replace $\boldsymbol{X}_{cv}$ with the deflated data matrix $\boldsymbol{E}_{cv}$;
13: **end for**
14: **end for**
15: The total number of misclassifications with m components, $N_m = \sum_{f=1}^{F} n_{f,m}$;
16: **Output**:
The optimal number of components $\tilde{m}$ which gives the smallest N_m;

So far the optimal α and the best number of components $\tilde{m}$ can be found. For a fixed α and $\tilde{m}$, the algorithm in the test phase is Algorithm 2. Here the training is done on the whole training set and then we use the training model to predict labels of a disjoint unused test set (see next page for more details).

Algorithm 2 Algorithm in the test phase

1: Centralise the data:
$\boldsymbol{X}_{train} = \boldsymbol{X}_{train} - \mathbf{1}_{train}\boldsymbol{\mu}_{train}^T$;
$\boldsymbol{X}_{test} = \boldsymbol{X}_{test} - \mathbf{1}_{test}\boldsymbol{\mu}_{train}^T$;
2: **for** the number of component m, $1 \leqslant m \leqslant \tilde{m}$ **do**
3: Draw an Randomise initial value $\boldsymbol{w}_m^0$ for $\boldsymbol{w}_m$;
4: Update $\boldsymbol{w}_m$ via mini-batch constrained stochastic gradient descent using $\boldsymbol{w}_m^{k_m+1} = \frac{\boldsymbol{w}_m^{k_m} - \boldsymbol{g}_m^{k_m}}{\sqrt{1-2(\boldsymbol{w}_m^{k_m})^T \boldsymbol{g}_m^{k_m} + ||\boldsymbol{g}_m^{k_m}||^2}}$, for $0 \leqslant k_m \leqslant K-1$;
5: Stop updating when the convergence is obtained or when the max number of iterations K is reached;
6: Repeat steps 3-5 until q optimal $\tilde{\boldsymbol{w}}_{ml}$ are found where $1 \leqslant l \leqslant q$;
Set $\tilde{\boldsymbol{w}}_m = \underset{\tilde{\boldsymbol{w}}_{ml}}{\operatorname{argmin}} \{\mathbb{J}(\tilde{\boldsymbol{w}}_{ml}) \text{ for } 1 \leqslant l \leqslant q\}$;
7: Compute the m-th generated factor:
$\boldsymbol{t}_{train,m} = \boldsymbol{X}_{train}\tilde{\boldsymbol{w}}_m$;
$\boldsymbol{t}_{test,m} = \boldsymbol{X}_{test}\tilde{\boldsymbol{w}}_m$;
8: Deflate and orthogonalise the data by:
$\boldsymbol{E}_{train} = \boldsymbol{X}_{train}(\boldsymbol{I}_p - \frac{\tilde{\boldsymbol{w}}_m \tilde{\boldsymbol{w}}_m^T \boldsymbol{X}_{train}^T \boldsymbol{X}_{train}}{||\tilde{\boldsymbol{w}}_m^T \boldsymbol{X}_{train}^T \boldsymbol{X}_{train} \tilde{\boldsymbol{w}}_m||})$;
$\boldsymbol{E}_{test} = \boldsymbol{X}_{test}(\boldsymbol{I}_p - \frac{\tilde{\boldsymbol{w}}_m \tilde{\boldsymbol{w}}_m^T \boldsymbol{X}_{train}^T \boldsymbol{X}_{train}}{||\tilde{\boldsymbol{w}}_m^T \boldsymbol{X}_{train}^T \boldsymbol{X}_{train} \tilde{\boldsymbol{w}}_m||})$;
9: Replace $\boldsymbol{X}_{train}$ with the deflated data matrix $\boldsymbol{E}_{train}$;
Replace $\boldsymbol{X}_{test}$ with the deflated data matrix $\boldsymbol{E}_{test}$;
10: **end for**
11: Build a QDA classifier on the training data with generated factors $\boldsymbol{t}_{train,1}, \boldsymbol{t}_{train,2}, \ldots, \boldsymbol{t}_{train,\tilde{m}}$;
Use this QDA model to predict the test data with $\boldsymbol{t}_{test,1}, \boldsymbol{t}_{test,2}, \ldots, \boldsymbol{t}_{test,\tilde{m}}$;
Record the number of misclassifications in the test set $n_{\tilde{m}}$;
12: **Output**:
The number of misclassifications in the test set $n_{\tilde{m}}$.

3.2.2.3 Explanation of the algorithm

Randomised Initialisation

The gradient descent method is commonly applied in nonconvex optimisation, though the global optimum cannot be guaranteed under non-convexity. Under non-convexity gradient descent tends to stop at a local optimum or even a saddle point. Nevertheless, repeated randomised initialisation can help in finding a better stable point. In this algorithm, for each $\boldsymbol{w}_m$ we initialise the searching algorithm at certain number of randomised initial points in the p-dimensional feature space and obtain

q stable points $\tilde{\boldsymbol{w}}_{ml}$ where $1 \leqslant l \leqslant q$. The selected $\tilde{\boldsymbol{w}}_m$ is the one corresponding to the smallest objective value $\mathbb{J}(\tilde{\boldsymbol{w}}_{ml})$ among all $\tilde{\boldsymbol{w}}_{ml}$'s. Here q can be regarded as a hyperparameter to tune. In general, the larger q is, the more comprehensive this method can be. Nevertheless, computation cost needs to be taken into consideration as well and thus this q cannot be too large. In practice we set q to be $2p$. In other words, we randomly initialise the optimisation of $\boldsymbol{w}_m$ and get $2p$ candidates, and the optimal $\tilde{\boldsymbol{w}}_m$ is selected from the $2p$ candidates.

Mini-batch Stochastic Gradient Descent

Gradient descent is a widely-used optimisation algorithm and it has many variations. Stochastic gradient descent updates the value of the parameter for each example in the training dataset, batch gradient descent updates the value of the parameters after the whole batch of training data have been evaluated, while mini-batch stochastic gradient descent is a balance between the efficiency of stochastic gradient descent and the robustness of batch gradient descent. It splits the training dataset into small batches that are used to calculate loss and then update model coefficients after each small batch. The batch size can also be determined via cross-validation or via some prior knowledge of the data. Since the sample size of most NIR data sets are limited to a hundred or couple of hundred most, the mini-batch size cannot be too large, while too small batch size will need extra time to obtain convergence. In our real data examples and simulations we set our batch size to be 24.

Constrained Gradient Descent

As in this case $\boldsymbol{w}_m$ needs to be of norm one, adaptation of the gradient descent method with unit-norm constraint can be employed (Douglas et al., 2000).

Following Douglas et al. (2000), in the constrained gradient descent $\boldsymbol{w}_m$ can be updated by:

$$\boldsymbol{w}_m^{k+1} = \frac{\boldsymbol{w}_m^k - \boldsymbol{g}_m^k}{\sqrt{1 - 2(\boldsymbol{w}_m^k)^T \boldsymbol{g}_m^k + ||\boldsymbol{g}_m^k||^2}}, \tag{3.2.19}$$

where the iteration time k, $0 \leqslant k \leqslant K-1$.

In formula (3.2.19) the scaled gradient

$$\boldsymbol{g}_m^k = r_k \bigtriangledown \mathbb{J}(\boldsymbol{w}_m | \boldsymbol{w}_m = \boldsymbol{w}_m^k), \tag{3.2.20}$$

where r_k is the learning rate in the k-th iteration of gradient descent and $\bigtriangledown \mathbb{J}(\boldsymbol{w}_m | \boldsymbol{w}_m = \boldsymbol{w}_m^k)$ is the gradient of the objective function when $\boldsymbol{w}_m = \boldsymbol{w}_m^k$.

In the training of gradient descent based algorithms the choice of the learning rate is one of the most tricky and important parts (Bishop et al., 1995). It is often useful to reduce learning rate as the training progresses. Learning rate can be reduced in a time-based decay, a step-based decay, or an exponential decay (Ruder, 2016b). Here we let the learning rate decay with the number of iterations, i.e. with time. We set $r^k = \frac{1}{k}$, where k is the number of iterations. As the learning rate decays with the iteration time the max iteration time cannot be too large. Here we set the max number of iterations to be 500.

Finally, as the analytic solution to the gradient of the above objective function is too complicated to be calculated directly, we use the numerical gradient to approximate it in practice (Quarteroni et al., 2010).

Convergence

As the optimised object $\boldsymbol{w}_m$ is of dimension p, here we have a high dimensional non-convex optimisation problem and it is not very likely to converge to the global optimum easily. Thus, to terminate the algorithm at a feasible point, the algorithm is deemed to have converged if the following conditions are satisfied.

In this high dimensional optimisation problem we set the condition of convergence to be in three aspects: 1) The value of the objective function $\mathbb{J}(\boldsymbol{w}_m)$ should be stabilised; 2) The loading vector $\boldsymbol{w}_m$ should be steady; 3) The norm of the gradient should be close to zero. To be more specific, in practice the criterion of convergence is set to be that in 5 successive iterations the following 3 conditions hold: 1) The five objective values cannot differ by more than 1%; 2) The norms of the differ-

ence of the 5 successive $\boldsymbol{w}_m$'s do not exceed 0.1, i.e. $\max||\boldsymbol{w}_m^{l_1} - \boldsymbol{w}_m^{l_2}|| \leqslant 0.1$ for any $k \leqslant l_1 < l_2 \leqslant k+4$; and 3). The norm of the gradient is no larger than 10.

If the above three conditions are satisfied, the algorithm is deemed to have converged and the $\boldsymbol{w}_m$ corresponding to the smallest objective value among the five is regarded as the local optimiser or as we call it, a stable point. Otherwise the convergence is not reached and we continue the iteration, until either we reach convergence or the max number of iterations is reached.

Deflation and Orthogonalisation

In the algorithm, the j-th and the m-th component $\boldsymbol{t}_j$ and $\boldsymbol{t}_m$ ($j \neq m$) are required to be orthogonal. The orthogonalisation is achieved by setting:

$$\boldsymbol{E} = \boldsymbol{X}(\boldsymbol{I} - \frac{\boldsymbol{w}_m \boldsymbol{w}_m^T \boldsymbol{X}^T \boldsymbol{X}}{\boldsymbol{w}_m^T \boldsymbol{X}^T \boldsymbol{X} \boldsymbol{w}_m}), \tag{3.2.21}$$

where $\boldsymbol{E}$ is the deflated matrix of $\boldsymbol{X}$. By doing this, the generated components $\boldsymbol{t}_j$ and $\boldsymbol{t}_m$ are guaranteed to be orthogonal, i.e. we have orthogonal scores. It is important to have orthogonal scores, otherwise when using the scores in subsequent QDA classification there might be collinearity problem.

To check the orthogonality of components, we consider two successive $\boldsymbol{t}_m$ and $\boldsymbol{t}_{m+1}$ where $\boldsymbol{t}_m = \boldsymbol{X}\boldsymbol{w}_m$ and $\boldsymbol{t}_{m+1} = \boldsymbol{E}\boldsymbol{w}_{m+1}$. Then we have

$$\begin{aligned}
&\boldsymbol{t}_m^T \cdot \boldsymbol{t}_{m+1} \\
=&(\boldsymbol{X}\boldsymbol{w}_m)^T \boldsymbol{E}\, \boldsymbol{w}_{m+1} \\
=&\boldsymbol{w}_m^T \boldsymbol{X}^T \boldsymbol{X}(\boldsymbol{I} - \frac{\boldsymbol{w}_m \boldsymbol{w}_m^T \boldsymbol{X}^T \boldsymbol{X}}{\boldsymbol{w}_m^T \boldsymbol{X}^T \boldsymbol{X} \boldsymbol{w}_m})\boldsymbol{w}_{m+1} \\
=&(\boldsymbol{w}_m^T \boldsymbol{X}^T \boldsymbol{X} - \frac{\boldsymbol{w}_m^T \boldsymbol{X}^T \boldsymbol{X} \boldsymbol{w}_m \boldsymbol{w}_m^T \boldsymbol{X}^T \boldsymbol{X}}{\boldsymbol{w}_m^T \boldsymbol{X}^T \boldsymbol{X} \boldsymbol{w}_m})\boldsymbol{w}_{m+1} \\
=&(\boldsymbol{w}_m^T \boldsymbol{X}^T \boldsymbol{X} - \boldsymbol{w}_m^T \boldsymbol{X}^T \boldsymbol{X})\boldsymbol{w}_{m+1} \\
=&\boldsymbol{0} \cdot \boldsymbol{w}_{m+1} = \boldsymbol{0},
\end{aligned}$$

i.e. $\boldsymbol{t}_m$ and $\boldsymbol{t}_{m+1}$ are orthogonal.

In short, every time when obtaining a new component $\boldsymbol{t}_m$ the data matrix $\boldsymbol{X}$ is deflated using formula (3.2.21) and then the following components are guaranteed to be orthogonal to $\boldsymbol{t}_m$.

3.3 Simulations

In this section we use one near infrared spectral data example and four simulations to illustrate our algorithm in high-dimensional classification.

As we discussed, QDA can well utilise the variance heterogeneity between the two classes, while LDA mainly uses the variance-regularised group mean difference information. Thus QDA-based methods are expected to work better than LDA-based methods when there is variance heterogeneity between the two classes and the mean difference is not discriminating enough. Figure 3.3.1 shows an illustrative example of this situation. In Figure 3.3.1, samples from class 1 and class 2 follow normal distributions with different means and variances. When LDA is used as the classifier, it implicitly assumes the two classes to have same variance in this direction, then we can hardly separate the two classes merely using the mean difference information. However, when QDA is used as the classifier it can use both the mean difference information and the variance heterogeneity of the two classes to better separate them.

In this section, we consider four simulation scenarios. In the first two scenarios, there is no mean difference between the two classes but heterogeneity in variance exists. In the third scenario, there is a highly noisy mean difference direction. While in the last scenario, the mean difference is sufficiently large to separate the two classes. We compare the performance of the proposed method and PCA-QDA, PLS-QDA, PCA-LDA and PLS-DA under these four scenarios.

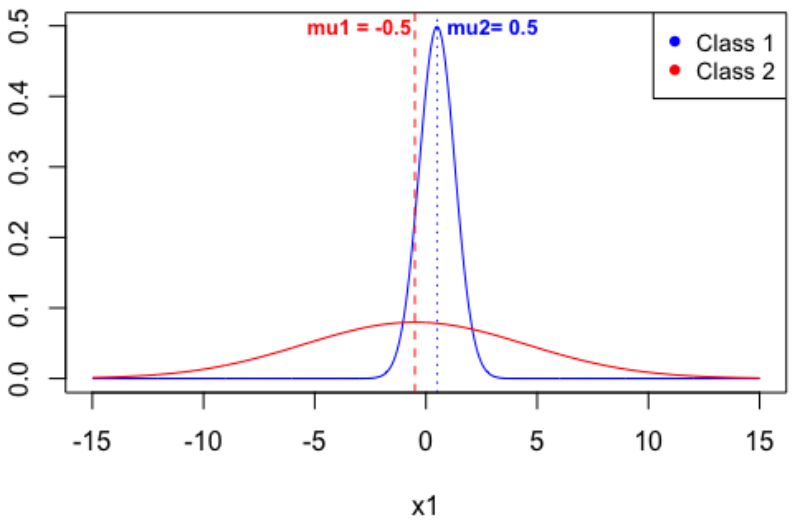


Figure 3.3.1. An illustrative example of the variance heterogeneity. In the $\boldsymbol{x}_1$ direction samples from class 1 follow $\boldsymbol{N}(0.5, 0.8^2)$ while samples from class 2 follow $\boldsymbol{N}(-0.5, 5^2)$.

3.3.1 Scenario 1

In this subsection we consider discrimination of two groups with same mean but distinct covariance structure. The composition of data is: 3 discriminative variables showing distinct covariance structure and 7 noise variables with larger variance than the discriminative variables. Here we assume our data is composed of 10 variables following normal distributions with zero means but different variances. In the first scenario, we assume the two classes to have the same population mean on these 10 variables, in other words, there is no significant mean difference between the two classes. Among these 10 variables we set the first three variables to be the potentially discriminating variables. Though not holding any difference in the mean location, the two classes have heterogeneous variance in these three directions. The remaining 7 variables contain Gaussian random noise of larger scale to the first 3 variables. Here we simulate a training set with 60 samples (30 of each class) and a test set with 60 samples (30 of each class). We repeat our simulation 10 times

to obtain the average classification performance. The detailed scenario setting is as follows.

Scenario Setting

For both training and test set:

Let $\mathbf{y} = (y_1, ..., y_{60})^T$, such that $y_i = 0$ for $1 \leq i \leq 30$ and $y_i = 1$ for $31 \leq i \leq 60$.

Let Σ_s, $1 \leq s \leq 3$ be 60×60 diagonal matrices with corresponding diagonals $\sigma_{s,i}$, where $1 \leq s \leq 3$ and $1 \leq i \leq 60$.

For $s = 1$, let $\sigma_{s,i} = 5^2$ for $1 \leq i \leq 30$, $\sigma_{s,i} = 0.5^2$ for $30 \leq i \leq 60$;

For $s = 2$, $\sigma_{s,i} = 0.5^2$ for $1 \leq i \leq 30$, and $\sigma_{s,i} = 5^2$ for $30 \leq i \leq 60$;

For $s = 3$, $\sigma_{s,i} = 5^2$ for $1 \leq i \leq 60$.

Consider 10 independent 60-dimensional vectors $\mathbf{x}_j$, $1 \leq j \leq 10$ such that

$$\begin{aligned} \mathbf{x}_j &\sim N(\mathbf{0}, \Sigma_1), j = 1, 2, \\ \mathbf{x}_j &\sim N(\mathbf{0}, \Sigma_2), j = 3, \\ \mathbf{x}_j &\sim N(\mathbf{0}, \Sigma_3), 4 \leq j \leq 10. \end{aligned}$$

Let $\boldsymbol{X}$ be a 60×10 matrix with $\boldsymbol{x}_j$ as its j-th column, i.e. $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{10}]$. Then a data matrix containing 60 samples (30 samples of each class) and 10 variables is generated. Both the training data $\boldsymbol{X}_{train}$ and the test data $\boldsymbol{X}_{test}$ are generated in this way. To better visualise and understand the structure of the simulated data, three illustrative scatter plots can be found in Figure 3.3.2, 3.3.3 and 3.3.4.

Though holding no difference in the group means, the heterogeneous variance structure of the two classes has discriminating power. However, large noise contained in directions $\boldsymbol{x}_4$, $\boldsymbol{x}_5$, ..., $\boldsymbol{x}_{10}$ is a challenge to feature extraction. Note that the noise directions $\boldsymbol{x}_4$ to $\boldsymbol{x}_{10}$ are set to have standard error 5 while the pooled sample standard deviation of $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ and $\boldsymbol{x}_3$ are 3.5. Then how to generate discriminative features from non-discriminative variation of larger scale is a challenge to all feature extraction and classification methods. Here we compare our QDA-adaptive method

with PCA-LDA, PCA-QDA, PLS-QDA and PLS-DA. Performance of them on the training set via 6-fold CV and on the test set can be found in Figure 3.3.5.

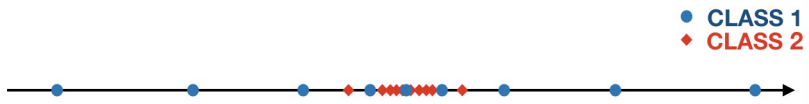


Figure 3.3.2. Illustrative scatter plot of the two classes in the direction of $\boldsymbol{x}_1$ or $\boldsymbol{x}_2$. Class 1 and class 2 are set to have same mean in direction $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ but class 1 is set to have higher variance than class 2. This heterogeneity in variance can be used to discriminate the two classes.

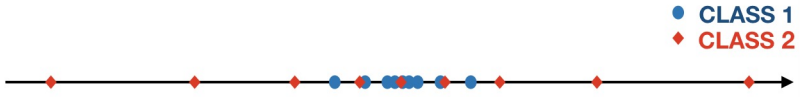


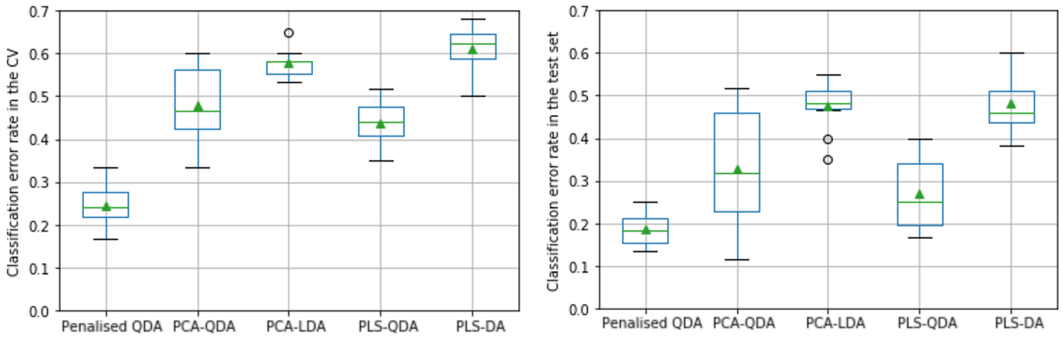
Figure 3.3.3. Illustrative scatter plot of the two classes in direction $\boldsymbol{x}_3$. Class 1 and class 2 are set to have same mean in direction $\boldsymbol{x}_3$ but class 2 is set to have greater variance than class 1. This heterogeneity in variance can be used to discriminate the two classes.

Figure 3.3.4. Illustrative scatter plot of the two classes in the other directions, $\boldsymbol{x}_4$, $\boldsymbol{x}_5$, ..., $\boldsymbol{x}_{10}$. The two classes are set to follow identical distribution on the remaining 7 directions. They are the noise directions.

Simulation Result

Figure 3.3.5 (a) shows the classification error rates of the five methods in 6-fold cross-validation while 3.3.5 (b) shows the error rate in the test set.

In the training phase, 60 training samples are randomly split into 6 folds, with each fold containing 5 samples from each class. Five folds are used to train the



(a) Classification error rate in 6-fold CV **(b)** Classification error rate on the test set

Figure 3.3.5. Classification performance of PCA-QDA, PCA-LDA, PLS-QDA, PLS-DA and our method in the first scenario. The triangular symbol in each box represents the average error rate over 10 simulations.

calibration model, one fold is used as a validation. The classification error rate in each CV fold is averaged and then recorded as the CV error rate in Figure 3.3.5 (a). The average CV error rate in 10 simulations is regarded as a measurement of the model performance. In the CV, our method achieves the lowest average error rate, 24.5%, followed by PLS-QDA with 43.7%, PCA-QDA with 48.0%, PCA-LDA with 57.8% and at last PLS-DA with 61.2% error rate. In this scenario, since there is no group mean difference LDA-based methods, PCA-LDA and PLS-DA, almost have no discriminative power. Figure 3.3.5(b) shows the classification error rate in the test set. The lowest average error rate 18.5% is obtained by our QDA-adaptive method, followed by PLS-QDA with 26.8%, PCA-QDA with 32.8%, PCA-LDA with 47.5% and at last PLS-DA with 48.2% error rate. Similar to the CV result, LDA-based methods have very low discriminating power in this scenario. Furthermore, among all three QDA-based methods our proposed method minimises the impact of the large noise by giving very small weight to the PCA part. The value of the weight α varies from 0.2 to 0.6 in 10 repetitions. Also, we can observe from Figure 3.3.5 that all QDA based methods achieve lower error rate in the test set than in the CV. This can be owing to the larger training sample size we have on the test

phase, known that QDA based methods are sensitive to sample size.

Conclusion

This simulation shows when there is no significant difference in the group mean but non-discriminative features containing large within group variability, LDA-based methods, such as PCA-LDA and PLS-DA can hardly distinguish the two classes, while PCA-QDA and PLS-QDA suffer from the large noise and thus discriminating features cannot be easily generated with these two methods. In the meanwhile, our QDA-adaptive method utilises the variance heterogeneity of the two groups and is free from the deficiency of PCA. This is accomplished by balancing between PCA and QDA. In this case a very small weight is given to the PCA part and the influence of the large and non-discriminative within group variation can be reduced.

3.3.2 Scenario 2

In this scenario we consider discrimination of two groups with same mean but distinct covariance structure. The composition of data is: 3 discriminative variables showing distinct covariance structure and 27 less noisy variables.

In the first scenario we show how dimension reduction and classification methods perform when there is no significant mean difference but a small number of noise variables which are of larger scales to the discriminative ones. In the second scenario, we will examine the performance of the above five methods when there are a large number of noise variables containing small disturbance. Also, in this scenario the mean difference is set to be zero.

We assume the data contain 30 variables with zero mean, 3 of them are discriminative variables as in the first scenario, 27 of them contain Gaussian noise, but of smaller scale. As before, both the training set and the test set contain 60 samples (30 samples each class). We repeat our simulation 10 times to obtain the average classification performance. The detailed simulation setting is found as follows.

Scenario Setting

For both training and test set:

Let $\mathbf{y} = (y_1, ..., y_{60})^T$, such that $y_i = 0$ for $1 \leq i \leq 30$, and $y_i = 1$ for $31 \leq i \leq 60$.

Let $\mathbf{\Sigma}_s$, $1 \leq s \leq 3$ be 60×60 diagonal matrices with corresponding diagonals $\sigma_{s,i}$, where $1 \leq s \leq 3$ and $1 \leq i \leq 60$.

For $s = 1$, let $\sigma_{s,i} = 5^2$ for $1 \leq i \leq 30$, $\sigma_{s,i} = 0.5^2$ for $30 \leq i \leq 60$;

For $s = 2$, $\sigma_{s,i} = 0.5^2$ for $1 \leq i \leq 30$, and $\sigma_{s,i} = 5^2$ for $30 \leq i \leq 60$;

For $s = 3$, $\sigma_{s,i} = 1$ for $1 \leq i \leq 60$.

Consider 30 independent 60-dimensional vectors $\mathbf{x}_j$, $1 \leq j \leq 30$ such that

$$\mathbf{x}_j \sim N(\mathbf{0}, \Sigma_1), j = 1, 2,$$

$$\mathbf{x}_j \sim N(\mathbf{0}, \Sigma_2), j = 3,$$

$$\mathbf{x}_j \sim N(\mathbf{0}, \Sigma_3), 4 \leq j \leq 30.$$

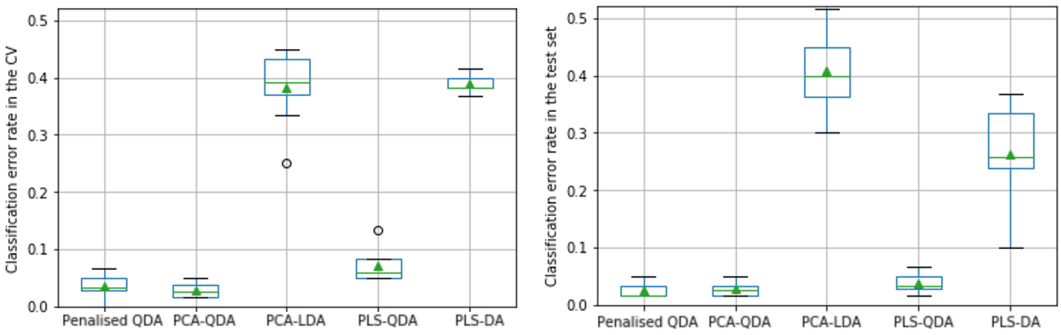
Let $\boldsymbol{X}$ be a 60×30 matrix with $\boldsymbol{x}_j$ as its $j-th$ column. Both the training data $\boldsymbol{X}_{train}$ and the test data $\boldsymbol{X}_{train}$ are generated in this way.

The difficulty in this scenario lies in two aspects. 1) There is no significant mean difference, 2) a large number of noise variables have been included in the data, though of small scale. How to extract discriminative features from a large number of non-discriminative features is the challenge to all dimension reduction and classification methods. Performance of the above five methods is described below.

Simulation Result

Performance of PCA-LDA, PCA-QDA, PLSQDA, PLS-DA and our QDA-based algorithm in the training set via 6-fold CV and in the test set can be found in Figure 3.3.6. As in the previous simulation the left subfigure shows the classification error rate of the five methods in the training set via 6-fold cross-validation over 10 repeti-

tions. The right subfigure shows the corresponding error rates in the test set over 10 repetitions. From Figure 3.3.6 (a), our method and PCA-QDA, PLS-QDA achieve comparably low average error rates on the training set, which are 3.5%, 2.9% and 7.1% respectively. The average CV classification error rates of PCA-LDA and PLS-DA are 38.3% and 39.0% respectively. In this scenario, both PCA-LDA and PLS-DA have low discriminative power. Figure 3.3.5 (b) shows the classification error rate in the test set. The lowest average error rate 2.5% is obtained by our QDA-adaptive method, which means this method only misclassified around 1.5 samples on average out of 60 samples. Meanwhile, PCA-QDA and PLS-QDA achieve 2.9% and 3.8% error rate on the test set, while that of PCA-LDA and PLS-DA are 40.8% and 26.3% respectively.



(a) Classification error rate in 6-fold CV

(b) Classification error rate on the test set

Figure 3.3.6. Classification performance of PCA-QDA, PCA-LDA, PLS-QDA, PLS-DA and our QDA-based method under the second scenario

Conclusion

This scenario shows when there is no difference in the group means, PCA-LDA and PLS-DA have low discriminative power, while PCA-QDA, PLS-QDA and our QDA-adaptive method utilise the heterogeneity in variance to separate the two groups and achieve high classification accuracy. Moreoever, generally speaking higher classification accuracy is obtained in this scenario, compared with the first scenario. This is because in this scenario discriminative features happen to be of

larger scale, i.e. containing higher variation than the non-discriminative variables and as a result PCA and PLS suffer less from the noise.

3.3.3 Scenario 3

In scenario 3 we consider discrimination of two groups with small mean difference and distinct covariance structure. The composition of data is: 2 discriminative variables showing small mean difference and distinct covariance structure, 1 variable showing no mean difference but distinct covariance, 5 non-discriminative variables containing large Gaussian noise, 5 variables containing small Gaussian noise.

In this scenario we simulate a situation in which the mean difference between the two groups exists but noise is also contained in this mean difference direction. In other words, there is a mean difference between the two groups but the difference is not sufficiently large when compared with the within-group variability in this direction. Specifically, in the direction of $\boldsymbol{x}_1$, class 1 follows a normal distribution with mean 0.5 and standard deviation 5, i.e. $\boldsymbol{x}_{1i}\,|\,y=0 \sim \boldsymbol{N}(0.5,25)$ for $1 \leqslant i \leqslant 30$ while class 2 follows a normal distribution with mean -0.5 and standard deviation 1, i.e. $\boldsymbol{x}_{1i}\,|\,y=0 \sim \boldsymbol{N}(-0.5,1)$ for $31 \leqslant i \leqslant 60$. A mean difference of 1 is contained in the $\boldsymbol{x}_1$ direction. Similarly, a mean difference of 1 is contained in the $\boldsymbol{x}_2$ direction. Meanwhile, $\boldsymbol{x}_3$ contains heterogeneity in variance but no mean difference, as in the previous scenarios. Apart from these, 5 variables containing large Gaussian noise and 5 variables containing small Gaussian noise are included in the data. In scenario 1 we only included non-discriminative variables containing large Gaussian noise (mean 0 and variance 25). In scenario 2 we only included non-discriminative variables with small Gaussian noise (mean 0 and variance 1), while in this scenario we included both variables with large variance and variables with small variance. Combining with $\boldsymbol{x}_1$, $\boldsymbol{x}_1$ and $\boldsymbol{x}_3$, the variance-covariance structure becomes more complex in this case. The detailed simulation setting is found as follows.

Scenario Setting

For both training and test set:

Let $\boldsymbol{y} = (y_1, ..., y_{60})^T$, such that $y_i = 0$ for $1 \leq i \leq 30$, and $y_i = 1$ for $31 \leq i \leq 60$.

Let $\boldsymbol{\mu} = (\mu_1, ..., \mu_{60})^T$, such that $\mu_i = 0.5$ for $1 \leq i \leq 30$, and $\mu_i = -0.5$ for $31 \leq i \leq 60$.

Let $\boldsymbol{\Sigma}_s$, $1 \leq s \leq 4$ be 60×60 diagonal matrices with corresponding diagonals $\sigma_{s,i}$, where $1 \leq s \leq 4$ and $1 \leq i \leq 60$.

For $s = 1$, let $\sigma_{s,i} = 5$ for $1 \leq i \leq 30$, $\sigma_{s,i} = 0.5$ for $30 \leq i \leq 60$;

For $s = 2$, $\sigma_{s,i} = 0.5$ for $1 \leq i \leq 30$, and $\sigma_{s,i} = 5$ for $30 \leq i \leq 60$;

For $s = 3$, $\sigma_{s,i} = 3.5$ for $1 \leq i \leq 60$.

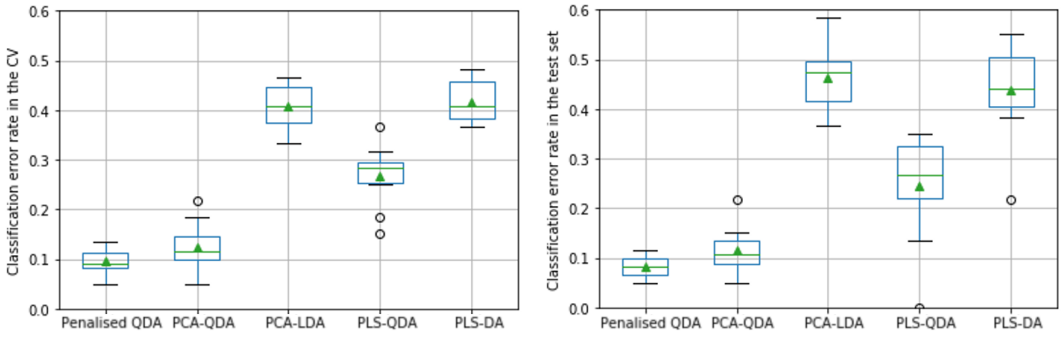
For $s = 4$, $\sigma_{s,i} = 1$ for $1 \leq i \leq 60$.

Consider 13 independent 60-dimensional vectors $\mathbf{x}_j$, $1 \leq j \leq 30$ such that

$$\begin{aligned}
\mathbf{x}_j &\sim N(\boldsymbol{\mu}, \Sigma_1), j = 1, \\
\mathbf{x}_j &\sim N(\boldsymbol{\mu}, \Sigma_2), j = 2, \\
\mathbf{x}_j &\sim N(\mathbf{0}, \Sigma_2), j = 3, \\
\mathbf{x}_j &\sim N(\mathbf{0}, \Sigma_3), 4 \leq j \leq 8 \\
\mathbf{x}_j &\sim N(\mathbf{0}, \Sigma_4), 9 \leq j \leq 13.
\end{aligned}$$

Let $\boldsymbol{X}$ be a 60×13 matrix with $\boldsymbol{x}_j$ as its j-th column. Both the training data $\boldsymbol{X}_{train}$ and the test data $\boldsymbol{X}_{test}$ are generated in this way. The difficulty in this scenario is, 1) how to identify discriminative features from non-discriminative ones, 2) as the mean difference direction is noisy, how to utilise heterogeneous variance-covariance information to assist the classification. Performance of the above five methods is displayed in Figure 3.3.7.

As in the previous simulation, Figure 3.3.7 (a) shows the classification error rates of the five methods in the training set via 6-fold cross-validation over 10 rep-



(a) Classification error rate in 6-fold CV **(b)** Classification error rate on the test set

Figure 3.3.7. Classification performance of PCA-QDA, PCA-LDA, PLS-QDA, PLS-DA and our QDA-based method under the third scenario

etitions. Figure 3.3.7 (b) shows the corresponding error rates in the test set over 10 repetitions. From Figure 3.3.7 (a) our method and PCA-QDA achieve relatively low average error rate in the training set, which are 8.2% and 12.3% respectively. The average CV classification error rates of classic PLS-QDA, PCA-LDA and PLS-DA are 26.8%, 40.8% and 41.7% respectively. Though there is a mean difference in this scenario, PCA-LDA and PLS-DA fail to use the heterogenous variance of the two groups and consequently misclassify more samples than the QDA-based methods. Among three QDA based methods, the proposed method is adaptive to the QDA classifier and hence obtains higher accuracy. Here the penalty weight of our algorithm varies from 2 to 3 over 10 simulations. The best number of components varies from 3 to 6 over the 10 simulations.

Figure 3.3.5 (b) shows the classification error rates in the test set. The lowest average error rate 8.8% is obtained by our QDA-adaptive method, followed by 11.5% by PCA-QDA. Classification error rates of PLS-QDA, PCA-LDA and PLS-DA are 24.5%, 46.2% and 47.8% respectively.

Conclusion

This scenario shows when there is a noisy mean difference direction and this direction also contains heterogeneity in variance, PCA-LDA and PLS-DA fail to utilise

this heterogeneity information and possess low discriminative power. QDA-based high dimensional classification methods utilise heterogeneous variance-covariance information to assist the classification and achieve higher classification accuracy.

3.3.4 Scenario 4

In this scenario, we consider discrimination of two groups with large mean difference and distinct covariance structure. The composition of data is: 2 discriminative variables showing large mean difference and distinct covariance structure, 2 variables showing no mean difference but distinct covariance structure, 8 non-discriminative variables containing large Gaussian noise, 8 variables containing small Gaussian noise.

Here we simulate a scenario where there is a large mean difference compared with the variability in that direction. In this situation we have a significant mean difference as well as more complex covariance structure than before. The first two variables $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ contain the mean difference with heterogeneous variance as in the previous scenarios, $\boldsymbol{x}_3$ and $\boldsymbol{x}_4$ contain heterogeneity in variance but no mean difference. Eight variables $\boldsymbol{x}_5$, $\boldsymbol{x}_6$, $\cdots$, $\boldsymbol{x}_{12}$ contain large Gaussian noise while eight variables $\boldsymbol{x}_{13}$, $\boldsymbol{x}_{14}$, $\cdots$, $\boldsymbol{x}_{20}$ contain small Gaussian noise. The detailed simulation setting is found as follows.

Scenario Setting

For both training and test set:

Let $\boldsymbol{y} = (y_1, ..., y_{60})^T$, such that $y_i = 0$ for $1 \leq i \leq 30$, and $y_i = 1$ for $31 \leq i \leq 60$.

Let $\boldsymbol{\mu} = (\mu_1, ..., \mu_{60})^T$, such that $\mu_i = 2$ for $1 \leq i \leq 30$, and $\mu_i = -2$ for $31 \leq i \leq 60$.

Let $\boldsymbol{\Sigma}_s$, $1 \leq s \leq 4$ be 60×60 diagonal matrices with corresponding diagonals $\sigma_{s,i}$, where $1 \leq s \leq 4$ and $1 \leq i \leq 60$.

For $s = 1$, let $\sigma_{s,i} = 5^2$ for $1 \leq i \leq 30$, $\sigma_{s,i} = 0.5^2$ for $30 \leq i \leq 60$;

For $s = 2$, $\sigma_{s,i} = 0.5^2$ for $1 \leq i \leq 30$, and $\sigma_{s,i} = 5^2$ for $30 \leq i \leq 60$;

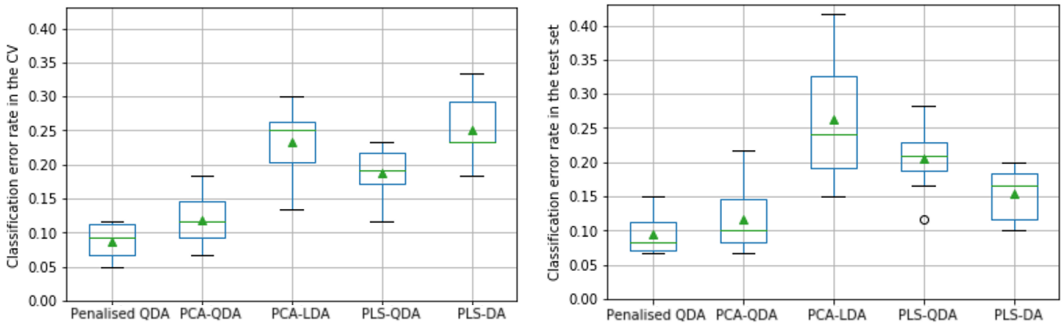
For $s = 3$, $\sigma_{s,i} = 3.5^2$ for $1 \leq i \leq 60$.

For $s = 4$, $\sigma_{s,i} = 1$ for $1 \leq i \leq 60$.

Consider 13 independent 60-dimensional vectors $\mathbf{x}_j$, $1 \leq j \leq 30$ such that

$$\mathbf{x}_j \sim N(\boldsymbol{\mu}, \Sigma_1), j = 1,$$
$$\mathbf{x}_j \sim N(\boldsymbol{\mu}, \Sigma_2), j = 2,$$
$$\mathbf{x}_j \sim N(\mathbf{0}, \Sigma_1), j = 3,$$
$$\mathbf{x}_j \sim N(\mathbf{0}, \Sigma_2), j = 4,$$
$$\mathbf{x}_j \sim N(\mathbf{0}, \Sigma_3), 5 \leq j \leq 12$$
$$\mathbf{x}_j \sim N(\mathbf{0}, \Sigma_4), 13 \leq j \leq 20.$$

Let $\boldsymbol{X}$ be a 60×20 matrix with $\boldsymbol{x}_j$ as its j-th column. Both the training data $\boldsymbol{X}_{train}$ and the test data $\boldsymbol{X}_{test}$ are generated in this way. The challenge is to identify a small number of discriminative variables $\boldsymbol{x}_1$ to $\boldsymbol{x}_4$ from a large number of noise variables from $\boldsymbol{x}_5$ to $\boldsymbol{x}_{20}$. Performance of the above five methods is shown in the following figure.



(a) Classification error rate in 6-fold CV

(b) Classification error rate on the test set

Figure 3.3.8. Classification performance of PC-DA (PCA-QDA, PCA-LDA), PLS-QDA, PLS-DA and our QDA-based method under the fourth scenario

Figure 3.3.8 (a) shows the classification error rates of the five methods in the training set via 6-fold cross-validation over 10 repetitions. Figure 3.3.8 (b) shows the corresponding error rates in the test set over 10 repetitions. From Figure 3.3.8

(a) our method and PCA-QDA achieve relatively low average error rates, which are 10.0% and 11.8% respectively, while the average CV classification error rates of classic PLS-QDA, PCA-LDA and PLS-DA are 18.8%, 23.3% and 25.2% respectively. In Figure 3.3.5 (b), the lowest average test error rate 10.2% is obtained by our QDA-adaptive method, followed by 11.7% by PCA-QDA. Classification error rates of the remaining three methods are 20.5%, 26.2% and 15.3% respectively.

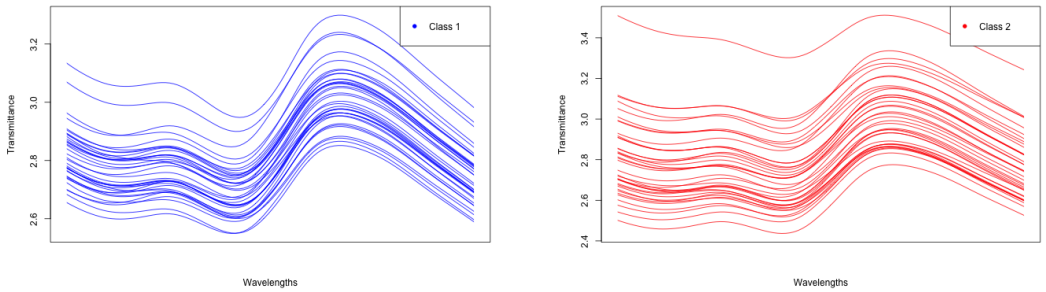
Conclusion

This scenario shows when there is a significant and large mean difference, LDA-based methods can utilise this information and obtain significantly higher classification accuracy than in the previous scenarios. Nevertheless, assistance of the second order information is still beneficial to classification. Furthermore, our method generate features specialised for QDA and accordingly achieve even higher classification accuracy with the QDA classifier than the other methods.

3.4 Examples

The real data example is based on the wheat NIR spectral data set. The wheat data set contains NIR transmittance spectra of 9 varieties of unground wheat, on the basis of known provenance. In this section variety 3 and variety 8 are selected as the two classes, so as to obtain two relative large and balanced groups. Another possible choice is variety 1 versus variety 5. However, these two groups can be well separated by classic PCA-QDA with 5.7% error rate, while the performance of PCA-QDA on separating variety 3 and variety 8 is much weaker, achieving only about 70% accuracy. Thus here we choose variety 3 and variety 8 as the two groups, to improve the performance of PCA-QDA on separating them.

Figure 3.4.1 shows the spectra of these two classes. In the figure, each curve represents the spectrum of a sample. The blue curves represent spectra of samples from class 1 (variety 3) while the red curves represent those from class 2 (variety 8). The two classes contain 36 and 37 samples respectively, while the number of wave-



(a) Spectra of samples from class 1 **(b)** Spectra of samples from class 2

Figure 3.4.1. Spectra plot of class 1 (variety 3) and class 2 (variety 8)

lengths is 100. The number of variables exceeds the number of observations and the variables are highly correlated, thus a feature extraction technique is required in this case.

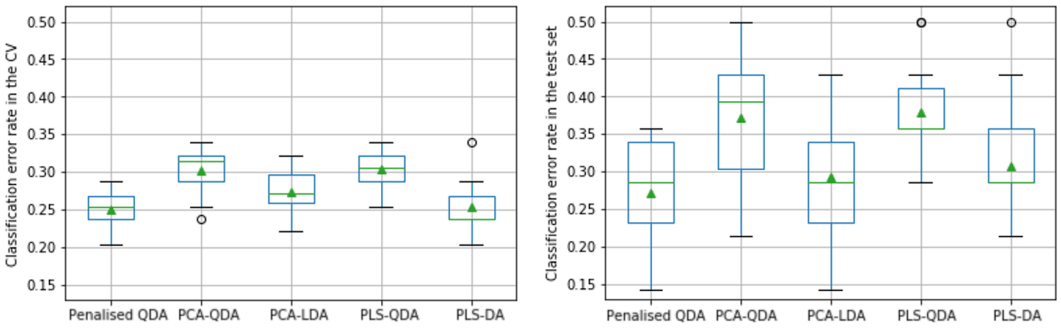
Here the number of wavelengths is 100. If implementing the above algorithm directly with the raw data, we will face a 100-dimensional optimisation problem. Further considering the randomised initiallisation and the cross validation required by the algorithm, the whole procedure will be computationally very expensive. As a result, though not ideally, we replace the raw spectra with the top 20 PCs of them, and use these PCs as the spectra data in the algorithm. Top 20 PCs contain most of the information (above 99% of the variability) thus we will not lose much information with 20 PCs. 99% variability indicates that most of the common information of the two groups as well as the discriminative information is retained. However, common information or noise are bound with the discriminative information in the PCs, thus a feature extraction or variable selection is better applied before the classification. Meanwhile, using 20 PCs simplifies the original 100-dimensional optimisation problem to 20-dimensional and significantly relieves the computation burden. This is the reason why we use 20 PCs. In this example, we use the raw spectra instead of the derivatives, as the derivatives will not lead to significantly better classification performance in this particular example. Note that the selection

of derivatives and raw spectra mostly depends on the corresponding classification performances.

Our QDA-adaptive algorithm is applied to this NIR data to extract QDA specialised features and simultaneously reduce dimension. When our penalised QDA-based method is selected as the feature extraction method, QDA is naturally selected as the subsequent classifier. The combined use of our feature extraction method and QDA is compared with PCA-QDA, PCA-LDA, PLS-QDA and PLS-DA. Classification performance of these methods in the wheat example can be found in Figure 3.4.2.

The data set is too limited to afford a separate training and test set. Thus we use double CV. In double CV, data are randomly split into a training set of 59 samples and a test set of 14 samples 10 times. In each split, 7 samples from each class are selected into the test set, in consideration of data balance. In terms of the training set, the remaining 59 samples are split into 6 folds with 9 or 10 samples, with each fold containing the same number of samples from each class, or as close as possible. We use 5 folds to train the calibration model, one fold to validate. Once all the folds are traversed as the cross-validation fold, the average error rates in the cross-validation fold can be obtained. The penalty weight α together with its best number of components which gives the smallest average error rate in the CV is selected. Then we use our QDA-adaptive model together with the selected parameter values to predict labels for the test samples. The classification error rate in the test set is regarded as the evaluation criterion of models. After 10 times of splitting, we compare the classification performance of our penalised QDA-based method, with PCA-QDA, PCA-LDA, PLS-QDA and PLS-DA. In PCA-QDA and PCA-LDA, PCs are obtained as solution to the eigendecomposition of the total sample covariance matrix and then used in QDA and LDA to classify samples. In PLS-QDA and PLS-DA, components are generated in the classic PLS manner and used in QDA and linear regression to predict the labels of samples. For PLS-DA, the predicted value

is compared with a threshold value. For example if the labels are set to be 1 and -1 then the threshold is often set to be 0. If the predicted value is greater than 0 the sample is set to class 1 otherwise it is set to class 2. All four methods as well as our QDA-based method are implemented under the same cross-validation structure and the best number of components used in each methods is chosen by cross-validation. Figure 3.4.2 shows the classification performance of the five methods in the training and the test set.



(a) Classification error rate in 6-fold CV **(b)** Classification error rate on the test set

Figure 3.4.2. Classification performance of PCA-QDA, PCA-LDA, PLS-QDA, PLS-DA and our QDA-based method in the training and test set. The triangular symbol in each box represents the average error rate over 10 simulations.

As shown in 3.4.2 (a), in 6-fold CV the average error rate decreases from 30.2% to 24.9% from the classic PCA-QDA to our QDA-adaptive algorithm. The corresponding error rates of PCA-LDA, PLS-QDA and PLS-DA are 27.3%, 30.3% and 25.3% respectively. Our method outperforms the classic PCA-QDA and PLS-QDA and achieves comparable error rate with PLS-DA. Here the best weight α of our algorithm varies, from 1 to 3, and the best number of components varies from 4 to 6 in these 10 times of splitting.

Figure 3.4.2 (b) shows the corresponding classification error rate on the test set. Our QDA-based method obtains the lowest test error rate, 27.1%, while that of the

original PCA-QDA is 37.1%. The error rates of PCA-LDA, PLS-QDA and PLS-DA are 29.3%, 37.9% and 30.7% respectively. Our method outperforms the classic PCA-QDA and PLS-QDA, obtains slightly lower average error rate compared with PCA-LDA and PLS-DA.

In this example we only consider models with no more than 6 features for all methods. The reasons are: 1) the first 6 PCs contain more than 99% of the total variability of the data and including more features have the risk of bringing in noise, 2) the classic PC-DA (PCA-LDA and PCA-QDA), PLS-QDA and PLS-DA will not obtain better result with more than 6 components in this example, 3) Last but not least, the computation time of our method is relatively high. The first two components take only couple of seconds to converge. However, the computation time increases significantly with the increase of the number of components. The sixth component takes couple of minutes to converge (or achieving the max iteration times). Also, we need to initialise the optimisation from a few different starting points, which further multiplies the computation time. In consideration of the computation time, we do not go further than 6 components.

3.4.1 Conclusion

As an unsupervised method PCA fails to use the label information of the observations. It gives high weights to features with higher variability irrespective of whether they contribute to further analysis, such as prediction and classification. One way to avoid this deficiency is to replace PCA with supervised feature extraction method. LDA-based methods such as Direct LDA, OLDA, NLDA and RDA are widely used in high dimensional feature extraction. However, contrary to PCA, these methods attach most of the importance to the discriminating power of features while neglecting their generalisation ability. It was argued in the literature (De Jong and Kiers, 1992) that two criteria should be taken into consideration and also balanced in feature extraction, one is the generalisation ability of features, one is the

predictive ability of them. Specifically, in classification tasks the predictive ability is expressed as the power of assigning the true label to the samples, i.e. the discriminating power. In this chapter we propose a penalised QDA-based feature extraction method to simultaneously maximise the generalisation ability and the discriminating power of generated features. In this algorithm the discriminating power of a feature is measured by the QDA loglikelihood and the generalisation ability is measured by the variability contained in this feature. Our method is based on QDA instead of LDA, so that we can utilise the heterogeneity in the second order structure of the data and build a nonlinear feature extraction method. A weight parameter α is employed to balance between generalisation and discrimination, which can be determined by cross-validation. Then we use mini-batch stochastic gradient descent to search for the best loading vector of each feature.

In this chapter we use one real data experiment and four simulations to illustrate the performance of our algorithm in high dimensional classification. In the wheat example, the average classification error rate on the training set via cross-validation decreases from 30.2% to 24.9% from the classic PCA-QDA to our QDA-based method, while the average error rate on the test set decreases from 37.1% to 27.1%. Results of four simulations show: 1) when there is no mean difference between the two classes and large noise, the proposed method can utilise variance heterogeneity to separate the two classes and is less impacted by the large noise than the regular PCA-QDA and PLS-QDA, 2) when there is a noisy mean difference direction and this direction also contains heterogeneity in variance, our method can combine the mean difference information and the heterogeneity information and achieve 91.2% classification accuracy on the test set, while the corresponding error rates of PCA-QDA, PLS-QDA, PCA-LDA and PLS-DA are 11.5%, 24.5%, 46.2% and 47.8% respectively, 3) when there exists a significant mean difference, all five methods obtain better separation. However the assistance of second order information can still benefit the classification. The lowest test error rate 10.2% is achieved

by our QDA-adaptive method, while that of PCA-QDA, PLS-QDA, PLS-DA and PCA-LDA are 11.7%,15.3%, 20.5% and 26.2% respectively.

Regardless of the advantage of our penalised QDA-based method, one obvious disadvantage of this method is its high requirement of computation time. In the search of the best loading vector $\boldsymbol{w}$, we need to solve a p-dimensional nonconvex optimisation problem with norm 1 constraint. This optimisation itself takes relatively long time, and the global optimum cannot be guaranteed. To address this, we randomly initialise the optimisation process at multiple different start points and try to find the best local optimum resulted from different start points. However, randomised initialisation further multiplies the computation time. Hence, future work can be done to enhance the computational efficiency of this method.

Chapter 4

General Conclusions

As the most well-known dimension reduction technique, PCA has been extensively applied in various fields, such as chemistry, bioscience, computer science, social science, and so forth. However, when PCA is used as a preliminary dimension reduction step in developing classification rules with high-dimensional data, it has a drawback that as an unsupervised method PCA fails to use the class labels when constructing the components. As a result, its maximisation of the variance of the projected patterns is not necessarily in favour of discrimination among classes. In this thesis, we propose five solutions to this problem from the following three perspectives.

Firstly, we can enhance the performance of PCA in high-dimensional classification by combining supervised information in the feature generation step. From this perspective, we propose two methods, reweighted PCA and between PCA. In c class classification, Reweighted PCA reweights c within group covariance matrices and 1 between group covariance with c weight parameters. Between PCA first extracts dimensions from between-group covariance and then reweights the other c within-group covariance matrices with $(c-1)$ weight parameters. These parameters, as well as the best number of components to use in the subsequent classification, can be determined via cross-validation. Results in two NIR data sets, the wheat data and the paddy rice data, show that these two methods outperform the

classic PCA in both binary classification and multi-class classification with QDA.

Secondly, PCs can be generated as usual. However, instead of ranking PCs by their associated eigenvalues, we can rank them by their discriminating power in a classifier. Here we use QDA as the classifier. Based on this idea we propose two methods, reordered PCA and stepwise-reordered PCA. In reordered PCA, PCs are ranked and selected individually by their discriminating power in a univariate QDA model, while in stepwise-reordered algorithm, PCs are ranked sequentially by their joint discriminating power in a multivariate QDA model. In these two methods both the cut-off point deciding the number of components taken into the re-ranking scheme and the best number of components to use in the subsequent classification are determined via cross-validation. By implementing them on the wheat data set and the paddy rice data set we find that both methods provide higher classification accuracy than the classic PCA in both binary and multi-class classification.

Moreover, supervised feature extraction methods such as DLDA, OLDA and NLDA are widely applied in face recognition. Borrowing this idea we propose a penalised QDA-based feature extraction method which simultaneously maximises the generalisation ability and the discriminating power of features. In this algorithm the discriminating power of a feature is measured by the QDA loglikelihood and the generalisation ability is measured by the variability contained in this feature. Here we use QDA instead of LDA, so as to utilise heterogeneity in the second order structure of the data and build a nonlinear feature extraction method. A weight parameter α is employed to balance between generalisation and discrimination, and it can be determined by cross-validation. Mini-batch stochastic gradient descent has been employed to search for the best loading vector for each feature. In the binary classification with wheat data, error rate on the test set decreases from 37.1% to 27.1% from the classic PCA-QDA to our method. Besides, simulation results show that when there is no mean difference between the two classes, or when the mean difference direction is highly noisy, our QDA-based method can utilise the

covariance heterogeneity to assist the classification and is free from the deficiency of PCA. Even when there is a large mean difference between the two groups, our method can generate features specialised for QDA and achieve better classification than the classic PCA-QDA, PLS-QDA, PCA-LDA and PLS-DA.

So far we have shown five modifications of PCA from three perspectives. We compare the performance of reweighted PCA, between PCA, reordered PCA and stepwise-reordered PCA on two NIR spectra data sets, the wheat data set and the paddy rice data set. Reweighted PCA performs the best among these four methods in the binary and the three-class classification of wheat samples. Stepwise-reordered PCA outperforms the others in the binary classification of paddy rice samples, while reweighted PCA and between PCA are the joint best performers in the three-class classification of paddy rice samples. Reweighted PCA-QDA is the best performer or the joint best performers in three classification examples out of four. In other words, reweighted PCA-QDA is the most accurate algorithm among the four proposed methods. Moreover, in both three-class classification examples, the highest accuracy is obtained by the reweighted algorithm. It gives different weights to different groups and this makes the reweighted algorithm inherently more adaptive to multi-class classification. In other words, when the classification accuracy is the driven reason of choosing a dimension reduction method, or when we deal with multi-class classification problems, the reweighted PCA-QDA is probably appropriate algorithm to apply.

However, one potential concern about the reweighted algorithm is on its computation time. Reweighted PCA gives different weights to different groups, and then search for the optimal combination of weights. The computation complexity of this algorithm (as well as the between PCA) does not depend on the dimension of the data (the number of variables), but the number of classes. In c class classification a c-dimensional bounded optimisation problem is implicitly contained in the procedure of the reweighted algorithm. As a result, the computation time of

the reweighted algorithm increases significantly with the increase of the number of classes. Similar concern is with the penalised QDA based method. In the penalised QDA based feature extraction method, though the computation cost does not grow with the number of classes, a p-dimensional optimisation problem is implicitly contained in the procedure of this method, where p is the number of variables. For high dimensional data, the number of variables is usually very large. Thus the QDA based method is usually computationally intensive. On the contrary, reordered methods (reordered PCA and stepwise-reordered PCA) add a filter step in the original PCA framework and have only two parameters to tune. They are computationally inexpensive compared with the other methods proposed in this thesis. Besides, the computation complexity of the reordered algorithm and the stepwise reordered method does not grow with the number of classes. The reordered algorithms only add a filter step under the classic PCA framework, which does not require complex computation or optimisation technique. This provides the reordered algorithms high potential to replace the reweighted algorithm in multi-class classification with a large number of classes. Moreover, in the binary classification of the paddy rice data, the stepwise reordered PCA-QDA achieved the highest accuracy with only 3.8 PCs on average to outperform the classic PCA-QDA with 10 components. This further verifies the computation efficiency of the stepwise reordered method. In other words, when the computation efficiency is the main concern of the users or when we have multi-class classification problems with a large number of classes, the stepwise reordered method is more likely to be the appropriate algorithm to apply.

In terms of interpretability, all of our proposed methods have high interpretability. As we discussed, the reweighting algorithms usually attach higher weights to the between covariance to uncover the difference of group means, and the group with distinct variation information, to help generating more discriminative PCs. The reordering algorithms extract PCs with high discriminative power first. In penalised

QDA based algorithm, the generated features are linear combination of the original variables which provide the best classification with QDA while maintaining large variability. The mechanism of all proposed methods is clear and easy to understand.

In this thesis, we implement the proposed dimension reduction methods with QDA. The reason of using QDA as the classifier is twofold. First, QDA can utilise the distinct variation information of each group to better separate them. Second, the reweighted PCA algorithm assumes each group to have different covariance matrix and gives asymmetric weights to each covariance matrix. This is consistent with the QDA setting. Thus the natural classifier used with the reweighted PCA is QDA rather than LDA. LDA assumes each group to have identical covariance. However, all of our proposed methods can be easily extended to work with other classifiers. Though reweighted PCA and between PCA naturally work with QDA, they can be undoubtedly applied to LDA. Higher weight can be attached to the between covariance, to uncover the difference in group means, and higher weight can be attached to the group contributing more to the classification with LDA. The idea of reordered PCA and stepwise reordered PCA can be extended to LDA as well. Instead of ranking PCs according to their discriminative power with QDA, we rank PCs according to their discriminative power with LDA. The penalised QDA based algorithm can take the loglikelihood of LDA as the indicator of discrimination, rather than the loglikelihood of QDA, and build a penalised LDA based algorithm. All of our models can be easily extended to LDA, and even other kinds of classifiers such as SVM and logistic regression.

Future work can be done based on the abovementioned methods. 1) As mentioned in the last chapter, our penalised QDA-based feature extraction method requires high computation time. On one hand, some acceleration techniques can be employed to expedite this algorithm, such as parallel computing. On the other hand, nonconvex optimisation is a rapidly developing field in the past few years. Advanced nonconvex optimisation methods can be employed to accelerate the con-

vergence of our method. 2) Here we only apply our methods with QDA. Future work can be done to combine the proposed methods with other kinds of classifiers, such as LDA and logistic regression. 3) In this thesis we only apply our high-dimensional classification methods in NIR spectral data. The NIR spectrum of a sample can be represented by a 1-D vector. Hyperspectral image data are 3-D data cubes that measure both spectral and spatial information. It is natural to extend the methods proposed in this thesis to classify hyperspectral image data. 4) In this thesis, we have only used a few examples. These examples have shown the potential of our new methods, but more examples will be needed to see how beneficial they are in general.

Appendix A

Decomposition of total covariance

Assume we have a c-class classification problem. Firstly, let us clarify the notation used in the following derivation.

Notations:

c	The number of classes in the classification problem
n	The total sample size;
n_i	The sample size of class i;
$\boldsymbol{x}_{ij}$	The j-th sample in the i-th class;
$\bar{\boldsymbol{x}}_i$	A column vector containing the mean of class i;
$\bar{\boldsymbol{x}}$	A column vector containing the mean of all samples.

Let us start the derivation by defining the covariance matrices we need.

The covariance of class i can be denoted as:

$$\boldsymbol{S}_i = \frac{1}{n_i - 1} \sum_{j=1}^{n_i} (\boldsymbol{x}_{ij} - \bar{\boldsymbol{x}}_i)(\boldsymbol{x}_{ij} - \bar{\boldsymbol{x}}_i)^T, \tag{A.1}$$

where the superscript T denotes transpose of the vector.

The between-class sum-of-squares and products (SSP) matrix can be denoted as,

$$\boldsymbol{S}_B = \sum_{i=1}^{c} n_i (\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})^T. \tag{A.2}$$

The total covariance:

$$\boldsymbol{S}_T = \frac{1}{n-1}\sum_{i=1}^{c}\sum_{j=1}^{n_i}(\boldsymbol{x}_{ij}-\bar{\boldsymbol{x}})(\boldsymbol{x}_{ij}-\bar{\boldsymbol{x}})^T. \tag{A.3}$$

Our task is to show:

$$\boldsymbol{S}_T = \frac{1}{n-1}\Big(\sum_{i=1}^{c}(n_i-1)\boldsymbol{S}_i + \boldsymbol{S}_B\Big). \tag{A.4}$$

Here the between SSP matrix:

$$\begin{aligned}
\boldsymbol{S}_B &= \sum_{i=1}^{c} n_i(\bar{\boldsymbol{x}}_i-\bar{\boldsymbol{x}})(\bar{\boldsymbol{x}}_i-\bar{\boldsymbol{x}})^T \\
&= \sum_{i=1}^{c} n_i(\bar{\boldsymbol{x}}_i\bar{\boldsymbol{x}}_i^T - \bar{\boldsymbol{x}}\bar{\boldsymbol{x}}_i^T - \bar{\boldsymbol{x}}_i\bar{\boldsymbol{x}}^T + \bar{\boldsymbol{x}}\bar{\boldsymbol{x}}^T) \\
&= \sum_{i=1}^{c} n_i\bar{\boldsymbol{x}}_i\bar{\boldsymbol{x}}_i^T - \bar{\boldsymbol{x}}\sum_{i=1}^{c} n_i\bar{\boldsymbol{x}}_i^T - (\sum_{i=1}^{c} n_i\bar{\boldsymbol{x}}_i)\bar{\boldsymbol{x}}^T + \sum_{i=1}^{c} n_i\bar{\boldsymbol{x}}\bar{\boldsymbol{x}}^T \\
&= \sum_{i=1}^{c} n_i\bar{\boldsymbol{x}}_i\bar{\boldsymbol{x}}_i^T - n\bar{\boldsymbol{x}}\bar{\boldsymbol{x}}^T
\end{aligned} \tag{A.5}$$

From formula (A.1) the covariance matrix $\boldsymbol{S}_i$ of class i:

$$\begin{aligned}
\boldsymbol{S}_i &= \frac{1}{n_i-1}\sum_{j=1}^{n_i}(\boldsymbol{x}_{ij}-\bar{\boldsymbol{x}}_i)(\boldsymbol{x}_{ij}-\bar{\boldsymbol{x}}_i)^T, \\
&= \frac{1}{n_i-1}\sum_{j=1}^{n_i}(\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T - \bar{\boldsymbol{x}}_i\boldsymbol{x}_{ij}^T - \boldsymbol{x}_{ij}\bar{\boldsymbol{x}}_i^T + \bar{\boldsymbol{x}}_i\bar{\boldsymbol{x}}_i^T), \\
&= \frac{1}{n_i-1}\Big(\sum_{j=1}^{n_i}\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T - \sum_{j=1}^{n_i}\bar{\boldsymbol{x}}_i\boldsymbol{x}_{ij}^T - \sum_{j=1}^{n_i}\boldsymbol{x}_{ij}\bar{\boldsymbol{x}}_i^T + \sum_{j=1}^{n_i}\bar{\boldsymbol{x}}_i\bar{\boldsymbol{x}}_i^T\Big), \\
&= \frac{1}{n_i-1}\Big(\sum_{j=1}^{n_i}\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T - 2n_i\bar{\boldsymbol{x}}_i\bar{\boldsymbol{x}}_i^T + n_i\bar{\boldsymbol{x}}_i\bar{\boldsymbol{x}}_i^T\Big), \\
&= \frac{1}{n_i-1}\Big(\sum_{j=1}^{n_i}\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T - n_i\bar{\boldsymbol{x}}_i\bar{\boldsymbol{x}}_i^T\Big).
\end{aligned} \tag{A.6}$$

Namely,

$$(n_i-1)\boldsymbol{S}_i = \sum_{j=1}^{n_i}\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T - n_i\bar{\boldsymbol{x}}_i\bar{\boldsymbol{x}}_i^T, \tag{A.7}$$

and

$$\sum_{i=1}^{c}(n_i-1)\boldsymbol{S}_i=\sum_{i=1}^{c}\sum_{j=1}^{n_i}\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T-\sum_{i=1}^{c}n_i\bar{\boldsymbol{x}}_i\bar{\boldsymbol{x}}_i^T. \tag{A.8}$$

Similarly, the total covariance matrix:

$$\boldsymbol{S}_T=\frac{1}{n-1}\Big(\sum_{i=1}^{c}\sum_{j=1}^{n_i}\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T-\sum_{i=1}^{c}n_i\bar{\boldsymbol{x}}\bar{\boldsymbol{x}}^T\Big). \tag{A.9}$$

According to equation (A.5) and equation (A.8), The total covariance:

$$\begin{aligned}\boldsymbol{S}_T&=\frac{1}{n-1}\Big(\sum_{i=1}^{c}\sum_{j=1}^{n_i}\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T-\sum_{i=1}^{c}n_i\bar{\boldsymbol{x}}\bar{\boldsymbol{x}}^T\Big)\\&=\frac{1}{n-1}\Big(\sum_{i=1}^{c}\sum_{j=1}^{n_i}\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T-\sum_{i=1}^{c}n_i\bar{\boldsymbol{x}}_i\bar{\boldsymbol{x}}_i^T+\sum_{i=1}^{c}n_i\bar{\boldsymbol{x}}_i\bar{\boldsymbol{x}}_i^T-\sum_{i=1}^{c}n_i\bar{\boldsymbol{x}}\bar{\boldsymbol{x}}^T\Big)\\&=\frac{1}{n-1}\Big(\sum_{i=1}^{c}(n_i-1)\boldsymbol{S}_i+\boldsymbol{S}_B)\Big).\end{aligned} \tag{A.10}$$

Appendix B

Comparison of classification error rates of the classic PCA-QDA and our reweighted PCA-QDA (rPCA-QDA) on the wheat data set

Varieties	PCA-QDA error rate	Data used in PCA-QDA	rPCA-QDA error rate	Data used in rPCA-QDA
Variety 1 vs 3	21.6%	Raw data	19.3%	Raw data
Variety 1 vs 4	17.3%	1st Derivative	14.8%	1st Derivative
Variety 1 vs 5	5.7%	1st Derivative	5.7%	1st Derivative
Variety 1 vs 8	11.2%	1st Derivative	9.0%	1st Derivative
Variety 1 vs 9	7.9%	Raw data	7.9%	2nd Derivative
Variety 3 vs 4	18.5%	Raw data	15.4%	1st Derivative
Variety 3 vs 5	11.5%	Raw data	10.6%	2nd Derivative
Variety 3 vs 8	28.8%	1st Derivative	26.0%	2nd Derivative
Variety 3 vs 9	14.3%	1st Derivative	9.5%	2nd Derivative
Variety 4 vs 5	16.5%	1st Derivative	12.4%	1st Derivative
Variety 4 vs 8	19.7%	1st Derivative	16.7%	1st Derivative
Variety 4 vs 9	25.0%	1st Derivative	21.4%	Raw data
Variety 5 vs 8	8.6%	Raw data	7.6%	Raw data
Variety 5 vs 9	15.8%	1st Derivative	13.7%	1st Derivative
Variety 8 vs 9	7.8%	1st Derivative	6.3%	1st Derivative

Table B1

Comparison of classification error rates of the classic PCA-QDA and our reweighted PCA-QDA (denoted as rPCA-QDA on the table) on 6 varieties of wheat

Bibliography

Aguilera, A. M., Escabias, M., Valderrama, M. J., and Aguilera-Morillo, M. C. (2013). Functional analysis of chemometric data. *Open Journal of Statistics*, 3(05):334.

Algazi, V. R., Brown, K. L., Ready, M. J., Irvine, D. H., Cadwell, C. L., and Chung, S. (1993). Transform representation of the spectra of acoustic speech segments with applications. i. general approach and application to speech recognition. *IEEE Transactions on Speech and Audio Processing*, 1(2):180–195.

Andrade-Garda, J. M., Carlosena-Zubieta, A., Boque-Marti, R., and Ferre-Baldrich, J. (2009). Partial least-squares regression. In *Basic Chemometric Techniques in Atomic Spectroscopy*, pages 181–243. Royal Society of Chemistry London,(UK).

Archambeau, C., Delannay, N., and Verleysen, M. (2006). Robust probabilistic projections. In *Proceedings of the 23rd International conference on machine learning*, pages 33–40. ACM.

Baffi, G., Martin, E., and Morris, A. (1999). Non-linear projection to latent structures revisited: the quadratic pls algorithm. *Computers & Chemical Engineering*, 23(3):395–411.

Balabin, R. M., Safieva, R. Z., and Lomakina, E. I. (2010). Gasoline classification using near infrared (NIR) spectroscopy data: Comparison of multivariate techniques. *Analytica Chimica Acta*, 671(1-2):27–35.

Berglund, A. and Wold, S. (1997). Inlr, implicit non-linear latent variable regression. *Journal of Chemometrics: A Journal of the Chemometrics Society*,

11(2):141–156.

Berrueta, L. A., Alonso-Salces, R. M., and Héberger, K. (2007). Supervised pattern recognition in food analysis. *Journal of chromatography A*, 1158(1-2):196–214.

Bhele, S. G. and Mankar, V. (2012). A review paper on face recognition techniques. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, 1(8):pp–339.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. springer.

Bishop, C. M. et al. (1995). *Neural Networks for Pattern Recognition*. Oxford university press.

Blanco, M. and Villarroya, I. (2002). Nir spectroscopy: a rapid-response analytical tool. *TrAC Trends in Analytical Chemistry*, 21(4):240–250.

Brent, R. P. (2013). *Algorithms for Minimization without Derivatives*. Courier Corporation.

Candès, E. J., Li, X., Ma, Y., and Wright, J. (2011). Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):11.

Caplan, J. D., Waxman, S., Nesto, R. W., and Muller, J. E. (2006). Near-infrared spectroscopy for the detection of vulnerable coronary artery plaques. *Journal of the American College of Cardiology*, 47(8 Supplement):C92–C96.

Cevikalp, H., Neamtu, M., Wilkes, M., and Barkana, A. (2005). Discriminative common vectors for face recognition. *IEEE Transactions on pattern analysis and machine intelligence*, 27(1):4–13.

Chen, L.-F., Liao, H.-Y. M., Ko, M.-T., Lin, J.-C., and Yu, G.-J. (2000). A new lda-based face recognition system which can solve the small sample size problem. *Pattern recognition*, 33(10):1713–1726.

Chen, S. and Sun, T. (2005). Class-information-incorporated principal component analysis. *Neurocomputing*, 69(1-3):216–223.

Chen, T., Martin, E., and Montague, G. (2009). Robust probabilistic pca with missing data and contribution analysis for outlier detection. *Computational Statistics*

& Data Analysis, 53(10):3706–3716.

Chiang, L. H., Russell, E. L., and Braatz, R. D. (2000). Fault diagnosis in chemical processes using Fisher discriminant analysis, discriminant partial least squares, and principal component analysis. *Chemometrics and intelligent laboratory systems*, 50(2):243–252.

Cho, J.-H., Lee, J.-M., Choi, S. W., Lee, D., and Lee, I.-B. (2005). Fault identification for process monitoring using kernel principal component analysis. *Chemical engineering science*, 60(1):279–288.

Christy, A. A., Kasemsumran, S., Du, Y., and OZAKI, Y. (2004). The detection and quantification of adulteration in olive oil by near-infrared spectroscopy and chemometrics. *Analytical Sciences*, 20(6):935–940.

Chu, D. and Thye, G. S. (2010). A new and fast implementation for null space based linear discriminant analysis. *Pattern Recognition*, 43(4):1373–1379.

Costa, F. S., Silva, P. P., Morais, C. L., Theodoro, R. C., Arantes, T. D., and Lima, K. M. (2017). Comparison of multivariate classification algorithms using eem fluorescence data to distinguish cryptococcus neoformans and cryptococcus gattii pathogenic fungi. *Analytical methods*, 9(26):3968–3976.

Dai, D.-Q. and Yuen, P. C. (2007). Face recognition by regularized discriminant analysis. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 37(4):1080–1085.

Daszykowski, M., Kaczmarek, K., Vander Heyden, Y., and Walczak, B. (2007). Robust statistics in data analysis—a review: basic concepts. *Chemometrics and intelligent laboratory systems*, 85(2):203–219.

De Jong, S. (1993). Simpls: an alternative approach to partial least squares regression. *Chemometrics and intelligent laboratory systems*, 18(3):251–263.

De Jong, S. and Kiers, H. A. (1992). Principal covariates regression: part i. theory. *Chemometrics and Intelligent Laboratory Systems*, 14(1-3):155–164.

Delwiche, S. R. and Norris, K. H. (1993). Classification of hard red wheat by near-

infrared diffuse reflectance spectroscopy. *Cereal Chemistry*, 70:29–29.

Devlin, S. J., Gnanadesikan, R., and Kettenring, J. R. (1981). Robust estimation of dispersion matrices and principal components. *Journal of the American Statistical Association*, 76(374):354–362.

Douglas, S. C., Amari, S.-i., and Kung, S.-Y. (2000). On gradient adaptation with unit-norm constraints. *IEEE Transactions on Signal Processing*, 48(6):1843–1847.

Duda, R. O., Hart, P. E., Stork, D. G., et al. (2001). Pattern classification. 2nd. *Edition. New York*, 55.

Egan, W. J. and Morgan, S. L. (1998). Outlier detection in multivariate analytical chemical data. *Analytical chemistry*, 70(11):2372–2379.

Erickson, S. J. and Godavarty, A. (2009). Hand-held based near-infrared optical imaging devices: a review. *Medical engineering & physics*, 31(5):495–509.

Fan, Z., Xu, Y., Zuo, W., Yang, J., Tang, J., Lai, Z., and Zhang, D. (2014). Modified principal component analysis: an integration of multiple similarity subspace models. *IEEE transactions on neural networks and learning systems*, 25(8):1538–1552.

Fearn, T., Brown, P., and Haque, M. (1999). Logistic discrimination with many variables. *Rev. R. Acad. Cienc. Exact. Fis. Nat.(Esp.)*, 93:337–342.

Ferré, L. (1995). Selection of components in principal component analysis: a comparison of methods. *Computational Statistics & Data Analysis*, 19(6):669–682.

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188.

Frank, L. E. and Friedman, J. H. (1993). A statistical view of some chemometrics regression tools. *Technometrics*, 35(2):109–135.

Friedman, J., Hastie, T., and Tibshirani, R. (2001a). *The elements of Statistical Learning*, volume 1. Springer series in statistics New York.

Friedman, J., Hastie, T., and Tibshirani, R. (2001b). *The Elements of Statistical*

Learning, volume 1. Springer series in statistics New York.

Friedman, J. H. (1989). Regularized discriminant analysis. *Journal of the American statistical association*, 84(405):165–175.

Geladi, P. (2003). Chemometrics in spectroscopy. part 1. classical chemometrics. *Spectrochimica Acta Part B: Atomic Spectroscopy*, 58(5):767–782.

Geladi, P., Isaksson, H., Lindqvist, L., Wold, S., and Esbensen, K. (1989). Principal component analysis of multivariate images. *Chemometrics and Intelligent Laboratory Systems*, 5(3):209–220.

Gendrin, C., Roggo, Y., and Collet, C. (2008). Pharmaceutical applications of vibrational chemical imaging and chemometrics: a review. *Journal of pharmaceutical and biomedical analysis*, 48(3):533–553.

Gowen, A., O'donnell, C., Taghizadeh, M., Cullen, P., Frias, J., and Downey, G. (2008). Hyperspectral imaging combined with principal component analysis for bruise damage detection on white mushrooms (agaricus bisporus). *Journal of Chemometrics: A Journal of the Chemometrics Society*, 22(3-4):259–267.

Grabner, H., Roth, P. M., and Bischof, H. (2007). Eigenboosting: Combining discriminative and generative information. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE.

Guan, Y. and Dy, J. (2009). Sparse probabilistic principal component analysis. In *Artificial Intelligence and Statistics*, pages 185–192.

Hadoux, X., Rutledge, D. N., Rabatel, G., and Roger, J.-M. (2015). Drop-d: dimension reduction by orthogonal projection for discrimination. *Chemometrics and Intelligent Laboratory Systems*, 146:221–231.

Higdon, R., Foster, N. L., Koeppe, R. A., DeCarli, C. S., Jagust, W. J., Clark, C. M., Barbas, N. R., Arnold, S. E., Turner, R. S., Heidebrink, J. L., et al. (2004). A comparison of classification methods for differentiating fronto-temporal dementia from alzheimer's disease using fdg-pet imaging. *Statistics in medicine*, 23(2):315–326.

Hong, H., Naghibi, S. A., Dashtpagerdi, M. M., Pourghasemi, H. R., and Chen, W. (2017). A comparative assessment between linear and quadratic discriminant analyses (lda-qda) with frequency ratio and weights-of-evidence models for forest fire susceptibility mapping in china. *Arabian Journal of Geosciences*, 10(7):167.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417.

Huang, J. Z., Shen, H., Buja, A., et al. (2008). Functional principal components analysis via penalized rank one approximation. *Electronic Journal of Statistics*, 2:678–695.

Huang, R., Liu, Q., Lu, H., and Ma, S. (2002). Solving the small sample size problem of lda. In *null*, page 30029. Citeseer.

Huang, S., Yang, D., Yongxin, G., and Zhang, X. (2015). Combined supervised information with pca via discriminative component selection. *Information Processing Letters*, 115(11):812–816.

Hubert, M. and Engelen, S. (2004). Robust pca and classification in biosciences. *Bioinformatics*, 20(11):1728–1736.

Hubert, M., Rousseeuw, P. J., and Vanden Branden, K. (2005). Robpca: a new approach to robust principal component analysis. *Technometrics*, 47(1):64–79.

Hubert, M., Rousseeuw, P. J., and Verboven, S. (2002). A fast method for robust principal components with applications to chemometrics. *Chemometrics and Intelligent Laboratory Systems*, 60(1-2):101–111.

Ince, H. and Trafalis, T. B. (2007). Kernel principal component analysis and support vector machines for stock price prediction. *IIE Transactions*, 39(6):629–637.

Jolliffe, I. T. (1986). Choosing a subset of principal components or variables. In *Principal Component Analysis*, pages 92–114. Springer.

Jolliffe, I. T. and Cadima, J. (2016). Principal component analysis: a review and recent developments. *Phil. Trans. R. Soc. A*, 374(2065):20150202.

Kallithraka, S., Arvanitoyannis, I., Kefalas, P., El-Zajouli, A., Soufleros, E., and Psarra, E. (2001). Instrumental and sensory analysis of greek wines; implementation of principal component analysis (pca) for classification according to geographical origin. *Food Chemistry*, 73(4):501–514.

Karoui, R. and De Baerdemaeker, J. (2007). A review of the analytical methods coupled with chemometric tools for the determination of the quality and identity of dairy products. *Food chemistry*, 102(3):621–640.

Khan, A. T., Afrin, S., and Rahman, T. (2018). Comparison of principal component analysis and partial least square discriminant analysis in the classification of eeg signals. In *2018 IEEE International Workshop on Signal Processing Systems (SiPS)*, pages 1–6. IEEE.

Kim, D. and Lee, I.-B. (2003). Process monitoring based on probabilistic pca. *Chemometrics and intelligent laboratory systems*, 67(2):109–123.

Kim, K. I., Jung, K., and Kim, H. J. (2002). Face recognition using kernel principal component analysis. *IEEE signal processing letters*, 9(2):40–42.

Kim, K. S., Choi, H. H., Moon, C. S., and Mun, C. W. (2011). Comparison of k-nearest neighbor, quadratic discriminant and linear discriminant analysis in classification of electromyogram signals based on the wrist-motion directions. *Current applied physics*, 11(3):740–745.

Kirby, M. and Sirovich, L. (1990). Application of the karhunen-loeve procedure for the characterization of human faces. *IEEE Transactions on Pattern analysis and Machine intelligence*, 12(1):103–108.

Kresta, J. V., Macgregor, J. F., and Marlin, T. E. (1991). Multivariate statistical monitoring of process operating performance. *The Canadian Journal of Chemical Engineering*, 69(1):35–47.

Krzanowski, W. (2000). *Principles of Multivariate Analysis*, volume 23. OUP Oxford.

Landau, S., Glasser, T., and Dvash, L. (2006). Monitoring nutrition in small rumi-

nants with the aid of near infrared reflectance spectroscopy (nirs) technology: a review. *Small Ruminant Research*, 61(1):1–11.

Lee, J.-M., Yoo, C., and Lee, I.-B. (2004). Fault detection of batch processes using multiway kernel principal component analysis. *Computers & chemical engineering*, 28(9):1837–1847.

Li, G. and Chen, Z. (1985). Projection-pursuit approach to robust dispersion matrices and principal components: primary theory and monte carlo. *Journal of the American Statistical Association*, 80(391):759–766.

Liu, L., Cozzolino, D., Cynkar, W., Gishen, M., and Colby, C. (2006). Geographic classification of spanish and australian tempranillo red wines by visible and near-infrared spectroscopy combined with multivariate analysis. *Journal of agricultural and food chemistry*, 54(18):6754–6759.

Lu, J., Plataniotis, K. N., and Venetsanopoulos, A. N. (2003). Face recognition using kernel direct discriminant analysis algorithms. *IEEE Transactions on Neural Networks*, 14(1):117–126.

Martens, H. and Naes, T. (1992). *Multivariate Calibration*. John Wiley & Sons.

Meher, L. C., Sagar, D. V., and Naik, S. (2006). Technical aspects of biodiesel production by transesterification—a review. *Renewable and sustainable energy reviews*, 10(3):248–268.

Mejdell, T. and Skogestad, S. (1991). Estimation of distillation compositions from multiple temperature measurements using partial-least-squares regression. *Industrial & Engineering Chemistry Research*, 30(12):2543–2555.

Moghimi, A., Aghkhani, M. H., Sazgarnia, A., and Sarmad, M. (2010). Vis/nir spectroscopy and chemometrics for the prediction of soluble solids content and acidity (ph) of kiwifruit. *Biosystems engineering*, 106(3):295–302.

Moreda, G., Ortiz-Cañavate, J., García-Ramos, F. J., and Ruiz-Altisent, M. (2009). Non-destructive technologies for fruit and vegetable size determination–a review. *Journal of Food Engineering*, 92(2):119–136.

Murugesan, A., Umarani, C., Chinnusamy, T., Krishnan, M., Subramanian, R., and Neduzchezhain, N. (2009). Production and analysis of bio-diesel from non-edible oils—a review. *Renewable and Sustainable Energy Reviews*, 13(4):825–834.

Naghibi, S. A., Pourghasemi, H. R., and Abbaspour, K. (2018). A comparison between ten advanced and soft computing models for groundwater qanat potential assessment in iran using r and gis. *Theoretical and applied climatology*, 131(3-4):967–984.

Nelder, J. A. and Mead, R. (1965). A simplex method for function minimization. *The computer journal*, 7(4):308–313.

Nyamundanda, G., Brennan, L., and Gormley, I. C. (2010). Probabilistic principal component analysis for metabolomic data. *BMC bioinformatics*, 11(1):571.

Nyström, J. and Dahlquist, E. (2004). Methods for determination of moisture content in woodchips for power plants—a review. *Fuel*, 83(7-8):773–779.

Olawale, F. and Garwe, D. (2010). Obstacles to the growth of new smes in south africa: A principal component analysis approach. *African journal of Business management*, 4(5):729–738.

Paliwal, K. K. and Sharma, A. (2010). Improved direct lda and its application to dna microarray gene expression data. *Pattern Recognition Letters*, 31(16):2489–2492.

Pearson, K. (1901). Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572.

Pechenizkiy, M., Tsymbal, A., and Puuronen, S. (2006). On combining principal components with Fisher's linear discriminants for supervised learning. *Foundations of Computing and Decision Sciences*, 31(1):59–74.

Peres-Neto, P. R., Jackson, D. A., and Somers, K. M. (2005). How many principal components? stopping rules for determining the number of non-trivial axes revisited. *Computational Statistics & Data Analysis*, 49(4):974–997.

Perez, D. R. and Narasimhan, G. (2018). So you think you can pls-da? *bioRxiv*, page 207225.

Pérez-Marín, D., Fearn, T., Guerrero, J. E., and Garrido-Varo, A. (2009). A methodology based on nir-microscopy for the detection of animal protein by-products. *Talanta*, 80(1):48–53.

Prasad, M. M., Sukumar, M., and Ramakrishnan, A. (2010). Orthogonal lda in pca transformed subspace. In *Frontiers in Handwriting Recognition (ICFHR), 2010 International Conference on*, pages 172–175. IEEE.

Preda, C., Saporta, G., and Lévéder, C. (2007). Pls classification of functional data. *Computational Statistics*, 22(2):223–235.

Prieto, N., Roehe, R., Lavín, P., Batten, G., and Andrés, S. (2009a). Application of near infrared reflectance spectroscopy to predict meat and meat products quality: A review. *Meat science*, 83(2):175–186.

Prieto, N., Roehe, R., Lavín, P., Batten, G., and Andrés, S. (2009b). Application of near infrared reflectance spectroscopy to predict meat and meat products quality: A review. *Meat science*, 83(2):175–186.

Qiao, H. (2019). Discriminative principal component analysis: A reverse thinking. *arXiv preprint arXiv:1903.04963*.

Qiu, J., Wang, H., Lu, J., Zhang, B., and Du, K.-L. (2012). Neural network implementations for pca and its extensions. *ISRN Artificial Intelligence*, 2012.

Quarteroni, A., Sacco, R., and Saleri, F. (2010). *Numerical Mathematics*, volume 37. Springer Science & Business Media.

Rao, C. R. (1948). The utilization of multiple measurements in problems of biological classification. *Journal of the Royal Statistical Society. Series B (Methodological)*, 10(2):159–203.

Ripley, B. D. (2007). *Pattern Recognition and Neural Networks*. Cambridge university press.

Rodriguez-Otero, J. L., Hermida, M., and Centeno, J. (1997). Analysis of dairy

products by near-infrared spectroscopy: A review. *Journal of Agricultural and Food chemistry*, 45(8):2815–2819.

Roggo, Y., Chalus, P., Maurer, L., Lema-Martinez, C., Edmond, A., and Jent, N. (2007). A review of near infrared spectroscopy and chemometrics in pharmaceutical technologies. *Journal of pharmaceutical and biomedical analysis*, 44(3):683–700.

Rosipal, R. and Trejo, L. J. (2001). Kernel partial least squares regression in reproducing kernel hilbert space. *Journal of machine learning research*, 2(Dec):97–123.

Rosipal, R., Trejo, L. J., and Matthews, B. (2003). Kernel pls-svc for linear and nonlinear classification. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 640–647.

Roweis, S. T. (1998). Em algorithms for pca and spca. In *Advances in Neural Information Processing Systems*, pages 626–632.

Ruder, S. (2016a). An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*.

Ruder, S. (2016b). An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*.

Saeys, W., De Ketelaere, B., and Darius, P. (2008). Potential applications of functional data analysis in chemometrics. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 22(5):335–344.

Sakudo, A., Suganuma, Y., Kobayashi, T., Onodera, T., and Ikuta, K. (2006). Near-infrared spectroscopy: promising diagnostic tool for viral infections. *Biochemical and biophysical research communications*, 341(2):279–284.

Sampson, D. L., Parker, T. J., Upton, Z., and Hurst, C. P. (2011). A comparison of methods for classifying clinical samples based on proteomics data: a case study for statistical and machine learning approaches. *PloS one*, 6(9):e24973.

Schölkopf, B., Smola, A., and Müller, K.-R. (1997). Kernel principal component

analysis. In *International Conference on Artificial Neural Networks*, pages 583–588. Springer.

Sharma, A. and Paliwal, K. K. (2012a). A new perspective to null linear discriminant analysis method and its fast implementation using random matrix multiplication with scatter matrices. *Pattern Recognition*, 45(6):2205–2213.

Sharma, A. and Paliwal, K. K. (2012b). A two-stage linear discriminant analysis for face-recognition. *Pattern Recognition Letters*, 33(9):1157–1162.

Sharma, A. and Paliwal, K. K. (2015a). A deterministic approach to regularized linear discriminant analysis. *Neurocomputing*, 151:207–214.

Sharma, A. and Paliwal, K. K. (2015b). Linear discriminant analysis for the small sample size problem: an overview. *International Journal of Machine Learning and Cybernetics*, 6(3):443–454.

Sharma, A., Paliwal, K. K., Imoto, S., and Miyano, S. (2014). A feature selection method using improved regularized linear discriminant analysis. *Machine vision and applications*, 25(3):775–786.

Shepherd, K. D. and Walsh, M. G. (2007). Infrared spectroscopy—enabling an evidence-based diagnostic surveillance approach to agricultural and environmental management in developing countries. *Journal of Near Infrared Spectroscopy*, 15(1):1–19.

Siqueira, L. F., Júnior, R. F. A., de Araújo, A. A., Morais, C. L., and Lima, K. M. (2017). Lda vs. qda for ft-mir prostate cancer tissue classification. *Chemometrics and Intelligent Laboratory Systems*, 162:123–129.

Sirovich, L. and Kirby, M. (1987). Low-dimensional procedure for the characterization of human faces. *Josa a*, 4(3):519–524.

Song, F., Zhang, D., Wang, J., Liu, H., and Tao, Q. (2007). A parameterized direct lda and its application to face recognition. *Neurocomputing*, 71(1-3):191–196.

Sturn, A., Quackenbush, J., and Trajanoski, Z. (2002). Genesis: cluster analysis of microarray data. *Bioinformatics*, 18(1):207–208.

Sutskever, I., Martens, J., Dahl, G., and Hinton, G. (2013). On the importance of initialization and momentum in deep learning. In *International Conference on Machine Learning*, pages 1139–1147.

Tewari, J. C. and Irudayaraj, J. M. (2005). Floral classification of honey using mid-infrared spectroscopy and surface acoustic wave based z-nose sensor. *Journal of agricultural and food chemistry*, 53(18):6955–6966.

Thomaz, C. E. and Giraldi, G. A. (2010). A new ranking method for principal components analysis and its application to face image analysis. *Image and Vision Computing*, 28(6):902–913.

Tipping, M. E. and Bishop, C. M. (1999). Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622.

Toczydlowska, D. (2020). *Machine Learning Developments in Dependency Modelling and Feature Extraction*. PhD thesis, UCL (University College London).

Tsuchikawa, S. (2007). A review of recent near infrared research for wood and paper. *Applied Spectroscopy Reviews*, 42(1):43–71.

Tsuchikawa, S. and Kobori, H. (2015). A review of recent application of near infrared spectroscopy to wood science and technology. *Journal of Wood Science*, 61(3):213–220.

Turk, M. and Pentland, A. (1991). Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1):71–86.

Vaid, T. P., Burl, M. C., and Lewis, N. S. (2001). Comparison of the performance of different discriminant algorithms in analyte discrimination tasks using an array of carbon black- polymer composite vapor detectors. *Analytical Chemistry*, 73(2):321–331.

Verdun, S., Hanafi, M., Cariou, V., and Qannari, E. M. (2012). Quadratic pls1 regression revisited. *Journal of Chemometrics*, 26(7):384–389.

Wang, L., Sun, D.-W., Pu, H., and Cheng, J.-H. (2017). Quality analysis, classifi-

cation, and authentication of liquid foods by near-infrared spectroscopy: A review of recent research developments. *Critical reviews in food science and nutrition*, 57(7):1524–1538.

Wang, P. and Yu, Z. (2015). Species authentication and geographical origin discrimination of herbal medicines by near infrared spectroscopy: A review. *Journal of pharmaceutical analysis*, 5(5):277–284.

Watkins, D. S. (2004). *Fundamentals of matrix computations*, volume 64. John Wiley & Sons.

Williams, P., Norris, K., et al. (1987). *Near-infrared Technology in the Agricultural and Food Industries.* American Association of Cereal Chemists, Inc.

Wise, B. M., Ricker, N., Veltkamp, D., and Kowalski, B. R. (1990). A theoretical basis for the use of principal component models for monitoring multivariate processes. *Process control and quality*, 1(1):41–51.

Wold, S. (1992). Nonlinear partial least squares modelling ii. spline inner relation. *Chemometrics and Intelligent Laboratory Systems*, 14(1-3):71–84.

Wold, S., Esbensen, K., and Geladi, P. (1987). Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52.

Wold, S., Kettaneh-Wold, N., and Skagerberg, B. (1989). Nonlinear pls modeling. *Chemometrics and intelligent laboratory systems*, 7(1-2):53–65.

Wold, S., Ruhe, A., Wold, H., and Dunn, III, W. (1984). The collinearity problem in linear regression. the partial least squares (pls) approach to generalized inverses. *SIAM Journal on Scientific and Statistical Computing*, 5(3):735–743.

Wold, S., Sjöström, M., and Eriksson, L. (2001). Pls-regression: a basic tool of chemometrics. *Chemometrics and intelligent laboratory systems*, 58(2):109–130.

Wu, B., Abbott, T., Fishman, D., McMurray, W., Mor, G., Stone, K., Ward, D., Williams, K., and Zhao, H. (2003). Comparison of statistical methods for classification of ovarian cancer using mass spectrometry data. *Bioinformatics*, 19(13):1636–1643.

Wu, W., Mallet, Y., Walczak, B., Penninckx, W., Massart, D., Heuerding, S., and Erni, F. (1996). Comparison of regularized discriminant analysis linear discriminant analysis and quadratic discriminant analysis applied to nir data. *Analytica Chimica Acta*, 329(3):257–265.

Xiaobo, Z., Jiewen, Z., Povey, M. J., Holmes, M., and Hanpin, M. (2010). Variables selection methods in near-infrared spectroscopy. *Analytica chimica acta*, 667(1-2):14–32.

Xie, Y.-L., Wang, J.-H., Liang, Y.-Z., Sun, L.-X., Song, X.-H., and Yu, R.-Q. (1993). Robust principal component analysis by projection pursuit. *Journal of Chemometrics*, 7(6):527–541.

Xu, X., Lai, Z., Chen, Y., and Kong, H. (2018). Robust discriminative principal component analysis. In *Chinese Conference on Biometric Recognition*, pages 231–238. Springer.

Yan, H. and Dai, Y. (2011). The comparison of five discriminant methods. In *2011 International Conference on Management and Service Science*, pages 1–4. IEEE.

Ye, J. (2005). Characterization of a family of algorithms for generalized discriminant analysis on undersampled problems. *Journal of Machine Learning Research*, 6(Apr):483–502.

Ye, J. (2007). Least squares linear discriminant analysis. In *Proceedings of the 24th International Conference on Machine Learning*, pages 1087–1093. ACM.

Ye, J. and Xiong, T. (2006). Computational and theoretical analysis of null space and orthogonal linear discriminant analysis. *Journal of Machine Learning Research*, 7(Jul):1183–1204.

Yeung, K. Y. and Ruzzo, W. L. (2001). Principal component analysis for clustering gene expression data. *Bioinformatics*, 17(9):763–774.

Yu, H. and Yang, J. (2001). A direct lda algorithm for high-dimensional data—with application to face recognition. *Pattern recognition*, 34(10):2067–2070.

Yu, S., Yu, K., Tresp, V., Kriegel, H.-P., and Wu, M. (2006). Supervised proba-

bilistic principal component analysis. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 464–473. ACM.

Zhang, P., Peng, J., and Riedel, N. (2005). Discriminant analysis: a least squares approximation view. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)-Workshops*, pages 46–46. IEEE.

Zhu, M. and Martinez, A. M. (2006). Selecting principal components in a two-stage lda algorithm. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 132–137. IEEE.

Zou, H., Hastie, T., and Tibshirani, R. (2006). Sparse principal component analysis. *Journal of computational and graphical statistics*, 15(2):265–286.