University of California
Santa Barbara

Classification methods for multivariate and functional data using features from density ratio estimation

A dissertation submitted in partial satisfaction
of the requirements for the degree

Doctor of Philosophy
in
Statistics and Applied Probability

by

Zachary Terner

June 2020

The Dissertation of Zachary Terner is approved.

Professor Wendy Meiring

Professor Yuedong Wang, Co-Chair

Professor Alexander Petersen, Co-Chair

June 2020

Classification methods for multivariate and functional data using features from density ratio estimation

To my parents, who wanted me to be happy and healthy

Acknowledgements

I want to thank my advisors, Professor Alexander Petersen and Professor Yuedong Wang, for their continued encouragement, support, and guidance throughout the course of my PhD. I have been incredibly fortunate to have two advisors who cared for my personal well-being as much as my academic success. I have learned much from each of you about not only statistics, but also work ethic, writing, and how to be encouraging. Thank you both for being such invaluable mentors. I also want to thank Professor Wendy Meiring, the third member of my committee, for always being available and eager to help in any way she can. In many ways, Professor Meiring has guided me as an unofficial advisor. She exudes kindness and positivity in every interaction, and has provided thoughtful and honest support during my time in the department. Thank you for always lending an ear.

I also want to thank two of my undergraduate professors from the University of Virginia, Professor Daniel Keenan and Professor Donald Brown. Professor Keenan first introduced me to statistical research and showed me how exciting and useful it could be. Professor Brown encouraged me to pursue my own ideas and gave me the confidence to do so. I am indebted to all of my mentors for helping and supporting me and am extremely grateful for their kindness.

I would like to thank the NSF for financial support and Wentian Huang and David Ruppert of Cornell for sharing their copula code and data. I would also like to thank the Center for Scientific Computing at UCSB for allowing me to use their computational resources.

I would also like to thank the many colleagues and friends who have taken this journey with me and helped me along the way. Thank you Yuanbo, Bret, Nhan, Jiaye, Mark, Youhong, Laura, Ya, Brennan, Esther, Dalia, and others for helping with countless assignments, questions, projects, and for being great friends. This journey would not have been as pleasant without you.

I owe a sincere thanks to the extended Loschak family and the larger community of Chabad of S. Barbara. Thank you for always providing support, food, and friendship during my time at UCSB, and for welcoming me into your homes.

Last, I want to thank my family for their endless support. Thank you to my California and New Jersey families for checking in on me, for sharing in my success, and for helping in all the ways that families do. Thank you to each of my brothers, Steven, Jeremy, and Benjamin, for always trying to help, for exemplifying hard work, and for feeding me. This milestone would not be happening without your support and love. And finally, thank you to my parents, who encouraged me to try harder and reach farther than I would have on my own.

Abstract

Classification methods for multivariate and functional data using features from density ratio estimation

by

Zachary Terner

We introduce a new method for estimating density ratios using splines, as a generalization of a method from Silverman [1]. This method applies to general domains and can be used to estimate joint density ratios. We then use the spline method to construct a new classifier named DAB, or Dependence-Adjusted naive Bayes. The DAB classifier estimates marginal log density ratios and uses them as features in a binary classification problem. We show that DAB may recover the optimal Bayes solution in certain Gaussian situations where naive Bayes cannot, and we also demonstrate its performance on simulated and empirical datasets. We also recreate a comparison of naive Bayes and logistic regression from Ng and Jordan [2] and show where DAB can outperform both methods. Last, we demonstrate DAB's effectiveness in the setting of functional data as an extension to the functional naive Bayes classifier [3].

Contents

Chapter 1

Introduction

The problem of classifying new observations into one of two or more classes is an old and ongoing problem in statistics. Many new statistical machine learning methods have been developed for this purpose, including the proliferation of neural network and deep learning techniques in recent years. The development of these methods has coincided with an increase in the types and quantities of data available to train reliable classifiers.

Similarly, the field of functional data analysis has also experienced a surge in the data collected and the subsequent methods developed. These data arise in a variety of application areas, including medicine, chemometrics, brain imaging, and finance [4, 5, 6, 7, 8, 9]. As a result, several classification methods designed for functional data have been published recently. Many of the more recent functional data classification papers translate classical ideas from multivariate data classification into a functional data framework. For example, Delaigle and Hall show that linear and quadratic discriminant analysis applied to functional data can achieve perfect classification, or an asymptotic misclassification rate of zero, under certain Gaussian conditions [10, 11]. Rossi and Villa present support vector machines applied to functional data [12], with Wu and Liu developing a robust version [13]. Similarly, there has been work in developing functional data methods for decision trees [14].

More recently, Dai, Mueller, and Yao proposed a nonparametric Bayes classifier for functional data [3]. Bayes classifiers are known to provide optimal misclassification rates when the joint densities are known and specified. Estimating joint densities is a tall task for multivariate data and even harder for functional data, since densities for functional data generally do not exist [15]. Dai, Mueller, and Yao overcome this by using the density ratios of functional projection scores in place of a traditional density ratio. Assuming the projection scores to be independent lets them reduce the Bayes classifier to the naive Bayes classifier, following the derivation of the naive Bayes classifier in multivariate data. Rather than treat these scores as independent, Huang and Ruppert include a Gaussian or t-copula in their Bayes classifier to model the dependence [16].

In this thesis, we propose new classification methods with either multivariate or functional data. The contribution of this thesis comes in two main parts: first, we propose a method for density ratio estimation, a special case of which can be traced back to Silverman [1]. We show that this technique of estimating the log density ratio directly can work better than the method used in [3] and [16] of taking the log of the quotient of two separate kernel density estimates. Second, we propose a generalization of the naive Bayes classifier to adjust for the dependence between variables. We show that this generalization can improve classification performance in both the traditional multivariate and functional data settings via simulation and real data applications. Additionally, we demonstrate that in certain Gaussian situations, this generalization can recover the optimal Bayes classifier, which is known to minimize misclassification rates.

1.1 Bayes and naive Bayes classifiers

In this section we introduce the general Bayes classifier, discuss its optimality, and explain its relationship to the naive Bayes classifier. We then show how naive Bayes is equivalent to logistic regression under a specific Gaussian setting. Both of these methods are fundamental to the contribution presented here, where we compare the classifier presented in Chapter 3 with naive Bayes and logistic regression solutions.

1.1.1 Naive Bayes

The general Bayes classifier for multivariate data is known to minimize misclassification rates [17, 18, 19, 20]. Let Y be our class labels and denote $\mathbf{X}$ as our p multivariate predictors. The Bayes classifier can be written as

$$\hat{y} = \arg\max_{y} P(Y = y|\mathbf{X} = \mathbf{x}). \tag{1.1}$$

In other words, we assign new observations x to the class which has the maximum posterior probability [21]. In the two-class problem, where $Y \in \{0, 1\}$, let

$$Q(\mathbf{x}) = \frac{P(Y = 1|\mathbf{X} = \mathbf{x})}{P(Y = 0|\mathbf{X} = \mathbf{x})}. \tag{1.2}$$

The Bayes classifier assigns observations to class $Y = 1$ if $Q(x) > 1$. In this case, we compute the posterior conditional probabilities of the two classes and take the ratio to determine which class has the higher posterior probability. By applying Bayes' theorem,

$$Q(\mathbf{x}) = \frac{P(Y = 1)P(\mathbf{X} = \mathbf{x}|Y = 1)}{P(Y = 0)P(\mathbf{X} = \mathbf{x}|Y = 0)} \tag{1.3}$$

where we have switched the conditioning to require a density of the data conditional on the class, $P(\mathbf{X} = \mathbf{x}|Y = y)$. The class probabilities, $P(Y = y)$, can be estimated by computing the number of instances in each class. However, the conditional probabilities in (1.3) can be very difficult to estimate in practice, especially when p is large, due to the curse of dimensionality [22].

One common assumption used to compute the joint densities in (1.3) is to assume that the predictor variables are independent, conditional on class membership. Let $g_y(\mathbf{x})$ be the density function of $\mathbf{X}$ conditional on $Y = y$. Let $\mathbf{X} = (X_1, \ldots, X_p)$ and g_{yk} be the marginal density function of X_k conditional on $Y = y$. By the factorization theorem, if the predictor variables are independent, we can write

$$Q(\mathbf{x}) = \frac{P(Y=1)g_1(\mathbf{x})}{P(Y=0)g_0(\mathbf{x})} = \frac{P(Y=1)\prod_{k=1}^{p} g_{1k}(x_k)}{P(Y=0)\prod_{k=1}^{p} g_{0k}(x_k)}. \tag{1.4}$$

Since (1.4) contains only products, we can take the log to convert the products into a sum. This creates the traditional naive Bayes classifier, which we denote as $Q_n(x)$:

$$Q_n(\mathbf{x}) = \log Q(\mathbf{x}) = \log \frac{P(Y=1)}{P(Y=0)} + \sum_{k=1}^{p} \log \frac{g_{1k}(x_k)}{g_{0k}(x_k)}. \tag{1.5}$$

In (1.5), we simply need to compute the log density ratio of each marginal variable. The class label is assigned based on whether the sum of the log density ratios and the prior probabilities is positive $(Y = 1)$ or negative $(Y = 0)$.

The naive Bayes classifier has been popular due to its strong performance. Multiple studies have sought to determine why the classifier performs well empirically even when the conditional independence assumption is violated. Hand and Yu provide a review of these explanations in [23]. One explanation given is that naive Bayes can provide lower variance estimates of $P(Y|\mathbf{X} = \mathbf{x})$ than more complicated models, especially in small sample sizes, because naive Bayes is traditionally restricted to computing only one estimand per feature, the log density ratio. This reduction in variance comes at a cost of having an estimate of $E[\hat{P}(Y|\mathbf{X})]$ taken with respect to $\mathbf{X}$ that may be more biased than the estimates from other models. However, this bias may not matter in terms of classification performance as long as $\hat{P}(Y = 1|\mathbf{X}) > \hat{P}(Y = 0|\mathbf{X})$ when $P(Y = 1|\mathbf{X}) > P(Y = 0|\mathbf{X})$ and vice versa. Additional information can be found in [23, 24, 2, 25].

1.1.2 Logistic regression

Consider the case where we are interested in modeling the posterior probability, $P(Y = 1|\mathbf{X} = \mathbf{x})$, as done in logistic regression. We show below that taking the marginal density ratios, as done in naive Bayes, results in a logistic regression model in the special case of Gaussian data where the only difference between the two classes is in the means [26, 27].

Recall that in (1.5), we write the conditional log odds, $Q_n(\mathbf{x})$, as the sum of the log of p marginal density ratios and a constant. If the only difference between the two marginal Gaussian densities comes in the means – $X_k|Y = 0 \sim N(\mu_{0k}, \sigma_k^2)$ and $X_k|Y = 1 \sim N(\mu_{1k}, \sigma_k^2)$ – we can write each individual log ratio as

$$\log\left(\frac{g_{0k}(x)}{g_{1k}(x)}\right) = \frac{(\mu_{0k} - \mu_{1k})}{\sigma_k^2} x_k + \frac{(\mu_{1k}^2 - \mu_{0k}^2)}{2\sigma_k^2}. \tag{1.6}$$

Let

$$\begin{aligned} w_0 &= \log\left(\frac{P(Y=0)}{P(Y=1)}\right) + \sum_{k=1}^{p} \frac{(\mu_{1k}^2 - \mu_{0k}^2)}{2\sigma_k^2} \\ w_k &= \frac{\mu_{0k} - \mu_{1k}}{\sigma_k^2} \end{aligned} \tag{1.7}$$

where we have grouped all constant terms as w_0 and terms depending on x_k as w_k. We can then rewrite (1.5) as

$$Q_n(x) = w_0 + \sum_{k=1}^{p} w_k x_k \tag{1.8}$$

which follows the form of a logistic regression presented in terms of the log odds.

Besides this clear connection, naive Bayes and logistic regression models comprise a Generative-Discriminative pair [2]. As Ng and Jordan note, generative classifiers learn a model of the joint probability $P(\mathbf{X}, Y)$ and generate predictions using Bayes rule to calculate $P(Y|\mathbf{X} = \mathbf{x})$, as done by the naive Bayes classifier. Discriminative classifiers attempt to model $P(Y|\mathbf{X} = \mathbf{x})$ directly, as done in the logistic regression example noted above. The authors note that generative classifiers approach their asymptotic error rate faster than discriminative classifiers, though discriminative classifiers have a lower asymptotic error rate. Therefore, there are two regions of performance: in smaller sample sizes, generative classifiers perform better since they more quickly approach their asymptotic error rate; in larger sample sizes, discriminative classifiers perform better since they have a lower asymptotic error rate. In this thesis, we develop a discriminative classifier that explicitly connects logistic regression to naive Bayes by using the log density ratios as features. We compare the performance of this model with naive Bayes and logistic regression.

1.2 Bayes classifier for functional data

In the section that follows, we borrow a significant portion of notation from Dai, Mueller, and Yao [3], but denote differences once they appear. We assume that our observed data arise from a common distribution (X, Y), where X is an observed square-integrable random function in $L^2(T)$. In this setup, we define T as a compact interval and $Y \in \{0, 1\}$ as a group label. Let $X_{(y)}$ be a random function which shares the same distribution as X if X is from population Π_y $(y = 0, 1)$, and let $\pi_y = P(Y = y)$ be the prior probability that an observation belongs to Π_y. We are interested in classifying a new observation X into one of two groups. The Bayes classifier assigns a new observation $X = x$ to Π_1 if

$$Q(x) = \frac{P(Y = 1|X = x)}{P(Y = 0|X = x)} > 1, \tag{1.9}$$

where X refers to a random predictor function and x a realized functional observation. Denote g_0 and g_1 as densities of X conditional on $Y = 0$ and $Y = 1$, respectively, where we assume the existence of a suitable dominating measure. Using Bayes' Theorem, we can equivalently write (1.9) as

$$Q(x) = \frac{\pi_1 g_1(x)}{\pi_0 g_0(x)}. \tag{1.10}$$

Dai, Mueller, and Yao approximate (1.10) by representing x and the random X by their projections onto an orthogonal basis $\{\psi_j\}_{j=1}^{\infty}$. Denote $\{x_j\}_{j=1}^{\infty}$ as projection scores via $x_j = \int_T x(t)\psi_j(t)dt$. Approximating the probabilities $P(Y = y|\ X = x)$ by $P(Y = y|x_1, \ldots, x_J)$, the approximate Bayes classifier is written as

$$Q(x) \approx \frac{P(Y = 1|x_1, \ldots, x_J)}{P(Y = 0|x_1, \ldots, x_J)} = \frac{\pi_1 f_1(x_1, \ldots, x_J)}{\pi_0 f_0(x_1, \ldots, x_J)} \tag{1.11}$$

where f_1 and f_0 represent the conditional densities, now with respect to the usual Lebesgue measure, for the first J random projection scores. We reserve f_y to refer only to conditional densities of random projection scores in the remainder of this work.

One method of simplifying (1.11) is to introduce assumptions that can ease the computation. Dai, Mueller, and Yao assumed that the covariances of the stochastic processes from which the functional data arise share the same eigenfunctions across classes. Let $\mu_y(t) = E\{X_{(y)}\}(t)$ and $G_y(s,t) = Cov\{X_{(y)}(s), X_{(y)}(t)\}$ be the mean and covariance functions conditional on class Y. Assume the covariances

$$G_y : L^2(T) \to L^2(T), \quad G_y(f)(t) = \int_T G_y(s,t)f(s)ds \tag{1.12}$$

are continuous. Then by Mercer's theorem [28],

$$G_y(s,t) = \sum_{j=1}^{\infty} \lambda_{yj}\phi_{yj}(s)\phi_{yj}(t), \tag{1.13}$$

where $\phi_{yj} = \phi_j$ are the shared orthonormal eigenfunctions ($j = 1, 2, \dots$) with corresponding eigenvalues λ_{yj} which remain nonnegative but monotonically decrease. The eigenvalues satisfy $\sum_{j=1}^{\infty} \lambda_{yj} < \infty$ where $y = 0, 1$ represents the class. One can set the shared eigenfunctions $\{\phi_j\}_{j=1}^{\infty}$ to be the projection directions $\{\psi_j\}_{j=1}^{\infty}$. An alternative approach is to use the partial least squares (PLS) basis for the covariance decomposition. See [29] and [30] for details.

As previously stated, a common assumption when constructing a Bayes classifier is to assume that the joint density of the features can be written as the product of the marginal densities via independence. This creates the naive Bayes classifier. For the functional Bayes classifier, under the assumption of a common eigenbasis $\{\phi_j\}_{j=1}^{\infty}$ between classes, projecting onto this basis yields uncorrelated scores within each class. Assuming these scores are not only uncorrelated but also independent, as would be the case for Gaussian functional data, yields the functional naive Bayes classifier. Taking the logarithm, (1.11) becomes

$$Q_J(x) = \log\left(\frac{\pi_1}{\pi_0}\right) + \sum_{j=1}^{J} \log\left(\frac{f_{1j}(x_j)}{f_{0j}(x_j)}\right), \tag{1.14}$$

denoting the density of the jth score under Π_y as f_{yj}. Using this classifier, one assigns a realization to Π_1 if $Q_J(x)$ is positive and to Π_0 otherwise. This classifier is presented in [3].

1.2.1 Functional Bayes classifier with copula

Huang and Ruppert proposed an extension of the functional Bayes classifier by adding a copula to the classifier $Q_J(x)$ [16]. We summarize this contribution here.

Note that (1.11) includes the computation of the joint density of projection scores. Depending on the choice of basis and the distribution of the data, these scores may exhibit some dependence or correlation between them. Copulas can be used to account for the dependence between these truncated projection scores.

As noted above, let $x_j = \int_T x(t)\phi_j(t)dt$ represent the jth projection score. We denote the copula for class y as a CDF C_y and its corresponding PDF c_y, with marginals all being uniform distributions on $[0, 1]$. Let F_{yj} and f_{yj} denote the marginal CDF and PDF of X_{yj}, the jth random projection score conditional on class $Y = y$. We can then define $F_y(x_1, \ldots, x_J)$ and $f_y(x_1, \ldots, x_J)$ as the joint CDF and PDF of $X_{y1}, \ldots, X_{yJ}$, respectively, as

$$\begin{aligned} F_y\left(x_1, \ldots, x_J\right) &= C_y\left\{F_{y1}\left(x_1\right), \ldots, F_{yJ}\left(x_J\right)\right\}, \\ f_y\left(x_1, \ldots, x_J\right) &= c_y\left\{F_{y1}\left(x_1\right), \ldots, F_{yJ}\left(x_J\right)\right\} f_{y1}\left(x_1\right) \cdots f_{yJ}\left(x_J\right). \end{aligned} \tag{1.15}$$

Our revised Bayes classifier then becomes

$$\log Q_J^*(x) = \log\left(\frac{\pi_1}{\pi_0}\right) + \sum_{j=1}^{J} \log\left\{\frac{f_{1j}\left(x_j\right)}{f_{0j}\left(x_j\right)}\right\} + \log\left\{\frac{c_1\left\{F_{11}\left(x_1\right), \ldots, F_{J1}\left(x_J\right)\right\}}{c_0\left\{F_{10}\left(x_1\right), \ldots, F_{J0}\left(x_J\right)\right\}}\right\} \tag{1.16}$$

where we assign new observations to class 1 if $\log Q_J^*(x) > 0$ and to class 0 otherwise.

1.3 Density ratio estimation

The estimation of density ratios is a classic and fundamental problem in the field of statistics. As demonstrated in Section 1.1, computing density ratios is the central aspect of building the Bayes classifier, which provides minimal misclassification rates, or the naive Bayes classifier, which is often developed in its place. Estimating density ratios pervades almost all areas of statistics, from building classification models, outlier detection [31], changepoint detection [32], deep learning [33], and machine learning in general [34]. See [35] for additional application areas and references regarding density ratio estimation.

In this section, we review two methods for estimating density ratios. We later compare a developed method for estimating density ratios with the first method presented here, taking the ratio of two kernel densities.

1.3.1 Ratio of two kernel density estimates

One of the most common methods of density ratio estimation is by taking the ratio of two kernel density estimates. Suppose that we obtain two independent samples of observations $X_{11}, \ldots, X_{1n_1} \overset{iid}{\sim} g_1$ and $X_{01}, \ldots, X_{0n_0} \overset{iid}{\sim} g_0$ where g_1 and g_0 have the same domain. Our goal is to estimate the ratio $d(x) = g_1(x)/g_0(x)$. One may estimate each density function first and then take the ratio as the estimate of d. In particular, the kernel estimate $\hat{d}_k$ of d is

$$\hat{d}_k(x) = \frac{\hat{g}_1(x)}{\hat{g}_0(x)}$$

where $\hat{g}_1$ and $\hat{g}_0$ are kernel density estimates of g_1 and g_0.

1.3.2 Silverman's approach

Next we detail an approach to density ratio estimation from Silverman [1], which has been updated in [36]. Suppose that we obtain two independent samples of observations on the real line $X_{11}, \ldots, X_{1n_1} \overset{iid}{\sim} g_1$ and $X_{01}, \ldots, X_{0n_0} \overset{iid}{\sim} g_0$ where g_1 and g_0 have the same domain. For the estimation of the density ratio $d(x) = g_1(x)/g_0(x)$, Silverman considered an almost equivalent problem: regard $X_{01}, \ldots, X_{0N_0}$ as the N_0 arrivals of an inhomogeneous Poisson process with intensity function $\lambda_0(x)$, and $X_{11}, \ldots, X_{1N_1}$ as the N_1 arrivals of an inhomogeneous Poisson process with intensity function $\lambda_1(x)$. Denote the combined and ordered samples as $Z_1, \ldots, Z_N$, where $N = N_1 + N_0$. Let $Y_i = 1$ if Z_i arises from the λ_1 process and $Y_i = 0$ if Z_i arises from the λ_0 process. We define $\lambda(x) = \lambda_0(x) + \lambda_1(x)$ and $\mu(x) = \lambda_1(x)/\lambda_0(x)$. Conditioning on the total number of events $N = n$, the $Z_1, \ldots, Z_N$ are order statistics. The joint density of $(Z_1, \ldots, Z_N, Y_1, \ldots, Y_N)$ can therefore be written as

$$\begin{aligned}
&f(z_1, \ldots, z_N, y_1, \ldots, y_N | N = n) \\
&= n! I(z_1 \leq z_2 \leq \cdots \leq z_n) \prod_{i=1}^{n} \lambda_{y_i}(z_i). \\
&= n! I(z_1 \leq z_2 \leq \cdots \leq z_n) \prod_{i=1}^{n} \lambda_0(z_i)(\lambda_1(z_i)/\lambda_0(z_i))^{y_i} \\
&= n! I(z_1 \leq z_2 \leq \cdots \leq z_n) \prod_{i=1}^{n} \frac{(\lambda_1(z_i) + \lambda_0(z_i))(\lambda_1(z_i)/\lambda_0(z_i))^{y_i}}{\frac{\lambda_0(z_i)+\lambda_1(z_i)}{\lambda_0(z_i)}} \\
&= n! I(z_1 \leq z_2 \leq \cdots \leq z_n) \prod_{i=1}^{n} \frac{\lambda(z_i)\mu(z_i)^{y_i}}{1 + \mu(z_i)}. \qquad (1.17)
\end{aligned}$$

We add that $N \sim Pois(\int_I \lambda(x))$ where I is the domain of functions g_0 and g_1. Therefore, the log-likelihood of λ and μ can be written as

$$l(\lambda, \mu) = -\int \lambda + N \log \int \lambda + \sum \log \lambda\left(Z_i\right) + l_1(\mu) \qquad (1.18)$$

where $l_1(\mu)$ depends only on the data and μ. When μ is given, we see that $(N, Z_1, \ldots, Z_N)$ form a minimal sufficient statistic for λ. We can treat the sum of the intensities λ as a nuisance parameter since the primary interest is in estimating the ratio μ. To write a likelihood for μ, note that the distribution of $(N, Z_1, \ldots, Z_N)$ depends only on the nuisance parameter λ. Thus, $(Y_1, \ldots, Y_N)$ is sufficient for μ following the definition by Fraser [37]. By conditioning on the sufficient statistic $(N, Z_1, \ldots, Z_N)$, we obtain the conditional log-likelihood l_1 of μ as

$$l_1(\mu) = \sum_{i=1}^{N} \left[y_i \log \mu\left(z_i\right) - \log\left\{1 + \mu\left(z_i\right)\right\}\right]. \tag{1.19}$$

Note that this expression is maximized when $\mu(z_i) = 0$ for each i where $y_i = 0$ and infinity for all other i. Maximizing this log-likelihood to obtain an estimate of μ would therefore provide poor estimates. Additionally, note that the log-likelihood is undefined when $\mu(z_i)$ is negative but, when $y_i = 0$, the likelihood increases as $\mu(z_i)$ approaches zero. To remedy this, we substitute $\alpha = \log \mu$ and write the conditional log-likelihood l_2 as

$$l_2(\mu) = \sum_{i=1}^{N} \left(y_i \alpha\left(z_i\right) - \log\left[1 + \exp\left\{\alpha\left(z_i\right)\right\}\right]\right). \tag{1.20}$$

This formulation yields a similar problem as (1.19) since (1.20) is maximized when $\alpha(z_i) = -\infty$ if $y_i = 0$ and $\alpha(z_i) = \infty$ if $y_i = 1$. However, one can use any spline model with a smoothing penalty to obtain a reasonable estimate for $\alpha(z_i)$. Further details on the estimation of $\alpha(z_i)$ using spline models can be found in [1, 38, 39, 40, 41] and in Chapter 2.

Finally, the estimate of log density ratio $\hat{d}_s(x)$ using a smoothing penalty can be written as

$$\hat{d}_s(x) = \frac{n_0}{n_1} \hat{\mu}(x) \tag{1.21}$$

where we scale our estimate $\hat{\mu}(x) = \exp(\hat{\alpha})$ by the ratio of number of observations in each class.

Chapter 2

Direct Density Ratio Estimation Using Splines

2.1 Introduction

The estimation of density ratios is a classic and important problem in the field of statistics. As demonstrated above, the Bayes classifier and many of its derivatives, including the naive Bayes classifier, depend on computing reliable estimates of density ratios between two classes. Modern machine learning approaches including generative adversarial networks (GANs) [42, 43, 33], reinforcement learning [44], and semi-supervised learning [45] also depend on estimating the ratio of two densities. Applications of density ratio estimation abound in changepoint detection [32, 46], outlier detection [31, 47], quantile estimation [48], variable selection [49], and other areas. Further details on density ratio estimation in machine learning can be found in [34].

Numerous methods for density ratio estimation exist [35]. The technique described in Section 2.2 differs from some previous procedures, such as the one described in Section 1.3.1, since it estimates the log density ratio directly. In general, estimating a desired quantity directly is preferred over estimating it via intermediate steps since direct estimates tend to have lower variance [50]. We continue with detailing this procedure before applying it to real and simulated datasets. Note that the method developed in [1] and summarized in Section 1.3.2 applies only to univariate density functions on real line. In Section 2.2 we propose a method for estimating density ratios defined on arbitrary domains.

2.2 Spline estimation of density ratio

Suppose that we observe (X,Y) where $Y = 1$ or $Y = 0$, $X \in \mathcal{X}$ and $\mathcal{X}$ is an arbitrary set. Let $g_1(x)$ denote the conditional density of $X|Y = 1$ and similarly let $g_0(x)$ denote the conditional density of $X|Y = 0$. Note that the domain of functions g_0 and g_1 are arbitrary sets which contain a subset of the real line as a special case. In particular, the domain could be a subset of a Euclidean $n-$space. Denote $\pi_y = P(Y = y)$ for $y = 0, 1$. Using Bayes' theorem, we can write

$$p(x) \triangleq P(Y = 1|X = x) = \frac{\pi_1 g_1(x)}{\pi_1 g_1(x) + \pi_0 g_0(x)} = \frac{rd(x)}{1 + rd(x)} \tag{2.1}$$

where r represents the ratio of prior probabilities, π_1/π_0, and $d(x)$ represents the density ratio, $g_1(x)/g_0(x)$.

The logistic probability, or the log odds ratio, can be written as

$$\eta(x) = \log \frac{p(x)}{1 - p(x)} = \log r + \log d(x). \tag{2.2}$$

Given data $\{(x_i, y_i),\ i = 1, \ldots, n\}$, $\eta(x)$ can be estimated nonparametrically. We assume that $\eta \in \mathcal{H}$ where $\mathcal{H}$ is a reproducing kernel Hilbert space (RKHS) and estimate

η as the minimizer of the penalized likelihood:

$$-\sum_{i=1}^{n} y_i \eta(x_i) + \sum_{i=1}^{n} \log(1 + \exp \eta(x_i)) + \lambda J(\eta) \tag{2.3}$$

where $J(\eta)$ is a penalty [51, 52]. The estimate of the log density ratio using the RKHS, $\hat{d}_h$, (referring to the Hilbert space h,) is defined as

$$\hat{d}_h(x) \triangleq \hat{\eta}(x) - \log \hat{r} \tag{2.4}$$

where $\hat{r}$ is an estimate of r. A simple estimate is $\hat{r} = \sum_{i=1}^{n} y_i/(n - \sum_{i=1}^{n} y_i)$. In comparison, Silverman [1] justified his method using the connection between inhomogeneous Poisson processes and density functions. He used the likelihood conditioned on sufficient statistics and treated the sum of intensity functions as a nuisance parameter. Therefore, his method is limited to density ratios defined on the real line. Note that the negative log-likelihood in the first two terms in (2.3) has the same form as the conditional log-likelihood (1.20). Therefore, with the same model space and penalty, these two approaches lead to the same estimate.

Here we consider the specific problem of classification. We use the likelihood conditional on $\{x_1, \ldots, x_n\}$ which is sufficient for the marginal density $f(x)$ where $f(x) = \pi_0 g_0(x) + \pi_1 g_1(x)$ is the nuisance parameter. There is no limitation regarding the domain of $f(x)$ as our method applies to general domains. For example, $\mathcal{X}$ can take the form of a circle or sphere. Additionally, $\mathcal{X}$ can be a subset of a Euclidean $n-$space. In the case where one knows that $k \leq p$ variables are correlated, one can estimate the joint density $f(x_1, \ldots, x_k)$ of the correlated variables rather than the marginal density for each variable. However, in the remainder of this thesis we only consider methods based on estimates of marginal densities.

The Bayes rule based on this estimated density ratio is equivalent to nonparametric logistic regression [51]. Specifically, the Bayes rule assigns an observation to Class 1 when $rd(x) \geq 1$ which is equivalent to $\eta(x) \geq 0$. In the dependence-adjusted naive Bayes method to be proposed in Section 3.1 later, we will estimate marginal density ratios and then fit logistic regression models with these estimated log density rations as features. Specifically, for multivariate $\mathbf{X} = (X_1, \ldots, X_p)$, we first estimate marginal density ratios $z_k = d_k(x_k)$ for $k = 1, \ldots, p$ where $d_k(x_k) = g_{1k}(x_k)/g_{0k}(x_k)$ and g_{1k} and g_{0k} are marginal densities of X_k conditional on classes 1 and 0 respectively. Therefore, we only need to solve (2.3) for univariate spline models for each marginal density ratio.

Define $\mathcal{X}_k$, an arbitrary set, as the domain of variable X_k. When $\mathcal{X}_k = [a, b]$, we may use the cubic spline Sobolev space $W_2^2[a, b]$ as $\mathcal{H}$. When $\mathcal{X}_k = \mathbb{R}$, we may use the cubic thin-plate spline space as $\mathcal{H}$. See [52] for choices of other model spaces.

We use the estimated marginal density ratios z_k as the features to logistic regression models. As illustrated in Section 3.3 and Section 3.4, using z_k as the features in a logistic regression may perform better than a traditional GLM. Note that the Bayes rule traditionally uses the full likelihood $f(y|x)f(x)$ while logistic regression uses the conditional likelihood, $f(y|x)$. In our approach, we estimate the density ratio using the conditional likelihood and treat the prior density $f(x)$ as a nuisance parameter.

2.3 Simulation studies

We present several simulations to evaluate and compare methods for the estimation of density ratios. The simulation settings include three distributional settings: 1) the ratio of two univariate Gaussians, where the only difference is in the mean, ($\sigma_0 = \sigma_1 = 2,\ \mu_0 = 2, \mu_1 = 4$), 2) the ratio of two univariate Gaussians, where the only difference is in the standard deviation ($\sigma_0 = 1.25,\ \sigma_1 = 2,\ \mu_0 = \mu_1 = 2$), and 3) the ratio of two Weibull(a, b) distributions, where the only difference is in a_y, the shape parameter ($b_0 = b_1 = 1,\ a_0 = 4,\ a_1 = 5$). These are evaluated at three sample sizes: small ($n = 50$), medium ($n = 300$), and large ($n = 1000$). In all settings, the two classes have the same number of observations. (For example, in the small settings, $n_0 = n_1 = 50$, where n_y is the number of training observations in class y.)

In each of these settings, we compare two methods of estimating the log density ratio: the spline method as described in Section 2.2, and the ratio of two kernel density estimates as described in Section 1.3.1. The spline estimates of the log density ratio were computed using a p-spline using the `gam` function from version 1.8.28 of the `mgcv` package in R [53]. These are fit by building a binomial GAM (setting `family = "binomial"`). The kernel density estimates were computed using the `density` function from the `stats` package in R. The Gaussian kernel and default bandwidth selection method were used.

To evaluate the estimates, we use a method of computing normalized mean square error (NMSE) as described in [36]. Denote r_i as the true density ratio estimate with $\hat{r}_i^{sp}$ and $\hat{r}_i^{kde}$ as its estimate from the spline and ratio of kernel density estimates, respectively. Then the NMSE for the spline estimate, NMSE_{sp} is defined as

$$\text{NMSE}_{sp} = \frac{1}{n}\sum_{i=1}^{n}\Big(\frac{\hat{r}_i^{sp}}{\sum_{i=1}^{n}\hat{r}_i^{sp}} - \frac{r_i}{\sum_{i=1}^{n} r_i}\Big)^2 \tag{2.5}$$

with an analogous formula for NMSE_{kde}. Normalizing in this way helps to prevent the mean square error from being skewed by large outliers. We present the results of these simulations in Table 2.1 and Figures 2.1 to 2.3 below. Note that the boxplots present the logarithm of NMSE to aid the comparison. As shown, the spline outperforms the KDE in all settings, and the density ratio estimation improves as n_0 increases.

Distribution	Method	n_0	Average NMSE	SD of NMSE
Gaussian1	KDE	50	2.2e-03	2.9e-03
Gaussian 1	Spline	50	4.3e-04	1.4e-03
Gaussian 1	KDE	300	5.8e-04	6.3e-04
Gaussian 1	Spline	300	1.5e-05	1.1e-04
Gaussian 1	KDE	1000	2.6e-04	2.1e-04
Gaussian 1	Spline	1000	3.7e-06	3.7e-05
Gaussian 2	KDE	50	2.6e-03	3.0e-03
Gaussian 2	Spline	50	6.4e-04	1.7e-03
Gaussian 2	KDE	300	7.9e-04	5.9e-04
Gaussian 2	Spline	300	2.8e-05	1.1e-04
Gaussian 2	KDE	1000	3.4e-04	1.6e-04
Gaussian 2	Spline	1000	9.3e-06	4.9e-05
Weibull	KDE	50	1.6e-03	2.9e-03
Weibull	Spline	50	4.4e-04	1.6e-03
Weibull	KDE	300	3.6e-04	5.6e-04
Weibull	Spline	300	2.3e-05	1.6e-04
Weibull	KDE	1000	1.6e-04	1.9e-04
Weibull	Spline	1000	3.1e-06	2.9e-05

Table 2.1: Simulation results reported as Average NMSE with associated standard deviations.
The Gaussian 1 setting calculates the ratio of univariate Gaussians, where the only difference is in the mean of the two classes: $\sigma_0 = \sigma_1 = 2,\ \mu_0 = 2, \mu_1 = 4$.
The Gaussian 2 setting calculates the ratio of univariate Gaussians, where the only difference is in the standard deviation of the two classes: $\sigma_0 = 1.25,\ \sigma_1 = 2,\ \mu_0 = \mu_1 = 2$.
The Weibull setting calculates the ratio of two Weibull(a, b) distributions: $b_0 = b_1 = 1,\ a_0 = 4,\ a_1 = 5$.

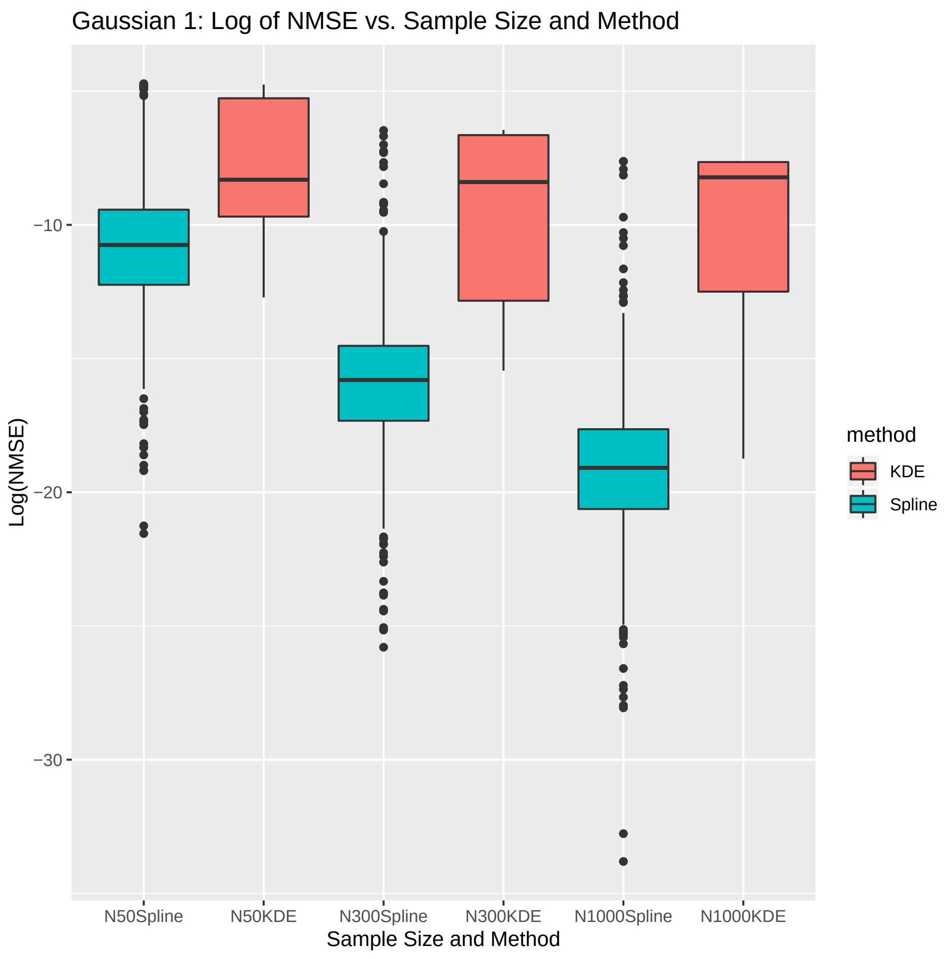

Figure 2.1: Log of the NMSE over 500 simulation runs for Gaussian 1. $\sigma_0 = \sigma_1 = 2,\ \mu_0 = 2, \mu_1 = 4$.

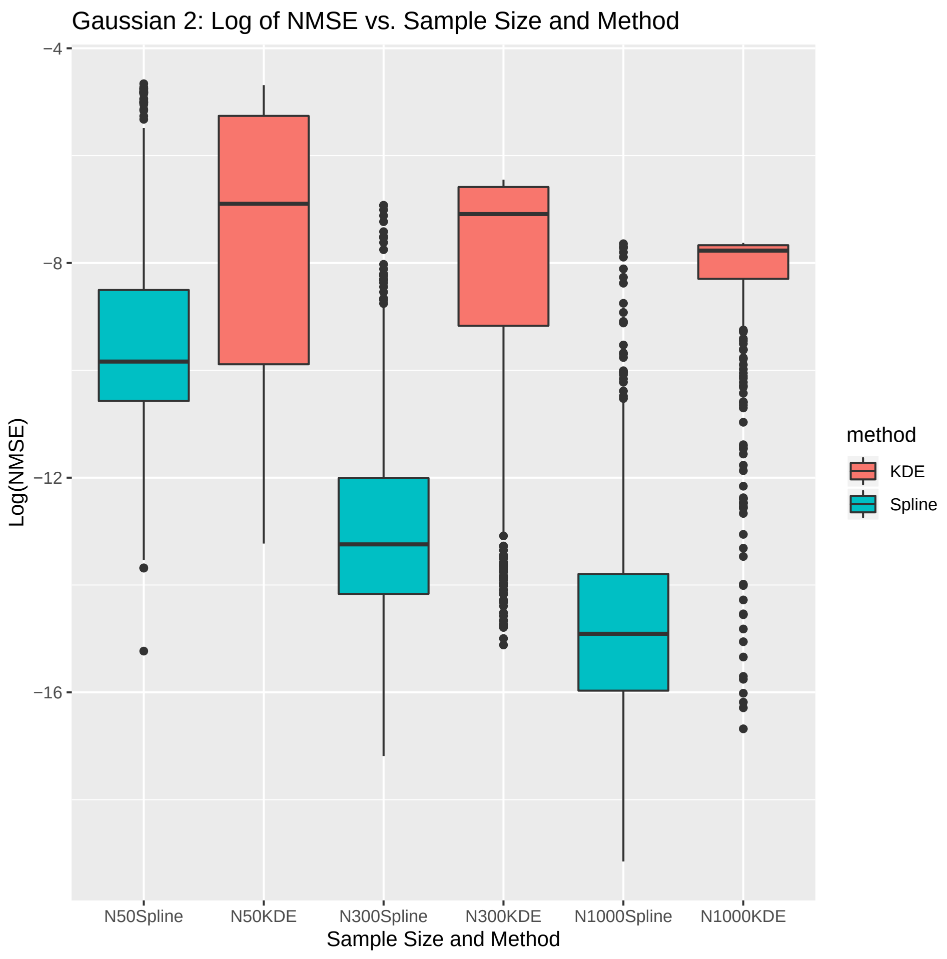

Figure 2.2: Log of the NMSE over 500 simulation runs for Gaussian 2. $\sigma_0 = 1.25,\ \sigma_1 = 2,\ \mu_0 = \mu_1 = 2$.

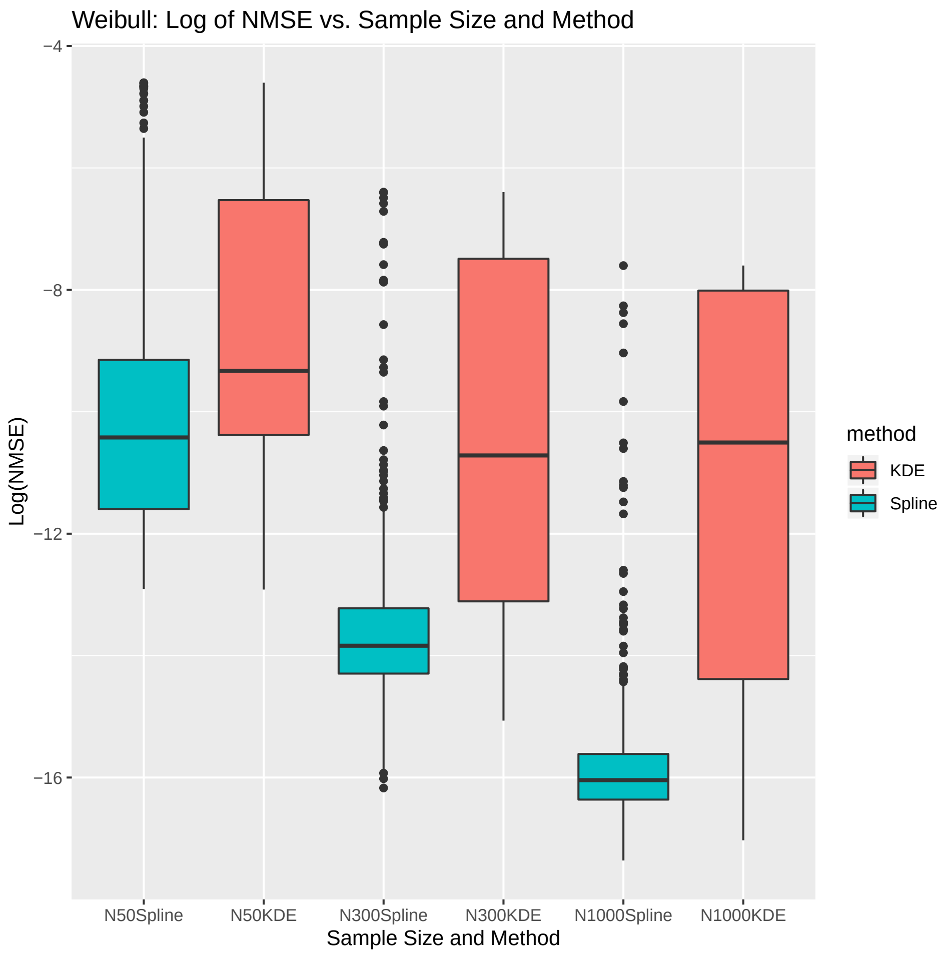

Figure 2.3: Log of the NMSE over 500 simulation runs for the Weibull setting. $b_0 = b_1 = 1,\ a_0 = 4,\ a_1 = 5$.

Chapter 3

DAB for Multivariate Data

We begin by introducing the DAB method in Section 3.1 and showing how it relates to both naive Bayes and logistic regression. Section 3.2 covers several interesting properties of the DAB method, including a case-by-case analysis of its ability to recover the optimal Bayes solution in certain Gaussian settings. We evaluate the performance of DAB using simulated and real data in Section 3.3 and Section 3.4.

3.1 The DAB method

Recall (1.5), the traditional naive Bayes classifier for multivariate data. As noted previously, the naive Bayes classifier results from assuming that the predictor variables in the optimal Bayes classifier are independent conditional on the class. Rather than assume the features are independent, we allow the marginal density ratios to be weighted to account for dependencies among them. Let Y be the class label and $\mathbf{X} = (X_1, \ldots, X_p)$ be p multivariate predictors. Let $Q_\alpha(\mathbf{x})$ denote a new classifier of multivariate observations $\mathbf{x} = (x_1, \ldots, x_p)$. Define z_k as the log ratio of marginal densities $z_k \triangleq \log \frac{g_{1k}(x_k)}{g_{0k}(x_k)}$, where g_{yk} is the density function of X_k conditional on $Y = y$. We seek $Q_\alpha(\mathbf{x})$ as a linear combination of derived features

$$Q_\alpha(\mathbf{x}) = \alpha_0 + \sum_{k=1}^{p} \alpha_k z_k \tag{3.1}$$

where the α_k represent weights that can be used to adjust for the dependencies among the marginal density ratios. This family of classifiers includes naive Bayes as a special case, where α_0 is the log ratio of the prior probabilities and $\alpha_k = 1$ for all $k > 0$. We estimate α_k by framing (3.1) as a logistic regression via the logit link

$$\operatorname{logit} p(\mathbf{x}) = \alpha_0 + \sum_{k=1}^{p} \alpha_k z_k. \tag{3.2}$$

In a traditional naive Bayes classifier, the only step is to estimate the log density ratio of each feature marginally. To estimate (3.2), we simply treat the estimated log density ratios as features and obtain α_k using a logistic regression with the logit link. Other link functions, such as the probit, can also be used. Additionally, other methods, such as support vector machines, can be used to estimate α_k.

The interpretation of coefficients α_k is straightforward. In a traditional logistic regression, $\operatorname{logit} p(\mathbf{x}) = \beta_0 + \sum_{k=1}^{p} \beta_k x_k$, we interpret β_k as the corresponding change in the log odds when x_k increases by 1, assuming all other features $x_j, j \neq k$ are held constant. We can interpret α_k in (3.2) as the corresponding change in the log-odds when z_k increases by 1, where z_k represents the log ratio of marginal densities. A change in the density ratio $\exp(z_k)$ of $100h\%$ will change the log-odds by $\alpha_k \log(1 + h)$. Thus, we explicitly connect the changes in the conditional density of $\mathbf{X}$ in the predictors to the conditional density of Y in the response. We name this classifier Dependence-Adjusted naive Bayes, or DAB, since the classifier retains the simplicity of naive Bayes while adjusting for the dependencies between features by estimating α_k.

Framing the classifier as a regression problem provides additional flexibility, namely the inclusion of interaction and higher order terms as well as regularization. We can generalize (3.1) as

$$Q_\alpha(\mathbf{x}) = \zeta(\mathbf{z}; \boldsymbol{\alpha}) \tag{3.3}$$

where ζ is a known function of $\mathbf{z} = (z_1, \ldots, z_p)^T$ with parameters $\boldsymbol{\alpha}$. In (3.1), ζ is merely the dot product of $\boldsymbol{\alpha}$ and $\mathbf{z}$ where $z_0 = 1$. However, ζ can also include interaction and higher-order terms as well as nonlinear functions of $\mathbf{z}$. In the sections that follow, we show several cases where the inclusion of higher-order terms provides further benefit than the naive Bayes classifier and in some cases recovers the form of the optimal Bayes classifier.

3.2 Properties of DAB

3.2.1 Approaching and recovering the optimal Bayes solution

One potential advantage of the DAB classifier comes through its ability to bridge the gap from the naive Bayes classifier to the optimal Bayes solution. We illustrate this below in the Gaussian setting of $\boldsymbol{X}|Y = y \sim N(\boldsymbol{\mu}_y, \Sigma_y)$, but it may hold in more general cases. We proceed by explaining in which Gaussian situations DAB may recover the optimal Bayes solution, and in which situations neither DAB nor naive Bayes can recover the optimal Bayes solution.

Assume that $\mathbf{X}|Y = y \sim N(\boldsymbol{\mu}_y, \Sigma_y)$ where $y \in \{0, 1\}$ and Σ_y is $p \times p$ and positive definite. Let g_y represent the multivariate density under class y. Define the vector $\mathbf{b}^T_{1\times p} \triangleq \boldsymbol{\mu}_0^T \Sigma_0^{-1} - \boldsymbol{\mu}_1^T \Sigma_1^{-1}$ and define $S \triangleq \Sigma_0^{-1} - \Sigma_1^{-1}$ with the ikth element S_{ik}. We can write the log ratio of the joint density as

$$\begin{aligned}\log\frac{g_1(\mathbf{x})}{g_0(\mathbf{x})} &= \frac{1}{2}\log\frac{|\Sigma_0|}{|\Sigma_1|} + \frac{1}{2}\Big((\mathbf{x}-\boldsymbol{\mu}_0)^T\Sigma_0^{-1}(\mathbf{x}-\boldsymbol{\mu}_0) - (\mathbf{x}-\boldsymbol{\mu}_1)^T\Sigma_1^{-1}(\mathbf{x}-\boldsymbol{\mu}_1)\Big)\\ &= b_0 + \frac{1}{2}\mathbf{x}^{\mathbf{T}}(\Sigma_0^{-1}-\Sigma_1^{-1})\mathbf{x} - (\boldsymbol{\mu}_0^T\Sigma_0^{-1} - \boldsymbol{\mu}_1^T\Sigma_1^{-1})\mathbf{x}\\ &= b_0 + \frac{1}{2}\mathbf{x}^{\mathbf{T}}S\mathbf{x} - \mathbf{b}^T\mathbf{x} \end{aligned} \tag{3.4}$$

where b_0 represents a constant term that does not depend on x.

We compare the form presented in (3.4) via the joint density with the log ratio of the marginal densities. We can write the marginal log density ratio z_k of X_k as

$$\begin{aligned} z_k &= \frac{1}{2}\log\frac{\sigma_0^2}{\sigma_1^2} + \frac{1}{2}\Big(\Big(\frac{x_k-\mu_{0k}}{\sigma_{0k}}\Big)^2 - \Big(\frac{x_k-\mu_{1k}}{\sigma_{1k}}\Big)^2\Big)\\ &= \alpha_{0k} + \frac{1}{2}\Big(\sigma_{0k}^{-2} - \sigma_{1k}^{-2}\Big)x_k^2 - \Big(\frac{\mu_{0k}}{\sigma_{0k}^2} - \frac{\mu_{1k}}{\sigma_{1k}^2}\Big)x_k \end{aligned} \tag{3.5}$$

where α_{0k} is a constant. Eq. (3.5) represents the marginal log density ratio of one variable. However, we are interested in computing a linear combination of marginal density ratios as shown in (3.1). Thus, when building the DAB classifier, we are actually computing

$$Q_\alpha(\mathbf{x}) = \alpha_0 + \sum_{k=1}^{p}\alpha_k\alpha_{0k} + \sum_{k=1}^{p}\alpha_k\Big(\frac{1}{2}\Big(\sigma_{0k}^{-2} - \sigma_{1k}^{-2}\Big)x_k^2 - \Big(\frac{\mu_{0k}}{\sigma_{0k}^2} - \frac{\mu_{1k}}{\sigma_{1k}^2}\Big)x_k\Big). \tag{3.6}$$

The weights α_k in (3.6) serve the same purpose as in (3.1). We endeavor to compare (3.4) with (3.6) to determine where (3.6), under the correct choice of α_k, can recover the optimal Bayes solution as a function of $\mathbf{x}$. Notice that both the log of the joint density ratio, (3.4), and the log of the marginal density ratio, (3.6), are quadratic functions of $\mathbf{x}$. Thus, comparing these two forms amounts to identifying the circumstances where the log ratios of the marginal densities can match terms with the log ratio of the joint density.

We present four cases below. These cases are 1) the classic case of a difference in means between the classes but a shared covariance; 2) the case where the quadratic terms in (3.6) cancel; 3) the case where the linear terms in (3.6) cancel; and 4) the case where no cancellation occurs.

Case 1: $\Sigma_0 = \Sigma_1 = \Sigma, \mu_{0k} \neq \mu_{1k}$ **for all** k

When the two classes share the same covariance but have different means, the optimal Bayes classifier is

$$\log \frac{g_1(\mathbf{x})}{g_0(\mathbf{x})} = b_0 - \mathbf{b}^T\mathbf{x} \tag{3.7}$$

where b_0 is a constant that does not depend on $\mathbf{x}$. Recall the vector $\mathbf{b}^T_{1\times p} \triangleq (\boldsymbol{\mu}_0^T - \boldsymbol{\mu}_1^T)\Sigma^{-1} = (b_1, \ldots, b_p)^T$ since $\Sigma_0 = \Sigma_1$. We can equivalently write (3.7) as

$$\log \frac{g_1(\mathbf{x})}{g_0(\mathbf{x})} = b_0 - \sum_{k=1}^{p} b_k x_k. \tag{3.8}$$

Define $\delta_k \triangleq \mu_{0k} - \mu_{1k}$ and set $\alpha_k = -b_k\sigma_k^2/\delta_k$ and $b_0 = \alpha_0 + \sum_{k=1}^{p}\alpha_k\alpha_{0k}$. Then the DAB classifier is written as

$$\begin{aligned} Q_\alpha(\mathbf{x}) &= \alpha_0 + \sum_{k=1}^{p} \alpha_k z_k \\ &= \alpha_0 + \sum_{k=1}^{p} \alpha_k \Big(\alpha_{0k} + \Big(\frac{\delta_k}{\sigma_k^2}\Big)x_k\Big) \\ &= b_0 - \sum_{k=1}^{p} b_k x_k. \end{aligned} \tag{3.9}$$

Note that (3.9) contains no quadratic terms since the quadratics cancel out of the marginal density ratio. By setting $\alpha_k = -b_k\sigma_k^2/\delta_k$, we can recover the form of the optimal Bayes classifier presented in (3.8). If any δ_k were to equal 0, the corresponding z_k would also be 0. Thus, we may not be able to recover the optimal Bayes solution.

If $\Sigma = \Sigma_0 = \Sigma_1$ is a diagonal matrix, $b_k = \delta_k/\sigma_k^2$. In this case, $\boldsymbol{\alpha} = \mathbf{1}$. Therefore, DAB recovers the naive Bayes solution, which is also the optimal Bayes rule.

Case 2: $\Sigma_0 \neq \Sigma_1, \sigma_{0k}^2 = \sigma_{1k}^2$ for all $k, \mu_{0k} \neq \mu_{1k}$ for all k

When the two classes have different covariances and different means, but the same variances, the optimal Bayes classifier is written as (3.4). The DAB classifier initially takes the form of (3.9) since the quadratic terms in the marginal log density ratios cancel. We can add higher-order terms to the DAB classifier in (3.9) to recover the joint density ratio completely. Let $\alpha_{jk} = S_{jk}\sigma_k^2\sigma_j^2/(2\delta_k\delta_j), \alpha_k = -(b_k\frac{\sigma_k^2}{\delta_k} + 2\sum_{j=1}^p \alpha_{kj}\alpha_{0j})$, and set $b_0 = \alpha_0 + \sum_{k=1}^p \alpha_k\alpha_{0k} + \sum_{k=1}^p\sum_{j=1}^p \alpha_{jk}\alpha_{0j}\alpha_{0k}$. We construct the DAB classifier for Case 2 below. We denote this classifier as $Q_{\alpha,2}(\mathbf{x})$ to note that we are extending DAB to include second order terms.

$$\begin{aligned}
Q_{\alpha,2}(\mathbf{x}) &= \alpha_0 + \sum_{k=1}^p \alpha_k z_k + \sum_{k=1}^p\sum_{j=1}^p \alpha_{jk} z_j z_k \\
&= \alpha_0 + \sum_{k=1}^p \alpha_k\Big(\alpha_{0k} + \Big(\frac{\delta_k}{\sigma_k^2}\Big)x_k\Big) + \sum_{k=1}^p\sum_{j=1}^p \alpha_{jk}\Big(\alpha_{0j} + \Big(\frac{\delta_j}{\sigma_j^2}\Big)x_j\Big)\Big(\alpha_{0k} + \Big(\frac{\delta_k}{\sigma_k^2}\Big)x_k\Big) \\
&= \alpha_0 + \sum_{k=1}^p \alpha_k\alpha_{0k} + \sum_{k=1}^p\sum_{j=1}^p \alpha_{kj}\alpha_{0k}\alpha_{0j} + \sum_{k=1}^p \Big(\alpha_k + 2\sum_{j=1}^p \alpha_{kj}\alpha_{0j}\Big)\frac{\delta_k}{\sigma_k^2}x_k \\
&\quad + \sum_{k=1}^p\sum_{j=1}^p \alpha_{kj}\frac{\delta_j\delta_k}{\sigma_j^2\sigma_k^2}x_j x_k \\
&= b_0 - \mathbf{b}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T S\mathbf{x}.
\end{aligned} \tag{3.10}$$

Under the proper choice of the linear terms α_k and higher order terms α_{jk}, we can recover the quadratic form of the joint density ratio in (3.4) exactly.

Case 3: $\Sigma_0 \neq \Sigma_1, \sigma_{0k}^2 \neq \sigma_{1k}^2, \frac{\mu_{0k}}{\sigma_{0k}^2} \neq \frac{\mu_{1k}}{\sigma_{1k}^2}$ **for some k**

The optimal Bayes classifier takes its full quadratic form specified in (3.4). The DAB classifier takes its full quadratic form specified in (3.6). We cannot recover the optimal Bayes solution since we cannot estimate separate α_k for the linear and quadratic coefficients using the marginal density ratios. This occurs because the linear and quadratic coefficients are linked together by α_k. However, the additional flexibility offered by DAB may allow it to outperform naive Bayes.

Case 4: $\Sigma_0 \neq \Sigma_1, \mu_{0k} = \mu_{1k}, \sigma_{0k}^2 \neq \sigma_{1k}^2$ **for some k**

Note that this is a special case of Case 3, since the requirement that $\mu_{0k} = \mu_{1k}$ for some k ensures that $\frac{\mu_{0k}}{\sigma_{0k}^2} \neq \frac{\mu_{0k}}{\sigma_{0k}^2}$. In this case, the optimal Bayes classifier takes its full form specified in (3.4). Let z_k be an example instance where $\mu_{0k} = \mu_{1k}$ and $\sigma_{0k}^2 \neq \sigma_{1k}^2$. The marginal log density ratio z_k is written as

$$z_k = \frac{1}{2}\log\frac{\sigma_0^2}{\sigma_1^2} + \frac{1}{2}(\sigma_{0k}^{-2} - \sigma_{1k}^{-2})x_k^2. \tag{3.11}$$

Here z_k only contains terms that are quadratic in x_k, but the joint log density ratio may contain terms that are linear in x_k. Therefore, we cannot recover the optimal Bayes solution since some features in the joint density ratio cannot be recovered from the marginals.

In general, DAB can recover the optimal Bayes solution when the marginal density ratios z_k are all linear in $\mathbf{x}$, as shown in Cases 1 and 2. When there are quadratic terms in some of the z_k, it may be impossible to recover the optimal Bayes solution. This is described in Cases 3 and 4.

3.2.2 Negative coefficients in the Bayes classifier

One notable disadvantage of the naive Bayes classifier is that it does not allow for the creation of fractional or negative coefficients in its sum. We show below that there are simple bivariate Normal cases where including negative coefficients for the marginal ratios would be appropriate.

Consider Case 1 from above, where $\Sigma_0 = \Sigma_1$ but $\mu_{0k} \neq \mu_{1k}$ for all k. Let $\delta_k \triangleq \mu_{0k} - \mu_{1k}$, let ρ denote the correlation between X_1 and X_2, and let σ_1 and σ_2 denote the corresponding standard deviations. As shown in (3.7), the optimal $\alpha_k = -b_k\sigma_k^2/\delta_k$ where $\mathbf{b}_{1\times p} \triangleq (\mu_0^T - \mu_1^T)\Sigma^{-1}$.

To compute $\boldsymbol{\alpha}$, we can write $\mathbf{b}^T = \boldsymbol{\delta}^T\Sigma^{-1}$ as

$$\boldsymbol{\delta}^T\Sigma^{-1} = \frac{1}{(\sigma_1\sigma_2)^2(1-\rho^2)} \begin{pmatrix} \sigma_2(\delta_1\sigma_2 - \delta_2\rho\sigma_1) \\ \sigma_1(\delta_2\sigma_1 - \delta_1\rho\sigma_2) \end{pmatrix}^T . \tag{3.12}$$

Let D denote the matrix with σ_k^2/δ_k on the diagonal and 0 in all remaining elements. We can calculate $\boldsymbol{\alpha}$ as a matrix computation:

$$\begin{pmatrix} \alpha_1 \\ \alpha_2 \end{pmatrix} = -\mathbf{b}^T D = \begin{pmatrix} \frac{1}{(\sigma_1\sigma_2)^2(1-\rho^2)} \frac{\sigma_2\sigma_1^2}{\delta_1}(\delta_2\rho\sigma_1 - \delta_1\sigma_2) \\ \frac{1}{(\sigma_1\sigma_2)^2(1-\rho^2)} \frac{\sigma_1\sigma_2^2}{\delta_2}(\delta_1\rho\sigma_2 - \delta_2\sigma_1) \end{pmatrix}^T . \tag{3.13}$$

The sign of each coefficient is determined by

$$\begin{aligned} \operatorname{sign}(\alpha_1) &= \operatorname{sign}\left(\frac{\delta_2\rho\sigma_1 - \delta_1\sigma_2}{\delta_1}\right), \text{ and} \\ \operatorname{sign}(\alpha_2) &= \operatorname{sign}\left(\frac{\delta_1\rho\sigma_2 - \delta_2\sigma_1}{\delta_2}\right). \end{aligned} \tag{3.14}$$

For α_1 in particular, if δ_1 and δ_2 share the same sign, then the condition $\rho|\delta_2|\sigma_1 < |\delta_1|\sigma_2$ implies that α_1 is negative. When δ_1 and δ_2 have opposite signs, $\rho|\delta_2|\sigma_1 > |\delta_1|\sigma_2$

will ensure a negative value of α_1. These two conditions can also be written as

$$\begin{aligned} &\text{If } \operatorname{sign}(\delta_1) = \operatorname{sign}(\delta_2),\ \frac{\delta_1\sigma_2}{\delta_2\sigma_1} > \rho \implies \alpha_1 < 0. \\ &\text{If } \operatorname{sign}(\delta_1) \neq \operatorname{sign}(\delta_2),\ \frac{\delta_1\sigma_2}{\delta_2\sigma_1} < \rho \implies \alpha_1 < 0. \end{aligned} \tag{3.15}$$

Similar conditions apply to obtain a negative coefficient for α_2. Thus, it is evident that there are common situations where negative coefficients would be appropriate to include when using Bayes' rule for classification. Fig. 3.1 below plots the eigenvectors of the covariance matrix Σ for four values of ρ where $\mu_0 = (0, 0), \mu_1 = (1, 1), \delta_1 = \delta_2 = 1, \sigma_1 = 1$ and $\sigma_2 = 1/2$. By (3.15), $\rho < 0.5$ would give a negative α_1. We plot the eigenvectors below for $\rho \in \{0, .25, .5, .75\}$ to demonstrate the change in the shape of the ellipse.

3.3 Simulation examples

We proceed by detailing simulation examples which demonstrate the performance of the DAB classifier on simulated datasets. We adopt a factorial design with two choices of dimension $p = 5$ and $p = 10$, three choices of sample size $n_2 = 50, 150$, and 500, and three choices of conditional distributions, multivariate Gaussian, mixture of multivariate Gaussians, and multivariate uniform. We use n_2 to denote the number of observations in the smaller class and $n_1 = 2n_2$ to denote the number of observations in the larger class. This setup allows us to evaluate the performance of the models on imbalanced classes. All tests were performed on sample sizes equal to those used to train the models.

In the multivariate normal setting, $\mathbf{X}|Y = y \sim N(\boldsymbol{\mu}_y, \Sigma_y)$ where $\boldsymbol{\mu}_0 = \mathbf{0}, \boldsymbol{\mu}_1 = 2\mathbf{1}$, where $\mathbf{0}$ and $\mathbf{1}$ denote a p-vector of 0s and 1s, respectively. We set $\Sigma_0 = I_p$, a $p \times p$ identity matrix, and $\Sigma_1 = (1 - \rho)I_p + \rho\mathbf{1}\mathbf{1}^T$, a $p \times p$ matrix with 1 on the diagonals and ρ on the off-diagonals. The $\boldsymbol{\mu}_y$ vector has length p for each simulation.

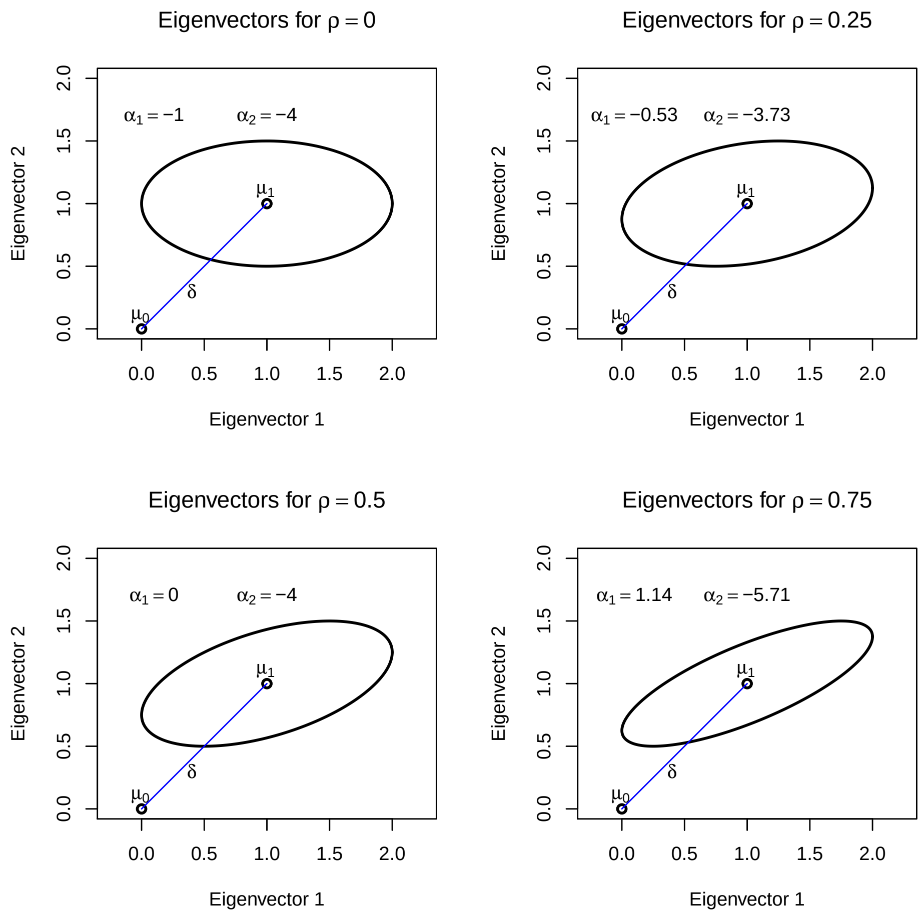

Figure 3.1: Plot of the eigenvectors for different values of ρ.

In the mixture of Gaussians setting, we set $g_y(\mathbf{x}) = \pi_{y1}\phi(\mathbf{x}|\boldsymbol{\mu}_{y1}, \Sigma_{y1})+\pi_{y2}\phi(\mathbf{x}|\boldsymbol{\mu}_{y2}, \Sigma_{y2})$ where $y = 0$ or 1, $\pi_{01} = \pi_{11} = \pi_{02} = \pi_{12} = 0.5$, $\boldsymbol{\mu}_{11} = \mathbf{0}, \boldsymbol{\mu}_{12} = \mathbf{1}, \boldsymbol{\mu}_{01} = \mathbf{1}, \boldsymbol{\mu}_{02} = 2\mathbf{1}, \Sigma_{01} = \Sigma_{02} = I_p, \Sigma_{11} = \Sigma_{12} = (1-\rho)I_p + \rho\mathbf{1}\mathbf{1}^T$, and $\phi(\mathbf{x}|\boldsymbol{\mu}, \Sigma)$ is the density function of a multivariate Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix Σ.

In the multivariate uniform setting, we generate $\mathbf{X}|Y = 0$ as a p-dimensional Uniform(0,1) with covariance $\Sigma_0 = I_p$. Similarly we generate $\mathbf{X}|Y = 1$ as a p-dimensional Uniform(0.1, 1.1) with covariance $\Sigma_1 = (1-\rho)I_p + \rho\mathbf{1}\mathbf{1}^T$. These were generated using the `MultiRNG` package in R [54], which implements an approach described in [55].

The simulations below are presented in the following order. The first plot contains simulations of the multivariate Gaussian setting in increasing sample size for $p = 5$. The second plot contains the same simulation but $p = 10$. The next two plots follow the same setup but are done using data from the mixture of Gaussians setting. The last two plots show results for the multivariate uniform. The graphs show error rates from ten methods for the Gaussian setting and error rates from eight methods for the non-Gaussian settings. The error rates represent the average misclassification rate over 10 replications on out-of-sample data. These are matched to the legend as follows:

- **Bayes-true**: This line represents the optimal error rates using Bayes rule by taking the true log density ratio of the two classes. This serves as a benchmark for all comparisons. For the multivariate Gaussian and mixture of Gaussian cases, we simply take the log density ratio of the PDFs as described above. For the multivariate uniform case, we characterize the joint density of the multivariate uniform using a Gaussian copula and the marginal densities. Similar to (1.15), let

$$f_y\left(x_1,\ldots,x_p\right) = c_y\left\{F_{y1}\left(x_1\right),\ldots,F_{yp}\left(x_p\right)\right\} f_{y1}\left(x_1\right)\cdots f_{yp}\left(x_p\right).$$

  where $f_y(x_1,\ldots,x_p)$ represents the joint density of $(x_1,\ldots,x_p)$ conditional on class y, c_y represents the Gaussian copula density conditional on class y, $F_{yp}(x_p)$ represents the pth marginal CDF of x_p conditional on class y, and $f_{yp}(x_p)$ represents the pth marginal density of x_p conditional on class y.

  Let Σ denote the correlation matrix of the Gaussian copula, and denote $\boldsymbol{q}_{1\times p} = (\Phi^{-1}(u_1),\ldots,\Phi^{-1}(u_p))$ where $\Phi^{-1}(u)$ denotes the inverse of a standard Gaussian CDF. The Gaussian copula density conditional on class y is provided by [56] as

$$c_y(u) = \frac{1}{\sqrt{\det\Sigma}}\exp\Big[-\frac{1}{2}\mathbf{q}(\Sigma^{-1}-I)\mathbf{q}^T\Big]. \tag{3.16}$$

This representation allows us to evaluate the joint density of the multivariate uniform in both classes.

- **NB-true (Gaussian only)** This line represents error rates using the naive Bayes classifier. However, we know the true marginal log density ratio, so we use the true mean and variance in calculating the ratio.

- **Bayes-est (Gaussian only)** This is similar to Bayes-true, but here we are using the sample values to estimate the log density ratio.

- **NB-est (Gaussian) / NB (others)** This is using the naive Bayes rule, but we use sample estimates of the mean and variance in the log density ratio.

- **GLM-linear** This line represents error rates from fitting a full GLM to the data itself without taking log density ratios. Here we only use the linear terms in the GLM.

- **GLM-quad** This line represents error rates from fitting a full GLM with quadratic and interaction terms to the data itself without taking log density ratios.

- **DAB1-linear** This line represents using kernel density estimation to estimate the marginal log density ratios and then fitting a GLM to the estimated log density ratios. The kernel density estimation was done using the `density` function in R with default arguments. Thus, we used the Gaussian kernel and a bandwidth chosen by Silverman's method described in (3.31) of [57]. The same settings were used for DAB1-quadratic. This GLM contains only linear terms. This is a version of the DAB model.

- **DAB1-quadratic** This line represents using kernel density estimation to estimate the marginal log density ratios and then fitting a GLM to the estimated log density ratios. The GLM contains linear and higher order (quadratic and interaction) terms. This is a version of the DAB model.

- **DAB2-linear** This line represents using a p-spline to estimate the marginal log density ratios and then fitting a GLM to the estimated log density ratios. The p-spline was fit using the `gam` function from version 1.8.28 of the `mgcv` package in R [58]. These are fit by building a binomial GAM (setting `family = "binomial"`). This implementation uses penalized regression splines to represent the smooth functions. The smoothing parameter used was the default setting of GCV.Cp, which uses Mallows' Cp in the case of a binomial family. The same settings were used for DAB2-linear. The GLM contains only linear terms. This is a version of the DAB model.

- **DAB2-quad** This line represents using a spline to estimate the marginal log density ratios and then fitting a GLM to the estimated log density ratios. The GLM contains linear and higher order (quadratic and interaction) terms. This is a version of the DAB model.

The Gaussian plots in Fig. 3.2 and Fig. 3.3 correspond to a special case of Case 1 ($\Sigma = I_p$) and Case 2 (all ρ) above. The true joint density ratio outperforms all other candidate models, as expected. At $p = 5$ and $n_1 = 100$, DAB1-linear performs similarly to GLM and NB. However, as the sample size increases, DAB1-quad and DAB2-quad, the density ratio estimates with all higher order and interaction terms, perform almost as well as the true joint density ratio in Fig. 3.2.

In Fig. 3.3 the trend is similar. At $p = 10$ and $n_1 = 100$, naive Bayes works about as well DAB2-quad. At $n_1 = 300$, DAB1-linear and GLM-linear are the best alternatives to the true Bayes ratio until $\rho \approx 0.8$, where the quadratic models do well. At $n_1 = 1000$, DAB1-linear and DAB2-linear do well until $\rho \approx .6$, at which point DAB1-quad and

DAB2-quad do better. However, these error rates do not match the true Bayes ratio as well as we see in the $p = 5$ case. Thus, when there are additional covariates in the model, it appears that we need a larger sample size to perform as well as the optimal Bayes classifier.

The mixture of Gaussians plots in Fig. 3.4 and Fig. 3.5 show similar behavior when $p = 5$ or $p = 10$. When $p = 5$ and $n_1 = 100$, GLM-linear, DAB1-linear, DAB2-linear, and naive Bayes all exhibit very similar performance. These models provide the lowest misclassification error until $\rho \approx 0.5$, where the quadratic models GLM-quad, DAB1-quad, and DAB2-quad do better. Of these, DAB2-quad outperforms the others once $\rho > 0.5$. When $p = 5$ and $n_1 = 300$, the linear and quadratic models do equally well until $\rho \approx 0.35$. From there onwards, the quadratic models do better with DAB1-quad and DAB2-quad each providing lower misclassification rates than GLM-quad. When $n_1 = 1000$, we see the same trend except that the ρ at which the quadratic models begin to do better is now ≈ 0.2.

When $p = 10$ and $n_1 = 100$, the linear models serve as the best alternatives until $\rho \approx 0.6$. At low values of ρ the quadratic models have the highest misclassification error but they provide the lowest misclassification error when $\rho > 0.6$. When $n_1 = 300$, the linear models again do better until $\rho \approx 0.35$, where the quadratic models outperform it. DAB2-quad provides the lower misclassification rates than the other two quadratic models in both cases. Last, when $n_1 = 1000$, DAB1-quad and DAB2-quad each do well for all values of ρ. They do better than the linear models at $\rho \approx .15$.

In all mixture of Gaussians cases, all models besides DR2-quad, DR1-quad, and GLM-quad perform worse as ρ increases. Estimating the density ratio and adding all quadratic and interaction terms generally performed better than adding all quadratic terms using a standard GLM on the data itself.

The multivariate uniform settings appeared to be more challenging to classify than the Gaussian and mixture of Gaussians settings. When $p = 5$, DAB1-linear provided the lowest misclassification until $\rho \approx 0.6$ when $n_1 = 100$, until $\rho \approx 0.5$ when $n_1 = 300$, and until $\rho \approx 0.6$ when $n_1 = 1000$. The DAB2-quadratic model provides the lowest misclassification error at high values of ρ. The same trend occurs when $p = 10$. DAB1-linear performs best until $\rho \approx 0.6$ when $n_1 = 100$ and it performs best until $\rho \approx 0.5$ when $n_1 = 300$ and $n_1 = 1000$. At high values of ρ, DAB2-quad does best along with GLM-quad. Interestingly, the worst option for the multivariate uniform case was DAB2-linear, or the linear model using spline estimates of the density ratio.

The DAB method generally outperforms the GLM variants in these simulations. The quadratic methods initially perform worse than the linear counterparts under small sample settings and in settings of small ρ. However, as the sample size and ρ increase, the quadratic DAB models outperform all other alternatives. This may arise due to not only the increased dependence in one of the two classes as indicated by increasing ρ, but also an improvement in the ability to estimate the log density ratio as the sample grows larger.

Of the two methods used for density ratio estimation, where DAB1 indicates the ratio of kernel density estimates and DAB2 indicates the spline method of density ratio estimation, DAB2-quadratic and DAB2-linear performed at least as well as their DAB1 counterparts in all Gaussian and mixture of Gaussian settings. Interestingly, in all settings of the multivariate uniform, DAB1-linear gave lower misclassification rates than DAB2-linear at all values of ρ. DAB1-quad also gave lower misclassification rates than DAB2-quad for small and medium values of ρ, but DAB2-quad did better for large values of ρ.

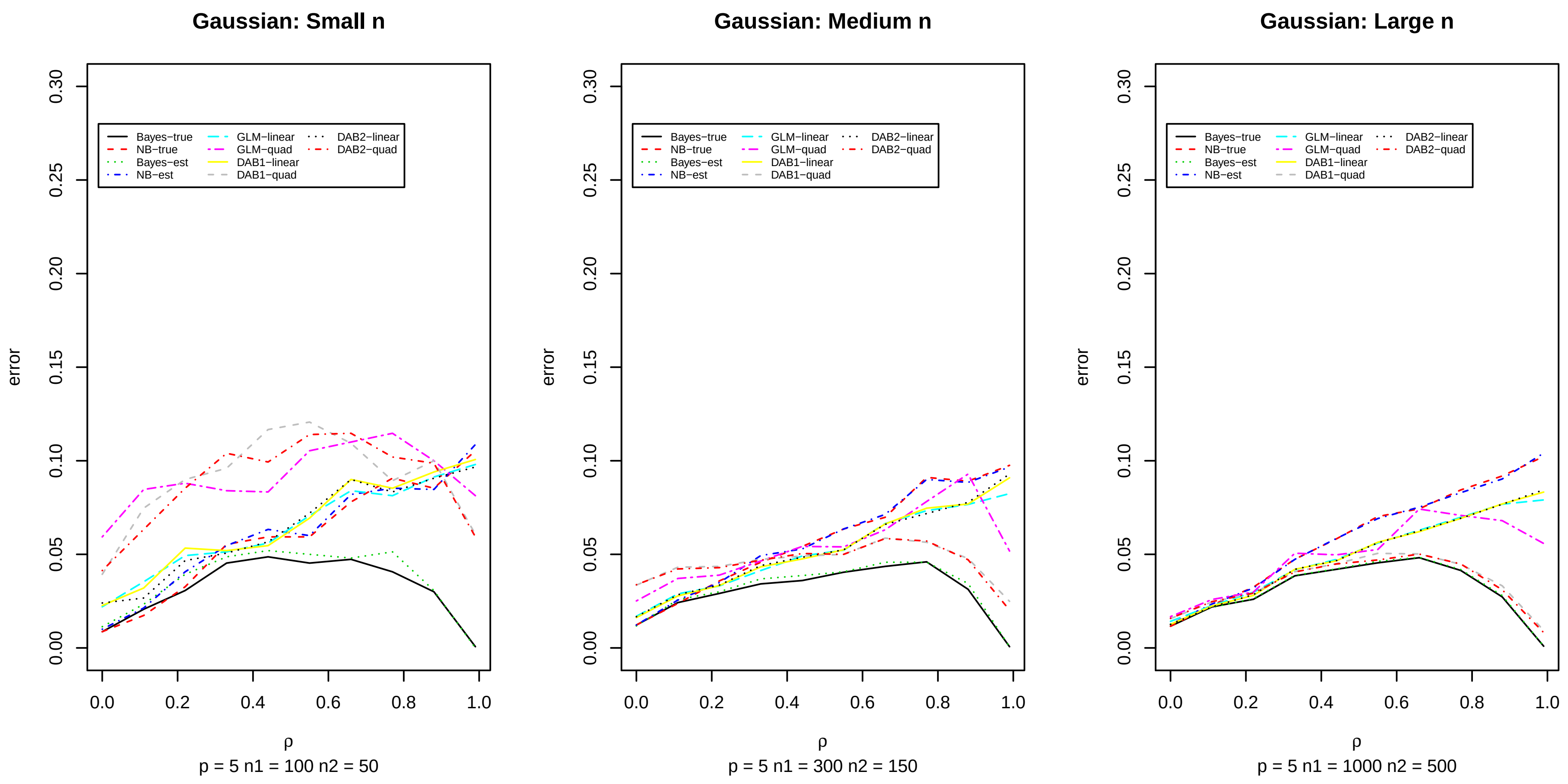

Figure 3.2: Simulation results from multivariate Gaussian data. The y-axis represents average misclassification error over 10 replications. The x-axis represents the ρ parameter between the two classes. $p = 5$ in all plots. The plots are ordered in increasing level of sample size.

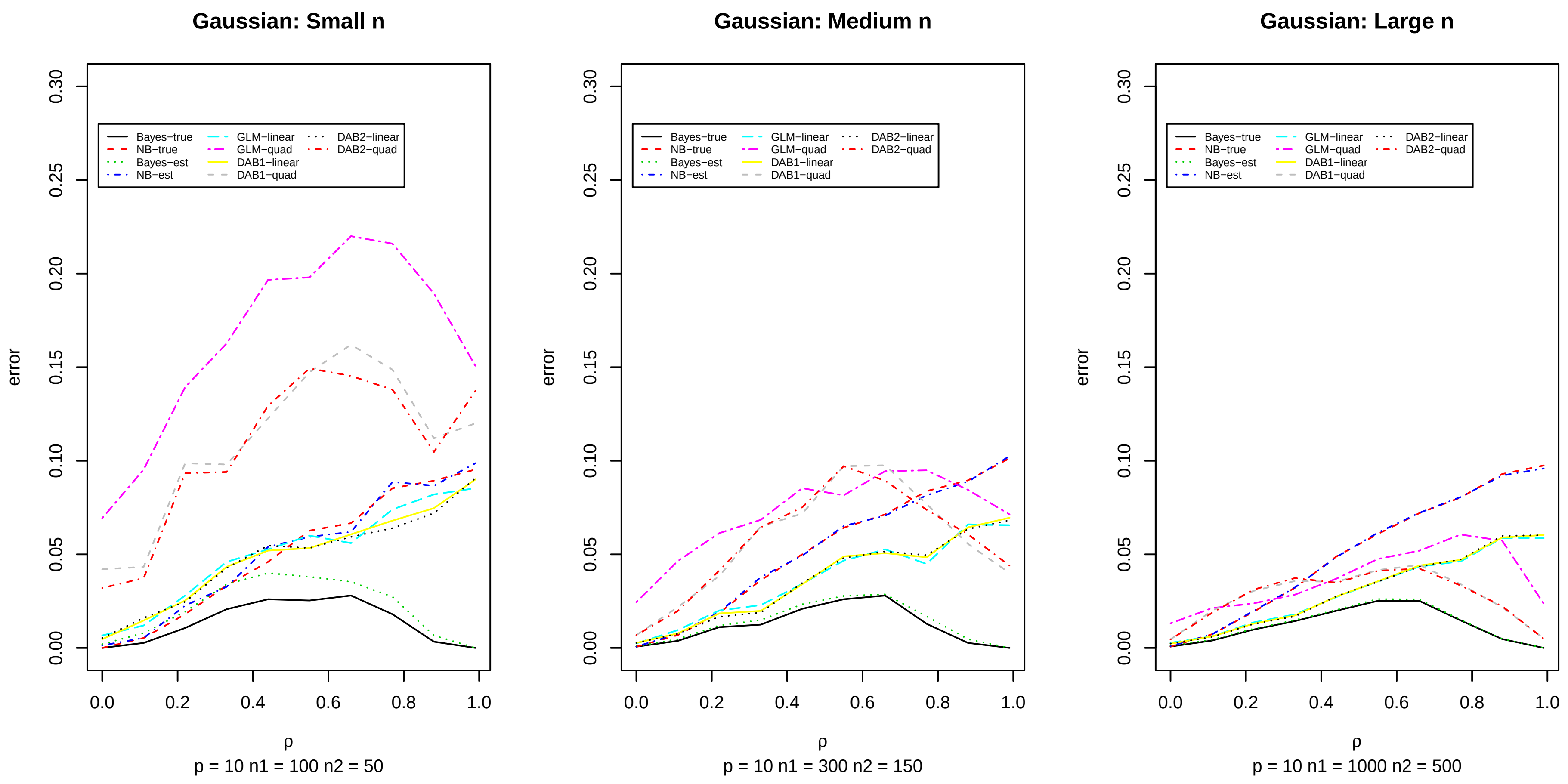

Figure 3.3: Simulation results from multivariate Gaussian data. The y-axis represents average misclassification error over 10 replications. The x-axis represents the ρ parameter between the two classes. $p = 10$ in all plots. The plots are ordered in increasing level of sample size.

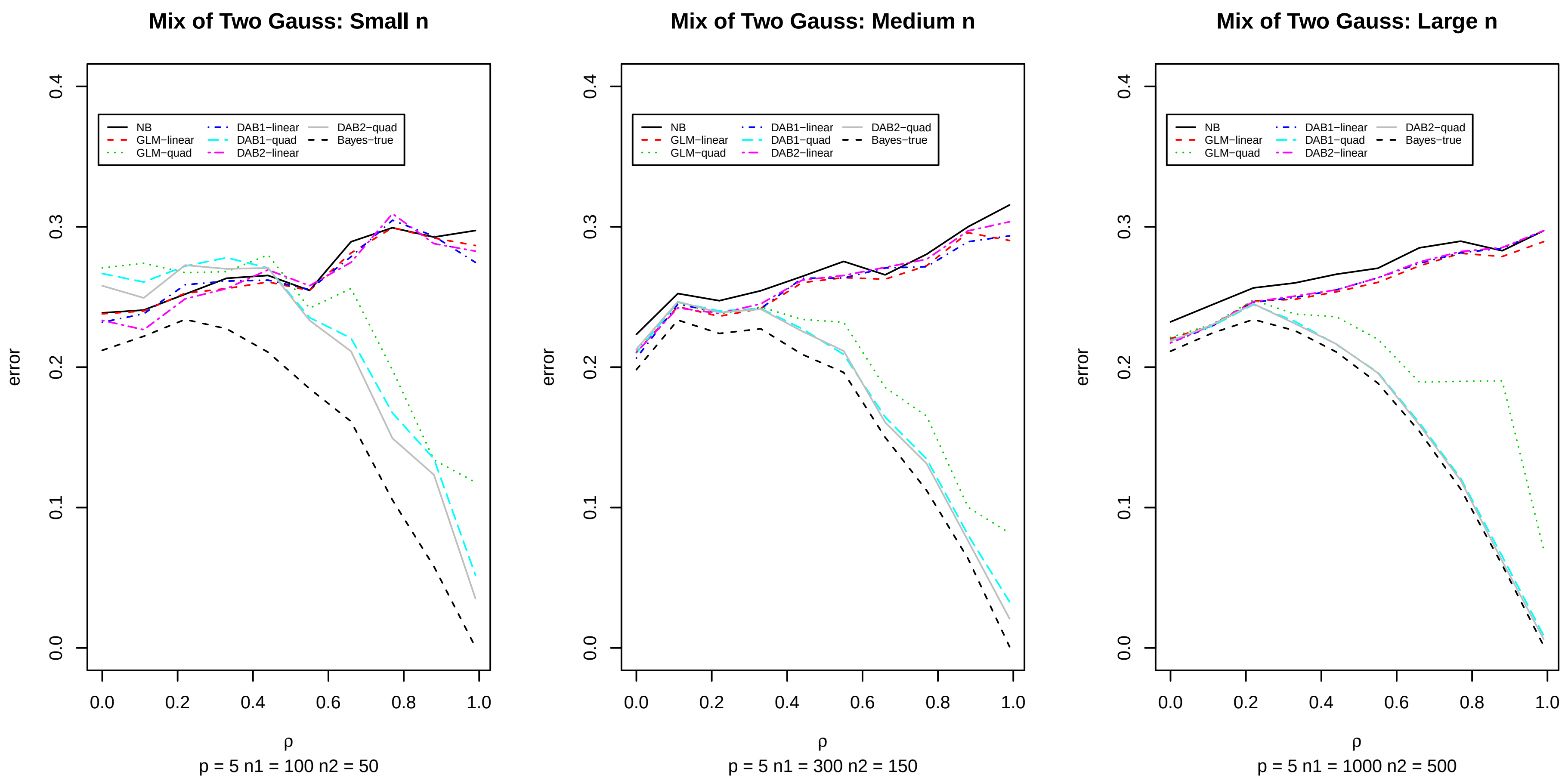

Figure 3.4: Simulation results from a mixture of Gaussian distributions. The y-axis represents average misclassification error over 10 replications. The x-axis represents the ρ parameter between the two classes. $p = 5$ in all plots. The plots are ordered in increasing level of sample size.

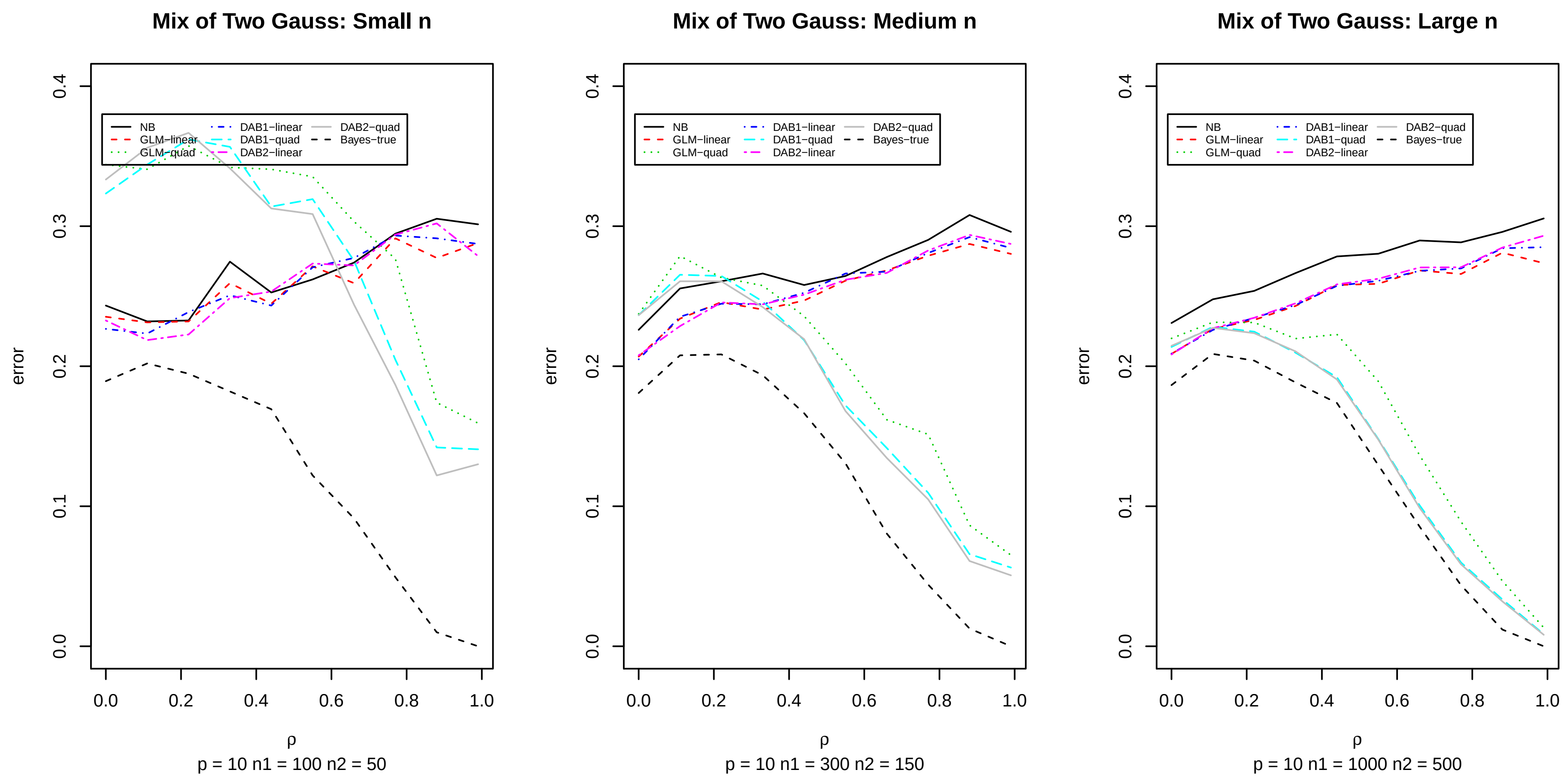

Figure 3.5: Simulation results from a mixture of Gaussian distributions. The y-axis represents average misclassification error over 10 replications. The x-axis represents the ρ parameter between the two classes. $p = 10$ in all plots. The plots are ordered in increasing level of sample size.

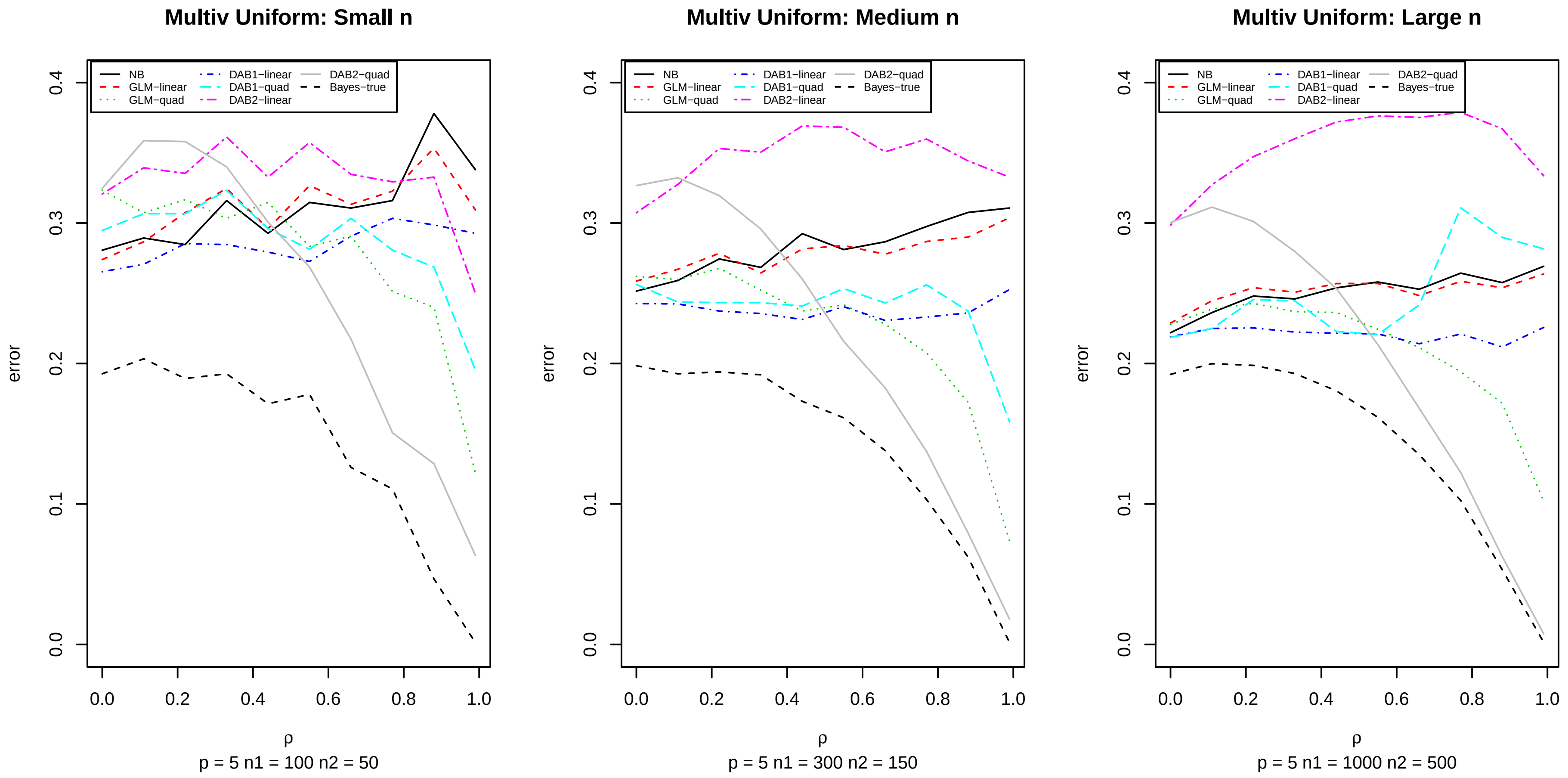

Figure 3.6: Simulation results from a multivariate uniform. The y-axis represents average misclassification error over 10 replications. The x-axis represents the ρ parameter between the two classes. $p = 5$ in all plots. The plots are ordered in increasing level of sample size.

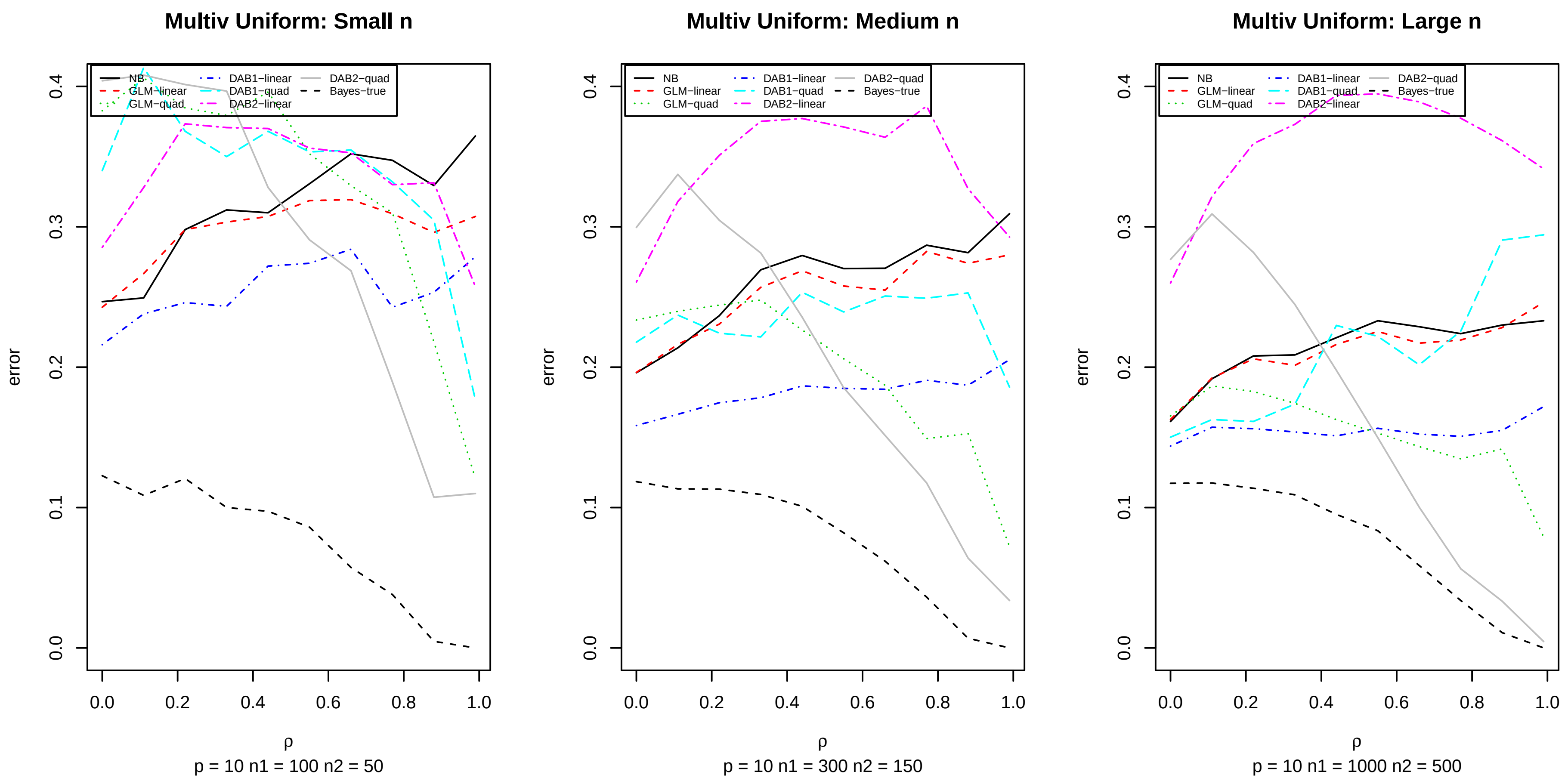

Figure 3.7: Simulation results from a multivariate uniform. The y-axis represents average misclassification error over 10 replications. The x-axis represents the ρ parameter between the two classes. $p = 10$ in all plots. The plots are ordered in increasing level of sample size.

3.4 Real data examples

We demonstrate the performance of this classifier on seven real datasets from the UCI machine learning repository [59]. All of these datasets besides the abalone dataset were used in the aforementioned paper by Ng and Jordan [2] to compare naive Bayes with logistic regression. We use many of the same datasets with continuous variables to recreate their comparisons while adding the linear DAB classifier using the spline method of density ratio estimation. As done by Ng and Jordan [2], we evaluate the performance of these classifiers at different training sizes n. At each training size, we conduct 250 replications and report the average misclassification rate on out of sample data. These are plotted in the odd-numbered figures in Figure 3.8 – Figure 3.23 below. The general shapes of the plots below match those of the same figures in [2], but there are minor differences. Although we attempted to choose the same variables as done by Ng and Jordan, it is possible that we may have omitted or included variables which they did not omit or include since their description only states that non-continuous variables were excluded from the analysis. It is also likely that their implementations of logistic regression and naive Bayes differ from those used here. All logistic regressions plotted below were fit using the standard `glm` function in R. The naive Bayes classifiers were built using the `naive_bayes` function from the `naivebayes` package in R [60].

For each dataset below, we have paired the misclassification rates below with correlation heatmaps of the features in the same dataset by each class using the `ggcorrplot` package in R [61]. These comprise the even-numbered figures below. We have also plotted the difference between the class correlation structures in the third panel. This is done to show how the difference in correlation structures relates to DAB's performance. In general, DAB appears to perform better than naive Bayes and logistic regression when there are large differences in the correlation structures in each class, as shown in the ionosphere and sonar datasets. If the correlation structure between the two classes is more similar, as in the liver and Boston housing datasets, then it is harder for DAB to outperform logistic regression. However, DAB can still outperform naive Bayes under all scenarios as the sample size grows large.

DAB clearly needs a large sample size to perform as well as logistic regression. This is largely because the spline method, which is used to estimate the log density ratios, requires a sufficient sample size. In the absence of a sufficient sample, one may prefer to use logistic regression or naive Bayes as classification models.

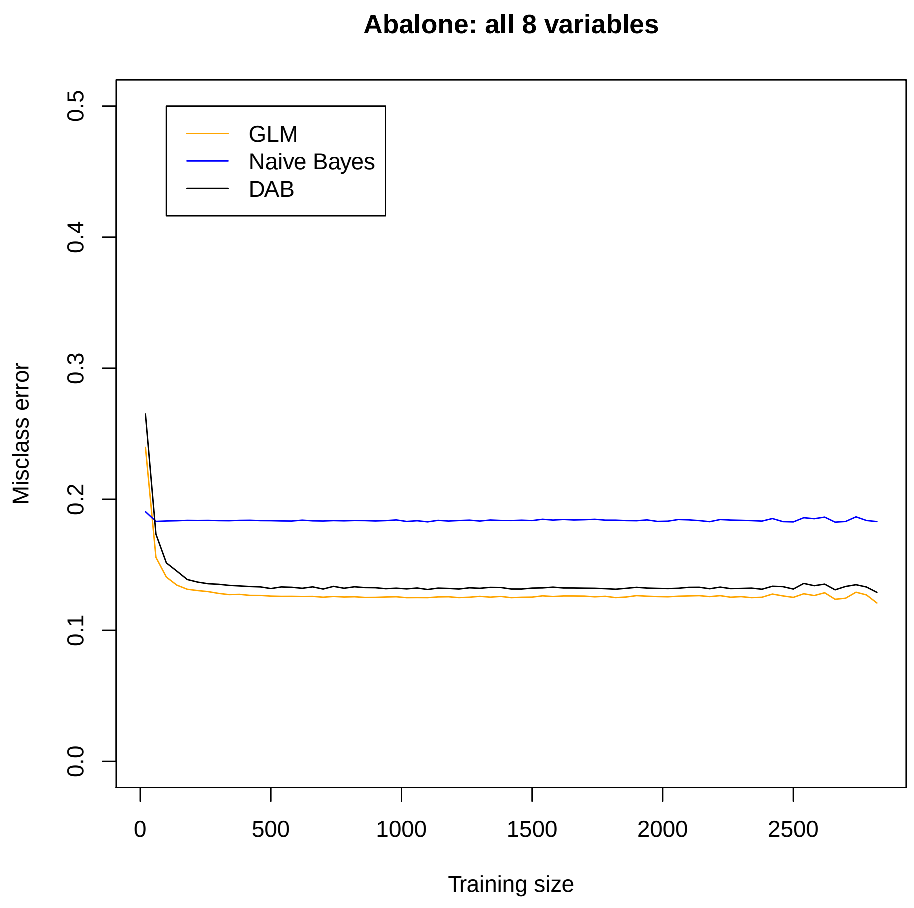

Figure 3.8: Average misclassification rate over 250 replications at different training sizes for the abalone dataset. Observations corresponding to the middle two quartiles of the response variable were removed from the dataset, so classification took place on data that were above the 3rd quartile (Class 1) or below the 1st quartile (Class 0).

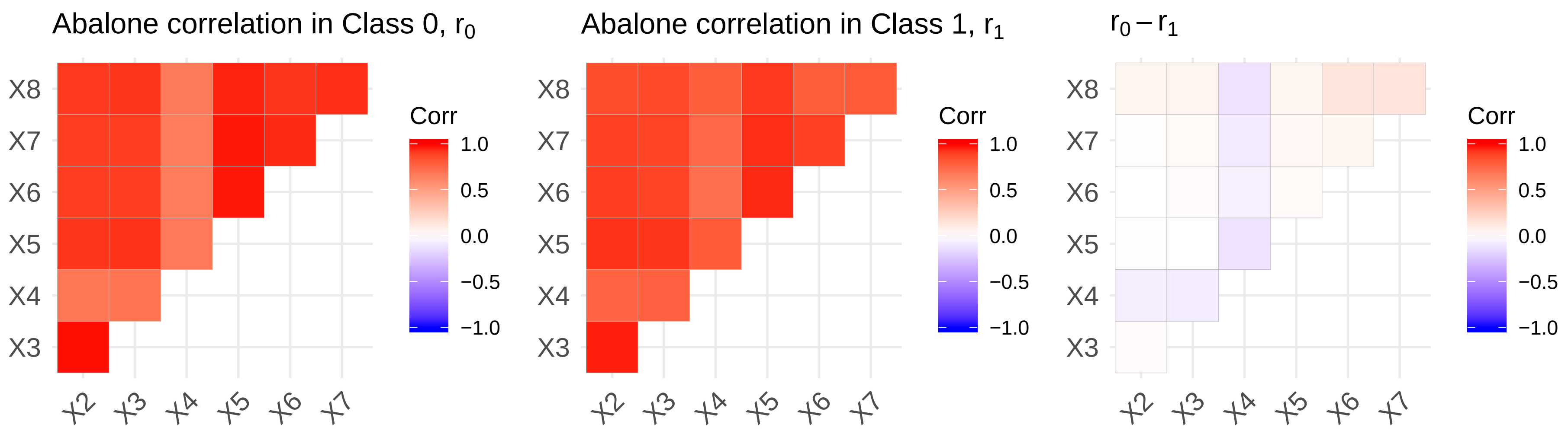

Figure 3.9: Correlation structure for Class 0 and Class 1 in the abalone dataset. We only present the upper half of the correlation matrix since the correlation matrix is symmetric. The rightmost image represents the differences in correlation between Class 0 and Class 1. In this image there are few differences in terms of the correlation between the classes.
Abalone is a type of shellfish. The classification task at hand is to predict a dichotomized age after removing the middle 50% of the data. The variables used refer to physical characteristics of the abalone, including its length, height, and various weights. More information can be found in [62].

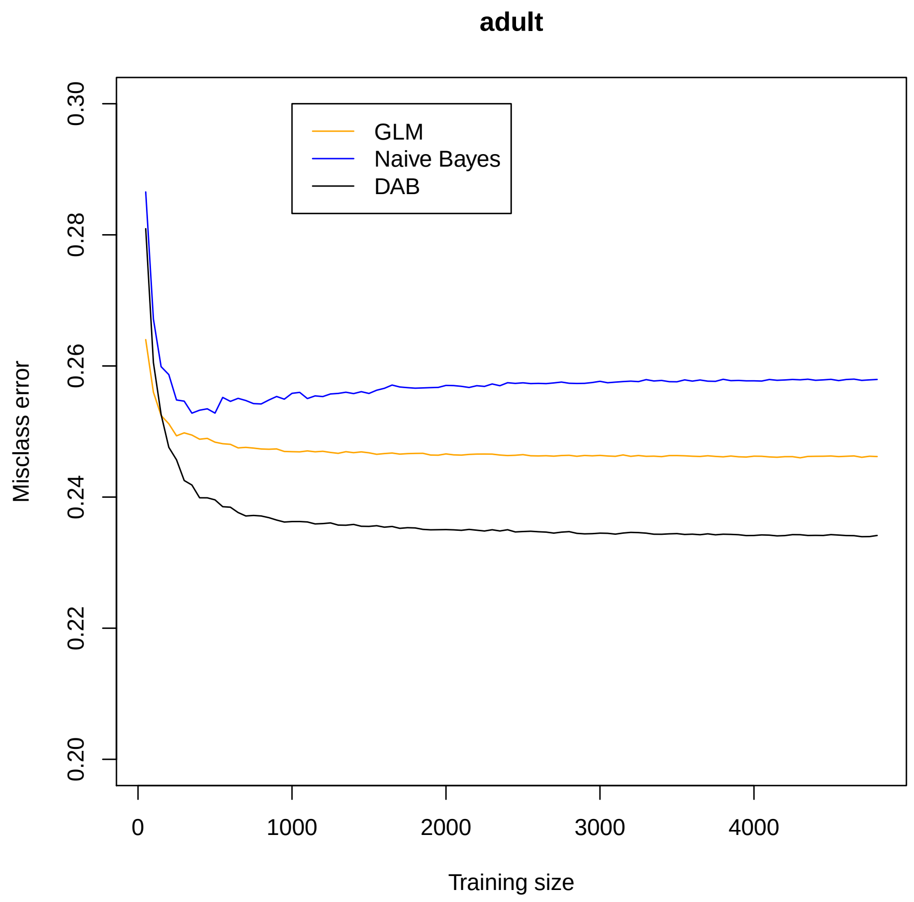

Figure 3.10: Average misclassification rate over 250 replications at different training sizes for the adult dataset. Non-continuous variables were removed from the classification.

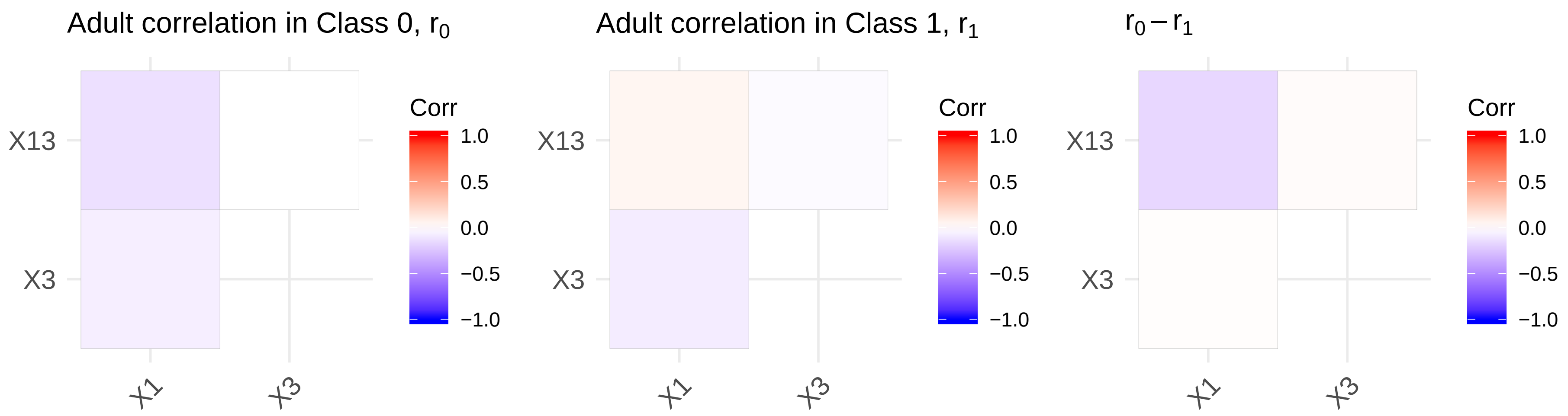

Figure 3.11: Correlation structure for Class 0 and Class 1 in the adult dataset. We only present the upper half of the correlation matrix since the correlation matrix is symmetric. The rightmost image represents the differences in correlation between Class 0 and Class 1. The only notable difference comes in the correlation of X1 and X13.
The response variable here in the adult dataset is whether a person makes $\geq$ 50K/year using data from the 1994 census. The covariates are age, the number of people represented by that entry in the database, and hours worked per week. More information can be found in [63].

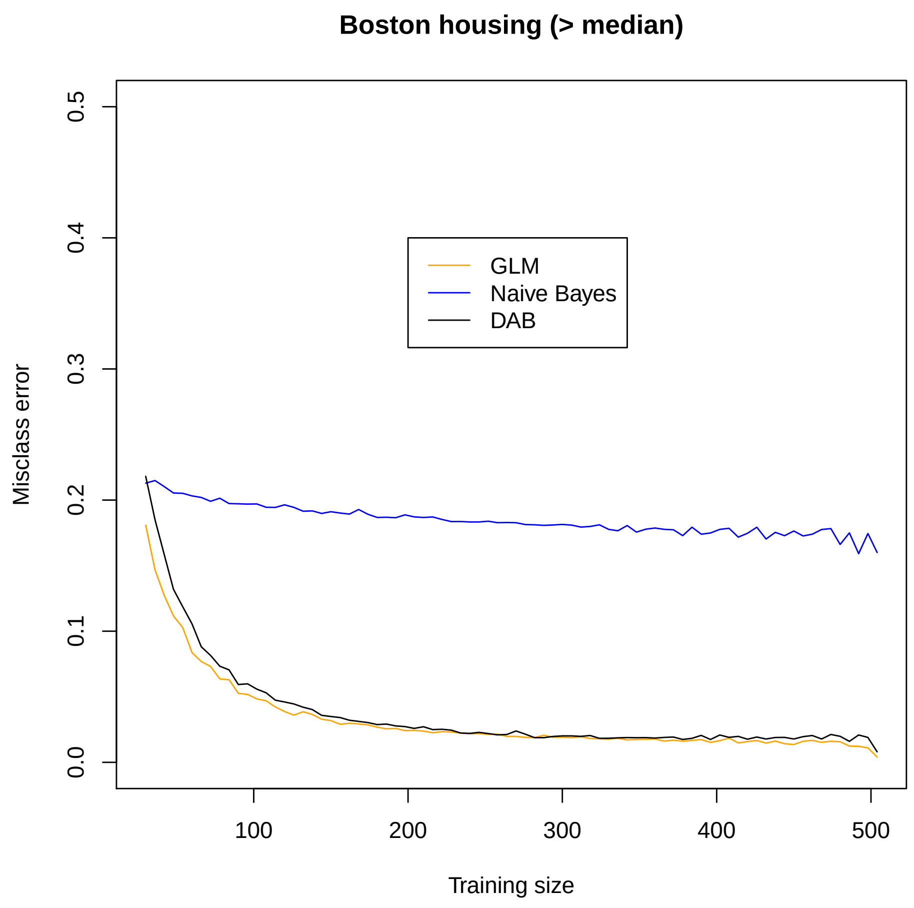

Figure 3.12: Average misclassification rate over 250 replications at different training sizes for the Boston housing dataset. The response variable was whether the median value of the home exceeded the median value of all homes. Non-continuous variables were removed from the classification.

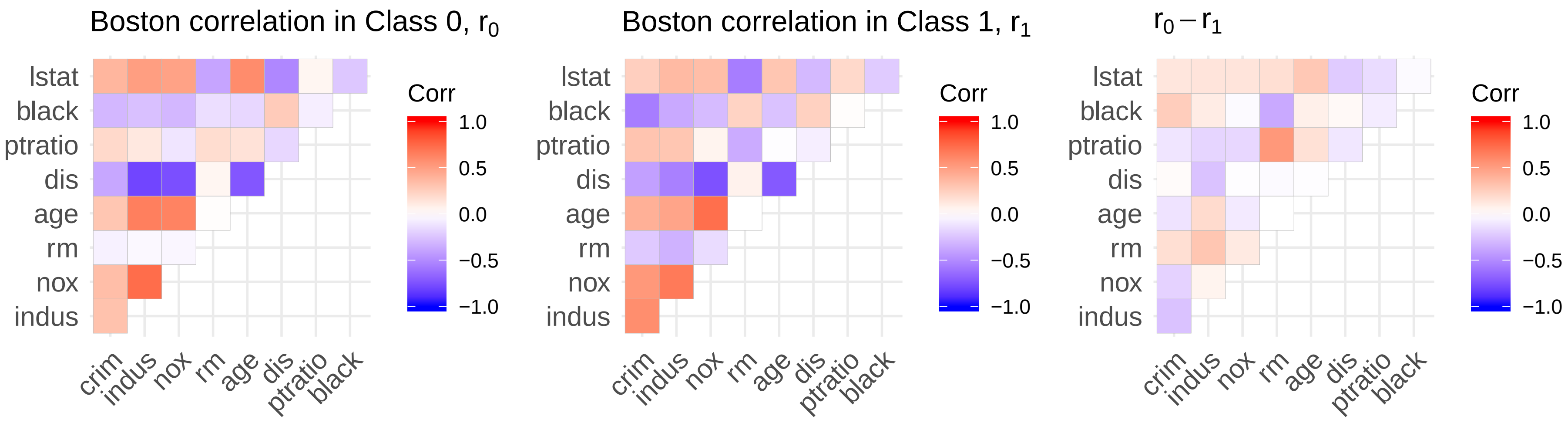

Figure 3.13: Correlation structure for Class 0 and Class 1 in the Boston housing dataset. We only present the upper half of the correlation matrix since the correlation matrix is symmetric. The rightmost image represents the differences in correlation between Class 0 and Class 1. The largest differences are in the correlation of rm with ptratio and rm with black. More information can be found in [64].

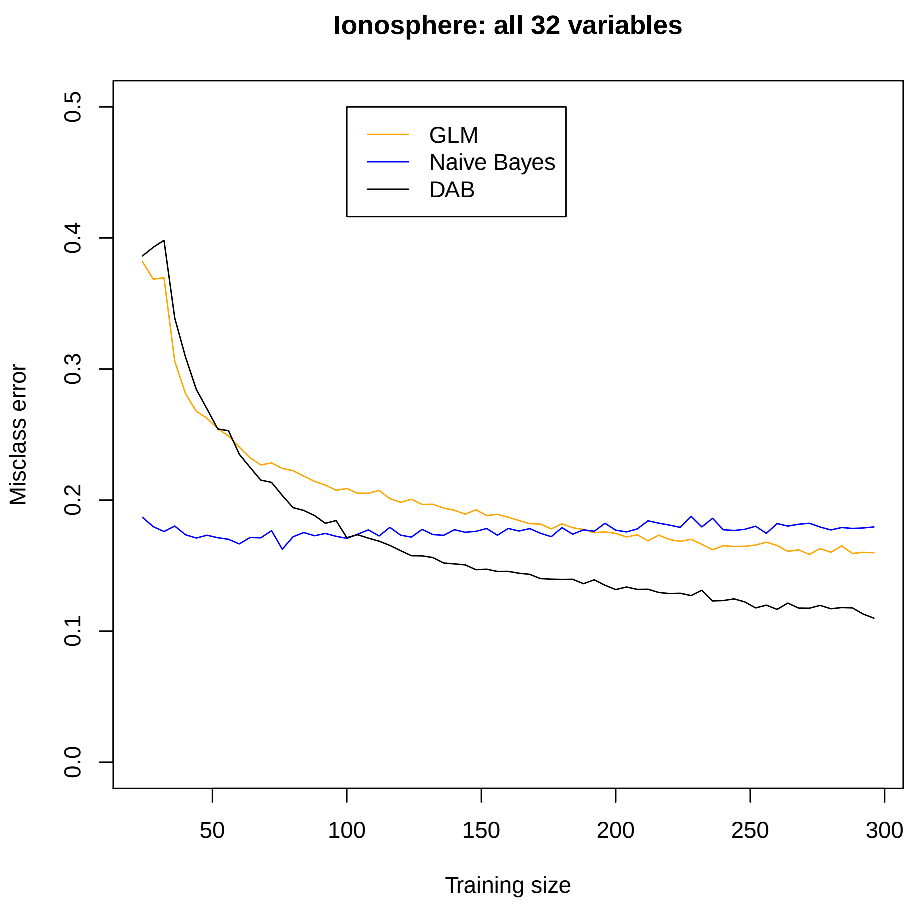

Figure 3.14: Average misclassification rate over 250 replications at different training sizes for the Ionosphere dataset. Non-continuous variables were removed from the classification.

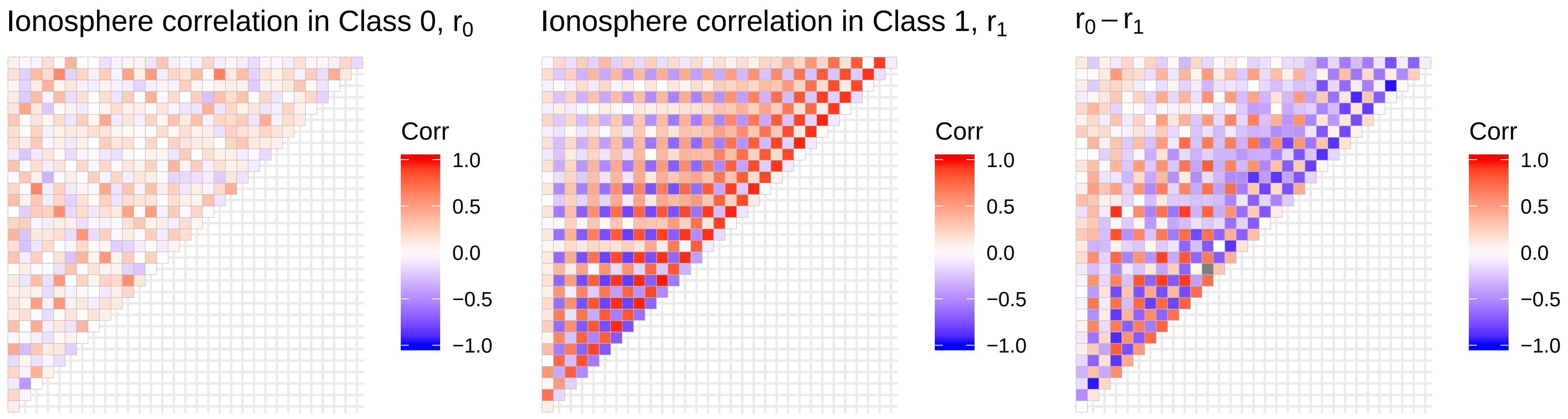

Figure 3.15: Correlation structure for Class 0 and Class 1 in the ionosphere dataset. We only present the upper half of the correlation matrix since the correlation matrix is symmetric. The rightmost image represents the differences in correlation between Class 0 and Class 1. There are many large differences in the correlation structure between the two classes.
The response variable in this dataset is whether the radar return showed evidence of structure in the ionosphere, or not. More information can be found in [65].

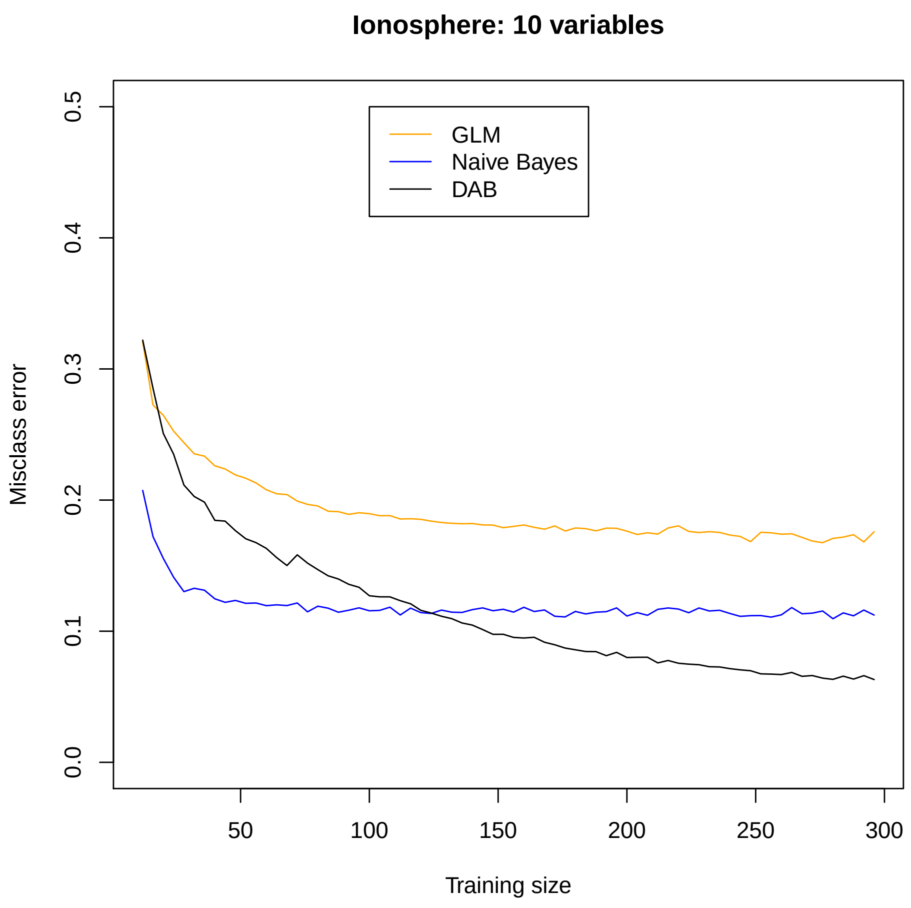

Figure 3.16: Average misclassification rate over 250 replications at different training sizes for the Ionosphere dataset. In this example we arbitrarily chose ten variables for the classification. Note that all methods achieve lower misclassification rates on a smaller dataset and DAB still does best as the training size increases. Non-continuous variables were removed from the procedure.

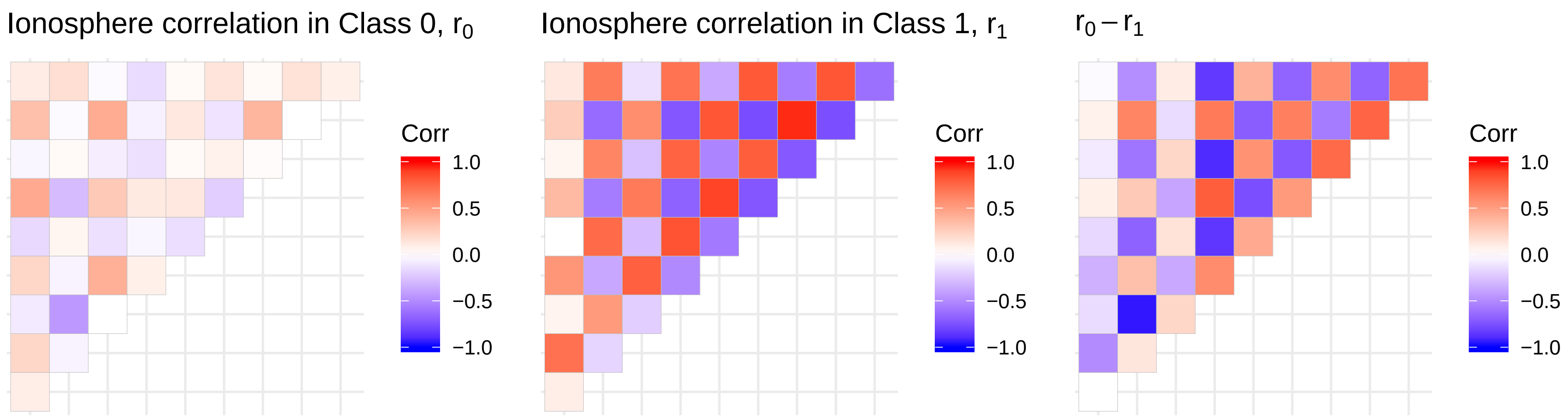

Figure 3.17: Correlation structure for Class 0 and Class 1 in the ionosphere dataset using only the first ten variables. We only present the upper half of the correlation matrix since the correlation matrix is symmetric. The rightmost image represents the differences in correlation between Class 0 and Class 1. There are many large differences in the correlation structure between the two classes.

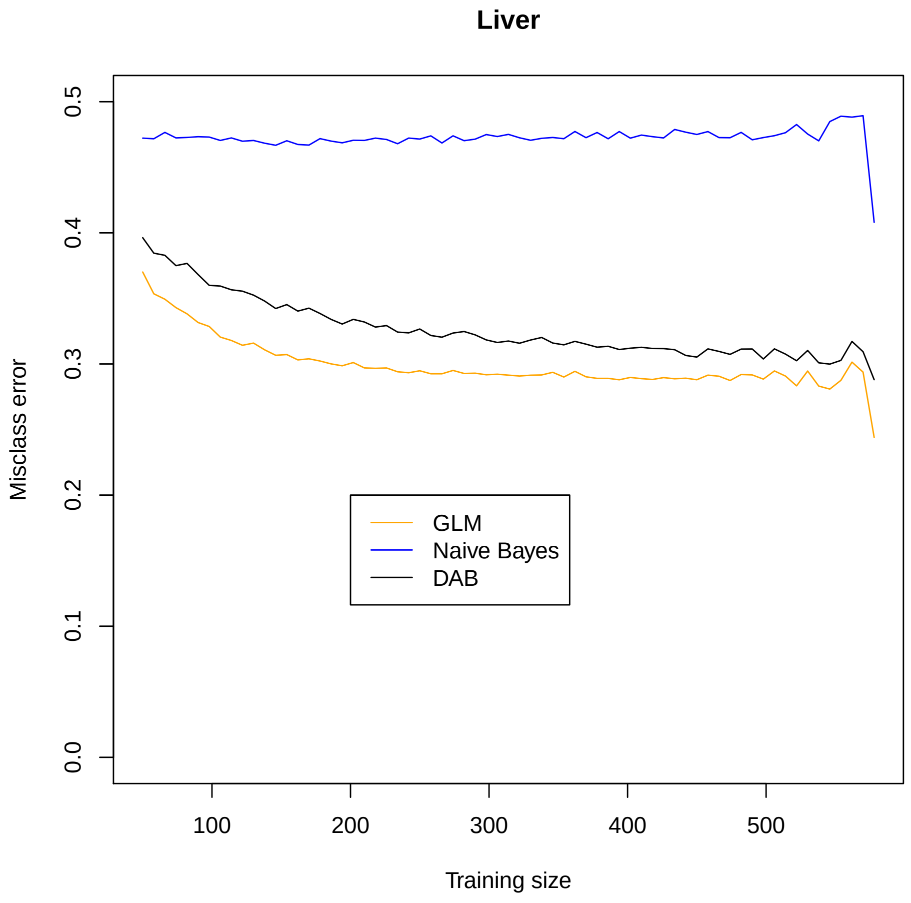

Figure 3.18: Average misclassification rate over 250 replications at different training sizes for the liver dataset. Non-continuous variables were removed from the classification.

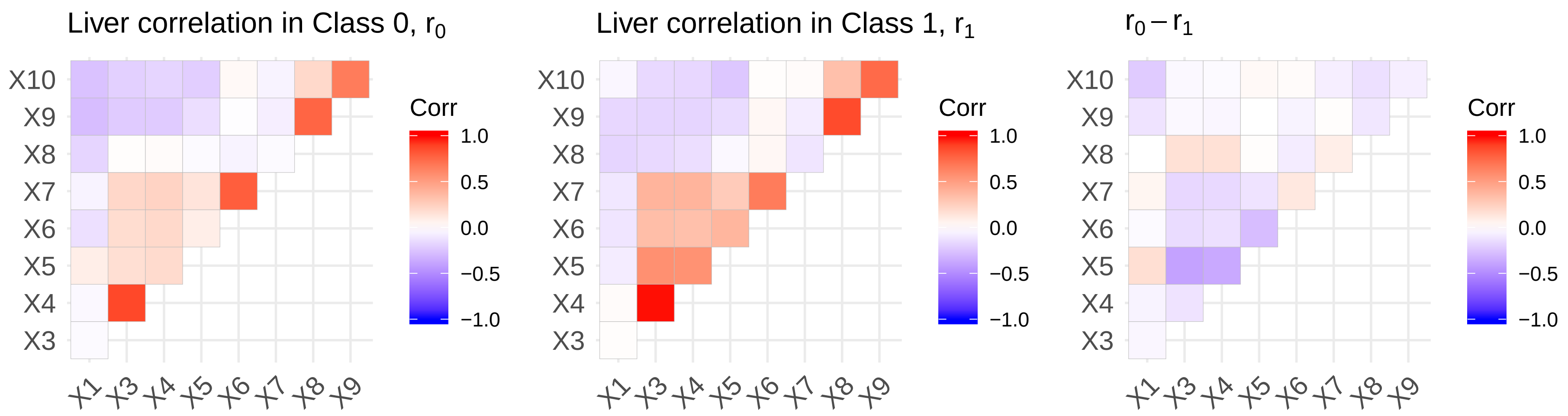

Figure 3.19: Correlation structure for Class 0 and Class 1 in the liver dataset. We only present the upper half of the correlation matrix since the correlation matrix is symmetric. The rightmost image represents the differences in correlation between Class 0 and Class 1. There are small differences in the correlation structure between the two classes.
This dataset, the Indian Liver Patient Dataset, contains measurements from 416 liver patients and 167 non-liver patients. The covariates presented above consist of age, bilirubin measurements, protein measurements, albumin, etc. More information can be found in [66].

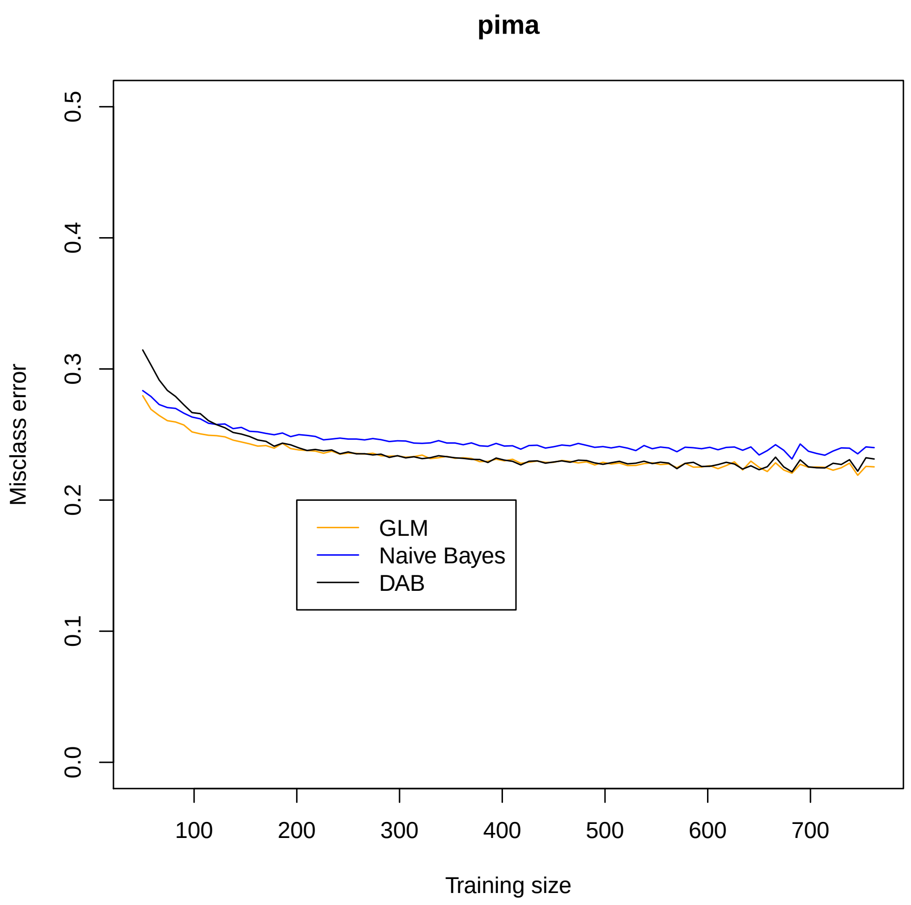

Figure 3.20: Average misclassification rate over 250 replications at different training sizes for the Pima dataset. Non-continuous variables were removed from the classification.

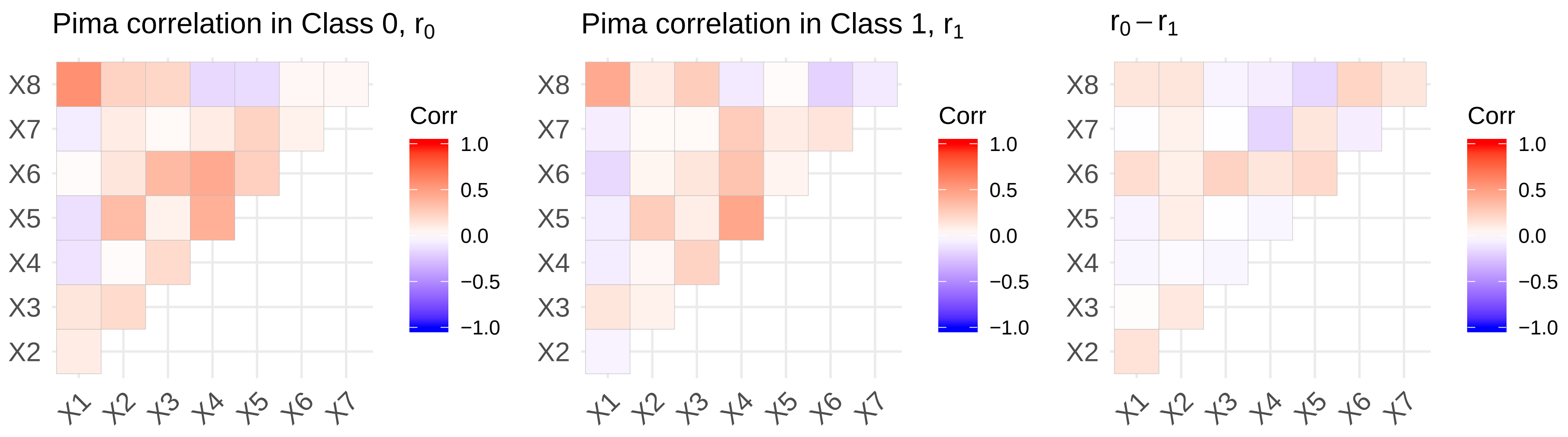

Figure 3.21: Correlation structure for Class 0 and Class 1 in the Pima dataset. We only present the upper half of the correlation matrix since the correlation matrix is symmetric. The rightmost image represents the differences in correlation between Class 0 and Class 1. There are minimal differences in the correlation structures between the two classes.
The response variable in the Pima dataset is whether a patient has diabetes. The covariates include age, BMI, blood pressure, and other medical measurements. More information can be found in [67].

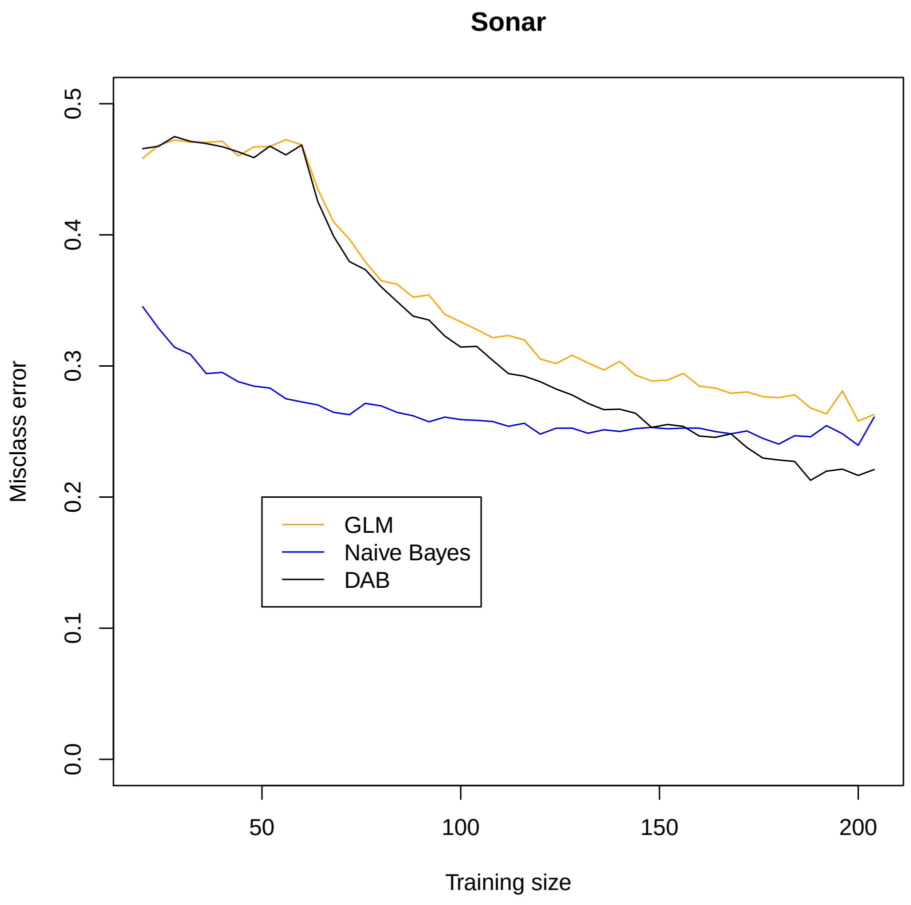

Figure 3.22: Average misclassification rate over 250 replications at different training sizes for the Sonar dataset. Non-continuous variables were removed from the classification.

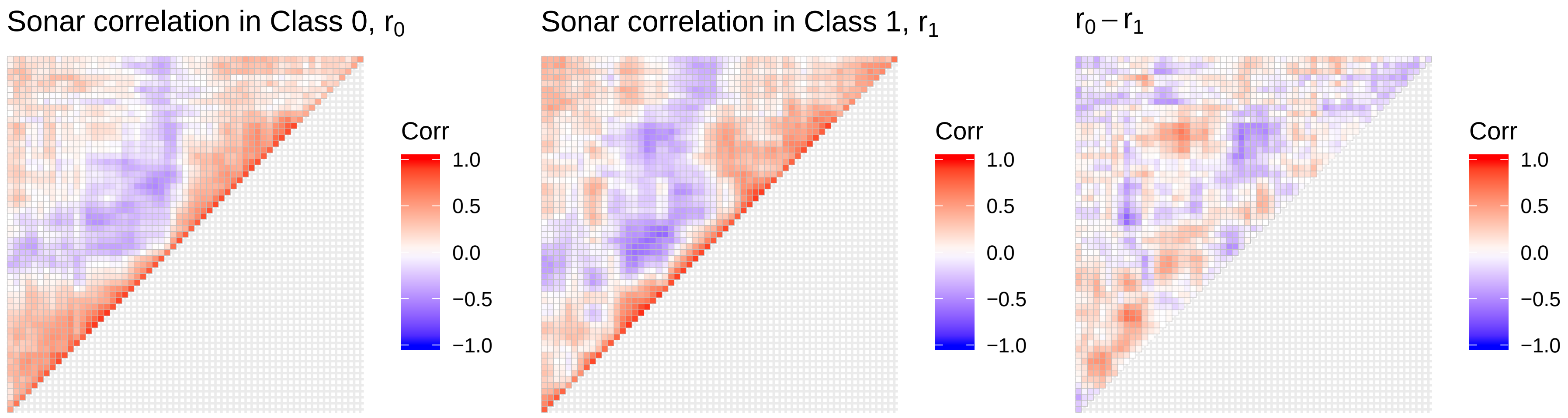

Figure 3.23: Correlation structure for Class 0 and Class 1 in the sonar dataset. We only present the upper half of the correlation matrix since the correlation matrix is symmetric. The rightmost image represents the differences in correlation between Class 0 and Class 1. There are clear differences in the correlation structure between the two classes.
The response variable in the sonar dataset is whether the object being studied is a rock or a mine. The covariates include measurements taken at various angles by bouncing sonar off a given object. More information can be found in [68].

Chapter 4

DAB for Functional Data

4.1 The fDAB method

The formulation of a naive Bayes classifier for functional data was presented in (1.14). Following the derivation in Section 3.1, we can similarly extend the dependence-adjusted Bayes model to functional data. Here we assume that our observed data arise from a common distribution (X, Y) where X is an observed square-integrable random function in $L^2(T)$, T is a compact interval, and $Y \in \{0, 1\}$ is a group label. Let $X_{(y)}$ be a random function which shares the same distribution as X if X arises from population Π_y $(y = 0, 1)$ and let $\pi_y = P(Y = y)$ be the prior probability than observation belongs to Π_y. Recall that to build the functional naive Bayes classifier, we project our functional data onto an orthogonal basis $\{\psi_j\}_{j=1}^{\infty}$ to get projection scores. The basis can be chosen a number of ways, including through a mixture of conditional covariances, as done in [3], or using the partial least squares decomposition [29]. If the covariance of the two classes share a common eigenbasis, and this is chosen as the orthogonal basis for projection, this creates projection scores that are uncorrelated. We compute the projections $\{x_j\}_{j=1}^{\infty}$ using $x_j = \int_T x(t)\psi_j(t)dt$. Therefore, in this chapter, $\mathbf{x}$ denotes a vector of projections of a functional random variable onto an orthogonal basis. We define $f_{yj}(x_j)$ as the density of the jth projection score x_j under population Π_y.

Consider an updated form of $Q_J(x)$, the functional Bayes classifier:

$$Q_J(x) = \alpha_0 + \sum_{j=1}^{J} \alpha_j \log\Big(\frac{f_{1j}(x_j)}{f_{0j}(x_j)}\Big). \tag{4.1}$$

In (4.1), we have updated (1.14) with α_0, a constant that does not depend on x, and an α_j coefficient that equals 1 for all j if the scores are independent, as the naive Bayes framework assumes. However, if there are dependencies among the projection scores x_j, we can adjust for those dependencies by estimating α_j. To estimate α_j, we once again frame this classifier as a generalized linear model (GLM), where we introduce the logit link to obtain a standard GLM form:

$$\operatorname{logit} p(\mathbf{x}) = \alpha_0 + \sum_{j=1}^{J} \alpha_j \log\Big(\frac{f_{1j}(x_j)}{f_{0j}(x_j)}\Big). \tag{4.2}$$

Eq. (4.2) can be seen as a more specific case of (3.2), the general formulation of the dependence adjusted naive Bayes classifier. In (4.2), the log ratio of the projection scores forms our log ratio of two marginal densities, defined as z_k in (3.2). Once again, α_0 includes the log ratio of the prior probabilities and any constant terms which may arise from taking the log ratio of the marginal densities of the projection scores. We name the classification method presented in (4.1) fDAB, or the functional dependence-adjusted naive Bayes classifier.

4.2 Simulation studies

We present two tables of classification results using simulated functional data. In the first table, we focus on functional data that are sample paths of Gaussian processes. In the second table, we generate functional data that do not contain any Gaussian components and demonstrate how existing methods do not perform as well in these circumstances.

The columns of Tables 4.1 and 4.2 below contain results from the different classifiers that were used. The first two columns contain results using the implementation and methods of the t and Gaussian copulas described in [16]. The Naive Bayes column contains misclassification rates for the Naive Bayes classifier as described in [3] and as implemented by Huang and Ruppert [16]. The last five columns display results from fDAB and its variants. Since fDAB is a GLM with log density ratio features, we can create alternative models by adding higher order terms to the model and including regularization in the model estimation. The L_1 and L_2 columns include results from using the `cv.glmnet` function from the `glmnet` package in R to select the appropriate tuning parameter and fit the lasso and ridge regression models, respectively [69]. The Quadratic model contained all possible main effects and quadratic terms but did not include interactions. The Interaction model contained all possible main effects and interaction terms but did not include quadratic terms. As shown in the results, the Interaction model generally performed poorly on all test data, so we do not expect additional benefit from fitting all interaction and quadratic terms.

For every simulation setting, we generate 200 functions from each class using the setups specified in the following sections. To identify the optimal number of components J, we let J vary from 1 to the maximum number of bases used in the generation of the functions. For most settings, this was five, but we let J reach eight in the last simulation setting. We performed 10-fold cross-validation at every level of J to identify the J that gives the minimal misclassification rate. Once the optimal J was chosen, we performed 10-fold cross-validation on the same data using a new train/test split. This was done to provide a better estimate of the error rate at the chosen J.

The estimation of the density ratios follows the procedure described in [3], where the training data are centered and pooled to obtain a joint covariance estimate. We write $G_y(s,t) = \text{Cov}\{X_{(y)}(s), X_{(y)}(t)\} = \sum_{j=1}^{\infty} \lambda_{jy}\phi_j(s)\phi_j(t)$, where ϕ_j are the eigenfunctions assumed to be common between the classes and λ_{jy} represents the jth eigenvalue from class y. Pooling the data allows us to obtain a joint covariance operator $G = \pi_0 G_0 + \pi_1 G_1$, where ϕ_j becomes the jth eigenfunction of G with eigenvalue $\lambda_j = \pi_0\lambda_{j0} + \pi_1\lambda_{j1}$. We use sample estimates of our mean and covariance functions, $\hat{\mu}_y(t)$ and $\hat{G}_y(s,t)$, to build the model. Thus, we create our sample estimate of the covariance as $\hat{G}(s,t) = \pi_0\hat{G}_0(s,t) + \pi_1\hat{G}_1(s,t)$ and its corresponding eigenvalues and eigenfunctions as $(\hat{\lambda}_j, \hat{\phi}_j)$. With these tools, we can estimate principal component scores as $\hat{\xi}_j = \int_T X(t)\hat{\phi}_j(t)dt, (j = 1, \ldots, J)$. Our projection scores for a given $X_i^{(y)}$ are written as $\hat{\xi}_{ijy}$ under the assumption of noise-free predictor trajectories.

4.2.1 Simulations of Gaussian functional data

Table 4.1 presents results from $N = 200$ runs of two different Gaussian or Gaussian-based simulation scenarios. We enumerate the settings below where each item corresponds to a row of the table in the simulation. In each setting, t is a vector of 200 equally spaced points from 0 to 1 unless otherwise specified.

1. **Gaussian: same covariance between the classes, different means.**
   All functions were generated using a basis of $\boldsymbol{\phi(t)} = [\cos(2\pi t), \sin(2\pi t), \cos(4\pi t), \sin(4\pi t), 1]$ and an expansion of $X_k(t) = \sum_{j=1}^{5} a_{yj}\phi_j(t)$ where y denotes class membership. In Class 0, the coefficients $\boldsymbol{a}_{0j} \overset{iid}{\sim} N(\boldsymbol{\mu}_0, \Sigma)$ where $\boldsymbol{\mu}_0 = (-.2, .2, .4, .6, .8)$. The co-

variance matrix Σ contained diagonal elements of $(.6, .7, .8, .9, 1)$ and off-diagonal elements of $\rho = 0.3$. In Class 1, the coefficients $\boldsymbol{a}_{1j} \overset{iid}{\sim} N(\boldsymbol{\mu}_1, \Sigma)$ where $\boldsymbol{\mu}_1 = (-1, 1, -.5, .5, .1)$.

2. **Gaussian: same means between the classes, different covariances.**
   All functions were generated using a basis of $\boldsymbol{\phi(t)} = [\cos(2\pi t), \sin(2\pi t), \cos(4\pi t), \sin(4\pi t), 1]$ and an expansion of $X_k(t) = \sum_{j=1}^{5} a_{yj}\phi_j(t)$ where y denotes class membership. In Class 0, the coefficients $\boldsymbol{a}_{0j} \overset{iid}{\sim} N(\mathbf{0}, \Sigma_0)$ where Σ_0 is the identity matrix. In Class 1, the coefficients $\boldsymbol{a}_{1j} \overset{iid}{\sim} MVN(\mathbf{0}, \Sigma_1)$ where Σ_1 contains 1s on the diagonals and $\rho = .8$ on the off-diagonals.

The results in these settings confirm the expected behavior of the classifiers under Gaussian situations. The Naive Bayes classifier does best when the only difference in the two classes comes in the mean of the Gaussian distribution, since it recovers the optimal Bayes solution. When the difference comes in the covariance, the copulas outperform the other methods since they explicitly model the dependence.

4.2.2 Simulations of functional data from other distributions

We enumerate the settings below where we generated functional data that do not have any Gaussian components.

1. **Rows 1-3 in Table 4.2: Multivariate uniform coefficients.**
   All functions were generated using a basis of $\boldsymbol{\phi(t)} = [\cos(\pi t), \sin(\pi t), \cos(2\pi t), \sin(2\pi t), 1]$ and an expansion of $X_k(t) = \sum_{j=1}^{5} a_{yj}\phi_j(t)$ where y denotes class membership. In Class 0, the coefficients were drawn independently and identically from a multivariate uniform with identity covariance matrix ($\Sigma_0 = I$) using the `MultiRNG` package in R [54] as described in [55]. In Class 1, the coefficients were drawn independently

and identically from a multivariate uniform where Σ_1 contained 1s as diagonal elements and $\rho > 0$ for the off-diagonal elements. Rows 1, 2, and 3 correspond to $\rho = .96, .8,$ and $.3$ respectively.

2. **Rows 4-6 in Table 4.2: Multivariate Laplace coefficients.**

   All functions were generated using a basis of $\boldsymbol{\phi(t)} = [\cos(\pi t), \sin(\pi t), \cos(2\pi t), \sin(2\pi t), 1]$ and an expansion of $X_k(t) = \sum_{j=1}^{5} a_{yj}\phi_j(t)$ where y denotes class membership. In Class 0, the coefficients were drawn independently and identically from a multivariate Laplace with identity covariance matrix ($\Sigma_0 = I$) using the `MultiRNG` package in R [54], which follows the method of [70]. In Class 1, the coefficients were drawn independently and identically from a multivariate Laplace where $\text{diag}(\Sigma_1) = \mathbf{1}$ and the off-diagonal elements were all $\rho > 0$. Rows 4, 5, and 6 correspond to $\rho = .96, .8,$ and $.3$ respectively. The shape parameter was set to 2 and the mean was set to 1 for all Laplace draws.

   The multivariate Laplace distribution can be written as

$$\begin{aligned} f(x|\boldsymbol{\mu}, \Sigma, \gamma) &= c \exp\Big(-((x-\boldsymbol{\mu})^T \Sigma^{-1}(x-\boldsymbol{\mu}))^{\gamma/2}\Big) \\ c &= \frac{\gamma\Gamma(d/2)}{2\pi^{d/2}\Gamma(d/\gamma)}|\Sigma|^{-1/2} \end{aligned} \tag{4.3}$$

   where Σ is symmetric and positive definite and $\boldsymbol{\mu}, \gamma$ and Σ are the mean vector, the shape parameter, and the covariance matrix.

3. **Row 7 in Table 4.2: Quadratic example.**

   All functions were generated using a basis of $\boldsymbol{\phi(t)} = [\cos(\pi t), \sin(\pi t), \cos(2\pi t), \sin(2\pi t)]$ and an expansion of $X_k(t) = \sum_{j=1}^{4} a_{yj}\phi_j(t)$. In Class 0, $a_{01}, \ldots, a_{04} \overset{iid}{\sim} Unif(0,1)$. In Class 1, a set of coefficients $b_{11}, \ldots, b_{14}$ were drawn from a $Unif(0, 3^{1/3})$. The coefficients used in the expansion a_{1j} depend on b_{1j} by the relationship $a_{1j}|b_{1j} \overset{iid}{\sim}$

$Unif(0, b_{1j}^2)$. This was done to generate coefficients that fall under the curve presented in Fig. 4.1.

4. **Row 8 in Table 4.2: Added basis.**

   In Class 0, functions were generated using a basis of $\boldsymbol{\phi(t)} = [\cos(\pi t), \sin(\pi t), \cos(3\pi t), \sin(3\pi t), \cos(4\pi t), \sin(4\pi t), 1]$ and an expansion of $X_k(t) = \sum_{j=1}^{7} a_{yj}\phi_j(t)$. The coefficients a_{yj} are drawn to be multivariate uniform with covariance matrix Σ_0 which has diagonal elements of $[8, 4, 4, 2, 2, 1, 1]$ and off-diagonal elements of $\rho = 0.5$, similar to a setting presented in [71]. In Class 1, functions were generated using the same basis as listed above with the addition of $\sin^2(\pi t)$ added to the basis vector. The first seven coefficients are multivariate uniform with the same covariance matrix Σ_0. The last function in the basis had coefficient $a_{18} \sim Unif(0, 1)$.

These simulations show the importance of modeling the dependence using the DAB classifier. In the multivariate uniform settings, we see that the DAB model does best when the coefficients are highly correlated; that improvement declines when ρ decreases to 0.3. In the Laplace settings, the copulas generally do best when $\rho = 0.96$ but the separation between the classifiers declines as ρ decreases. Taken together, these results indicate that a t or Gaussian copula can model the dependence when that dependence is strong and the distribution is bell-shaped. However, if the copula does not match the distribution of the coefficients in shape, the DAB may perform better, especially when the coefficients in one of the classes are highly correlated.

The last two rows of Table 4.2 highlight the flexibility of fitting a GLM to the log-density ratios. We show that fitting a GLM using the quadratic terms of the log density ratios can improve classification when one class of the functional data are generated with coefficients that exhibit strong nonlinear dependence. Additionally, we present an interesting case where DAB and its variants strongly outperform both copulas and the

Naive Bayes classifier, merely by adding an extra basis to the coefficient expansion. Thus, the DAB fits have the ability to model differences in the two classes which a copula cannot capture and a Naive Bayes model ignores.

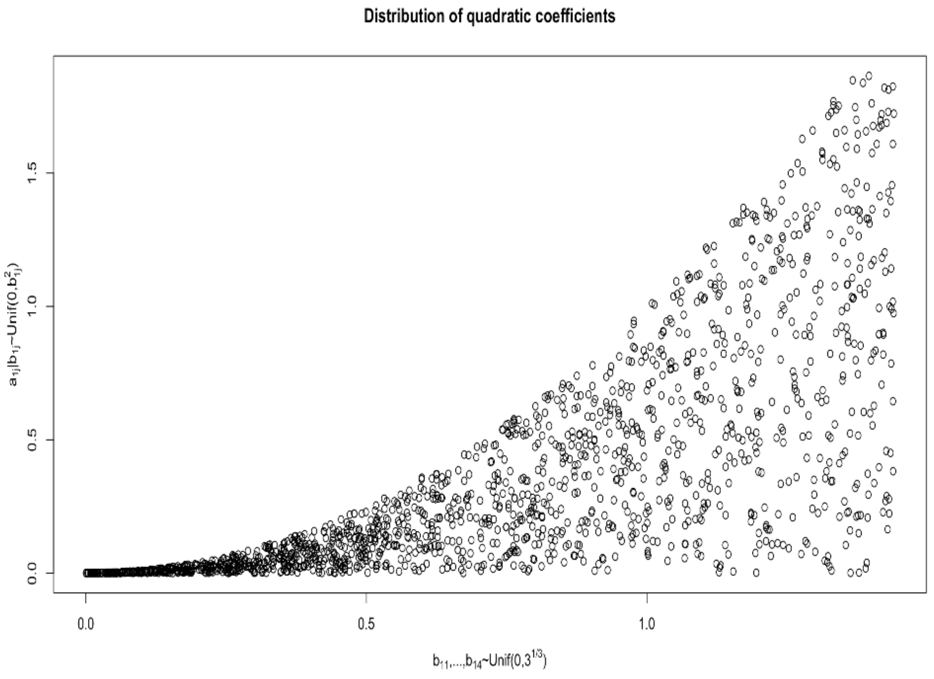

Figure 4.1: Distribution of the coefficients of Class 1 in Row 7 of Table 4.2.

Table 4.1: Simulation results from Gaussian cases. Misclassification rates are reported in percent error and accompanied by standard deviation. The new methods here are fDAB, the DAB classifier applied to functional data, and the variants that follow it as described in the text. We use the ratio of kernel density estimates to estimate the density ratio in these simulations. Rates within .05 of the smallest misclassification rate are highlighted in green.
When the functional data follow sample paths of Gaussian processes and $\Sigma_0 = \Sigma_1$, naive Bayes gives the lowest misclassification error. When $\Sigma_0 \neq \Sigma_1$, the t and Gaussian copulas give the lowest misclassification error since they model the dependence explicitly.

	t Copula	G Copula	Naive Bayes	fDAB	L_1	L_2	Quadratic	Interaction
$\boldsymbol{\mu}_1 \neq \boldsymbol{\mu}_0, \Sigma_1 = \Sigma_0$	13.85 (1.66)	13.83 (1.67)	13.74 (1.77)	13.88 (1.74)	13.80 (1.71)	13.82 (1.71)	14.22 (1.84)	14.44 (1.83)
$\boldsymbol{\mu}_1 = \boldsymbol{\mu}_0, \Sigma_1 \neq \Sigma_0$	13.71 (1.72)	13.66 (1.72)	14.28 (1.73)	14.18 (1.73)	14.59 (1.77)	14.61 (1.79)	14.97 (1.92)	16.62 (2.43)

Table 4.2: Simulation results from non-Gaussian distributions. Misclassification rates are reported in percent error and accompanied by standard deviation. The new methods here are fDAB, the DAB classifier applied to functional data, and the variants that follow it as described in the text. We use the ratio of kernel density estimates to estimate the density ratio in these simulations. Rates within .05 of the smallest misclassification rate are highlighted in green.
In these examples, fDAB performs best in a variety of scenarios: when the data are uniform and ρ is high, in a few Laplace settings, and in the added basis setting. fDAB appears to be more flexible than the t and Gaussian copulas when the functional data are not Gaussian. The Quadratic model also performs well in two settings.

	t Copula	G Copula	Naive Bayes	fDAB	L_1	L_2	Quadratic	Interaction
Unif: .96	3.5 (0.94)	3.5 (0.93)	3.27 (0.9)	2.78 (0.84)	2.98 (0.97)	2.94 (0.92)	6.84 (2.47)	10.04 (3.02)
Unif: .8	15.59 (1.91)	15.59 (1.89)	15.41 (1.97)	14.58 (1.98)	15.12 (1.98)	15.05 (1.94)	14.46 (1.93)	15.3 (1.97)
Unif : .3	39.58 (3.02)	39.67 (3.02)	39.06 (2.92)	39.14 (2.77)	39.48 (2.84)	39.39 (2.86)	39.42 (2.89)	39.51 (2.82)
Laplace: .96	0.71 (0.44)	0.71 (0.44)	0.74 (0.46)	1.31 (0.73)	1.07 (0.57)	1.05 (0.56)	3.85 (2.09)	3.1 (1.81)
Laplace: .8	8.7 (1.37)	8.73 (1.35)	8.54 (1.29)	8.44 (1.4)	8.61 (1.42)	8.6 (1.46)	8.9 (1.35)	10.57 (1.85)
Laplace: .3	39.5 (2.64)	39.48 (2.66)	39.04 (2.6)	39.09 (2.63)	39.62 (2.76)	39.55 (2.71)	39.15 (2.48)	39.61 (2.63)
Quadratic	19.96 (2.18)	19.66 (2.1)	19.02 (2.19)	18.78 (2)	19.36 (2.12)	19.38 (2.18)	18.2 (1.98)	19.17 (2.15)
Added Basis	14.97 (2.02)	15.04 (1.77)	12.08 (1.37)	7.1 (1.54)	7.66 (1.64)	7.68 (1.65)	8.77 (2.59)	15.07 (2.4)

4.3 Real data examples

We present empirical classification results obtained on four real data examples. No smoothing was applied to any of the datasets since all datasets appear smooth when plotted. To evaluate the performance, we used $N = 200$ repetitions of 10-fold cross-validation on each dataset. Each data example was run using both the PCA basis and the PLS basis, as done in [16] and [10].

The first example that we use consists of wine spectra data. This dataset contains 124 samples of wine spectra measured on 256 points, with $n_1 = 78$ with alcohol content greater than 12 and $n_0 = 46$ with alcohol content less than 12. These data were provided online by Professor Marc Meurens of the Université Catholique de Louvain.

The second included dataset is the DTI dataset from the `refund` package in R [72]. The MRI/DTI data in the refund package were collected at Johns Hopkins University and the Kennedy-Krieger Institute. This dataset includes 142 subjects with fractional anisotropy measurements taken at 93 points on the corpus callosum. The outcome variable in this case is a diagnosis of multiple sclerosis (MS), since MS can create lesions in white matter tracts that decrease fractional anisotropy. As done in [16], we selected the first visits of all patients and used only the 141 cases without any missing values. There were $n_1 = 42$ healthy patients and $n_0 = 99$ unhealthy patients.

The third example consists of the phoneme data from the `fds` package in R [73]. This dataset contains five separate phonemes based on digitized speech. As done in [10], we try to classify the two sounds "aa" and "ao." There are 400 functions in each class and each function contains 150 measurements.

The fourth dataset describes particulate matter (PM) emissions from heavy trucks. This dataset was used in [74], extracted from [75], and provided by Wentian Huang and David Ruppert [16]. As done in [16], we dichotomize log PM and classify the $n_1 = 41$

cases with log PM above average as high and the $n_0 = 67$ cases with log PM below average as low. Each function contains 91 measurements.

In each repetition, we first performed 10-fold cross-validation for every option of the number of components, J, to be used on the whole dataset. For wine and DTI, the maximum J was set to 10; for the truck and phoneme datasets, the maximum J was set to 30. The higher limit was used since [16] and [10] note that the differences in these datasets occur in very large expansions of the functional principal component scores. Preliminary examinations of the other datasets indicated that J was not needed beyond 10. All predicted probabilities were thresholded at 0.5.

Once the number of components were selected, 10-fold cross-validation was repeated on the whole dataset using a new split of the data and the selected number of components. We report the average misclassification rates along with their associated standard deviations in percent error.

As seen in the tables below, the DAB classifier performs best on the wine data using the PCA basis. All variants of DAB besides the interaction model outperform the Naive Bayes or copula models on the phoneme data using the PCA basis. The copula and Naive Bayes models perform best on the DTI and truck data, respectively.

The PLS basis tells a different story. Interestingly, the Naive Bayes implementation on the PLS basis outperforms all other methods on the wine data. The DAB variants generally do better than the copula and Bayes methods on the DTI data, with the quadratic model doing best. The Gaussian and t-copulas exhibit the best performance on the phoneme data, though all methods are substantially improved over the PCA basis on this dataset. Last, the copulas outperform all other methods on the truck dataset, validating the results in [16].

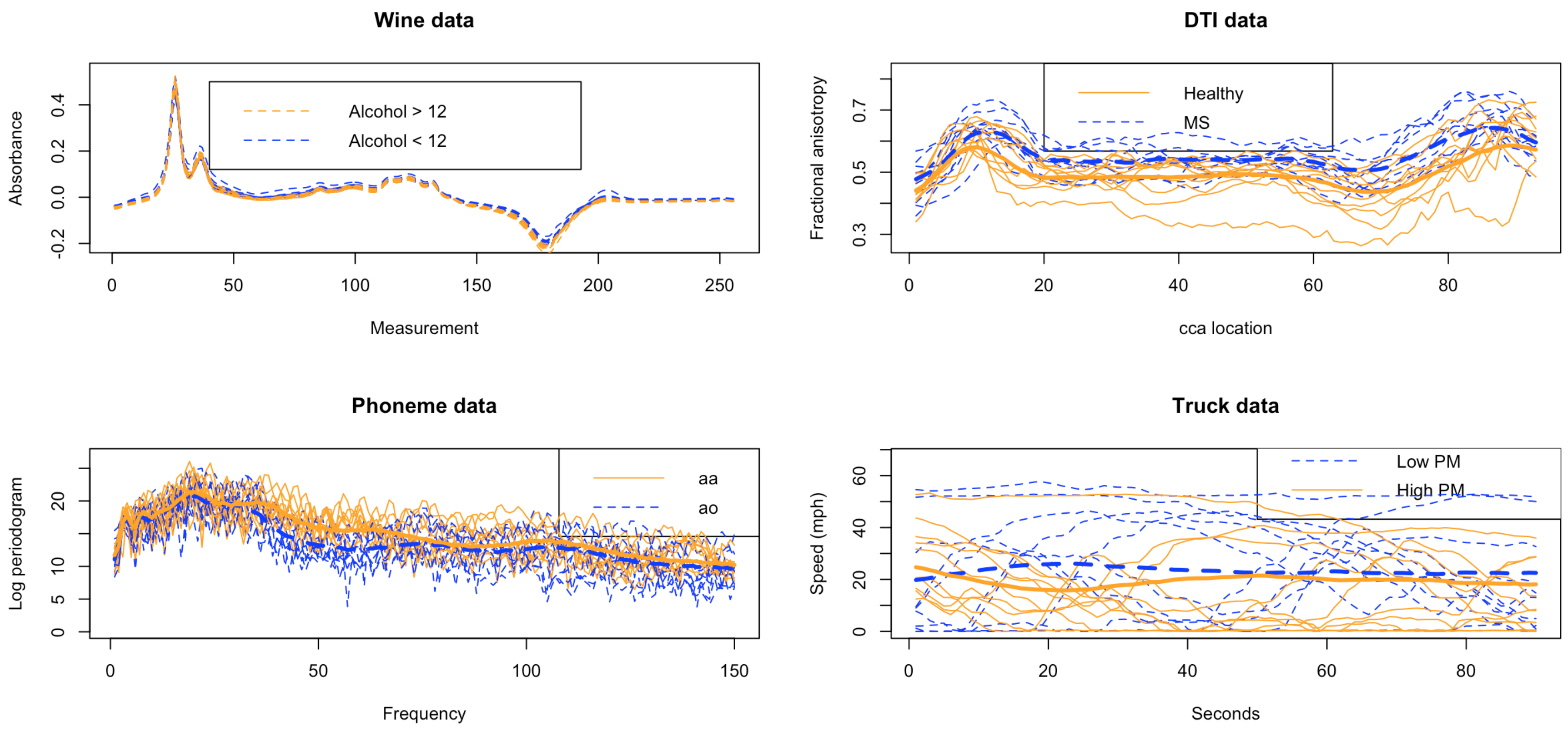

Figure 4.2: Plots of the four functional datasets. Thick lines denote the class average.

Table 4.3: Empirical results using PCA basis and spline density ratio estimation reported as average percent error (standard deviation). The L_2 and Quadratic variants do best on the Wine and DTI datasets, while the Gaussian copula and naive Bayes do best on the Truck and Phoneme datasets.

	t Copula	Gaussian Copula	Naive Bayes	fDAB	L_1	L_2	Quadratic	Interaction
Wine	9.5 (1.62)	8.83 (1.6)	8.77 (1.55)	8.04 (1.55)	7.33 (1.34)	7.15 (1.31)	8.71 (1.45)	8.8 (1.42)
DTI	25.08 (2.07)	25.44 (2.04)	27.06 (1.79)	22.67 (1.85)	23.11 (1.77)	23.17 (1.87)	22.54 (1.76)	23.09 (1.91)
Truck	29.94 (2.84)	29.7 (2.86)	30.28 (3.47)	31.31 (3.59)	31.44 (3.05)	31.18 (3.1)	34.22 (4.25)	35.18 (3.74)
Phoneme	21.09 (0.47)	21.09 (0.45)	19.64 (0.63)	20.61 (0.87)	20.25 (0.74)	20.22 (0.71)	21.2 (0.88)	22.92 (0.6)

Table 4.4: Empirical results using PLS basis and spline density ratio estimation reported as average percent error (standard deviation). The Quadratic model and DAB variants outperform naive Bayes and the copulas on the DTI datasets, but naive Bayes and the Gaussian copula have the lowest misclassification rates on the Wine, Truck, and Phoneme datasets. Switching to the PLS basis improves misclassification results substantially for the copulas and naive Bayes on the Wine, Truck, and Phoneme datasets.

	t Copula	Gaussian Copula	Naive Bayes	fDAB	L_1	L_2	Quadratic	Interaction
Wine	7.18 (1.5)	6.75 (1.42)	6.12 (1.2)	7.63 (1.78)	6.9 (1.18)	6.84 (1.21)	6.77 (1.56)	6.6 (1.2)
DTI	27.54 (2.31)	27.67 (2.15)	27.47 (1.64)	22.96 (1.31)	23.62 (1.67)	23.55 (1.74)	22.27 (1.14)	23.21 (1.38)
Truck	25.59 (3.12)	25.44 (3.25)	29.77 (4.1)	28.15 (3.59)	29.63 (3.22)	29.55 (3.41)	28.82 (3.95)	25.56 (3.53)
Phoneme	18 (0.49)	17.96 (0.49)	17.69 (0.47)	18.32 (0.56)	17.83 (0.43)	17.78 (0.42)	18.21 (0.56)	18.1 (0.52)

Table 4.5: Empirical results using PCA basis and ratio of KDE estimates reported as average percent error (standard deviation). Rates within .05 of the smallest misclassification rate are highlighted in green.
The L_1 and L_2 models do best on the Phoneme dataset. The Gaussian copula and fDAB do about equally well on the wine dataset, but the Gaussian copula has less variance in its misclassification rate. The Gaussian copula does best here on the DTI data and naive Bayes does best on the Truck data.

	t Copula	Gaussian Copula	Naive Bayes	fDAB	L_1	L_2	Quadratic	Interaction
Wine	8.29 (1.3)	7.46 (1.09)	8.66 (1.41)	7.47 (1.57)	8.15 (1.48)	8.34 (1.44)	8.68 (1.86)	8.27 (1.48)
DTI	23.11 (2.01)	22.88 (2.01)	25.2 (1.61)	23.48 (1.9)	23.6 (1.97)	23.55 (1.9)	23.17 (1.62)	23.37 (1.88)
Truck	30.77 (2.65)	30.9 (2.67)	30.37 (3.13)	31.35 (3.52)	31.07 (3.17)	31.13 (3)	33.85 (3.65)	35.72 (4.29)
Phoneme	21.21 (0.42)	21.24 (0.48)	21.05 (0.71)	20.64 (0.84)	20.25 (0.77)	20.25 (0.76)	21.1 (0.88)	22.9 (0.72)

Table 4.6: Empirical results using PLS basis and ratio of KDE estimates reported as average percent error (standard deviation). The naive Bayes model does best here on the Wine data. The Quadratic model and variants of fDAB do better on the DTI data than naive Bayes or the copulas. Switching to the PLS basis improves misclassification results substantially for the copulas and naive Bayes on the Truck and Phoneme datasets.

	t Copula	Gaussian Copula	Naive Bayes	fDAB	L_1	L_2	Quadratic	Interaction
Wine	8.23 (1.65)	7.41 (1.48)	6.3 (1.52)	7.88 (2.62)	7.93 (1.7)	8.01 (1.64)	17.51 (3.54)	7.45 (2.06)
DTI	25.32 (1.92)	24.82 (2.06)	24.13 (1.73)	22.33 (1.47)	22.4 (1.62)	22.36 (1.54)	22.23 (1.41)	22.53 (1.57)
Truck	23.43 (2.72)	23.35 (2.61)	26.67 (3.14)	26.91 (3.69)	27.35 (3.39)	27.64 (3.43)	27.94 (3.48)	24.05 (3.23)
Phoneme	18.46 (0.55)	18.42 (0.52)	18.52 (0.5)	19.32 (0.52)	18.64 (0.5)	18.63 (0.5)	19.36 (0.6)	18.98 (0.55)

Chapter 5

Discussion

In this work, we have demonstrated a new method for log density ratio estimation using splines in Chapter 2. Although we only estimate univariate log density ratios, the methodology can be applied in more general cases. For example, if $k \leq p$ features in a dataset are known to be independent of the others, one may estimate their joint density as a component of the full joint density of all variables, simplifying overall computation. We show in Section 2.3 that the spline method consistently outperforms the standard log ratio of kernel density estimates in a variety of simulation settings.

The spline methodology of estimating the log density ratio also demonstrates advantages compared to kernel density estimation in the classification methods of Chapters 3 and 4, where the marginal log density ratios are treated as features. In Chapter 3, we show that there are situations where the multivariate DAB classifier can recover the optimal Bayes solution in a Gaussian setting by taking linear or quadratic combinations of marginal log density ratios. These theoretical results are confirmed by simulation analyses done in Section 3.3, where we show that DAB outperforms a variety of other methods, including using GLMs on the actual data. In Section 3.4, we show that DAB can outperform naive Bayes and logistic regression as the size of the training samples increase. We offer a qualitative explanation of where multivariate DAB can outperform a standard logistic regression, explaining that DAB can better account for correlations between the features when there are differences in correlations within each class.

Chapter 4 extends the DAB model to functional data through the functional dependence-adjusted naive Bayes classifier, or fDAB. In fDAB, the features are once again marginal log density ratios, but these are marginal log density ratios of functional projection scores. We compare fDAB's performance versus both the functional naive Bayes classifier, as developed by [3], and the naive Bayes classifier augmented by a copula, as derived by [16] in simulation and real data settings. In simulation settings, DAB cannot match the performance of naive Bayes or the copula models when the data are sample paths of Gaussian processes. A possible explanation for this is that the chosen basis, which is formed by performing some variant of functional PCA on the observations, often has the effect of removing or at least weakening correlations amongst the projection scores. However, fDAB can provide lower misclassification results when the data are non-Gaussian and when there are significant dependencies among the coefficients used to generate the data.

In the real data setting, we compare fDAB to naive Bayes and the copula methods using both functional PCA and PLS bases on four datasets. These comparisons also compare kernel density estimates for estimating the log density ratio with spline estimates. Across these four settings, DAB performs consistently well on the DTI data and occasionally presents the lowest misclassification rate on the wine data. The truck and phoneme datasets exhibit their lowest misclassification results using the PLS basis expansion and the Gaussian copula, possibly indicating that these datasets resemble paths of Gaussian processes more so than the wine and DTI datasets.

In general, DAB can be useful in both multivariate and functional data settings. In the context of multivariate data, the performance improvements come through its ability to more closely approach the optimal Bayes classifier through estimating the log of the marginal density ratios. By treating these marginal log density ratios as features, DAB can account for dependencies in the dataset more explicitly than naive Bayes and standard logistic regression. In particular, a logistic regression with only main effects terms assumes that, in the Gaussian setting, the two classes share the same covariance and differ only in terms of the mean. DAB does not make this assumption and can perform better than logistic regression by adjusting for these dependencies. In terms of functional data, DAB similarly can model the dependencies between features. There are cases where Gaussian or t copulas outperform DAB, since the functional data resemble sample paths of Gaussian processes; however, when the dependence is non-Gaussian, DAB can provide lower misclassification rates.

5.1 Future work

There are several avenues of future research to pursue. In the multivariate setting, DAB can be extended to the case of discrete covariates or used in the presence of categorical covariates. It would also be interesting to compare DAB's performance in the multivariate real data setting when the model includes higher order terms. We showed in Section 3.2 that adding higher order terms can recover the optimal Bayes classifier in certain Gaussian simulation settings, but we did not add higher order terms in multivariate real data settings. Comparing DAB to logistic regression in this setting could determine whether DAB can outperform logistic regression empirically even in the presence of higher order terms.

Additionally, more theoretical analysis of DAB's performance relative to logistic regression and naive Bayes would be worthwhile. Ng and Jordan [2] show that logistic regression's asymptotic misclassification rate is lower than naive Bayes's asymptotic misclassification rate since they form a generative/discriminative pair. In Section 3.4, we show that DAB can outperform logistic regression on some datasets but on other datasets it appears to mirror logistic regression's performance. We qualitatively attribute this to differences in the correlation structures in the two classes, but a more formal argument could determine specifically when DAB can be expected to outperform logistic regression.

One could also extend DAB in the functional data setting to be used in the context of multivariate functions. Creating interaction terms in this setting would be particularly useful, as one can learn how two functional components vary together relative to a response variable. Additionally, one could use DAB in a setting where a dataset includes both functional and multivariate data. These hybrid settings are an interesting use case for DAB since it is flexible and can accommodate different types of covariates.

Last, it would be interesting to understand how sensitive DAB's performance is based on the method of density ratio estimation. In Section 3.3, we find that the spline method of density ratio estimation generally outperforms the ratio of kernel density estimates with the exception of the multivariate uniform case. Understanding DAB's sensitivity to the method of density ratio estimation could help one decide when to use DAB and which method of density ratio estimation would best suit a given dataset.

Bibliography

[1] B. W. Silverman, *Density ratios, empirical likelihood and cot death*, *Journal of the Royal Statistical Society: Series C (Applied Statistics)* **27** (1978), no. 1 26–33.

[2] A. Y. Ng and M. I. Jordan, *On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes*, in *Advances in neural information processing systems*, pp. 841–848, 2002.

[3] X. Dai, H.-G. Müller, and F. Yao, *Optimal Bayes classifiers for functional data and density ratios*, *Biometrika* **104** (2017), no. 3 545–560.

[4] S. Ullah and C. F. Finch, *Applications of functional data analysis: a systematic review*, *BMC medical research methodology* **13** (2013), no. 1 43.

[5] B. M. Wise, N. B. Gallagher, S. W. Butler, D. D. White, and G. G. Barna, *A comparison of principal component analysis, multiway principal component analysis, trilinear decomposition and parallel factor analysis for fault detection in a semiconductor etch process*, *Journal of Chemometrics* **13** (1999), no. 3-4 379–396.

[6] H. Wang and M. Yao, *Fault detection of batch processes based on multivariate functional kernel principal component analysis*, *Chemometrics and Intelligent Laboratory Systems* **149** (2015) 78–89.

[7] R. Burfield, C. Neumann, and C. P. Saunders, *Review and application of functional data analysis to chemical data—the example of the comparison, classification, and database search of forensic ink chromatograms*, *Chemometrics and Intelligent Laboratory Systems* **149** (2015) 97–106.

[8] A. Petersen, C.-J. Chen, and H.-G. Müller, *Quantifying and visualizing intraregional connectivity in resting-state functional magnetic resonance imaging with correlation densities*, *Brain connectivity* **9** (2019), no. 1 37–47.

[9] P. Kokoszka, H. Miao, A. Petersen, and H. L. Shang, *Forecasting of density functions with an application to cross-sectional and intraday returns*, *International Journal of Forecasting* **35** (2019), no. 4 1304–1317.

[10] A. Delaigle and P. Hall, *Achieving near perfect classification for functional data*, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **74** (2012), no. 2 267–286.

[11] A. Delaigle and P. Hall, *Classification using censored functional data*, *Journal of the American Statistical Association* **108** (2013), no. 504 1269–1283.

[12] F. Rossi and N. Villa, *Support vector machine for functional data classification*, *Neurocomputing* **69** (2006), no. 7-9 730–742.

[13] Y. Wu and Y. Liu, *Functional robust support vector machines for sparse and irregular longitudinal data*, *Journal of computational and Graphical Statistics* **22** (2013), no. 2 379–395.

[14] S. Balakrishnan and D. Madigan, *Decision trees for functional variables*, in *Sixth International Conference on Data Mining (ICDM'06)*, pp. 798–802, IEEE, 2006.

[15] A. Delaigle, P. Hall, *et. al.*, *Defining probability density for a distribution of random functions*, *The Annals of Statistics* **38** (2010), no. 2 1171–1193.

[16] W. Huang and D. Ruppert, *Copula-based functional Bayes classification with principal components and partial least squares*, *arXiv preprint arXiv:1906.00538* (2019).

[17] P. J. Bickel, E. Levina, *et. al.*, *Some theory for Fisher's linear discriminant function, naive Bayes', and some alternatives when there are many more variables than observations*, *Bernoulli* **10** (2004), no. 6 989–1010.

[18] T. Y. Young and T. W. Calvert, *Classification, estimation, and pattern recognition.* Elsevier, 1974.

[19] P. Domingos and M. Pazzani, *On the optimality of the simple Bayesian classifier under zero-one loss*, *Machine learning* **29** (1997), no. 2-3 103–130.

[20] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An introduction to statistical learning*, vol. 112. Springer, 2013.

[21] E. Fox, *Bayes optimal classifier and naive Bayes*, 2017. Course notes from University of Washington.

[22] C. R. Shalizi, *Estimating distributions and densities*, 2016. Course notes from Carnegie Mellon University.

[23] D. J. Hand and K. Yu, *Idiot's Bayes—not so stupid after all?*, *International statistical review* **69** (2001), no. 3 385–398.

[24] J. H. Friedman, *On bias, variance, 0/1—loss, and the curse-of-dimensionality*, *Data mining and knowledge discovery* **1** (1997), no. 1 55–77.

[25] H. Zhang, *The optimality of naive Bayes*, *AA* **1** (2004), no. 2 3.

[26] R. Schapire and D. Blei, "Logistic regression and naive Bayes."

[27] T. Mitchell, "Generative and discriminative classifiers: Naive Bayes and logistic regression."

[28] J. Mercer, *Xvi. functions of positive and negative type, and their connection the theory of integral equations*, *Philosophical transactions of the royal society of London. Series A, containing papers of a mathematical or physical character* **209** (1909), no. 441-458 415–446.

[29] C. Preda, G. Saporta, and C. Lévéder, *PLS classification of functional data*, *Computational Statistics* **22** (2007), no. 2 223–235.

[30] C. Preda and G. Saporta, *Clusterwise PLS regression on a stochastic process*, *Computational Statistics & Data Analysis* **49** (2005), no. 1 99–108.

[31] S. Hido, Y. Tsuboi, H. Kashima, M. Sugiyama, and T. Kanamori, *Statistical outlier detection using direct density ratio estimation*, *Knowledge and information systems* **26** (2011), no. 2 309–336.

[32] S. Liu, M. Yamada, N. Collier, and M. Sugiyama, *Change-point detection in time-series data by relative density-ratio estimation*, *Neural Networks* **43** (2013) 72–83.

[33] M. Uehara, I. Sato, M. Suzuki, K. Nakayama, and Y. Matsuo, *Generative adversarial nets from a density ratio estimation perspective*, *arXiv preprint arXiv:1610.02920* (2016).

[34] M. Sugiyama, T. Suzuki, and T. Kanamori, *Density ratio estimation in machine learning*. Cambridge University Press, 2012.

[35] M. Sugiyama, M. Yamada, P. Von Buenau, T. Suzuki, T. Kanamori, and M. Kawanabe, *Direct density-ratio estimation with dimensionality reduction via least-squares hetero-distributional subspace search*, *Neural Networks* **24** (2011), no. 2 183–198.

[36] T. Kanamori, S. Hido, and M. Sugiyama, *A least-squares approach to direct importance estimation*, *Journal of Machine Learning Research* **10** (2009), no. Jul 1391–1445.

[37] D. Fraser *et. al.*, *Sufficient statistics with nuisance parameters*, *The Annals of Mathematical Statistics* **27** (1956), no. 3 838–842.

[38] I. Good and R. A. Gaskins, *Nonparametric roughness penalties for probability densities*, *Biometrika* **58** (1971), no. 2 255–277.

[39] C. H. Reinsch, *Smoothing by spline functions*, *Numerische mathematik* **10** (1967), no. 3 177–183.

[40] S. N. Wood, *Thin plate regression splines*, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **65** (2003), no. 1 95–114.

[41] D. Ruppert, M. P. Wand, and R. J. Carroll, *Semiparametric regression.* No. 12. Cambridge University Press, 2003.

[42] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, *Generative adversarial nets*, in *Advances in neural information processing systems*, pp. 2672–2680, 2014.

[43] I. Goodfellow, *Nips 2016 tutorial: Generative adversarial networks*, *arXiv preprint arXiv:1701.00160* (2016).

[44] E. Uchibe, *Model-free deep inverse reinforcement learning by logistic regression*, *Neural Processing Letters* **47** (2018), no. 3 891–905.

[45] M. Kawakita and T. Kanamori, *Semi-supervised learning with density-ratio estimation*, *Machine learning* **91** (2013), no. 2 189–209.

[46] H. Khan, L. Marcuse, and B. Yener, *Deep density ratio estimation for change point detection*, *arXiv preprint arXiv:1905.09876* (2019).

[47] H. Nam and M. Sugiyama, *Direct density ratio estimation with convolutional neural networks with application in outlier detection*, *IEICE TRANSACTIONS on Information and Systems* **98** (2015), no. 5 1073–1079.

[48] J. Chen, Y. Liu, *et. al.*, *Quantile and quantile-function estimations under density ratio model*, *The Annals of Statistics* **41** (2013), no. 3 1669–1692.

[49] T. Suzuki, M. Sugiyama, T. Kanamori, and J. Sese, *Mutual information estimation reveals global associations between stimuli and biological processes*, *BMC bioinformatics* **10** (2009), no. S1 S52.

[50] V. Vapnik, *The nature of statistical learning theory.* Springer science & business media, 2013.

[51] G. Wahba, Y. Wang, C. Gu, R. Klein, B. Klein, *et. al.*, *Smoothing spline ANOVA for exponential families, with application to the Wisconsin Epidemiological Study of Diabetic Retinopathy: the 1994 Neyman Memorial Lecture*, *The Annals of Statistics* **23** (1995), no. 6 1865–1895.

[52] Y. Wang, *Smoothing splines: methods and applications.* CRC Press, 2011.

[53] S. Wood, *Generalized Additive Models: An Introduction with R.* Chapman and Hall/CRC, 2 ed., 2017.

[54] H. Demirtas, R. Allozi, and R. Gao, *MultiRNG: Multivariate Pseudo-Random Number Generation*, 2019. R package version 1.2.2.

[55] M. Falk, *A simple approach to the generation of uniformly distributed random variables with prescribed correlations*, *Communications in Statistics-Simulation and Computation* **28** (1999), no. 3 785–791.

[56] P. Xue-Kun Song, *Multivariate dispersion models generated from gaussian copula*, *Scandinavian Journal of Statistics* **27** (2000), no. 2 305–320.

[57] B. W. Silverman, *Density estimation for statistics and data analysis*, vol. 26. CRC press, 1986.

[58] S. Wood and M. S. Wood, *Package 'mgcv'*, *R package version* **1** (2015) 29.

[59] D. Dua and C. Graff, *UCI machine learning repository*, 2017.

[60] M. Majka, *naivebayes: High Performance Implementation of the Naive Bayes Algorithm in R*, 2019. R package version 0.9.6.

[61] A. Kassambara, *ggcorrplot: Visualization of a correlation matrix using'ggplot2'*, *R package version 0.1* **1** (2016).

[62] W. J. Nash, T. L. Sellers, S. R. Talbot, A. J. Cawthorn, and W. B. Ford, *The population biology of abalone (haliotis species) in tasmania. i. blacklip abalone (h. rubra) from the north coast and islands of bass strait*, *Sea Fisheries Division, Technical Report* **48** (1994) p411.

[63] R. Kohavi, *Scaling up the accuracy of naive Bayes classifiers: A decision-tree hybrid.*, in *Kdd*, vol. 96, pp. 202–207, 1996.

[64] D. Harrison Jr and D. L. Rubinfeld, *Hedonic housing prices and the demand for clean air*, .

[65] V. G. Sigillito, S. P. Wing, L. V. Hutton, and K. B. Baker, *Classification of radar returns from the ionosphere using neural networks*, *Johns Hopkins APL Technical Digest* **10** (1989), no. 3 262–266.

[66] B. V. Ramana, M. S. P. Babu, and N. Venkateswarlu, *A critical comparative study of liver patients from usa and india: an exploratory analysis*, *International Journal of Computer Science Issues (IJCSI)* **9** (2012), no. 3 506.

[67] J. W. Smith, J. Everhart, W. Dickson, W. Knowler, and R. Johannes, *Using the adap learning algorithm to forecast the onset of diabetes mellitus*, in *Proceedings of the Annual Symposium on Computer Application in Medical Care*, p. 261, American Medical Informatics Association, 1988.

[68] R. P. Gorman and T. J. Sejnowski, *Analysis of hidden units in a layered network trained to classify sonar targets*, *Neural networks* **1** (1988), no. 1 75–89.

[69] T. Hastie and J. Qian, *Glmnet vignette*, *Retrieve from http://www. web. stanford. edu/˜ hastie/Papers/Glmnet_Vignette. pdf. Accessed September* **20** (2014) 2016.

[70] M. D. Ernst, *A multivariate generalized laplace distribution*, *Computational Statistics* **13** (1998), no. 2 227–232.

[71] J. Usset, A.-M. Staicu, and A. Maity, *Interaction models for functional regression*, *Computational statistics & data analysis* **94** (2016) 317–329.

[72] J. Goldsmith, F. Scheipl, L. Huang, J. Wrobel, J. Gellar, J. Harezlak, M. W. McLean, B. Swihart, L. Xiao, C. Crainiceanu, and P. T. Reiss, *refund: Regression with Functional Data*, 2018. R package version 0.1-17.

[73] H. L. Shang and R. J. Hyndman, *fds: Functional Data Sets*, 2018. R package version 1.8.

[74] M. W. McLean, G. Hooker, and D. Ruppert, *Restricted likelihood ratio tests for linearity in scalar-on-function regression*, *Statistics and Computing* **25** (2015), no. 5 997–1008.

[75] N. N. Clark, M. Gautam, W. S. Wayne, D. W. Lyons, G. Thompson, and B. Zielinska, *Heavy-duty vehicle chassis dynamometer testing for emissions inventory, air quality modeling, source apportionment and air toxics emissions inventory*, *Coordinating Research Council, incorporated* (2007).