1

Flexible Discriminant and Mixture Models

Trevor Hastie
Stanford University
http://stat.stanford.edu/~trevor

Robert Tibshirani
University of Toronto
http://utstat.toronto.edu/~tibs

Andreas Buja
AT&T Laboratories, Research
http://www.research.att.com/~ andreas

1 Introduction

In the generic classification or discrimination problem, the outcome of interest G falls into J unordered classes, which for convenience we denote by the set $\{1, 2, \cdots J\}$. We wish to build a rule for predicting the class membership of an item based on p measurements of predictors or features $X \in R^p$. Our training sample consists of the class membership and predictors for N items. This is an important practical problem with applications in many fields. Traditional statistical methods for this problem include linear discriminant analysis (LDA) and multiple logistic regression, nearest neighbor methods and classification trees. Neural network classifiers have become a powerful alternative, with the ability to incorporate a very large number of features in an adaptive nonlinear model. Ripley (1994) gives an informative review from a statistician's viewpoint.

This chapter is about linear discriminant analysis, and a variety of ways of enhancing it as tool for classification and data analysis. Some of the virtues of LDA are as follows:

- It is a simple prototype classifier. A new observation is classified to the class with closest centroid. A slight twist is that distance is measured in the Mahalanobis metric, using a pooled covariance estimate.
- LDA is the estimated Bayes classifier if the observations are multivariate Gaussian in each class, with a common covariance matrix. Since this assumption is unlikely to be true, this might not seem to be much of a virtue.
- The decision boundaries created by LDA are linear, leading to decision rules that are simple to describe and implement.

- LDA provides natural low-dimensional views of the data. For example, Fig. 7 is an informative two–dimensional view of data in 256 dimensions with 10 classes.
- Often LDA produces the best classification results, because of its simplicity and low variance. LDA was among the top 3 classifiers for 11 of the 22 datasets studied in the STATLOG project (Michie et al. 1994).

Unfortunately the simplicity of LDA causes it to fail in a number of situations as well:

- Often linear decision boundaries do not adequately separate the classes. When N is large, it is possible to estimate more complex decision boundaries. Quadratic discriminant analysis (QDA) is often useful here, and allows for quadratic decision boundaries. More generally we would like to be able to model irregular decision boundaries.
- The aforementioned shortcoming of LDA can often be paraphrased by saying that a single prototype per class is insufficient. LDA uses a single prototype (class centroid) plus a common covariance matrix to describe the spread of the data in each class. In many situations, several prototypes are more appropriate.
- At the other end of the spectrum, we may have way too many (correlated) predictors, for example in the case of digitized analogue signals and images. In this case LDA uses too many parameters, which are estimated with high variance, and its performance suffers. In cases such as this we need to restrict or regularize LDA even further.

In this chapter we describe a class of techniques that attend to all these issues by generalizing the LDA model. This is achieved largely by three different ideas.

- The first idea is to recast the LDA problem as a linear regression problem. Many techniques exist for generalizing linear regression to more flexible, nonparametric forms of regression. This in turn leads to more flexible forms of discriminant analysis, which we call FDA. In most cases of interest, the regression procedures can be seen to identify an enlarged set of predictors via basis expansions. FDA amounts to LDA in this enlarged space.
- In the case of too many predictors, such as the pixels of a digitized image, we do not want to expand the set: it is already too large. The second idea is to fit an LDA model, but penalize its coefficients to be smooth or otherwise coherent in the spatial domain, i.e. as an image. We call this procedure *penalized discriminant analysis* or PDA. With FDA itself, the expanded basis set is often so large that regularization is also required. Both of these can be achieved via a suitably regularized regression in the context of the FDA model.
- The third idea is to model each class by a mixture of two or more Gaussians with different centroids, but with every component Gaussian, both within and between classes, sharing the same covariance matrix. This allows for

more complex decision boundaries, and allows for subspace reduction as in LDA. We call this extension *mixture discriminant analysis* or MDA.

The ideas behind FDA were originally proposed in Breiman and Ihaka (1984), and are developed further in Hastie et al. (1994) and Ripley (1996), and along with PDA in Hastie et al. (1995). The MDA model is developed in Hastie and Tibshirani (1996). In this chapter we describe the essential aspects of each of these techniques, and illustrate each with examples. We also describe in some detail software we provide in Splus for fitting all these models. We also discuss some further extensions and modifications not contained in the previously published work.

2 Linear Discriminant Analysis

Here we review some relevant details of LDA. We assume that the conditional density of the predictors in each class, denoted by $P(X|G)$, is multivariate Gaussian with each class having its own mean vector, but sharing a common covariance matrix. The density in class j is

$$\phi(X;\mu_j,\Sigma) = \frac{1}{(2\pi)^{\frac{p}{2}}|\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(X-\mu_j)^T\Sigma^{-1}(X-\mu_j)}. \tag{2.1}$$

The class prior probabilities are $P(G=j)=\Pi_j$. In this idealised setting, where everything is known, we can also obtain the ideal or *Bayes optimal* classifier. We will use Bayes formula to flip the densities into class posterior probabilities $P(G|X)$. Knowing $P(G|X)$ exactly is the best one can do in classification. If the new observations to be classified arise from this same joint distribution, the rule

$$C(x) = j \text{ if } P(G=j|x) = \max_\ell P(G=\ell|x) \tag{2.2}$$

achieves the minimum misclassification rate. In this case we have

$$\begin{aligned} P(G=j|X=x) &= \frac{\phi(x;\mu_j,\Sigma)\Pi_j}{\sum_\ell \phi(x;\mu_\ell,\Sigma)\Pi_\ell} \\ &\propto \exp\left(x^T\Sigma^{-1}\mu_j - \frac{1}{2}\mu_j^T\Sigma^{-1}\mu_j + \log\Pi_j\right) \\ &= \exp(x^T\beta_j+\alpha_j). \end{aligned} \tag{2.3}$$

The $\propto$ in the second line denotes proportional; since we are interested in which is largest, we are concerned only with the numerators since the denominators do not depend on the class label. Note also that the quadratic terms cancel.

The decision boundary between class i and j is defined as the set of points having equal posterior probability: $\{x \in R^p | P(G=i|x) = P(G=j|x)\}$. From (2.3), we see that, in the case of LDA, this is linear.

The *discriminant function* for class j is denoted by

$$\delta_j(x) = -x^T\Sigma^{-1}\mu_j + \frac{1}{2}\mu_j^T\Sigma^{-1}\mu_j - \log\Pi_j \tag{2.4}$$

$$= -(x^T\beta_j+\alpha_j) \text{ (bias-corrected)}, \tag{2.5}$$

and the equivalent rule is to classify to the class for which $\delta_j(x)$ is smallest.

In practice we have to estimate the parameters β_j and α_j using the training data. LDA uses the maximum–likelihood estimates (MLEs), which amounts to plugging the MLEs $\hat{\mu}_j$, $\hat{\pi}_j$ and $\hat{\Sigma}$ into the formula (2.4). Here

$$\hat{\Sigma} = \frac{1}{N-p}\sum_{j=1}^{J}\sum_{g_i=j}(x_i-\hat{\mu}_j)(x_i-\hat{\mu}_j)^T. \tag{2.6}$$

Some remarks:

- Even if the Gaussian assumptions are correct, the *estimated* Bayes classifier need not be optimal, since it will have bias and variance.
- When the dimension p of the space is large, it might seem that the number of parameters in LDA can become a problem, since Σ has $O(p^2)$ parameters. From (2.4), however, we see that only $J(p+2)$ parameters, nonlinear transformations of the originals, are required. In fact, since we *compare* discriminant functions, we can look at differences, and a more precise number of parameters is $(J-1)(p+2)$.
- Examination of (2.3) shows that the quadratic term $x^T\Sigma x$ plays no role, since it is common to all classes. If each of the classes has a different covariance matrix, this no longer holds. In this case the discriminant functions have quadratic components as well, and the decision boundaries are quadratic surfaces. This is known as *quadratic discriminant analysis.*

Figure 1 shows the linear and quadratic decision boundaries of LDA and QDA for a simulated example. In this case, the true density is a mixture of Gaussians in each class, so neither LDA nor QDA is ideal.

2.1 Reduced rank LDA

LDA operates by comparing Mahalanobis distances from the target point to the estimated class centroids, with an adjustment for unequal class priors. All the relevant distance information is contained in the at most $J-1$ dimensional subspace of R^p spanned by the J group centroids. A reduced form of LDA due to Fisher and Rao adds a graphical component to the procedure. One finds the $K < J-1$ dimensional subspace of R^p in which the group centroids are most separated, once again using the Mahalanobis metric confined to this subspace, and then classifies new data to the closest centroid in the reduced space. For small K the data can be plotted in the reduced space, giving a graphical representation of the group separation. Figure 7 shows such a projection for digitized handwritten images, where $p = 256$ and $J = 10$.

Without going into the details, we note that the reduced subspace is obtained from a generalized principal component analysis of the J class centroids, using as metric the within–class covariance matrix:

$$\max u^T\Sigma_{Bet}u \text{ subject to } u^T\Sigma_W u = 1. \tag{2.7}$$

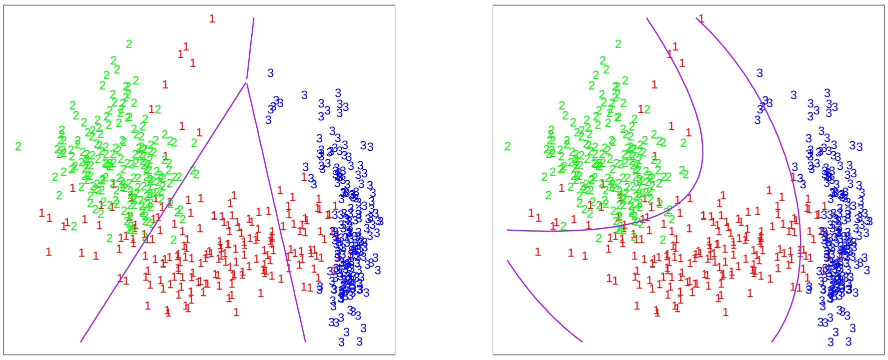

Fig. 1: *The data in these plots are generated from a mixture of Gaussians in each class. The left plot shows the linear decision boundaries produced by the LDA model. The right plot shows the quadratic decision boundaries produced by QDA*

Here Σ_{Bet} denotes the covariance matrix of the class centroids, and $\Sigma_W = \Sigma$, the common or *within class* covariance matrix; see Hastie et al. (1994, 1995) for details. By successively optimizing (2.7) we get a sequence of u_k, orthogonal in Σ, known as the *discriminant* or *canonical coefficients*. The coordinates or *canonical variates* in the reduced space are given by $z_k = u_k^T x,\ k = 1, \ldots, K$. In fact the procedure is a version of generalized canonical correlation analysis (Hastie et al. 1995).

Hastie and Tibshirani (1996) show that this reduced rank formulation of LDA is equivalent to the Gaussian/Bayes procedure above, but where the class densities are estimated by maximum likelihood subject to rank constraints on the centroids: $\text{rank}\{\mu_j,\ j = 1, \ldots, J\} = K$.

3 Flexible Discriminant Analysis

In this section we describe a method for performing LDA using linear regression on derived responses. This in turn leads to nonparametric and flexible alternatives to LDA.

Suppose $\theta : \{1, \ldots, J\} \mapsto R^1$ is a function that assigns scores to the classes, such that the transformed class labels are optimally predicted by linear regression on X. This produces a one–dimensional separation between the classes. More generally, we can find K sets of independent scorings for the class labels, $\theta_1, \theta_2, \cdots \theta_K$, and K corresponding linear maps $\eta_k(X) = X^T \beta_k,\ k = 1, \ldots, K$, chosen to be optimal for multiple regression in R^p. If our training sample has the form $(g_i, x_i),\ i = 1, 2, \cdots N$, then the scores $\theta_k(g)$ and the maps β_k are chosen to minimize the average squared residual,

$$ASR = \frac{1}{N}\sum_{k=1}^{K}\left[\sum_{i=1}^{N}\left(\theta_k(g_i) - x_i^T\beta_k\right)^2\right]. \tag{3.1}$$

The set of scores are assumed to be mutually orthogonal and normalized with respect to an appropriate inner product to prevent trivial zero solutions. It can be shown that the sequence of canonical vectors u_k are identical to the sequence β_k up to a constant (Mardia et al. 1979, Hastie et al. 1994).

Moreover, the Mahalanobis distance of a test point x to the jth class centroid $\hat{\mu}_k$, confined to the subspace defined by the first K canonical vectors, is given by

$$\delta_K(x, \hat{\mu_j}) = \sum_{k=1}^{K} w_k(\eta_k(x) - \bar{\eta}_k^j)^2, \tag{3.2}$$

where $\bar{\eta}_k^j$ is the mean of $\eta_k(x_i)$ for $g_i = j$. Here w_k are coordinate weights that are defined in terms of the mean squared residual r_k^2 of the kth optimally scored fit:

$$w_k = \frac{1}{r_k^2(1 - r_k^2).} \tag{3.3}$$

To summarise: LDA can be performed by a sequence of linear regressions, followed by classification to the closest class centroid in the space of fits. The analogy applies both to the reduced rank version, or the full rank case when $K = J - 1$.

The real power of this result is in the generalizations that it invites. We can replace the linear regression fits $\eta_k(x) = x^T\beta_k$ by far more flexible, nonparametric fits, and by analogy achieve a more flexible classifier than LDA. We have in mind generalized additive fits, spline functions, MARS models and the like. In this more general form the regression problems are defined via the criterion

$$ASR(\{\theta_k, \eta_k\}_{k=1}^K) = \frac{1}{N}\sum_{k=1}^{K}\left[\sum_{i=1}^{N}\left(\theta_k(g_i) - \eta_k(x_i)\right)^2 + \lambda L(\eta_k)\right], \tag{3.4}$$

where L is a regularizer appropriate for some forms of nonparametric regression, such as smoothing splines, additive splines and lower–order anova spline models.

Before we describe the computations involved in this generalization, let us consider a very simple example. Suppose we use degree–two polynomial regression for each η_k. The decision boundaries implied by the (3.2) will be quadratic surfaces, since each of the fitted functions is quadratic, and as in LDA their squares cancel out when comparing distances. We could have achieved *identical* quadratic boundaries in a more conventional way, by augmenting our original predictors with their squares and cross products. In the enlarged space one performs an LDA, and the linear boundaries in the enlarged space map down to quadratic boundaries in the original space. A classic example is a pair of multivariate Gaussians centred at the origin, one having covariance matrix I, and the other cI for $c > 1$. The Bayes decision boundary is the sphere $\|x\|^2 = \frac{pc \log c}{c-1}$, which is a linear boundary in the enlarged space.

Many nonparametric regression procedures operate by generating a basis expansion of derived variables, and then performing a linear regression in the enlarged space. Friedman's MARS procedure (Friedman, 1991) is exactly of this form. Smoothing splines and additive spline models generate an extremely large basis set ($N \times p$ basis functions for additive splines), but then perform a penalized regression fit in the enlarged space. FDA in this case can be shown to perform a *penalized linear discriminant analysis* in the enlarged space. We elaborate in Section 4. Hastie et al. (1994) illustrate FDA on a tough speech recognition problem, with $J = 11$ classes and $p = 10$ predictors. The classes correspond to 11 vowel sounds, each contained in 11 different words. Here are the words, preceded by the symbols that represent them:

i	heed	O	hod	I	hid	C:	hoard	E	head	U	hood
A	had	u:	who'd	a:	hard	3:	heard	Y	hud		

Each of 8 speakers spoke each word 6 times in the training set, and likewise 7 speakers in the test set. The 10 predictors are derived from the digitised speech in a rather complicated way, but standard in the speech recognition world. There are thus 528 training observations, and 462 test observations. Figure 2 shows two–

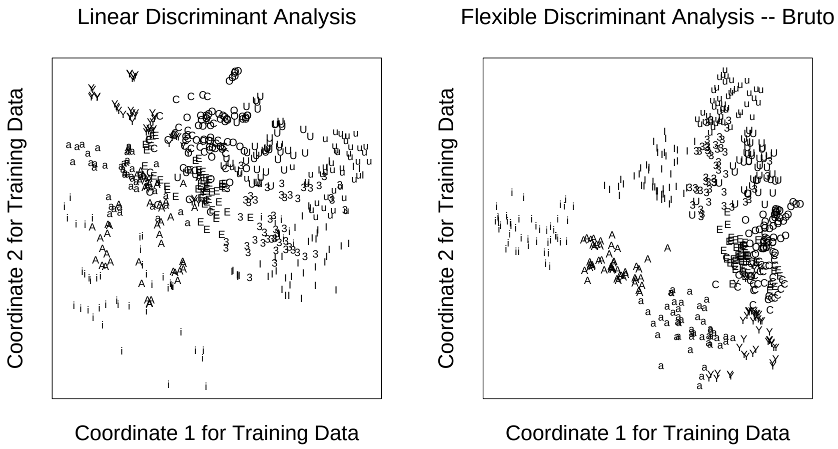

Fig. 2: *The left plot shows the first two LDA canonical variates for the vowel training data. The right plot shows the corresponding projection when FDA/BRUTO is used to fit the model. Notice the improved separation. The letters label the vowel sounds.*

dimensional projections produced by LDA and FDA. The FDA model used adaptive additive–spline regression functions to model the $\eta_k(x)$, and the two coordinates

Table 1 *Vowel recognition data performance results. The results for neural networks are the best among a much larger set, taken from a neural network archive. The notation FDA/BRUTO refers to the regression method used with FDA.*

	Technique	Error rates	
		Training	Test
(1)	LDA	0.32	0.56
	Softmax	0.48	0.67
(2)	QDA	0.01	0.53
(3)	CART	0.05	0.56
(4)	CART (linear combination splits)	0.05	0.54
(5)	Single–layer Perceptron		0.67
(6)	Multi–layer Perceptron (88 hidden units)		0.49
(7)	Gaussian Node Network (528 hidden units)		0.45
(8)	Nearest Neighbour		0.44
(9)	FDA/BRUTO	0.06	0.44
	Softmax	0.11	0.50
(10)	FDA/MARS (degree = 1)	0.09	0.45
	Best reduced dimension (=2)	0.18	0.42
	Softmax	0.14	0.48
(11)	FDA/MARS (degree = 2)	0.02	0.42
	Best reduced dimension (=6)	0.13	0.39
	Softmax	0.10	0.50

plotted in the right plot correspond to $k = 1, 2$. The routine used in Splus is called `bruto`, hence the heading on the plot and in table 1. We see that flexible modelling has helped to separate the classes in this case. Table 1 shows training and test error rates for a number of classification techniques. FDA/MARS refers to Friedman's multivariate adaptive regression splines; degree=2 means second–degree tensor products are permitted. Notice that, for FDA/MARS, the best classification results are obtained in a reduced rank subspace.

3.1 Computing the FDA model

The computations for the FDA coordinates can be simplified in many important cases, in particular when the nonparametric regression procedure can be represented as a linear operator. We will denote this operator by S_λ; i.e. $\hat{y} = S_\lambda y$, where y is the vector of responses and $\hat{y}$ the vector of fits. Additive splines have this property, if the smoothing parameters are fixed, as does MARS once the basis functions are selected. The subscript λ denotes the entire set of smoothing parameters.

We create an $N \times J$ *indicator response matrix* Y from the responses g_i, such that $Y_{ij} = 1$ if $g_i = j$, otherwise $Y_{ij} = 0$. For a five–class problem Y might look like

$$
\begin{array}{c}
 \\ g_1 = 2 \\ g_2 = 1 \\ g_3 = 1 \\ g_4 = 5 \\ g_5 = 4 \\ \vdots \\ g_n = 3
\end{array}
\begin{array}{c}
\begin{array}{ccccc} C_1 & C_2 & C_3 & C_4 & C_5 \end{array} \\
\begin{pmatrix}
0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 \\
 & & \vdots & & \\
0 & 0 & 1 & 0 & 0
\end{pmatrix}
\end{array}
$$

1. *Multivariate nonparametric regression.* Fit a multi–response, adaptive nonparametric regression of Y on X, giving fitted values $\hat{Y}$. Let S_λ be the linear operator that fits the final chosen model, and $\eta(x)$ be the vector of fitted regression functions.
2. *Optimal scores.* Compute the eigen–decomposition of $Y^T\hat{Y} = Y^T S_\lambda Y$, where the eigenvectors Θ are normalised: $\Theta^T D_\pi \Theta = I$. Here $D_\pi = Y^T Y/N$ is a diagonal matrix of the class priors, often estimated by $Y^T Y/N$.
3. *Update* the final model from step 1 using the optimal scores: $\eta(x) \leftarrow \Theta^T \eta(x)$.

Again S_λ can correspond to any regression method. When $S_\lambda = H_X$, the linear regression projection operator, then FDA is LDA. The software we describe in Section 6 makes good use of this modularity; the `fda` function has a `method=` argument which allows one to supply *any* regression function, as long as it follows some natural conventions. The regression functions we provide allow for polynomial regression, adaptive additive models and MARS. They all efficiently handle multiple responses, so step (1) is a single call to a regression routine. The eigen–decomposition in step (2) simultaneously computes all the optimal scoring functions.

3.2 Indicator matrix regression versus FDA

For jointly distributed random variables Y and X, $E(Y|X)$ is known as the regression function. Classical parametric regression procedures, such as linear regression, assume this function has some parametric form and then estimate the parameters. Many nonparametric regression procedures are motivated as ways to estimate this function directly and with minimal assumptions.

In our context, if $Y = Y(G)$ is a coding of the random class variable G, then $E(Y|X) = P(G|X)$, the vector of posterior class probabilities. This suggests that one can apply regression methods directly to the indicator response matrix, and classify to the class corresponding to the largest fitted values, a procedure known in the machine–learning community as *softmax.* In the context of the FDA model, this raises the question: why go beyond step 1: $\hat{Y} = S_\lambda Y$?

Figure 3 shows how disastrous this can be for linear regression, in that one class is completely masked. Figure 4 shows what goes wrong; without loss of generality we have projected the data on to the diagonal line passing through the centroids of the three classes.

The classes are perfectly separated, yet when we perform the indicator variable

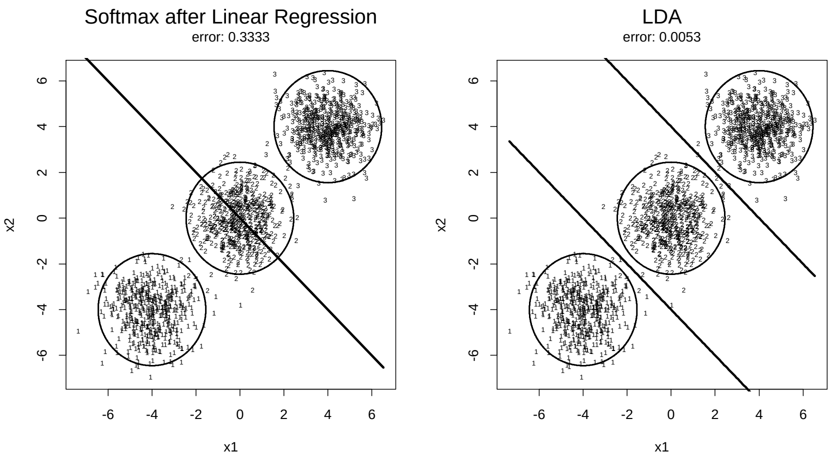

Fig. 3: *Masking effects of Softmax. The data consist of 500 samples each from 3 spherical bivariate Gaussian distributions, whose centroids line up along a line. The centre class is completely masked by the outside two when SOFTMAX is used, while LDA has no such problem.*

regressions, using linear regression, we see that the middle class never dominates. Of course, this problem can be easily solved by using quadratic regressions rather than linear, and, since we anticipate adaptive regression procedures, why the concern?

- Suppose there are ten predictors and the three classes line up along a particular direction α in predictor space. In order to solve the problem via quadratic polynomials, we would need to fit a general quadratic surface with all the bilinear terms included. Of course, if we knew about projection pursuit regression, we could be a bit smarter than that.
- If four classes line up, then quadratic curves do not drop down sufficiently fast, and cubic curves are more appropriate. In general, if M classes line up, order M polynomials tend to be needed to completely untangle them.

When the number of classes is large relative to the number of predictors, masking or partial masking of this kind is relatively frequent. Procedures like MARS will struggle in general to achieve the untangling, because they have difficulty in creating the type of general interaction terms required here.

The post–processing of FDA overcomes this masking without the need for unnecessary transformations. See Hastie et al. (1994) for more details.

4 Penalised Discriminant Analysis

Although FDA is motivated by generalizing optimal scoring, it can also be viewed directly as a form of regularized discriminant analysis. Suppose the regression procedure used in FDA amounts to a linear regression on to a basis expansion

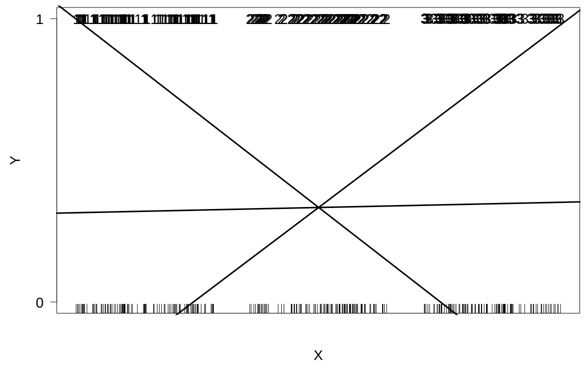

Fig. 4: *The three classes are perfectly separated by the single predictor X (the rug plot shows the distribution of the data). The three lines represent the linear regression fits of each of the three columns of the indicator response matrix Y on X. The 1s, 2s and 3s at the top of the plot indicate the three response indicators Y_i, $i = 1, \dots, 3$, each to be matched with the zeros for each of the other two classes. The middle class is completely masked, in that its regression line (fitted probabilities) never dominates.*

$h(X)$, with a quadratic penalty on the coefficients:

$$ASR(\{\theta_k, \beta_k\}_{k=1}^K) = \frac{1}{N}\sum_{i=1}^N (\theta_k(g_i) - h^T(x_i)\beta_k)^2 + \lambda\beta_k^T\Omega\beta_k. \tag{4.1}$$

The choice of Ω depends on the problem. If $\eta_k(X) = h(X)\beta_k$ is an expansion on spline basis functions, Ω might constrain η_k to be smooth over the domain of X. In the case of additive splines, there are N spline basis functions for each coordinate, resulting in a total of Np basis functions in $h(X)$; Ω in this case is $Np \times Np$ and block diagonal.

The steps in FDA can then be viewed as a generalized form of LDA, which we call *penalised Discriminant Analysis* or PDA:

- Enlarge the set of predictors X via a basis expansion $h(X)$.
- Use (penalised) LDA in the enlarged space, where the penalised Mahalanobis distance is given by

  $$D(x, \mu) = (h(x) - h(\mu))^T(\Sigma_W + \Omega)^{-1}(h(x) - h(\mu)) :$$

  Σ_W is the within–class covariance matrix of the derived variables $h(x_i)$.
- Decompose the classification subspace using a penalised metric:

  $$\max u^T\Sigma_{Bet}u \text{ subject to } u^T(\Sigma_W + \Omega)u = 1$$

Loosely speaking, the penalised Mahalanobis distance tends to give less weight to 'rough' coordinates, and more weight to smooth ones; since the penalty is not diagonal, the same applies to linear combinations that are rough or smooth.

For some classes of problems, the first step, involving the basis expansion, is not needed; we already have far too many (correlated) predictors. A leading example is when the objects to be classified are digitised analog signals:

- the log–periodogram of a fragment of spoken speech, sampled at a set of 256 frequencies
- the greyscale pixel–values in a digitised image of a handwritten digit.

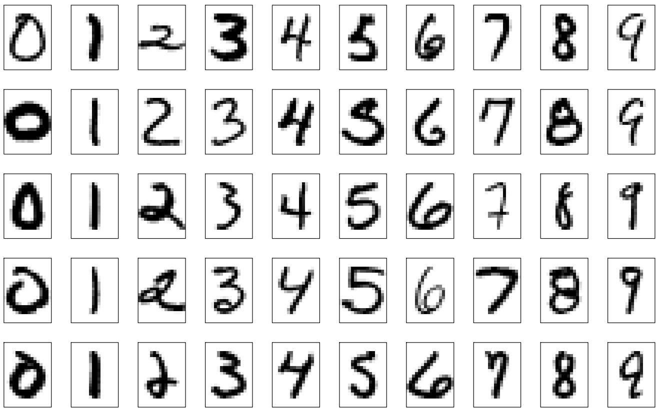

Fig. 5: *Some examples of handwritten digits.*

It is also intuitively clear in these cases why regularization is needed. Take the digitised image as an example. Neighbouring pixel values will tend to be correlated, being often almost the same. This implies that the pair of corresponding LDA coefficients for these pixels can be wildly different and opposite in sign, and thus cancel when applied to similar pixel values. Positively correlated predictors lead to noisy, negatively correlated coefficient estimates, and this noise results in unwanted sampling variance. A reasonable strategy is to regularize the *coefficients* to be smooth over the spatial domain, as with images. This is what PDA does. The computations proceed just as for FDA, except that an appropriate penalised regression method is used. Here $h^T(X)\beta_k = X\beta_k$, and Ω is chosen so that $\beta_k^T\Omega\beta_k$ penalizes roughness in β_k when viewed as an image. Figure 5 shows some examples of handwritten digits. Figure 6 shows the discriminant variates using LDA and PDA. Those produced by LDA appear as *salt and pepper* images, while those produced by PDA are smooth images. The first smooth image can be seen as the coefficients of a linear contrast functional for separating images with a dark central vertical strip (ones, possibly sevens) from images that are hollow in the middle (zeros, some fours). Figure 7 supports this interpretation, and with more difficulty allows an interpretation of the second coordinate. This and other examples are discussed in more detail in Hastie et al. (1995), who also show that the regularization improves the classification performance of LDA on independent test data, by a factor of around 25%, in

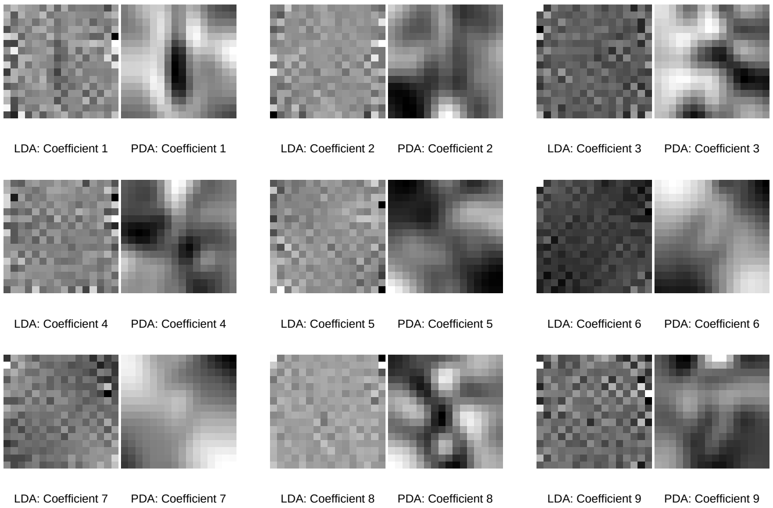

Fig. 6: *The images appear in pairs, and represent the nine discriminant coefficient functions for the digit recognition problem. The left member of each pair is the LDA coefficient, while the right member is the PDA coefficient, regularised to enforce spatial smoothness.*

the cases they tried.

5 Mixture Discriminant Analysis

LDA can be viewed as a *prototype* classifier; each class is represented by its centroid, and we classify to the closest using an appropriate metric. In many situations a single prototype is not sufficient to represent inhomogeneous classes, and mixture models are more appropriate. In this section we review Gaussian mixture models and show how they can be generalized via the FDA and PDA methods discussed earlier. A Gaussian mixture model for the jth class has density

$$P(X|G=j) = \sum_{r=1}^{R_j} \pi_{jr}\phi(X;\mu_{jr},\Sigma), \tag{5.1}$$

where the *mixing proportions* π_{jr} sum to one. This has R_j prototypes for the jth class, and in our specification, the same covariance matrix Σ is used as the metric throughout. Given such a model for each class, the class posterior probabilities are given by

$$P(G=j|X=x) = \frac{\sum_{r=1}^{R_j} \pi_{jr}\phi(X;\mu_{jr},\Sigma)\Pi_j}{\sum_{\ell=1}^{J}\sum_{r=1}^{R_\ell} \pi_{\ell r}\phi(X;\mu_{\ell r},\Sigma)\Pi_\ell}. \tag{5.2}$$

As in LDA, we estimate the parameters by maximum likelihood; again the joint

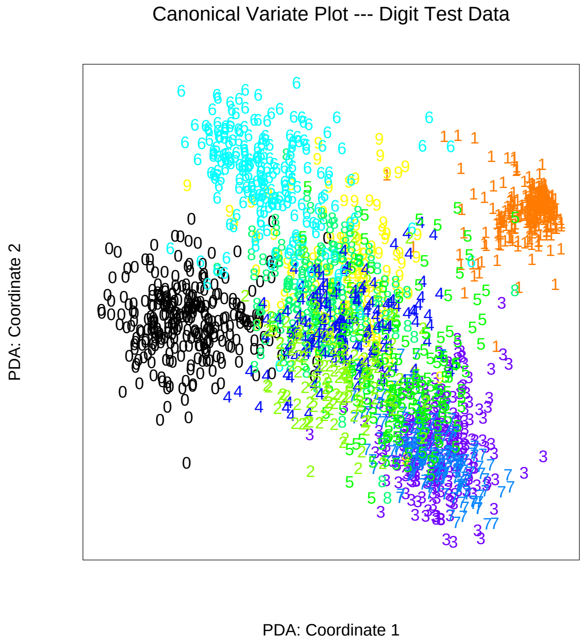

Fig. 7: *The first two penalised canonical variates, evaluated for the test data. The circles indicate the class centroids. The first coordinate contrasts mainly 0s and 1s, while the second contrasts 6s and 7/9s.*

log–likelihood based on $P(G, X)$ is used:

$$\sum_{j=1}^{J} \sum_{g_i=j} \log\left(\sum_{r=1}^{R_j} \pi_{jr} \phi(x_i; \mu_{jr}, \Sigma) \Pi_j\right) \tag{5.3}$$

The sum within the log makes this a rather messy optimization problem if tackled directly. The classical and natural method for computing the MLEs for mixture distributions is the EM algorithm (Dempster et al. 1977), which is known to converge to possess good convergence properties. EM alternates between the two steps:

E–step: Given the current parameters, compute the *responsibility* of subclass c_{jr} within class j for each of the class–j observations ($g_i = j$):

$$W(c_{jr}|x_i, g_i) = \frac{\pi_{jr}\phi(x_i; \mu_{jr}, \Sigma)}{\sum_{k=1}^{R_j} \pi_{jk}\phi(x_i; \mu_{jk}, \Sigma)}. \tag{5.4}$$

M–step: Compute the weighted MLEs for the parameters of each of the component Gaussians within each of the classes, using the weights from the E–step.

In the E–step, the algorithm apportions the unit weight of an observation in class j to the various subclasses assigned to that class. If it is close to the centroid of a particular subclass, and far from the others, it will receive a mass close to one for that subclass. On the other hand, observations halfway between two subclasses will get approximately equal weight for both.

In the M–step, an observation in class j is used R_j times, to estimate the parameters in each of the R_j component densities, with a different weight for each.

The algorithm requires initialization, which can have an impact, since mixture likelihoods are generally multi–modal. Our software allows several strategies; here we describe the default. The user supplies the number R_j of subclasses per class. Within class j a *k–means clustering model* (Hartigan and Wong 1979), with multiple, random starts, is fitted to the data. This partitions the observations into R_j disjoint groups, from which an initial weight matrix, consisting of zeros and ones, is created.

Our assumption of equal component covariance matrix Σ throughout buys an additional simplicity; we can incorporate the rank restrictions in the mixture formulation. We thus maximize the log–likelihood (5.3) subject to rank constraints on *all* the $\sum_j R_j$ centroids: $\text{rank}\{\mu_{jk}\} = K$.

Again the EM algorithm is available, and the M–step turns out to be a weighted version of LDA, with $R = \sum_{j=1}^{J} R_j$ 'classes'. Furthermore, we can use optimal scoring as before to solve the weighted LDA problem, which allows us to use a weighted version of FDA or PDA at this stage. One would expect, in addition to an increase in the number of 'classes', a similar increase in the number of 'observations' in the jth class by a factor of R_j. It turns out that, if linear operators are used for the optimal scoring regression, we use a *blurred* response matrix Z rather than the indicator matrix Y. For example, suppose there are $J = 3$ classes, and $R_j = 3$ subclasses per class. Then Z might be

$$\begin{array}{rc}
 & \begin{array}{ccccccccc} c_{11} & c_{12} & c_{13} & c_{21} & c_{22} & c_{23} & c_{31} & c_{32} & c_{33} \end{array} \\
\begin{array}{r} g_1 = 2 \\ g_2 = 1 \\ g_3 = 1 \\ g_4 = 3 \\ ,\, g_5 = 2 \\ \vdots \\ g_n = 3 \end{array} &
\begin{pmatrix}
0 & 0 & 0 & 0.3 & 0.5 & 0.2 & 0 & 0 & 0 \\
0.9 & 0.1 & 0.0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.1 & 0.8 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0.4 & 0.1 \\
0 & 0 & 0 & 0.7 & 0.1 & 0.2 & 0 & 0 & 0 \\
 & & & \vdots & & & & & \\
0 & 0 & 0 & 0 & 0 & 0 & 0.1 & 0.1 & 0.8
\end{pmatrix}
\end{array},$$

where the entries in a class–j row correspond to $W(c_{jr}|x, g_i)$.

The remaining steps are the same:

$$\left.\begin{array}{l} \hat{Z} = SZ \\ Z^T\hat{Z} = \Theta D \Theta^T \\ \text{Update } \pi\text{s and } \Pi\text{s} \end{array}\right\} \Leftrightarrow \text{M–step of MDA.}$$

These simple modifications to the mixture model add considerable flexibility:

- The dimension reduction step in LDA, FDA or PDA is limited by the number of classes; in particular, for $J = 2$ classes no reduction is possible. MDA substitutes subclasses for classes, and then allows us to look at low dimensional views of the subspace spanned by these subclass centroids. This subspace will tend to be important for discrimination.

- By using FDA or PDA in the M–step, we can adapt even more to particular situations. For example, we can fit MDA models to digitised analogue signals and images, with smoothness constraints built in.

Figure 8 shows the two dimensional discriminant plot for a language recognition task. The data are Japanese characters which are grouped as Hiragana or not. The predictors are eight features extracted from the images of each character. For these

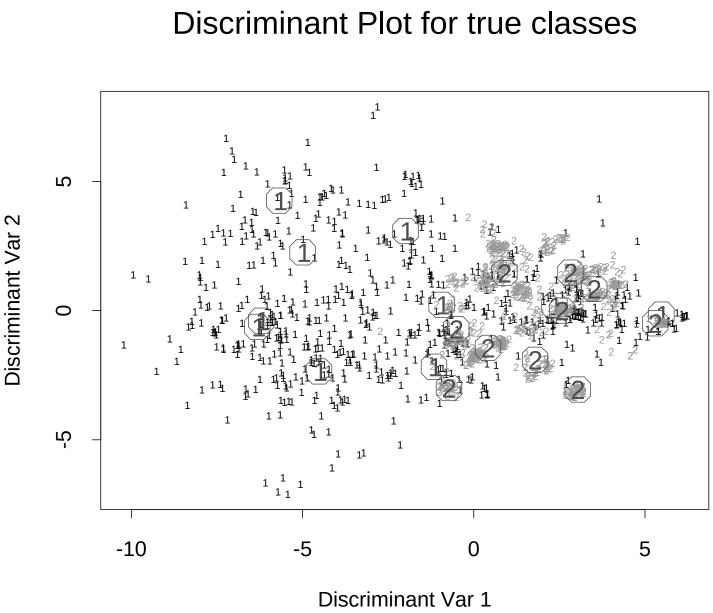

Fig. 8: *Optimal two–dimensional subspace for representing the mixture model fit to the two–class Japanese character recognition task. The numbered disks represent the subclass centres. The Hiragana class tend to occur in many clusters, and mixture models seem suitable for representing them.*

data the test error using 10 mixture centres per class is 10.7%, and with 5 per class is 16%. When combined with adaptive additive modelling in the regression step, the test error drops to 8.2% with 5 centres per class.

We now illustrate some of these ideas on a popular simulated example, taken from Breiman et al. (1994, p. 49–55), and used in Hastie and Tibshirani (1994) and elsewhere. It is a three–class problem with 21 variables, and is considered to

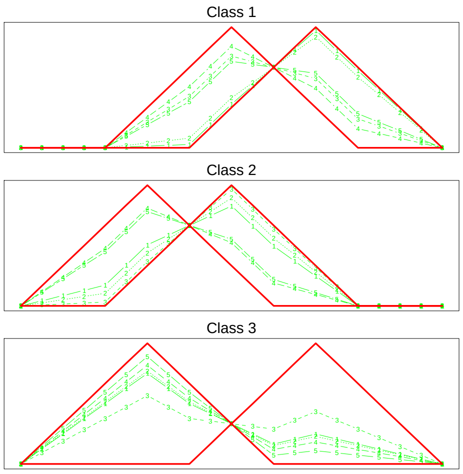

Fig. 9: *Some examples of the waveforms generated from model (5.5) before the Gaussian noise is added.*

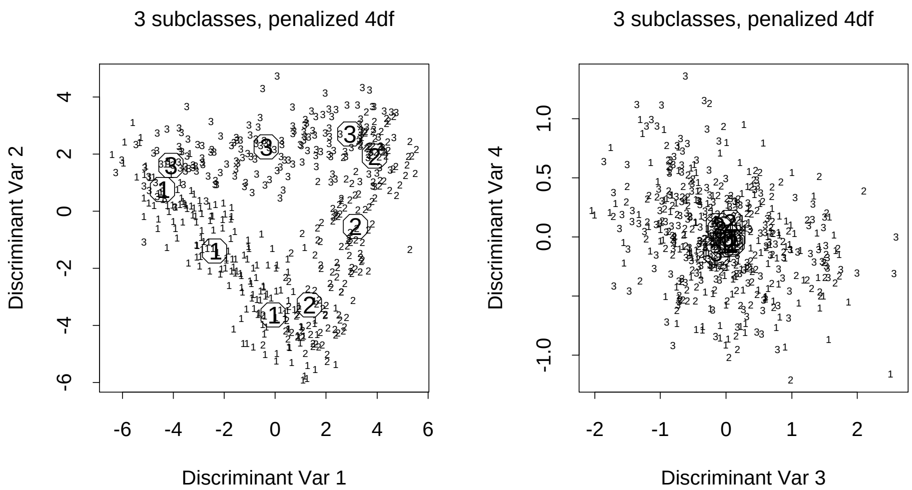

Fig. 10: *Some two–dimensional views of the MDA model fitted to a sample of the waveform model. The points are independent test data, projected on to the leading two canonical coordinates (left panel), and the third and fourth (right panel). The subclass centres are indicated.*

Table 2 *Results for waveform data. The values are averages over 10 simulations, with the standard error of the average in parentheses. The five entries above the line are taken from Hastie et al. (1994). The first model below the line is MDA with 3 subclasses per class. The next line is the same, except that the discriminant coefficients are penalised via a roughness penalty to effectively 4df. The third is the corresponding penalised LDA or PDA model.*

Technique	Error Rates	
	Training	Test
LDA	0.121(0.006)	0.191(0.006)
QDA	0.039(0.004)	0.205(0.006)
CART	0.072(0.003)	0.289(0.004)
FDA/MARS (degree = 1)	0.100(0.006)	0.191(0.006)
FDA/MARS (degree = 2)	0.068(0.004)	0.215(0.002)
MDA (3 subclasses)	0.087(0.005)	0.169(0.006)
MDA (3 subclasses, penalised 4df)	0.137(0.006)	0.157(0.005)
PDA (penalised 4df)	0.150(0.005)	0.171(0.005)

be a difficult pattern recognition problem. The predictors are defined by

$$\begin{aligned} x_i &= uh_1(i) + (1-u)h_2(i) + \epsilon_i && \text{Class 1} \\ x_i &= uh_1(i) + (1-u)h_3(i) + \epsilon_i && \text{Class 2} \\ x_i &= uh_2(i) + (1-u)h_3(i) + \epsilon_i && \text{Class 3,} \end{aligned} \tag{5.5}$$

where $i = 1, 2, \ldots 21$, u is uniform on $(0, 1)$, ϵ_i are standard normal variates, and the h_i are the shifted triangular waveforms: $h_1(i) = \max(6-|i-11|, 0)$, $h_2(i) = h_1(i-4)$ and $h_3(i) = h_1(i+4)$.

Each training sample has 300 observations, and equal priors were used, so there are roughly 100 observations in each class. We used test samples of size 500. The two MDA models are described in the caption.

Figure 10 shows the leading canonical variates for the penalised MDA model, evaluated at the test data. As we might have guessed, the classes appear to lie on the edges of a triangle. This is because the $h_j(i)$ are represented by 3 points in 21–space, thereby forming vertices of a triangle, and each class is represented as a convex combination of a pair of vertices, and hence lie on an edge. Also it is clear visually that all the information lies in the first two dimensions; the percentage of variance explained by the first two coordinates is 99.8%, and we would lose nothing by truncating the solution there. The Bayes risk for this problem is about 0.14 (Breiman et al. 1994); MDA comes close to the optimal rate, which is not surprising since the structure of the MDA model is similar to the generating model.

6 Software for fitting FDA, PDA and MDA models in Splus

Software for fitting FDA, PDA and MDA models is available in the S/Splus language (Becker et al. 1988, Chambers and Hastie 1991).

The function `fda()` fits FDA and PDA models. A `method` argument allows the user to specify the multi–response regression method to be used; the default is linear regression and thus LDA. Other regression methods provided for FDA are polynomial regression, ridge regression, BRUTO and MARS. The following examples illustrate the usage of `fda`, and the associated functions for producing the plots and making predictions. The data are supplied as a list or data–frame, in this case including components `y` , the class variable, and `x`.

```
fit.lda <- fda(y  ~ x, data = vowel.train)
fit.coef <-coef(fit.lda)
fit.coord <- predict(fit.lda, type = 'variates')
```

fits an LDA model, and computes the canonical coefficients and coordinates. Simply typing `fit.lda` will print out a short summary of the model.

```
fit.disc <- coef(fit.lda, type = 'discriminant')
```

computes instead the discriminant function coefficients.

```
plot(fit.lda, fit.train)
```

produces a plot of the first two canonical variates for the training data, while

```
plot(fit.lda, vowel.test, coords = c(3,4))
```

plots coordinates 3 and 4 for the test data; note that these functions will look for components `x` and `y` in `vowel.test`.

```
fit.bruto <- fda(y ~ x, data = vowel.train, method = bruto)
```

or simply

```
fit.brut <- update(fit.lda, method = bruto)
```

fits an FDA/BRUTO model, and assumes the function `bruto()` is available. Likewise `method = 'mars'` will fit FDA/MARS models, and `method = 'polyreg'` will fit polynomial LDA models. Each of these functions has additional, optional arguments, which can also be supplied by name in the call to `fda()`. The `update()` function is very useful for modifying a model by adding, changing or deleting one of the arguments that created its first argument.

```
confusion(fit.brut, vowel.test)
```

computes the $J \times J$ confusion matrix which results from applying the classification rule in `fit.brut` to the test data. One can produce the predicted classes for new data:

```
fit.predict <-predict(fit.brut, vowel.test)
```

where the default `type = 'class'` is implicit. To compute the confusion matrix in a more direct fashion, use

```
confusion(fit.predict, vowel.test$g)
```

There are also `plot()` methods for MDA objects, and these label the subclasses as in Figure 10.

For PDA models a generalized form of ridge regression is provided, called `gen.ridge()`. This is simply used in the `method =` argument to `fda()`. Users supply a penalty matrix and target `df`, and the procedure derives the appropriate penalty constant.

```
fit.pda <- fda(g ~ x, data = zip.train, method = gen.ridge,
               omega = Omega.zip, df = 80)
```

fits the PDA model for the zip–code data. The user has to supply an appropriate PSD penalty matrix `omega` in factored form. The function `laplacian()` produced the penalty matrix `Omega` for this image example, and it is provided with the software. The user also supplies the `df`, which generates a ridge penalty to achieve that many equivalent degrees of freedom.

The `mda()` function has additional arguments for controlling the number of subclasses and the initialization.

```
fit.wave <- mda( g ~ x, data = wave.train, subclasses = 3,
                 method = gen.ridge, omega = Omega.wf, df = 4, trace = T)
plot(fit.wave, wave.test)
```

Here we fit an MDA model with 3 subclasses per class. By default the `kmeans()` clustering algorithm is used to initialize the weights, and also by default 5 random starts to `kmeans` are used. The `trace = T` argument traces the iterations, and prints out the *conditional* log–likelihood at each iteration; in general the full log–likelihood is not easily available for MDA models. Another choice for initialization is `start = 'lvq'`, which uses *Learning Vector Quantization*, a variant of k–means focussed more on classification.

Additional features of the software can be found in the online documentation. The `mda` software is publicly available from the statistics archive at Carnegie–Mellon University with URL: `http://lib.stat.cmu.edu/S/mda`. The software and technical report are also available from the first author's home page:
`http://stat.stanford.edu/~trevor`

7 Further extensions

Our mixture formulation uses a separate mixture of Gaussians for each class. Here we propose a variation that attempts to allocate mixture centres where they are needed for representing the joint distribution $P(G, X)$. We refer to our first model as MDA1 and this new formulation as MDA2. We consider the mixture model

$$P(G,X) = \sum_{r=1}^{R} \pi_r P_r(G,X), \tag{7.1}$$

a mixture of *joint densities.* Furthermore we assume

$$P_r(G,X) = P_r(G)\phi(X;\mu_r,\Sigma). \tag{7.2}$$

This model consists of regions centred at μ_r, and for each there is a class profile $P_r(G)$. The posterior class distribution is given by

$$P(G=j|X=x) = \frac{\sum_{r=1}^{R} \pi_r P_r(G=j)\phi(x;\mu_r,\Sigma)}{\sum_{r=1}^{R} \pi_r \phi(x;\mu_r,\Sigma)}, \tag{7.3}$$

where the denominator is the marginal distribution $P(X)$. MDA2 can also be viewed as a version of MDA1, since

$$P(X|G=j) = \frac{\sum_{r=1}^{R} \pi_r P_r(G=j)\phi(x;\mu_r,\Sigma)}{\sum_{r=1}^{R} \pi_r P_r(G=j)}, \tag{7.4}$$

where $\pi_{rj} = \pi_r P_r(G=j)/\sum_{r=1}^{R} \pi_r P_r(G=j)$ corresponds to the mixing proportions for the j class, and all classes use *the same Gaussian distributions.* In fact, it turns out that MDA2 contains MDA1; see the comments on initialization below.

Once again there is a natural EM algorithm for fitting this model:

E–step: Each mixture component is assigned a responsibility for the ith observation:

$$W(c_r|x_i,g_i) = \frac{\pi_r P_r(g_i,x_i)}{\sum_{k=1}^{R} \pi_k P_k(g_i,x_i)}. \tag{7.5}$$

Note that in addition to the distance of x_i from μ_r, here the class label of the observation contributes to the weight determination; the log of the numerator is

$$-\frac{1}{2}\|x_i-\mu_r\|_{\Sigma}^2 + \log P_r(g_i), \tag{7.6}$$

after cancellation of common factors. This encourages centroids to favour particular classes.

M–step: These compute appropriately weighted MLEs for the parameters of each of the component Gaussians. $\hat{P}_r(j) = \sum_{g_i=j} W(c_r|x_i,g_i)/\sum_{i=1}^{n} W(c_r|x_i,g_i)$, and the π_r are estimated similarly.

Again we can fruitfully resort to optimal scoring to perform the M–step. This allows us to fit the models with rank constraints on the space spanned by the centroids, and also permits more flexible versions of the procedure by using nonparametric regression. In this case the blurred response matrix Z is slightly different, since each observation has potentially some weight for each mixture component; Fig. 11 illustrates the difference. We fitted this model to the Japanese character data. The results were similar to those obtained for the original mixture formulation and are not presented here.

We initialize this algorithm by using an unsupervised k–means clustering algorithm on all the data. This provides an initial weight matrix W, and the algorithm starts at the M–step.

It is not hard to see that, if this weight matrix is constructed as in MDA1, which involves separate k–means cluster models in each class, then MDA2 is identical to

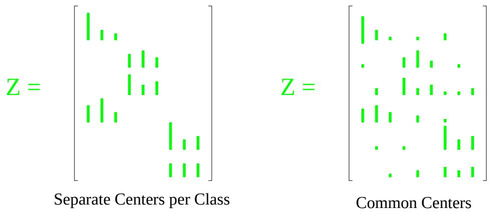

Fig. 11: *Depiction of the blurred response matrices Z for the two types of mixture models. On the left, each class has its own mixture components, while on the right, classes compete for and share mixture components.*

MDA1. In this case the initial versions of $P_k(g)$ will each be devoted to a single class (all zeros, with a one in the appropriate position.) Furthermore, once in this state, it remains like this through the iterations.

Another variation that we are currently exploring is to use *discriminative learning* to fit the model. This amounts to using a conditional likelihood based on $P(G|X)$ to fit the model. It turns out that the resulting algorithm is very similar to the one presented here. Instead of a multiple regression with Z as the response in the M–step, one uses a polychotomous logistic regression instead, with observations the rows of the weight matrix. Details of this will appear elsewhere.

Bibliography

1. R. Becker, J. Chambers, and A. Wilks. *The New S Language.* Wadsworth International Group, 1988.
2. L. Breiman and R. Ihaka. Nonlinear discriminant analysis via scaling and ACE. Technical report, Univ. of California, Berkeley, 1984.
3. L. Breiman, J.H. Friedman, R. Olshen, and C.J. Stone. *Classification and Regression Trees.* Wadsworth, Belmont, California, 1984.
4. J. Chambers and T. Hastie. *Statistical Models in S.* Wadsworth/Brooks Cole, Pacific Grove, California, 1991.
5. Dempster, A.P., Laird, N.M., and Rubin, D.B. Maximum likelihood from incomplete data via the EM algorithm (with discussion). *J. R. Statist. Soc. B*, 39:1–38, 1977.
6. J.H. Friedman. Multivariate adaptive regression splines (with discussion). *Annals of Statistics*, 19(1):1–141, 1991.
7. J. A. Hartigan and M. A. Wong. [algorithm As 136] A k-means clustering algorithm (as R39: 81v30 p355-356). *Applied Statistics*, 28:100–108, 1979.
8. T. Hastie, A. Buja, and R. Tibshirani. Penalized discriminant analysis. *Annals of Statistics*, 23(1):73–102, 1995.
9. T Hastie and R. Tibshirani. Discriminant adaptive nearest neighbor classification. Technical report, Statistics Department, Stanford University, 1994.
10. T. Hastie and R. Tibshirani. Discriminant analysis by gaussian mixtures. *J.*

Royal Statist. Soc. (Series B), 58:155–176, 1996.

11. T. Hastie, R. Tibshirani, and A. Buja. Flexible discriminant analysis by optimal scoring. *Journal of the American Statistical Association*, 89:1255–1270, 1994.
12. K.V. Mardia, J. Kent, and J.M. Bibby. *Multivariate Analysis.* Academic Press, 1979.
13. D. Michie, D. Spiegelhalter, and C. Taylor, editors. *Machine Learning, Neural and Statistical Classification.* Ellis Horwood series in Artificial Intelligence. Ellis Horwood, 1994.
14. B. D. Ripley. Pattern recognition and neural networks. Cambridge University Press, 1996.
15. B.D. Ripley. Neural networks and related methods for classification. *J. Royal Statist. Soc. (Series B)*, pages 409–456, 1994. (with discussion).