ICS 663: Homework 1

Christopher Mullins

September 15, 2011

Contents

1 Introduction

Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA) are techniques for classifying a set of feature vectors by computing linear and quadratic surfaces that separate classes. They can be applied when the likelihood probabilities for the feature vectors ($p(\mathbf{x}|\omega_i)$) are normally distributed.

For classes $\omega_1, \omega_2, \ldots, \omega_c$, one discriminant function $d_i(\mathbf{x})$ is produced for each class i. For a feature vector $\mathbf{x}$, the classifier need only maximize $d_i(\mathbf{x})$. That is, select class i if $\forall i \neq j \, d_i(\mathbf{x}) > d_j(\mathbf{x})$.

In each case (LDA and QDA), the discriminant functions are determined using a mean, μ_i and a covariance matrix $\mathbf{\Sigma}_i$:

$$d_i(\mathbf{x}) = -\frac{1}{2}(\mathbf{x} - \mu_i)^t \mathbf{\Sigma}_i^{-1}(\mathbf{x} - \mu_i) - \frac{1}{2}\ln|\mathbf{\Sigma}_i| + \ln P(\omega_i)$$

The mean vector, μ_i is the mean of the feature vectors in the training set with class i. In QDA, $\mathbf{\Sigma}_i$ is the covariance matrix for feature vectors with class i. In LDA, $\mathbf{\Sigma}_i$ is fixed for each class. Depending on the application, it may be more appropriate to use the covariance matrix for the whole dataset, or $\sigma^2 I_c$, where σ^2 is the mean variance across all of the dimensions.

2 Implementation

I implemented LDA and QDA in R (www.r-project.org). It is designed to work in an arbitrary number of dimensions, but it was only tested with two. It uses matrix operations wherever possible as an attempt to improve performance.

There are three files. `discriminant_analysis.R` contains functionality for loading, parsing, and preparing data, and classifying feature vectors using LDA and QDA. `plots.R` contains helper methods for creating the plots shown later in this report. This includes painting decision regions, means, and classes. Finally, `hw1.R` is a script that seamlessly runs all of the tasks required for this assignment.

2.1 Installation and System Setup

Setup should be fairly straightforward. First, install R from www.r-project.org. Then, open and execute `hw1.R`. This should run all of the required tasks for assignment 1. It is probably best to run it from the commandline, as follows:

```
$ R --no-save --slave < hw1.R
Reading data...
Training...
TRAIN ERROR:
        Case 1 : 0.120000
        Case 2 : 0.133333
        Case 3 : 0.100000
Running discriminant functions on test data...
Case 1 (linear):
  | d_1: 10%  20%  30%  40%  50%  60%  70%  80%  90%  100%
  | d_2: 10%  20%  30%  40%  50%  60%  70%  80%  90%  100%
  | d_3: 10%  20%  30%  40%  50%  60%  70%  80%  90%  100%
Case 2 (linear):
  | d_1: 10%  20%  30%  40%  50%  60%  70%  80%  90%  100%
  | d_2: 10%  20%  30%  40%  50%  60%  70%  80%  90%  100%
  | d_3: 10%  20%  30%  40%  50%  60%  70%  80%  90%  100%
Case 3 (quadratic):
  | d_1: 10%  20%  30%  40%  50%  60%  70%  80%  90%  100%
  | d_2: 10%  20%  30%  40%  50%  60%  70%  80%  90%  100%
  | d_3: 10%  20%  30%  40%  50%  60%  70%  80%  90%  100%
TEST ERROR:
        Case 1 : 0.115497
        Case 2 : 0.110817
        Case 3 : 0.098907
Generating plots...
null device
          1
null device
          1
null device
          1
```

On a two year old MacBook Pro, this takes about five minutes to complete (including generating plots with a fairly high resolution).

2.2 Decision Boundaries

The plots in the following section depict *decision regions* instead of decision boundaries. Areas in which a particular class is selected are painted with the color that corresponds to that class. The `settings` map in `plots.R` controls settings that affect the region generation.

Because this process involves computing the most likely class for small blocks on the plot canvas, this is a very costly operation. One can modify the resolution and brush size in `plots.R` in order to decrease the runtime. For the plots in this report, a resolution of 0.01 and a brush size of 1 were used. To increase the performance by approximatley 100x, one could decrease the granularity by changing the resolution to 0.1 and the brush size to 2.

3 Results

3.1 Error Rates

The error rates for the training and testing data are shown below:

	Train Error	**Test Error**
Case 1	0.1200	0.1155
Case 2	0.1333	0.1108
Case 3	0.1000	0.0989

3.2 Plots

The plots for cases 1, 2, and 3 are shown in figures 3.2, 3.2, and 3.2, respectively.

The plots for cases 1 and 2 clearly demonstrate that using $\mathbf{\Sigma}_i = \sigma^2\mathbf{I}$ and $\mathbf{\Sigma}_i = \text{cov}(D)$ result in linear classifiers. Therefore, these should only be used if the data are linearly separable.

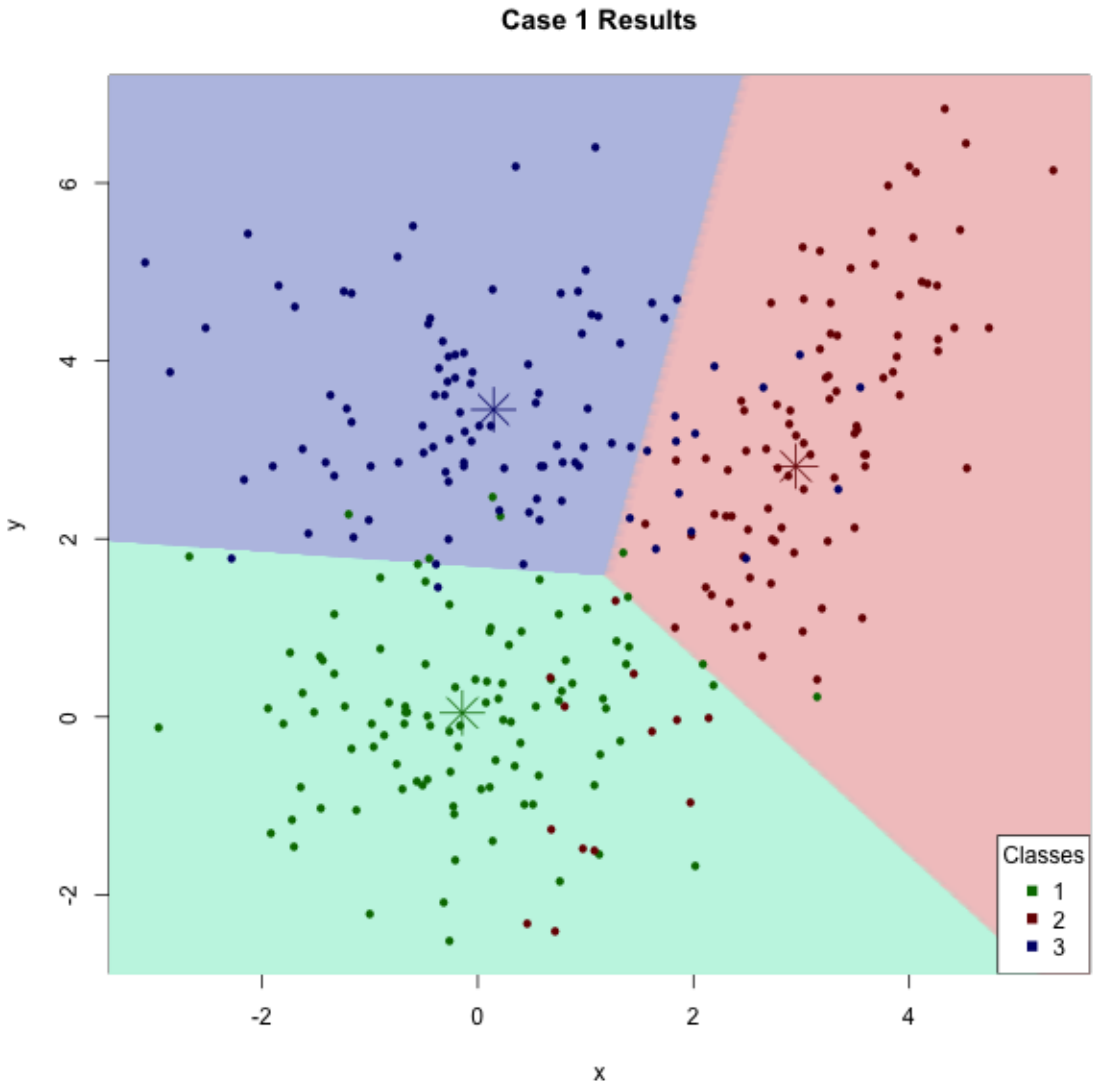

Figure 1: Results for Case 1: $\mathbf{\Sigma}_i = \sigma^2\mathbf{I}$

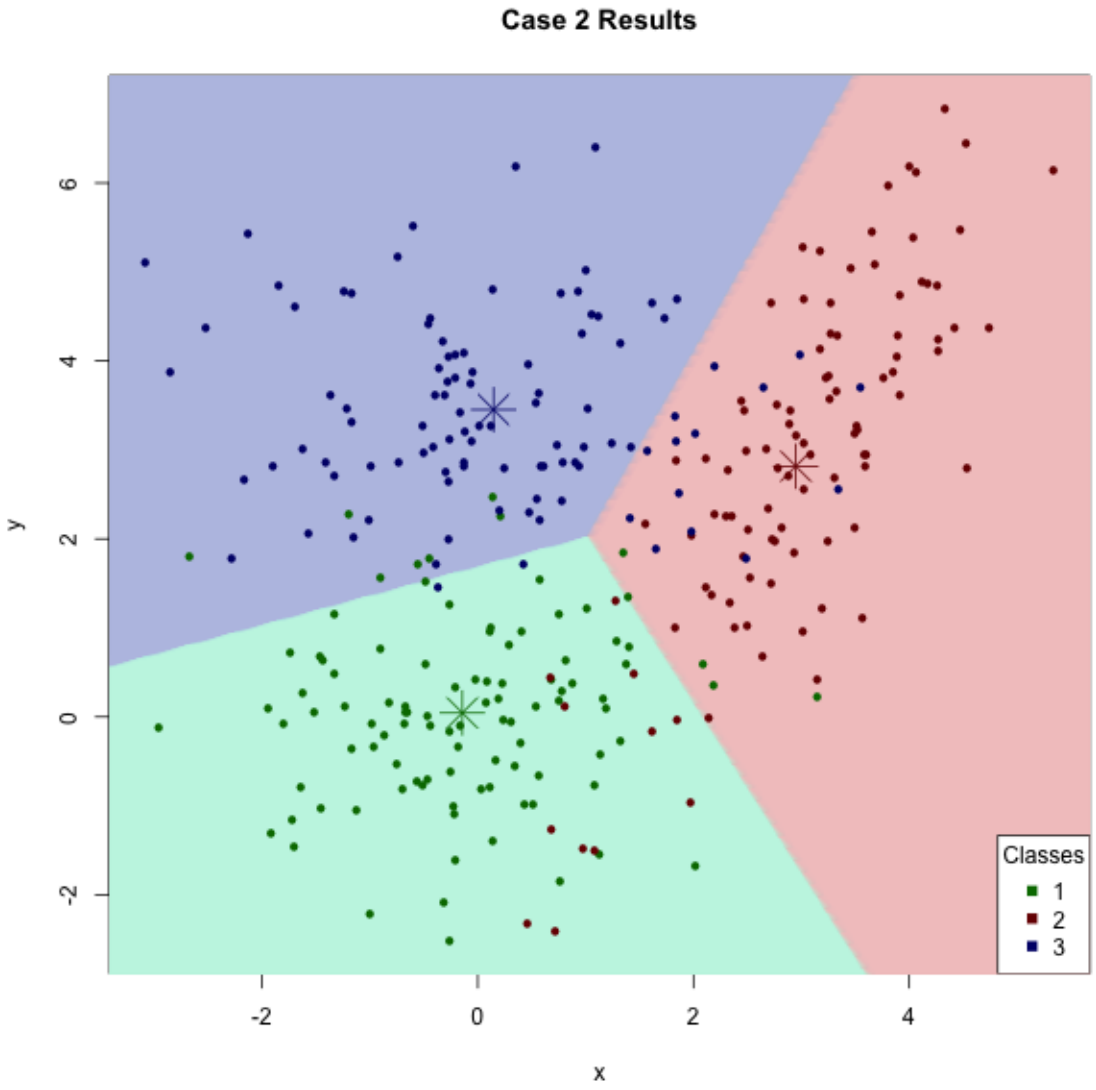

Figure 2: Results for Case 2: $\mathbf{\Sigma}_i = \text{cov}(D)$

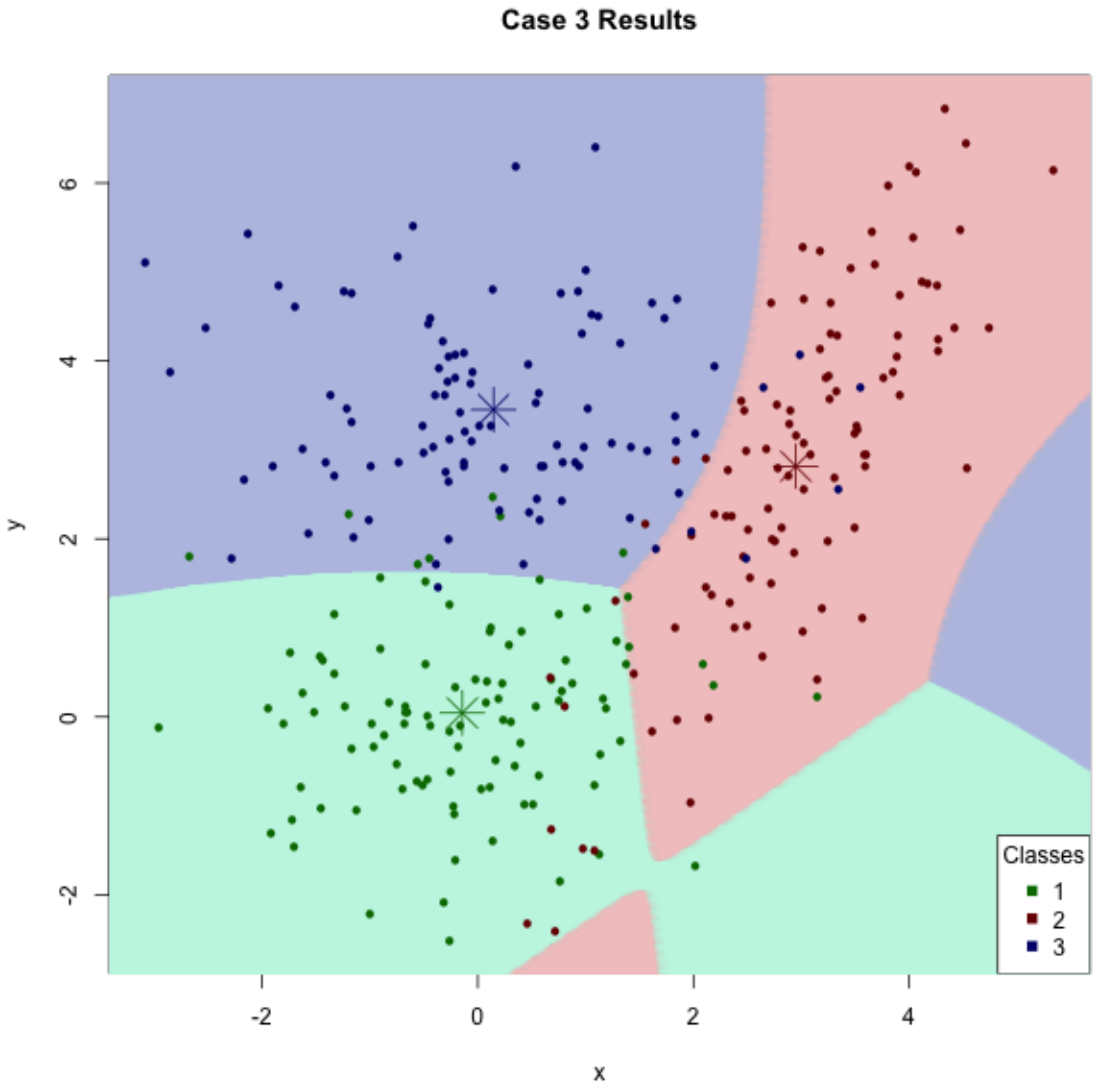

Figure 3: Results for Case 3: $\mathbf{\Sigma}_i = \text{cov}(D_i)$