APPENDIX A

Performing Linear Discriminant Analysis in R

OBJECTIVE

The main objective of this appendix is to show you step by step, how to implement Linear Discriminant Analysis (LDA) using R. The R built-in dataset `iris` will be used to illustrate how the calculations are done. `MASS` is the R package that provides the function needed to perform LDA, and `ggplot2` is the package I will use for visualizing the analysis results.

Linear discriminant analysis is a method you may need if you have a set of predictor variables and want to use them to guide the classification of records into two or more predefined classes. This appendix provides a step-by-step example of how to perform linear discriminant analysis in R. An R script, which you can download using the link https://bit.ly/3MPCbJn will be discussed one chunk of code at a time, throughout this appendix.

I am going to use the R built-in dataset `iris` to illustrate the LDA implementation in R.

Step 1: Load Necessary Libraries

First, I must load two R packages that I will need for this example. These are the `MASS` and `ggplot2` packages. You may need to first install these packages if they are not yet installed in your system. The loading of these 2 packages are done as follows:

```
library(MASS)
library(ggplot2)
```

Step 2: Load the Data

The following code shows how to load and view the `iris` dataset:

```
#attach iris dataset to make it easy to work with
attach(iris)

#view structure of dataset
str(iris)

'data.frame': 150 obs. of  5 variables:
 $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
 $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
 $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
 $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
 $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1..
```

You can see that the dataset contains 5 variables and 150 observations. For this example I will build a linear discriminant analysis model to classify which

species a given flower belongs to, and will use the following predictor variables in the model:

- Sepal.length,
- Sepal.Width,
- Petal.Length,
- Petal.Width.

The prediction model will predict the response variable "Species", which takes one of the following three possible classes:

- setosa,
- versicolor,
- virginica.

Step 3: Scale the Data

Note that whether the predictor variables have the same variance (i.e. are standardized) or not, discriminant analysis will yield the same outcome. Nevertheless, some practitioners like myself, often standardize all predictor variables before they are analyzed. This will generally ensure the stability of many algorithms, because they will not need to process large numbers with a possible loss of precision. The conventional way of standardizing is to center and scale each variable so that it has a mean of 0 and a standard deviation of 1. You can do so in R by using the `scale()` function as follows:

```
#scale each predictor variable (i.e. first 4 columns)
iris[1:4] <- scale(iris[1:4])
```

Step 4: Create Training and Test Samples

Next, I want to split the dataset into a training set to train the model on, and a testing set to test the model on. This is done as follows:

```
#make this example reproducible
set.seed(1)

#Use 70% of dataset for training and the remaining 30% for testing
sample <- sample(c(TRUE, FALSE), nrow(iris), replace=TRUE,
                 prob=c(0.7,0.3))
train <- iris[sample, ]
test <- iris[!sample, ]
```

Step 5: Fit the LDA Model

Next, I want to use the `lda()` function from the MASS package to fit the LDA model to our data:

```
#fit LDA model
model <- lda(Species~., data=train)

#view model output
model

Call:
lda(Species ~ ., data = train)

Prior probabilities of groups:
    setosa versicolor  virginica
 0.3207547  0.3207547  0.3584906

Group means:
           Sepal.Length Sepal.Width Petal.Length Petal.Width
setosa       -1.0397484   0.8131654   -1.2891006  -1.2570316
versicolor    0.1820921  -0.6038909    0.3403524   0.2208153
virginica     0.9582674  -0.1919146    1.0389776   1.1229172

Coefficients of linear discriminants:
                    LD1        LD2
Sepal.Length  0.7922820  0.5294210
Sepal.Width   0.5710586  0.7130743
Petal.Length -4.0762061 -2.7305131
```

```
Petal.Width  -2.0602181  2.6326229

Proportion of trace:
   LD1    LD2
0.9921 0.0079
```

Here is how to interpret the output of the model:

Prior probabilities of group: These represent the proportions of each Species in the training set. For example, 35.8% of all flowers in the training set are of `virginica` species.

Group means: These numbers are the mean values of each predictor variable, broken down by species.

Coefficients of linear discriminants: Each column is a linear combination of the predictor variables. These linear combinations also known as the Linear Discriminants (LD) are used to form decision rules for the LDA model. For example:

- **LD1:** $0.792 \times$ `Sepal.Length` $+ 0.571 \times$ `Sepal.Width` $-$
  $-4.076 \times$ `Petal.Length` $- 2.06 \times$ `Petal.Width`
- **LD2:** $0.529 \times$ `Sepal.Length` $+ 0.713 \times$ `Sepal.Width` $-$
  $-2.731 \times$ `Petal.Length` $+ 2.63 \times$ `Petal.Width`
- **Proportion of trace:** Each of these numbers represents the percentage separation achieved by each linear discriminant function. In other words, it is the proportion of between-class variance that is explained by the discriminant function.

Step 6: Use the Model to Make Predictions

Once you have fit the model using training data, you can use the trained model for making predictions with test data as follows:

```
#use LDA model to make predictions on test data
predicted <- predict(model, test)
```

```
names(predicted)
```

```
[1] "class"     "posterior" "x"
```

The `predict()` function used above willreturn a list containing the following elements:

- **class:** The predicted class,
- **posterior:** The posterior probability that an observation belongs to each class,
- **x:** The linear discriminants.

You can view each of these results for the first six observations in our test dataset:

```
#view predicted class for first six observations in test set
head(predicted$class)
```

```
[1] setosa setosa setosa setosa setosa setosa
Levels: setosa versicolor virginica
```

```
#view posterior probabilities for first six observations in test
set head(predicted$posterior)
```

```
   setosa   versicolor    virginica
4       1 2.425563e-17 1.341984e-35
6       1 1.400976e-21 4.482684e-40
7       1 3.345770e-19 1.511748e-37
15      1 6.389105e-31 7.361660e-53
17      1 1.193282e-25 2.238696e-45
18      1 6.445594e-22 4.894053e-41
```

```
#view linear discriminants for first six observations in test set
head(predicted$x)
```

```
         LD1        LD2
4   7.150360 -0.7177382
6   7.961538  1.4839408
7   7.504033  0.2731178
```

```
15 10.170378  1.9859027
17  8.885168  2.1026494
18  8.113443  0.7563902
```

To see the percentage of observations that the LDA model correctly predicted the Species for, you can execute the following command:

```
#find accuracy of model
mean(predicted$class==test$Species)

[1] 1
```

It turns out that the model correctly predicted `Species` for 100% of the observations in our test dataset.

In the real-world an LDA model will rarely predict every class outcome correctly, but this `iris` dataset is built in a way that machine learning algorithms can understand.

Step 7: Visualize the Results

Lastly, you can create an LDA plot to view the linear discriminants produced by the model and visualize how well it separated the three different species in our dataset. Using the `ggplot()` function, you can do it as follows:

```
#define data to plot
lda_plot <- cbind(train, predict(model)$x)

#create plot
ggplot(lda_plot, aes(LD1, LD2)) +
  geom_point(aes(color = Species))
```

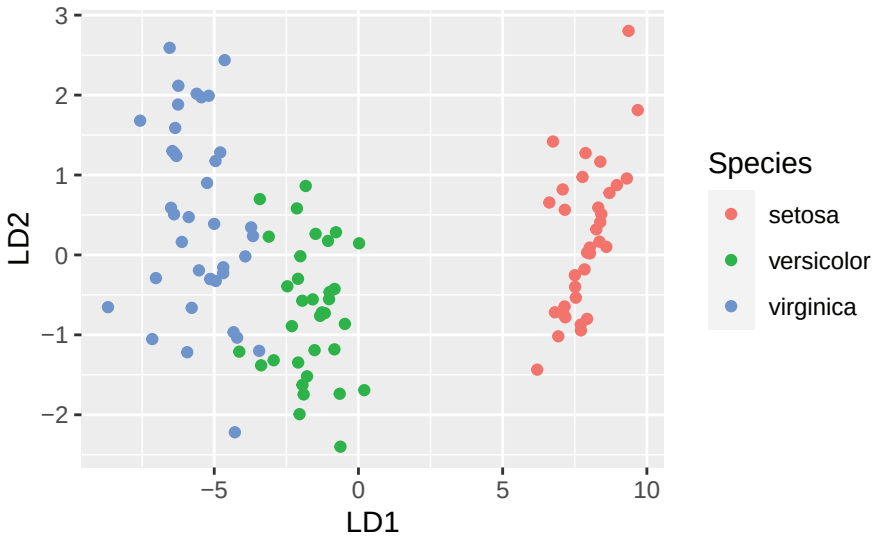

Figure A.1: Scatterplot of the two discriminant functions `LD1` and `LD2`.