Discriminant Analysis in R

* [Data and Required Packages](https://rstudio-pubs-static.s3.amazonaws.com/298913_9bd76dd24a9241cfa112d19a5e50610e.html#data-and-required-packages)
* [Linear Discriminant Analysis](https://rstudio-pubs-static.s3.amazonaws.com/298913_9bd76dd24a9241cfa112d19a5e50610e.html#linear-discriminant-analysis)
* [Quadratic Discriminant Analysis](https://rstudio-pubs-static.s3.amazonaws.com/298913_9bd76dd24a9241cfa112d19a5e50610e.html#quadratic-discriminant-analysis)
* [Exercises](https://rstudio-pubs-static.s3.amazonaws.com/298913_9bd76dd24a9241cfa112d19a5e50610e.html#exercises)

Discriminant Analysis (DA) is a multivariate classification technique that separates objects into two or more mutually exclusive groups based on measurable features of those objects. The measurable features are sometimes called predictors or independent variables, while the classification group is the response or what is being predicted.

DA uses data where the classes are known beforehand to create a model that may be used to predict future observations. It’s useful as an analytic technique trying to understand the relationship between independent variables and a discrete dependent variable. It differs from regression analysis in that the dependent variable must be discrete. It differs from cluster analysis in that the classes must be known beforehand to create the model.

This tutorial will discuss two types of DA and how to leverage R to conduct this type of analysis:

1. **Linear Discriminant Analysis (LDA)** which assumes that the covariance of the independent variables is equal across all classes.
2. **Quadratic Discriminant Analysis (QDA)** which does not assume equal covariance across the classes.

Both LDA and QDA require the number of independent variables to be less than the sample size and both assume multivariate normality among the independent variables. That is, the independent variables come from a normal (or Gaussian) distribution.

Data and Required Packages

We will use Fisher’s famous iris data set to walk through LDA and QDA during this tutorial. This data set is automatically available in R. You can type view(iris) into your console for a spreadsheet view of the data. The iris data set includes length and width measurements (in centimeters) of the sepals and petals from 150 flowers that can be classified into one of three different species: *iris setosa*, *iris versicolor*, and *iris virginica*. The length and width measurements are the independent variables that will be used to predict the flower’s species, the discrete dependent variable.

You will also need to load the following packages:

* tidyverse for data manipulation and visualization
* MASS for LDA and QDA functions
* klaR for partition plot function

**library**(tidyverse)

**library**(MASS)

**library**(klaR)

set.seed(101)

sample\_n(iris, 10)

## Sepal.Length Sepal.Width Petal.Length Petal.Width Species

## 56 5.7 2.8 4.5 1.3 versicolor

## 7 4.6 3.4 1.4 0.3 setosa

## 106 7.6 3.0 6.6 2.1 virginica

## 97 5.7 2.9 4.2 1.3 versicolor

## 37 5.5 3.5 1.3 0.2 setosa

## 44 5.0 3.5 1.6 0.6 setosa

## 85 5.4 3.0 4.5 1.5 versicolor

## 48 4.6 3.2 1.4 0.2 setosa

## 89 5.6 3.0 4.1 1.3 versicolor

## 77 6.8 2.8 4.8 1.4 versicolor

Preparing the Data

After confirming the data set is available and the packages loaded, there is one last step to prepare our data for analysis. That is to separate the data into two subsets: a training set and a testing set. We will use the training set to build our predictive model and then we will use our testing set to evaluate the accuracy of that model. For convenience sake we will use a 60/40 split, using 60% of the data as the training set and the remaining 40% for the testing set.

training\_sample <- sample(c(TRUE, FALSE), nrow(iris), replace = T, prob = c(0.6,0.4))

train <- iris[training\_sample, ]

test <- iris[!training\_sample, ]

Linear Discriminant Analysis

LDA looks for linear combinations of the independent variables to best explain the data and predict the different classes. Discriminant scores are calculated for each observation for each class based on these linear combinations. The scores are calculated using the below equation:

δi(X)=−12μTiΣ−1μi+μTiΣ−1X+ln(πi)δi(X)=−12μiTΣ−1μi+μiTΣ−1X+ln⁡(πi)

where

* δiδi is the discriminant score for class ii
* XX is the matrix of independent variables
* μiμi is a vector containing the means of each variable for class ii
* ΣΣ is the covariance matrix of the variables (assumed to be the same for all classes)
* πiπi is the prior probability that an observation belongs to class ii

A detailed derivation of this equation can be found [here](http://www.real-statistics.com/multivariate-statistics/discriminant-analysis/linear-discriminant-analysis/). Each observation will have a score for each class. The class with the largest score will be the classification prediction for that observation. Below is a visual of what LDA is actually doing. Think of the data as if it were projected onto a linear combination of the variables (or linear discriminant) in one-dimensional space. LDA finds the linear discriminants where the greatest distinction between classes can be observed. In this picture the green line divides the space between the two classes. Observations on one side of the line will be predicted to be from the blue class while observations on the other side will be predicted to be from the red class. Some predictions may be wrong. The more overlap between the classes, the greater the error rate will be.

Applying LDA

Now let’s try using the iris training data set to create a model to predict the species. The lda function from the MASS package does most of the leg work for us. Try using the below code and lets analyze the outputs.

lda.iris <- lda(Species ~ ., train)

lda.iris *#show results*

## Call:

## lda(Species ~ ., data = train)

##

## Prior probabilities of groups:

## setosa versicolor virginica

## 0.3333333 0.2857143 0.3809524

##

## Group means:

## Sepal.Length Sepal.Width Petal.Length Petal.Width

## setosa 5.046429 3.403571 1.478571 0.2607143

## versicolor 5.933333 2.770833 4.295833 1.3125000

## virginica 6.603125 2.984375 5.509375 2.0437500

##

## Coefficients of linear discriminants:

## LD1 LD2

## Sepal.Length 0.5081196 -0.925690

## Sepal.Width 1.9631006 -1.206187

## Petal.Length -2.1214689 1.868896

## Petal.Width -2.9806917 -3.850355

##

## Proportion of trace:

## LD1 LD2

## 0.9901 0.0099

The Prior probabilities of groups show πiπi, the probability of randomly selecting an observation from class ii from the total training set. Because there are 50 observations of each species in the original data set (150 observations total) we know that the prior probabilities should be close to 33.3% for each class.

The Group means shows μiμi, the mean value for each of the independent variables for each class ii.

The Coefficients of linear discriminants are the coefficients for each discriminant. For example the first linear discriminant (LD1) is the linear combination:

(0.51∗Sepal.Length)+(1.96∗Sepal.Width)+(−2.12∗Petal.Length)+(−2.98∗Petal.Width)(0.51∗Sepal.Length)+(1.96∗Sepal.Width)+(−2.12∗Petal.Length)+(−2.98∗Petal.Width)

The Proportions of trace describes the proportion of between-class variance that is explained by successive discriminant functions. As you can see LD1 explains 99%99% of the variance.

Plotting this data using the basicplot function illustrates how the observations are grouped together.

plot(lda.iris, col = as.integer(train$Species))

As you can see, there are three distinct groups with some overlap between *virginica* and *versicolor*. Plotting again, but adding the code dimen = 1will only plot in one dimension (LD1). Think of it as a projection of the data onto LD1 with a histogram of that data.

plot(lda.iris, dimen = 1, type = "b")

These plots illustrate the separation between groups as well as overlapping areas that are potential for mix-ups when predicting classes.

LDA Partition Plots

Using the partimat function from the klaR package provides an alternate way to plot the linear discriminant functions. partimat outouts an array of plots for every combination of two variables. Think of each plot as a different view of the same data. Colored regions delineate each classification area. Any observation that falls within a region is predicted to be from a specific class. Each plot also includes the apparent error rate for that view of the data.

partimat(Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width, data=train, method="lda")

LDA Predictions

Next let’s evaluate the prediction accuracy of our model. First we’ll run the model against the training set used to verify the model fits the data properly by using the command predict. The table output is a confusion matrix with the actual species as the row labels and the predicted species at the column labels.

lda.train <- predict(lda.iris)

train$lda <- lda.train$class

table(train$lda,train$Species)

##

## setosa versicolor virginica

## setosa 28 0 0

## versicolor 0 22 1

## virginica 0 2 31

The total number of correctly predicted observations is the sum of the diagonal. So this model fit the training data correctly for almost every observation. Verifying the training set doesn’t prove accuracy, but a poor fit to the training data could be a sign that the model isn’t a good one.

Now let’s run our test set against this model to determine its accuracy.

lda.test <- predict(lda.iris,test)

test$lda <- lda.test$class

table(test$lda,test$Species)

##

## setosa versicolor virginica

## setosa 22 0 0

## versicolor 0 25 0

## virginica 0 1 18

Based on the earlier plots, it makes sense that a few *iris versicolor* and *iris virginica* observations may be miscategorized. Overall the model performs very well with the testing set with an accuracy of 98.5%.

Quadratic Discriminant Analysis

QDA, like LDA, assumes the independent variables are normally distributed (multivariate normality). Unlike LDA, QDA does not assume equal covariance between the classes. QDA does not find linear combinations of the independent variables, but finds a quadratic function of the independent variables. Below is the equation to calculate discriminant scores:

δi(X)=−12ln(|Σi|)−12(X−μ)TΣ−1i(X−μ)+ln(πi)δi(X)=−12ln⁡(|Σi|)−12(X−μ)TΣi−1(X−μ)+ln⁡(πi)

where

* δiδi is the discriminant score
* XX is the matrix of independent variables
* μμ is a vector containing the means of each variable
* ΣiΣi is the covariance matrix of the variables for class ii
* πiπi is the prior probability that an observation belongs to class ii

A detailed explanation of this equation can be found [here](http://www.real-statistics.com/multivariate-statistics/discriminant-analysis/quadratic-discriminant-analysis/). Like LDA, the class with the largest discriminant score will be the prediction for a given observation. The below image depicts the difference between how LDA and QDA discriminant functions separate the classes.

Applying QDA

The function qda from the MASS package is used. The output looks very similar to the lda output with the same prior probabilities and class means, but there are no linear discriminants calculated with QDA.

qda.iris <- qda(Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width, train)

qda.iris *#show results*

## Call:

## qda(Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width,

## data = train)

##

## Prior probabilities of groups:

## setosa versicolor virginica

## 0.3333333 0.2857143 0.3809524

##

## Group means:

## Sepal.Length Sepal.Width Petal.Length Petal.Width

## setosa 5.046429 3.403571 1.478571 0.2607143

## versicolor 5.933333 2.770833 4.295833 1.3125000

## virginica 6.603125 2.984375 5.509375 2.0437500

QDA Partition Plots

Using the partimat function again provides a way to plot the quadratic discriminant functions. The only difference in code from the LDA section [above](https://rstudio-pubs-static.s3.amazonaws.com/298913_9bd76dd24a9241cfa112d19a5e50610e.html#lda-partition-plots) is replacing method="lda" with method="qda". These plots provide a good visual of the difference between the linear functions used in [LDA](https://rstudio-pubs-static.s3.amazonaws.com/298913_9bd76dd24a9241cfa112d19a5e50610e.html#lda-partition-plots) and the quadratic functions used in QDA. Again, colored regions delineate each classification area. Any observation that falls within a region is predicted to be from a specific class. Each plot also includes the apparent error rate for that view of the data.

partimat(Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width, data=train, method="qda")

QDA Predictions

The predict function works exactly the same way as before and can be used to create a confusion matrix for the training data.

qda.train <- predict(qda.iris)

train$qda <- qda.train$class

table(train$qda,train$Species)

##

## setosa versicolor virginica

## setosa 28 0 0

## versicolor 0 23 1

## virginica 0 1 31

This model fit the training data very well. Again verifying the training set does not prove accuracy, but a poor fit could be a sign that the model isn’t a good one. Next we’ll generate the same confusion matrix with the test data.

qda.test <- predict(qda.iris,test)

test$qda <- qda.test$class

table(test$qda,test$Species)

##

## setosa versicolor virginica

## setosa 22 0 0

## versicolor 0 24 0

## virginica 0 2 18

The model correctly predicted the class of observations 97% of the time. This is slightly less accurate than the LDA model which is not always the case, but overall still quite impressive.

Exercises

1. What variable has the least effect on LD1? ***Hint:*** *Which variable has the smallest LD1 coefficient?*
2. Try removing this variable and rerunning the model. ***Hint:*** *Try replacing the* ~ . *in the line of code* lda.iris <- lda(Species ~ ., train)*with something like* ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width
3. What is the accuracy of this new model?
4. Continue removing variables until there is only one left. How accurate is LDA with a single variable?