Lab 8 Discriminant Analysis (DA)

The goal of this lab is to become familiar with **Discriminant Analysis (DA)**. DA allows you to determine what variables are important in separating groups. It is an eigenanalysis that reduces dimensionality of multivariate data in to new “canonical axes”.

# Set up R session

## Data

Today you will be using the iris data set found in the package datasets in R. To access this data set, simply type:

iris

To learn more about the data set:

`?`(iris)

## Download packages

We will be using the following packages:

library(MASS)  
library(candisc)  
library(ade4)  
library(vegan)

# Testing the assumptions of DA

## Homogeneity of within-group variance-covariance matrices

We will use the Fligner-Killeen test of homogeneity of variances. This test has been shown to be robust to departures from normality (which most of our data does to some extent)

`?`(fligner.test)

You want to test the variance of each variable (n=4) across all three groups (i.e.’ species). **Remember, you don’t want there to be significant differences**

fligner.test(iris$Sepal.Length, iris$Species)  
fligner.test(iris$Sepal.Width, iris$Species)  
fligner.test(iris$Petal.Length, iris$Species)  
fligner.test(iris$Petal.Width, iris$Species)

Should you transform the iris variables?

If you think yes, use the code below. If you think no, skip ahead.

log <- cbind.data.frame(apply(iris[, 1:4] + 1, 2, log), iris$Species)  
names(log)[5] <- "Species"

Re-run the Fligner-Killeen test on the transformed data:

fligner.test(log$Sepal.Length, log$Species)  
fligner.test(log$Sepal.Width, log$Species)  
fligner.test(log$Petal.Length, log$Species)  
fligner.test(log$Petal.Width, log$Species)

## Multivariate Normality

To look for multivariate normality, we would look at the distribution of each variable for each group. However, (hint for previous section) you may have already transformed your data and not need this step in this case.

## Multicolinearity

To test for multicolinearity, you should look at all pairwise correlations. Remember, correlations > 0.7 can be trouble.

cor(iris[, 1:4])

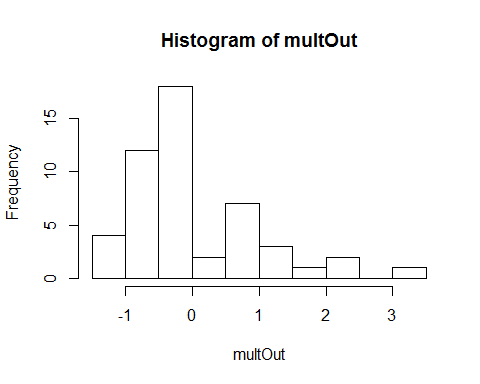
## Sepal.Length Sepal.Width Petal.Length Petal.Width  
## Sepal.Length 1.0000000 -0.1175698 0.8717538 0.8179411  
## Sepal.Width -0.1175698 1.0000000 -0.4284401 -0.3661259  
## Petal.Length 0.8717538 -0.4284401 1.0000000 0.9628654  
## Petal.Width 0.8179411 -0.3661259 0.9628654 1.0000000

We have some co-linearity. Let’s move ahead using all four variables, but we will also run the analysis without Petal.Length.

## Outliers

Outliers can have a large influence on canonical axes in DA. You are going to take a multivariate approach to identifying outliers (we did this in lab 2):

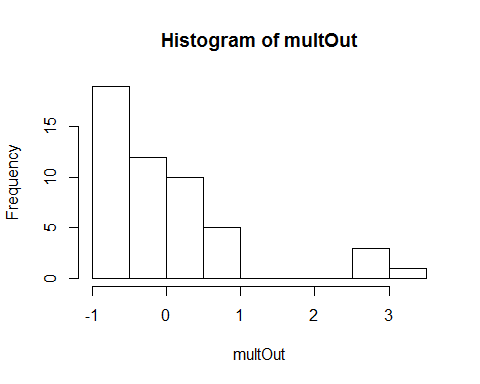
# Calculate a withi-group distance matrix:  
  
brayDist <- vegdist(iris[1:50, 1:4], "bray")  
  
# Calculate the average distance of each sample to all other samples (i.e.  
# column average) and turn the means in z-scores:  
  
multOut <- scale(colMeans(as.matrix(brayDist)))  
  
# Now look at a histogram of these data to identify samples that are > 3 sd  
# from the mean:  
  
hist(multOut)



# and get the number of those samples:  
  
multOut[multOut > 3, ]

## 42   
## 3.428475

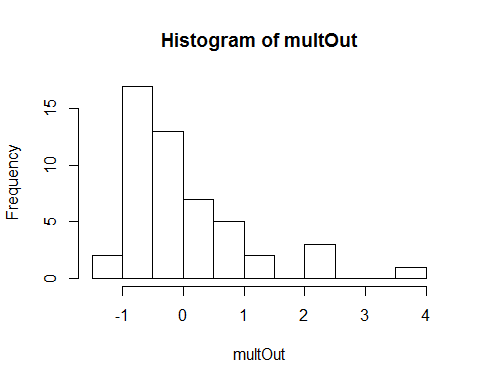
# Repeat for the other two groups (i.e.,species):  
  
brayDist <- vegdist(iris[51:100, 1:4], "bray")  
multOut <- scale(colMeans(as.matrix(brayDist)))  
hist(multOut)



multOut[multOut > 3, ]

## 61   
## 3.07037

brayDist <- vegdist(iris[101:150, 1:4], "bray")  
multOut <- scale(colMeans(as.matrix(brayDist)))  
hist(multOut)



multOut[multOut > 3, ]

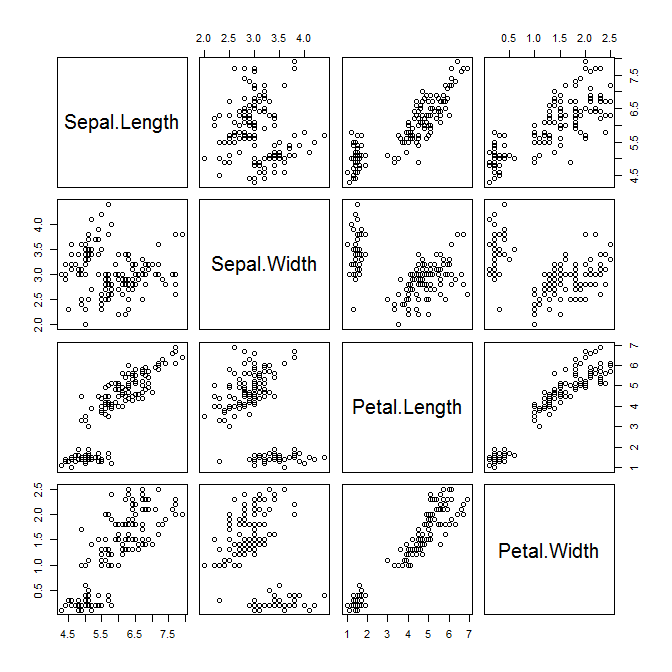
## 107   
## 3.816638

# Finally, make a vector of the outliers to pull out of the data set later:  
  
Outliers <- c(42, 61, 107)

## Linearity

Next we test for linear relationship between variables. This is key to making sure that the variables change in a linear fashion along underlying gradients (i.e. canonical axes):

pairs(iris[, 1:4])



Believe it or not, those pass as linear relationships!

# Discriminant analysis

You will use the LDA function in the *MASS* package to conduct DA and will also use the candisc function in package *candisc*.

`?`(lda)  
  
`?`(candisc)

First run the analysis on the un-transformed data with all of the variables. We are going to split our data set into “training” data and “testing” data. We randomly select 75 samples from the iris data set:

set.seed(11)  
train <- sample(1:150, 75)

Check the frequency of each species in the training data, which will help set your priors:

prior <- table(iris$Sp[train])

Next run the DA for the training data:

iris.LDA <- lda(Species ~ ., iris, prior = cbind(prior/75), subset = train)

**“Species ~ .” is the model and the “dot” stands for all of the variables (so you don’t have to type them all)**

What do the results tell you about the number of meaningful canonical axes and the absolute contribution of each variable to those axes?

# Assessing and interpreting canonical axes

## Relative % Criterion

The “proportion of trace” from the lda output is the **Relative % criterion**.

## Canonical Correlation Criterion

Here, you want to test the correlation between the canonical scores and the grouping variable. First you need to predict the membership (i.e. calculate canonical scores) of each sample in the training set. You will use the predict.lda function from the *MASS* package:

`?`(predict.lda)

iris.LDA.p <- predict(iris.LDA, iris[train, ])

You want to test the canonical scores (iris.LDA.P$x) with the grouping variable (i.e. species):

corTest <- lm(iris.LDA.p$x ~ iris$Sp[train])  
summary(corTest)

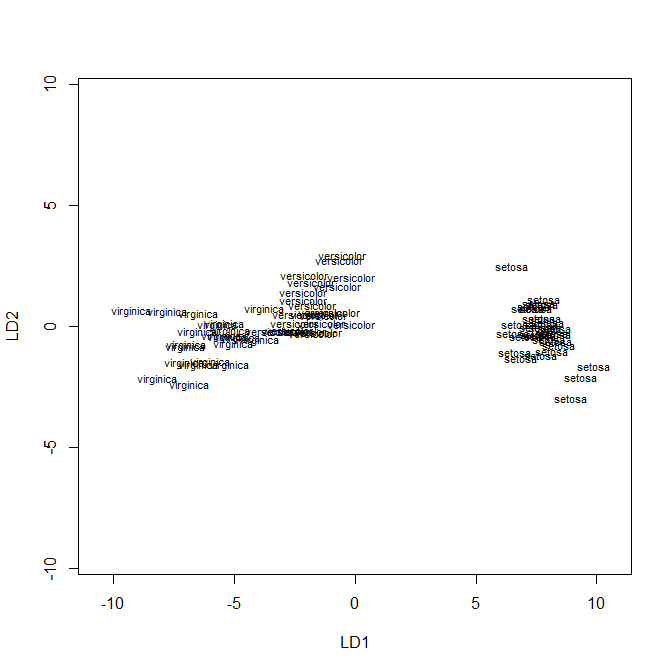
## Response LD1 :  
##   
## Call:  
## lm(formula = LD1 ~ iris$Sp[train])  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.16072 -0.56066 -0.02883 0.55679 2.35039   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.7877 0.1796 37.79 <2e-16 \*\*\*  
## iris$Sp[train]versicolor -9.4464 0.2752 -34.33 <2e-16 \*\*\*  
## iris$Sp[train]virginica -13.8958 0.2826 -49.17 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1 on 72 degrees of freedom  
## Multiple R-squared: 0.9736, Adjusted R-squared: 0.9728   
## F-statistic: 1326 on 2 and 72 DF, p-value: < 2.2e-16

How does the correlation and significance look for each canonical axis?

## Plot canonical scores and axes

Now visually asses the discrimination of the three species of iris by the canonical axes:

plot(iris.LDA, xlim = c(-11, 11), ylim = c(-6, 6))



## Classification accuracy

You now want to measure how well the axes discriminate. The higher the correct classification rate, the greater degree of group discrimination achieved by the canonical axes. We will classify the training data first. Create a classification matrix:

ct <- table(iris[train, ]$Species, predict(iris.LDA, iris[train, ])$class)  
  
# Change to a table of proportions:  
  
pct <- prop.table(ct)  
pct

##   
## setosa versicolor virginica  
## setosa 0.41333333 0.00000000 0.00000000  
## versicolor 0.00000000 0.30666667 0.00000000  
## virginica 0.00000000 0.01333333 0.26666667

# Calculate classification rate by summing the diagonal:  
  
sum(diag(pct))

## [1] 0.9866667

## Interpreting canonical axes (raw coefficients, standardized weights, and structure coefficients)

Unfortunately, lda doesn’t provide an easy way to calculate the standardized weights, and structure coefficients. For this, you will use candisc in the *candisc* package.

`?`(candisc)

For candisc, you build a linear model using lm:

iris.mod <- lm(cbind(Sepal.Length, Sepal.Width, Petal.Length, Petal.Width) ~   
 Species, data = iris[train, ])

And run the candisc function:

iris.can <- candisc(iris.mod, data = iris[train, ])

Next pull the raw coefficients, standardized weights, and structure coefficients from the output:

iris.can$coeffs.raw

## Can1 Can2  
## Sepal.Length 0.8423099 -0.1330735  
## Sepal.Width 1.0422173 2.6272422  
## Petal.Length -2.3499555 -0.4386999  
## Petal.Width -2.8149766 1.9869027

iris.can$coeffs.std

## Can1 Can2  
## Sepal.Length 0.4097631 -0.06473701  
## Sepal.Width 0.3412844 0.86031658  
## Petal.Length -0.8947022 -0.16702689  
## Petal.Width -0.5258196 0.37114068

iris.can$structure

## Can1 Can2  
## Sepal.Length -0.8192511 0.20342162  
## Sepal.Width 0.5464200 0.80742872  
## Petal.Length -0.9916072 0.03274444  
## Petal.Width -0.9806997 0.16929089

Note that the raw coefficients are the same as the lda output:

iris.LDA$scaling

## LD1 LD2  
## Sepal.Length 0.8423099 0.1330735  
## Sepal.Width 1.0422173 -2.6272422  
## Petal.Length -2.3499555 0.4386999  
## Petal.Width -2.8149766 -1.9869027

*notice that the signs are flipped but coefficients are the same*

Looking at the structure coefficients, how would you “define” the first canonical axis?

# Validating canonical axes

A DA is only as good as its ability to classify new data correctly. Here, you are going to use the “testing” data from your split sample to see how well the canonical axes predict group membership of “new” samples. You are going to use the predict.lda function again. This time with the testing data (i.e., iris[-train, ])

iris.LDA.new <- predict(iris.LDA, iris[-train, ])

Next, make a classification table and calculate the classification rate:

ynew.table <- table(iris[-train, ]$Species, iris.LDA.new$class)  
ynew.table

##   
## setosa versicolor virginica  
## setosa 19 0 0  
## versicolor 0 25 2  
## virginica 0 1 28

sum(diag(prop.table(ynew.table)))

## [1] 0.96

How did the canonical axes handle the new data?

# MANOVA, *the other side of the coin of DA*

In MANOVA, we are interested if groups differ in their measured variables. In DA, we are interested in a linear combination of variables that maximize differences between groups. In this case, MANOVA can provide a test of whether or not the groups are different.

`?`(manova)

Y <- as.matrix(iris[, 1:4])  
Sp <- factor(iris[, 5])  
  
fit <- manova(Y ~ Sp)  
summary(fit, test = "Wilks")

## Df Wilks approx F num Df den Df Pr(>F)   
## Sp 2 0.023439 199.15 8 288 < 2.2e-16 \*\*\*  
## Residuals 147   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

## Post-hoc tests:

Yset <- as.matrix(iris[1:50, 1:4])  
Yversi <- as.matrix(iris[51:100, 1:4])  
Yvirg <- as.matrix(iris[101:150, 1:4])  
Sp <- factor(iris[, 5])  
  
fit1 <- manova(rbind(Yset, Yversi) ~ Sp[1:100])  
summary(fit1, test = "Hotelling-Lawley")

## Df Hotelling-Lawley approx F num Df den Df Pr(>F)   
## Sp[1:100] 1 26.335 625.46 4 95 < 2.2e-16 \*\*\*  
## Residuals 98   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

fit2 <- manova(rbind(Yversi, Yvirg) ~ Sp[51:150])  
summary(fit2, test = "Hotelling-Lawley")

## Df Hotelling-Lawley approx F num Df den Df Pr(>F)   
## Sp[51:150] 1 3.6273 86.148 4 95 < 2.2e-16 \*\*\*  
## Residuals 98   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

fit3 <- manova(rbind(Yset, Yvirg) ~ Sp[-c(51:100)])  
summary(fit3, test = "Hotelling-Lawley")

## Df Hotelling-Lawley approx F num Df den Df Pr(>F)   
## Sp[-c(51:100)] 1 49.792 1182.6 4 95 < 2.2e-16 \*\*\*  
## Residuals 98   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

If you have time, now run the analysis on the log transformed data without the variable “Petal.Length” and the outliers. **Remove the Petal.Length column and outliers before analyzing**.

Does meeting the assumptions of DA change our interpretation of our results?