Discriminant analysis - Practice 4

José Luis Romero Béjar Guillermo Arturo Cañadas De la Fuente

Junio, 2024

© This material is licensed under a **Creative Commons CC BY-NC-ND** attribution which allows *works to be downloaded and shared with others, as long as they are referenced, but may not be modified in any way or used commercially.*

This practice illustrates an example of classification with a **linear discriminant** model and another example with a **quadratic classifier**.

To carry out this practice you must download the following files available on the PRADO platform of the course:

- *DA_4_en.Rmd*

This brief guide is intended to familiarize the reader with the following:

- R packages required.
- Graphical exploration of data.
- Assumptions: normality and homogeneity of variance.
- Discriminant functions.
- Model validation.
- Visualization of the classifications.

Loading/installation of R packages necessary for this practice.

The following source code module is responsible for loading, if they are already installed, all the packages that will be used in this R session. While an R package can be loaded at any time when it is to be used, it is advisable to optimize its calls with this code chunk at the beginning.

Loading a package into an R session **requires it to be already installed**. If it is not, the first step is to run the sentence:

install.packages("name_of_the_library")

```
########################################
# Loading necessary packages and reason #
########################################

# This is an example of the first installation of a package
# Only runs once if the package is not installed
# Once it is installed this sentence has to be commented (not to run again)
# install.packages("ggplot2")

# Package required to call 'ggplot' function
library(ggplot2)

# Package required to call 'ggarrange' function
library(ggpubr)
```

```r
# Package required to call 'scatterplot3d' function
library(scatterplot3d)

# Package required to call 'melt' function
library(reshape2)

# Package required to call 'mvn' function
library(MVN)

# Package required to call 'boxM' function
library(biotools)

# Package required to call 'partimat' function
library(klaR)

# Package required to call 'summarise' function
library(dplyr)

# Package required to call 'createDataPartition' function
library(caret)
```

Linear discriminant analysis

A team of biologists wants to generate a statistical model that allows classifying which species, a or b, a certain insect belongs to based on the length of its legs, the diameter of its abdomen and that of its sexual organ.

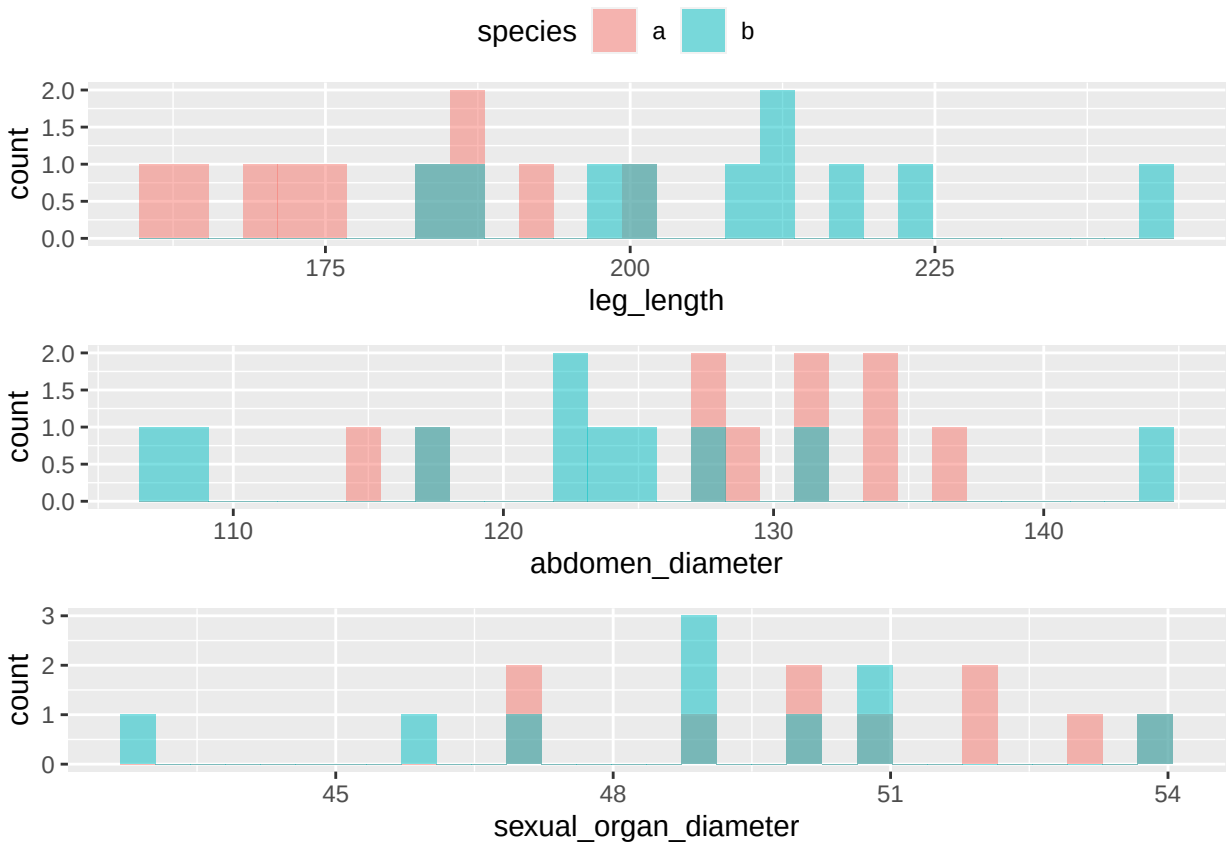
As **training data** these three variables (**length of the legs, diameter of the abdomen and diameter of the sexual organ** in millimeters) have been measured in 10 individuals of each of the two species.

```r
# Response variable
species<-c("a","a","a","a","a","a","a","a","a","a","b","b","b","b","b","b","b","b","b","b")

# Explanatory variables
leg_length<-c(191,185,200,173,171,160,188,186,174,163,186,211,201,242,184,211,
              217,223,208,199)
abdomen_diameter<-c(131,134,137,127,128,118,134,129,131,115,107,122,144,131,108,
                    118,122,127,125,124)
sexual_organ_diameter<-c(53,50,52,50,49,47,54,51,52,47,49,49,47,54,43,51,49,51,
                         50,46)

# The whole dataset
datos<-data.frame(species,leg_length,abdomen_diameter,sexual_organ_diameter)

# The response variable has to be an object of the class 'factor' of R language
datos$species<-as.factor(datos$species)
```

Graphical exploration of data

First, we explore how well (or poorly) **each of the explanatory variables considered independently classifies the species**. To do this, we draw the superimposed histograms. If **the histograms are separated**, the variable considered would be a **good individual classifier** for the species.

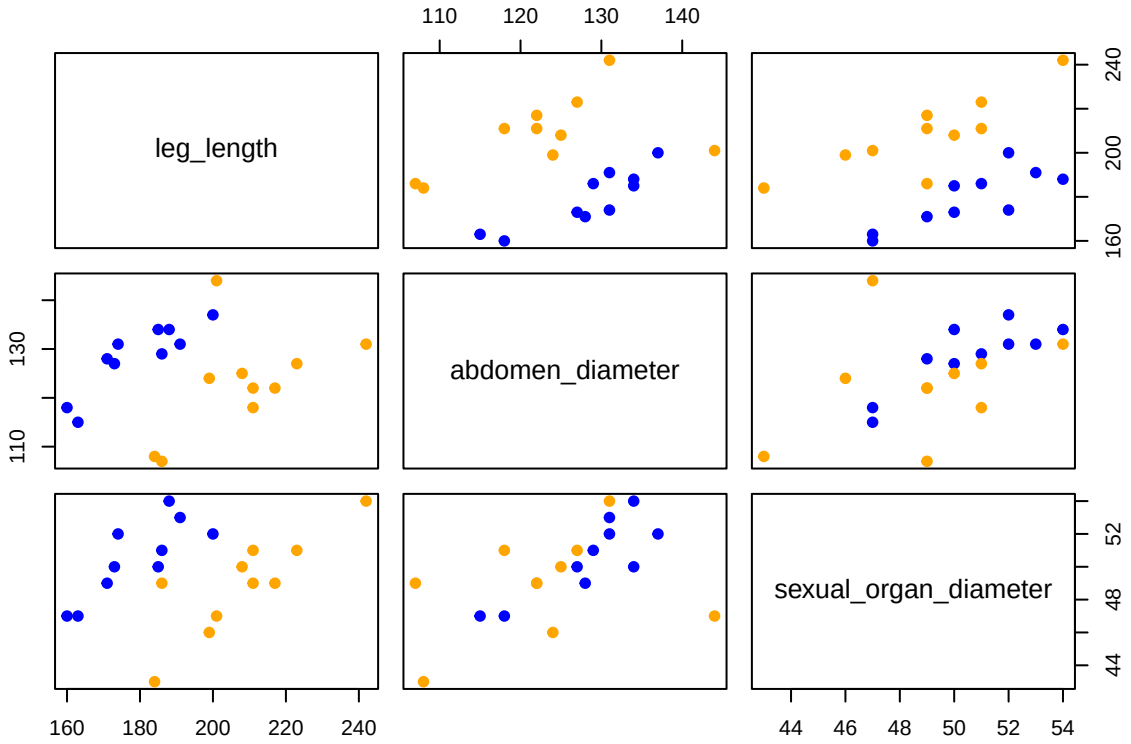
```
p1 <- ggplot(data = datos, aes(x = leg_length, fill = species)) +
      geom_histogram(position = "identity", alpha = 0.5)
p2 <- ggplot(data = datos, aes(x = abdomen_diameter, fill = species)) +
      geom_histogram(position = "identity", alpha = 0.5)
p3 <- ggplot(data = datos, aes(x = sexual_organ_diameter, fill = species)) +
      geom_histogram(position = "identity", alpha = 0.5)

ggarrange(p1, p2, p3, nrow = 3, common.legend = TRUE)
```



In this sense, it seems that the variable **leg__length** is the one that best differentiates between species (least overlapping).

Next, we explore which **pairs of variables** best separate between species. We draw the bivariate scatterplots. If the scatterplots separate the colors assigned to each species well, it means that the pair of variables would provide a good classifier model based on them.

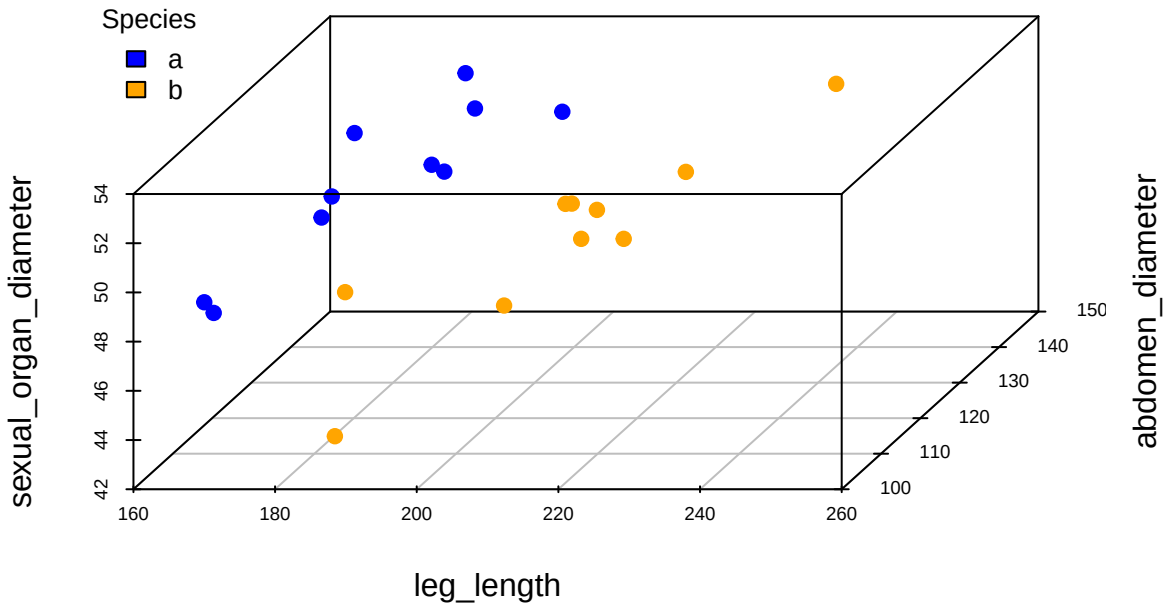
```
pairs(x = datos[, c("leg_length","abdomen_diameter","sexual_organ_diameter")],
      col = c("blue", "orange")[datos$species], pch = 19)
```



The combination of variables **abdomen_diameter - leg_length** and **leg_length - sexual_organ_diameter** seem to adequately separate between species.

Finally, we explore whether the **three variables** together adequately differentiate between the two species. We draw the three-dimensional scatterplot trying to identify if the graphical representation separates the species according to the colors assigned to each of them.

```
scatterplot3d(datos$leg_length, datos$abdomen_diameter, datos$sexual_organ_diameter,
              color = c("blue", "orange")[datos$species], pch = 19,
              grid = TRUE, xlab = "leg_length", ylab = "abdomen_diameter",
              zlab = "sexual_organ_diameter", angle = 65, cex.axis = 0.6)
legend("topleft",
       bty = "n", cex = 0.8,
       title = "Species",
       c("a", "b"), fill = c("blue", "orange"))
```



Indeed, the three variables simultaneously perfectly separate the two species in the three-dimensional space generated. It makes sense to consider building a discriminant model for classification, for example.

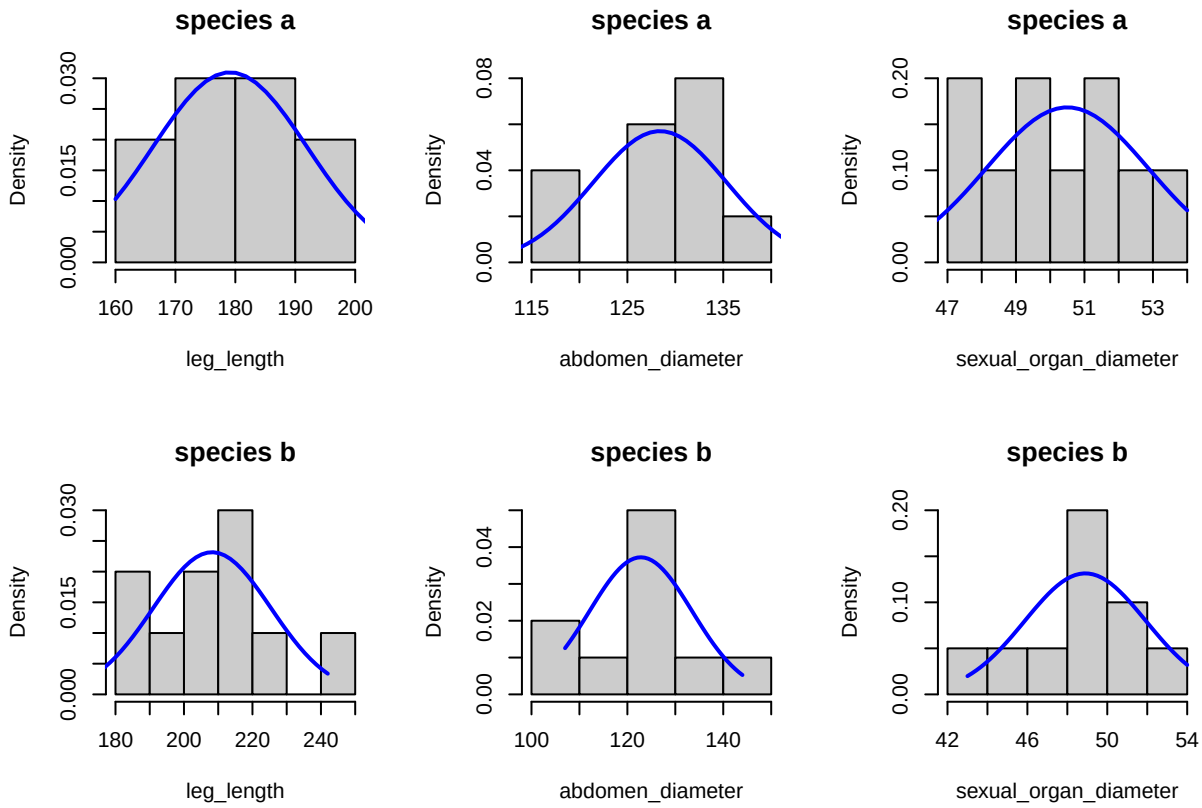
Univariate and multivariate normality

Univariate normality

Next we make a graphical exploration of the **normality** of the **univariate distributions** of our predictors by representing the **histograms** and the **qqplots**.

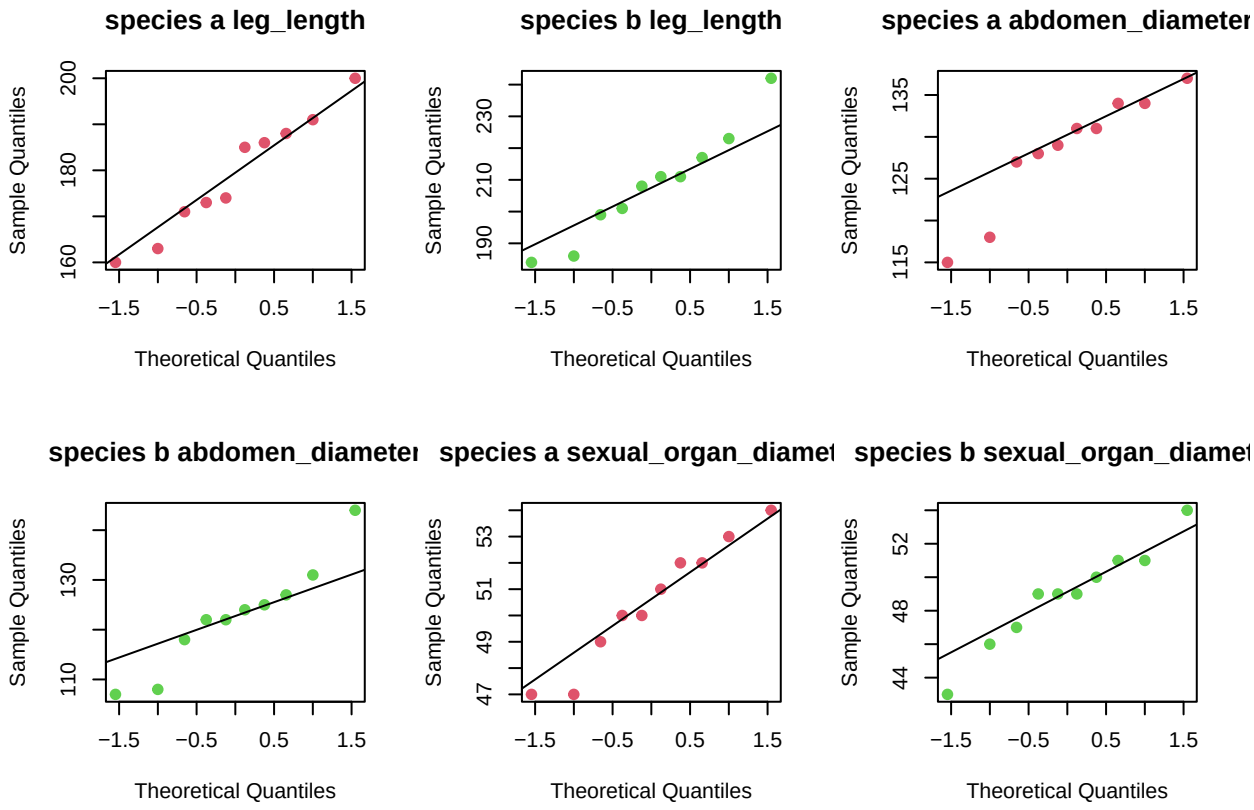
Univariate histograms

```
# Histogram representation of each variable for each species
par(mfcol = c(2, 3))
for (k in 2:4) {
  j0 <- names(datos)[k]
  x0 <- seq(min(datos[, k]), max(datos[, k]), le = 50)
  for (i in 1:2) {
    i0 <- levels(datos$species)[i]
    x <- datos[datos$species == i0, j0]
    hist(x, proba = T, col = grey(0.8), main = paste("species", i0), xlab = j0)
    lines(x0, dnorm(x0, mean(x), sd(x)), col = "blue", lwd = 2)
  }
}
```



Qqplot graphics

```
# Representation of normal quantiles of each variable for each species
par(mfrow=c(2,3))
for (k in 2:4) {
  j0 <- names(datos)[k]
  x0 <- seq(min(datos[, k]), max(datos[, k]), le = 50)
  for (i in 1:2) {
    i0 <- levels(datos$species)[i]
    x <- datos[datos$species == i0, j0]
    qqnorm(x, main = paste("species", i0, j0), pch = 19, col = i + 1)
    qqline(x)
  }
}
```



This exploratory analysis can give us an idea of the possible normal distribution of the univariate variables, but it is always better to do the respective normality tests.

Univariate normality test (Shapiro-Wilk)

The **null hypothesis** that the data **follow a univariate normal distribution** is tested. This hypothesis is **rejected** if the **p-value** given by the Shapiro-Wilk test is **less than 0.05**. Otherwise the assumption of normality of the data is not rejected.

```
datos_tidy <- melt(datos, value.name = "value")
aggregate(formula = value ~ species + variable, data = datos_tidy,
          FUN = function(x){shapiro.test(x)$p.value})
```

```
##   species              variable     value
## 1       a           leg_length 0.7763034
## 2       b           leg_length 0.7985711
## 3       a      abdomen_diameter 0.1845349
## 4       b      abdomen_diameter 0.5538213
## 5       a sexual_organ_diameter 0.6430844
## 6       b sexual_organ_diameter 0.8217855
```

No evidence of lack of univariate normality is found (p-value > 0.05).

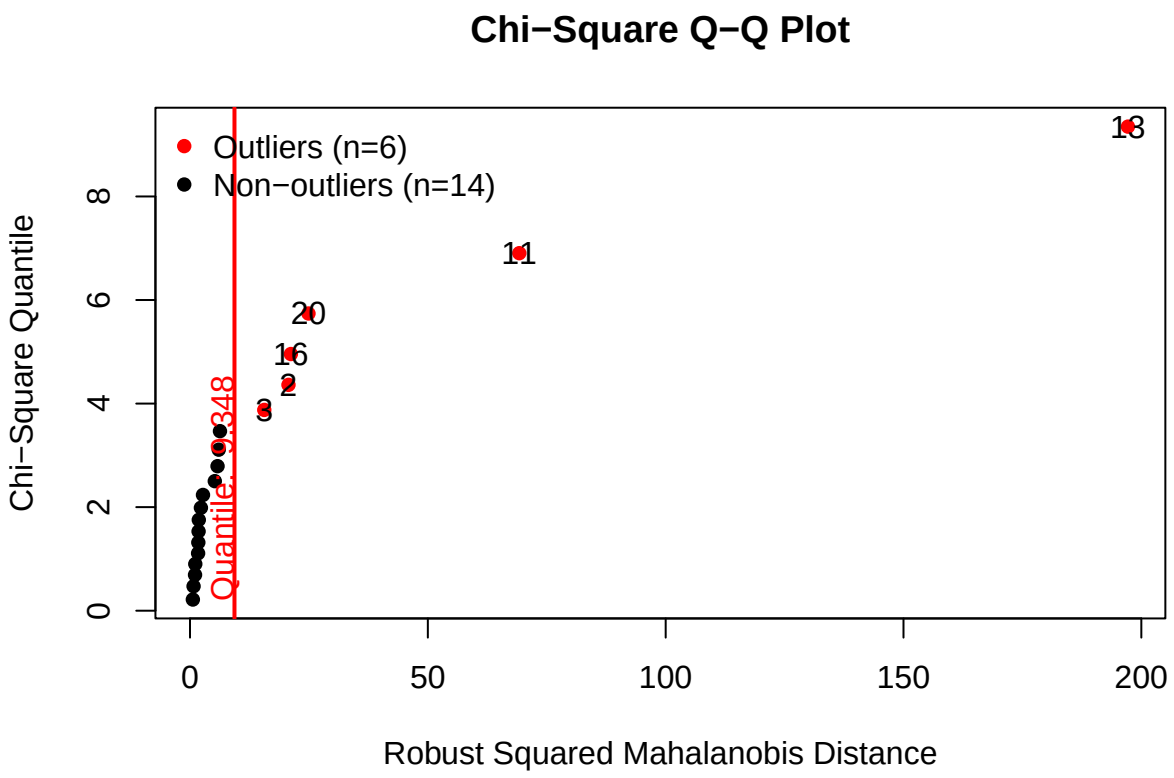
Multivariate normality

Multivariate normality test (Mardia, Henze-Zirkler and Royston)

The **MVN** package contains functions that allow you to perform the three tests that are commonly used to **test multivariate normality**.

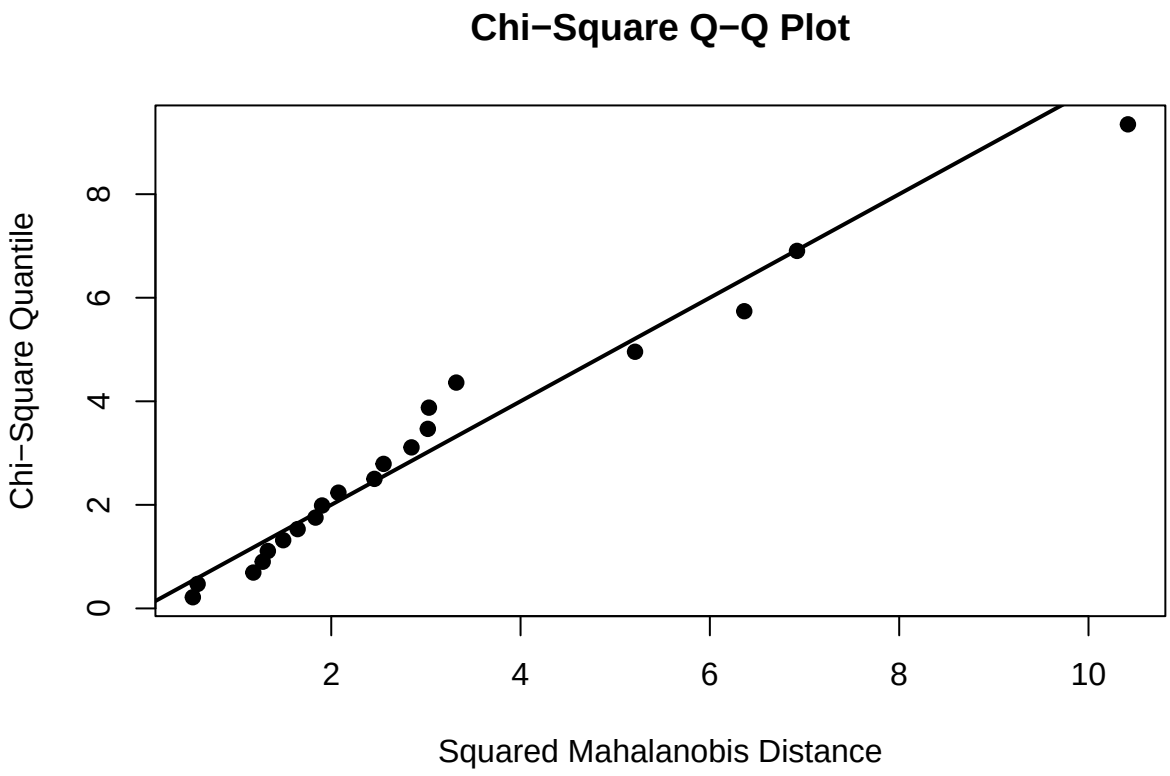
This multivariate normality can be affected by the presence of multivariate outliers. In this package we also find **functions for the analysis of multivariate outliers**.

```
outliers <- mvn(data = datos[,-1], mvnTest = "hz", multivariateOutlierMethod = "quan")
```



Six outliers are detected in observations 2, 3, 11, 13, 16 and 20. However, none of the two tests carried out below find evidence of a lack of multivariate normality at the 5% level.

```
# Royston multivariate normality test
royston_test <- mvn(data = datos[,-1], mvnTest = "royston", multivariatePlot = "qq")
```




```
royston_test$multivariateNormality
```

```
##      Test         H   p value MVN
## 1 Royston 0.4636176 0.9299447 YES
```

```
# Henze-Zirkler multivariate normality test
hz_test <- mvn(data = datos[,-1], mvnTest = "hz")
hz_test$multivariateNormality
```

```
##            Test        HZ    p value MVN
## 1 Henze-Zirkler 0.7870498 0.07666139 YES
```

Homogeneity of variance

For the analysis of homogeneity of variance:

- When there is **a single predictor** the most recommended test is the **Barttlet test**, already used in previous practices.
- When **multiple predictors** are used, it must be verified that the covariance matrix is constant in all groups. In this case it is also advisable to check the homogeneity of the variance for each predictor at the individual level. The most recommended test is the **Box M test**, which is an extension of the Barttlet test for multivariate scenarios. It must be taken into account that it is **very sensitive to whether the data are actually distributed according to a multivariate normal**. For this reason, it is recommended to use a significance (p-value) <0.001 to reject the null hypothesis.

The **null hypothesis** to be tested is that of **equality of covariance matrices** in all groups.

```r
boxM(data = datos[, 2:4], grouping = datos[, 1])
```

```
##
##  Box's M-test for Homogeneity of Covariance Matrices
##
## data:  datos[, 2:4]
## Chi-Sq (approx.) = 9.831, df = 6, p-value = 0.132
```

In this case we do not reject the null hypothesis since **p-value = 0.132 > 0.001** and therefore **we assume homogeneity of variances**.

It is important to remember that for this **conclusion to be reliable** the assumption of **multivariate normality** must be met, which, in this case, was the case.

Discriminant function

Given that the **assumptions of multivariate normality and homogeneity of variance are satisfied**, it makes sense to **fit a linear discriminant** classification model.

The **lda** function of the **MASS** package adjusts this model.

```r
modelo_lda <- lda(formula = species ~ leg_length + abdomen_diameter + sexual_organ_diameter,data = datos
modelo_lda
```

```
## Call:
## lda(species ~ leg_length + abdomen_diameter + sexual_organ_diameter,
##     data = datos)
##
## Prior probabilities of groups:
##   a   b
## 0.5 0.5
##
## Group means:
##   leg_length abdomen_diameter sexual_organ_diameter
## a      179.1            128.4                  50.5
## b      208.2            122.8                  48.9
##
## Coefficients of linear discriminants:
##                              LD1
## leg_length             0.13225339
## abdomen_diameter      -0.07941509
## sexual_organ_diameter -0.52655608
```

The output of this object shows us the **prior probabilities** of each group, in this case 0.5 for each species (10/20), the **means of each regressor per group** and the **coefficients of the linear discriminant classification model**, which in this case would have the form:

odds = 0.13225339leg_length - 0.07941509abdomen_diameter - 0.52655608sexual_organ_diameter

Once the classifier is built, we can classify new data based on its measurements by simply calling the **predict** function. For example, let's consider a new specimen whose measurements are: *leg_length = 194, abdomen_diameter = 124, sexual_organ_diameter = 49.*

```r
nuevas_observaciones <- data.frame(leg_length = 194, abdomen_diameter = 124,sexual_organ_diameter = 49)
predict(object = modelo_lda, newdata = nuevas_observaciones)
```

```
## $class
## [1] b
```

```
## Levels: a b
##
## $posterior
##            a         b
## 1 0.05823333 0.9417667
##
## $x
##         LD1
## 1 0.5419421
```

According to the discriminant function, the posterior probability that the specimen is of species **b** is 94.2% while that of it is of species **a** is only 5.8%. Therefore this specimen would be classified in species **b**. Similarly, taking $\mathbf{p = 0.5}$ as *cut-point*, given that the discriminant function takes the value **0.5419421**, it would also invite classification into the species **b**.

Model validation

A first basic form of **model validation** consists of trying to see how well it classifies in a test set from which we know the species to which each record belongs. The usual way to proceed is to divide the original dataset into two subsets: the first, known as **training set**, with approximately 80% of the records, which will be used to fit the model; and the second, known as **test set**, with 20% of the remaining records, which will be used for cross-validation.

For this illustration, however, we take the entire dataset as the training and test set, given that we have only 20 records.

The **confusionmatrix** function of the **biotools** package performs cross-validation of the classification model.

```
pred <- predict(modelo_lda, dimen = 1)
confusionmatrix(datos$species, pred$class)
```

```
##   new a new b
## a    10     0
## b     0    10
```

```
# Classification error percentage
trainig_error <- mean(datos$species != pred$class) * 100
paste("trainig_error=", trainig_error, "%")
```

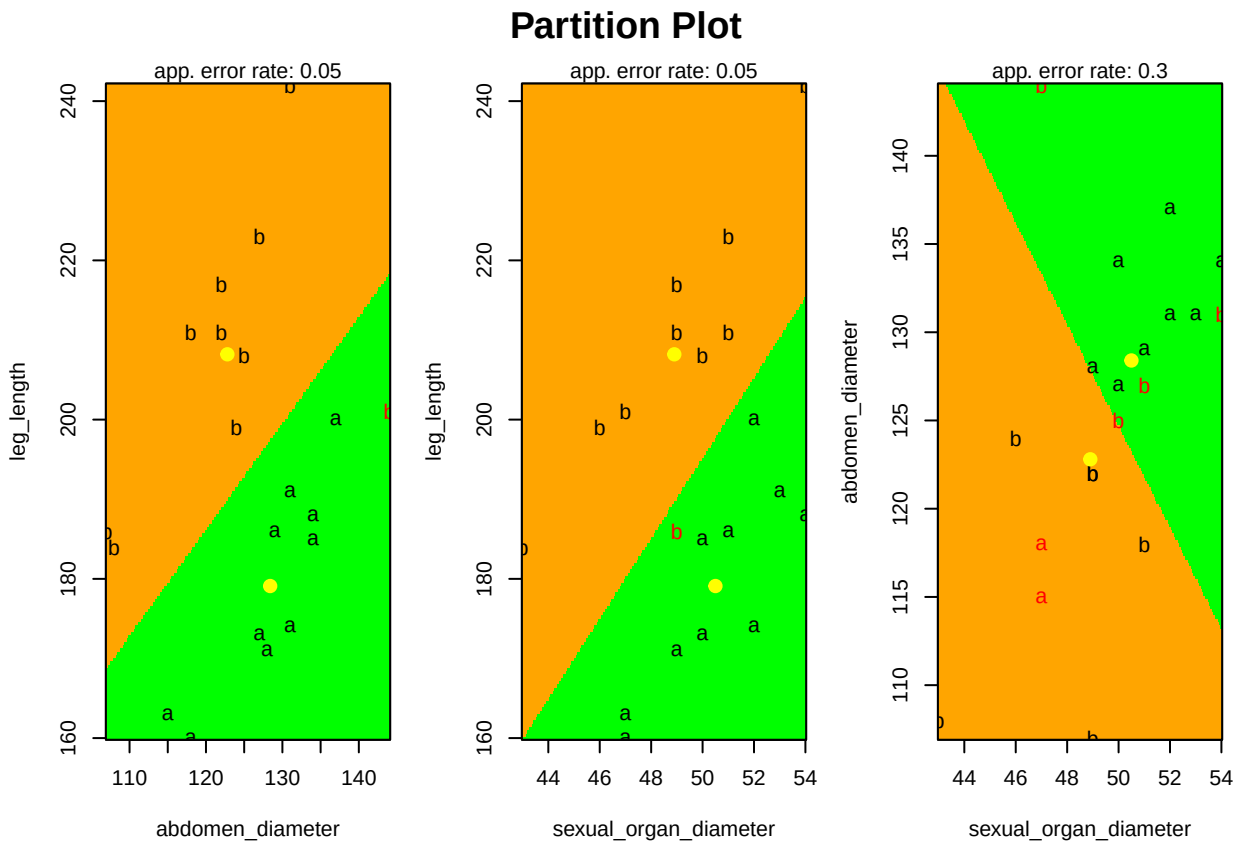
```
## [1] "trainig_error= 0 %"
```

In this case the **correct classifications rate is 100%** (this is not usual).

Displaying rankings

From a geometric point of view, linear discriminant analysis separates space using a straight line. In this sense, the **partimat** function of the **klaR** package allows us to represent the **classification limits** of a linear or quadratic discriminant model for each pair of predictors. Each color represents a classification region according to the model, **the centroid** of each region and the **real value of the observations** are shown.

```
partimat(species ~ leg_length + abdomen_diameter + sexual_organ_diameter,
         data = datos, method = "lda", prec = 200,
         image.colors = c("green", "orange"),
         col.mean = "yellow",nplots.vert =1, nplots.hor=3)
```



Unlike the classification with the three explanatory variables, which has an error of 0%, when considering the classification according to each pair of variables, larger errors are made, although as was intuited from the beginning **the pairs *abdomen_diameter - leg_length* and *sexual_organ_diameter - leg_length* make little error in the classification (5%)** while the error with the pair *abdomen_diameter - sexual_organ_diameter* shoots up to 30%.

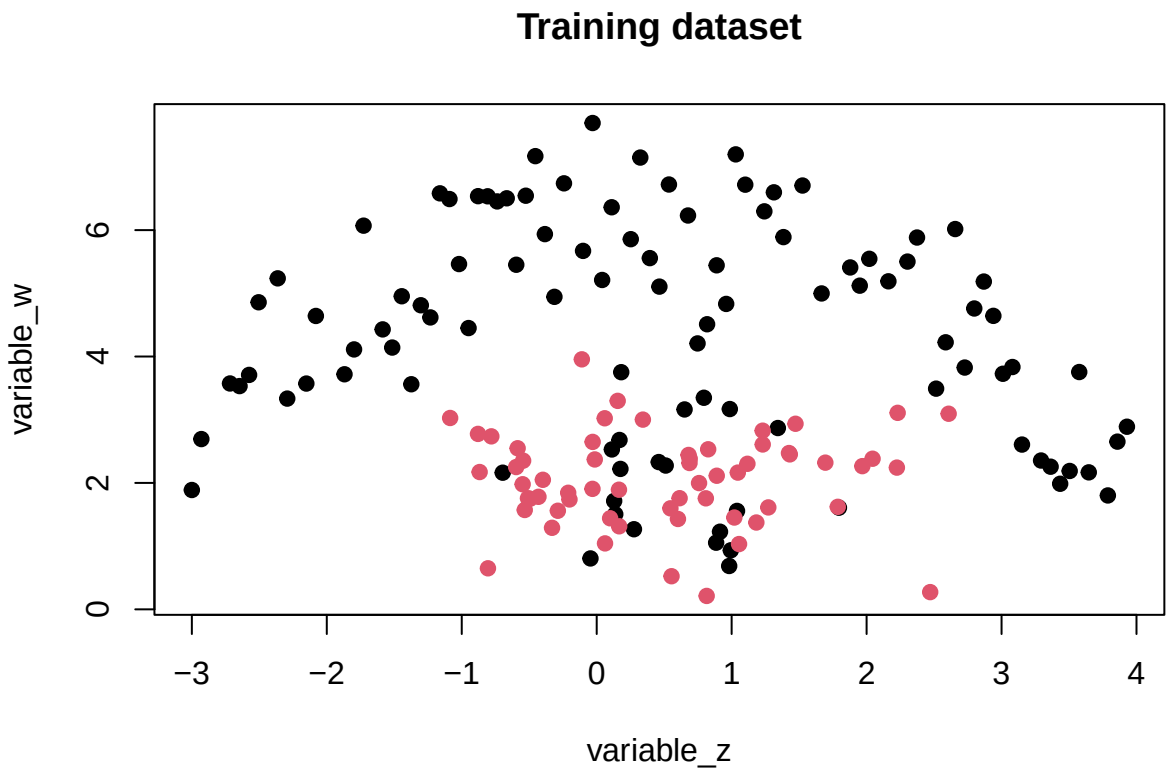
Quadratic discriminant analysis

An example of quadratic discriminant analysis using a set of **simulated training data** is illustrated below. 200 records of two explanatory variables, *variable_z and variable_w*, are simulated, which are classified into two groups, *group A and group B*.

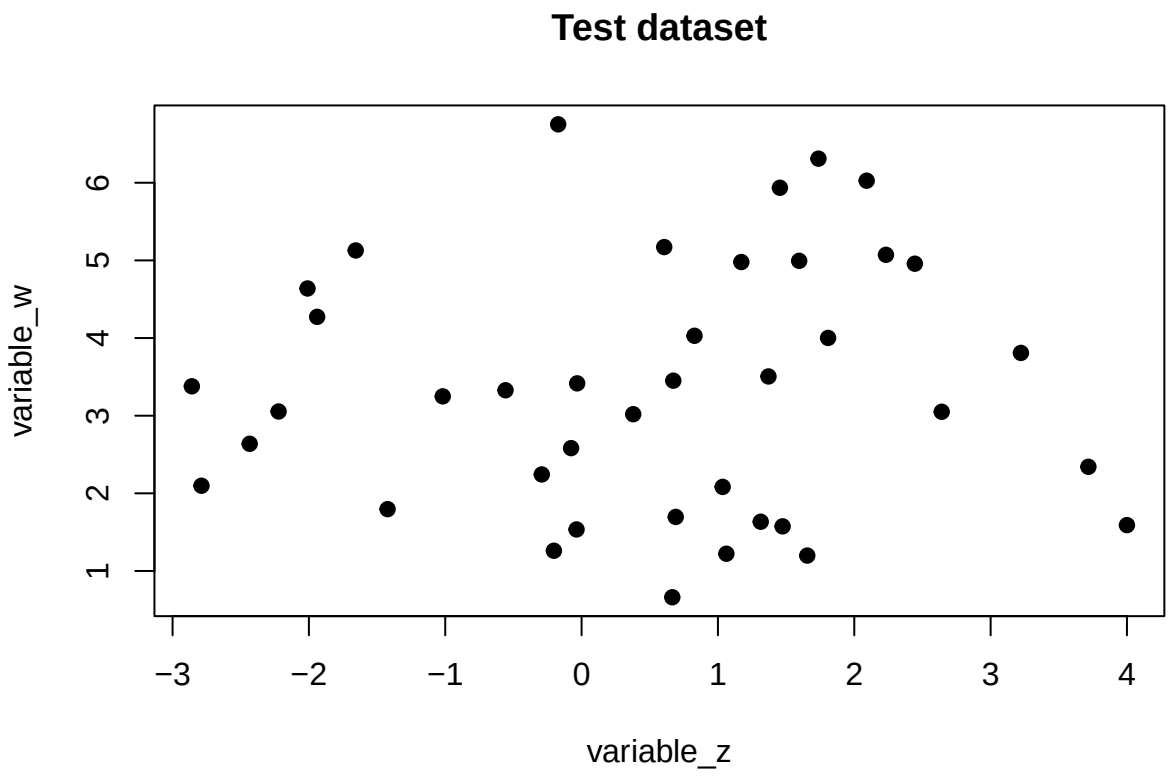
In this example, **we will work with a partition of the dataset, as a training set,** on which we will perform the quadratic discriminant analysis, and **another partition, as a test set,** with which we perform the validation of the model.

```
set.seed(3)
grupoA_x <- seq(from = -3, to = 4, length.out = 100)
grupoA_y <- 6 + 0.15 * grupoA_x - 0.3 * grupoA_x^2 + rnorm(100, sd = 1)
grupoA <- data.frame(variable_z = grupoA_x, variable_w = grupoA_y, grupo = "A")

grupoB_x <- rnorm(n = 100, mean = 0.5, sd = 0.8)
grupoB_y <- rnorm(n = 100, mean = 2, sd = 0.8)
grupoB <- data.frame(variable_z = grupoB_x, variable_w = grupoB_y, grupo = "B")

datos <- rbind(grupoA, grupoB)
```

```r
datos$grupo <- as.factor(datos$grupo)

# Partitioning the dataset: training (80%) + test (20%)
trainIndex<-createDataPartition(datos$grupo,p=0.80)$Resample1
datos_train<-datos[trainIndex,]
datos_test<-datos[-trainIndex,]

plot(datos_train[, 1:2], col = datos$grupo, pch = 19, main="Training dataset")
```



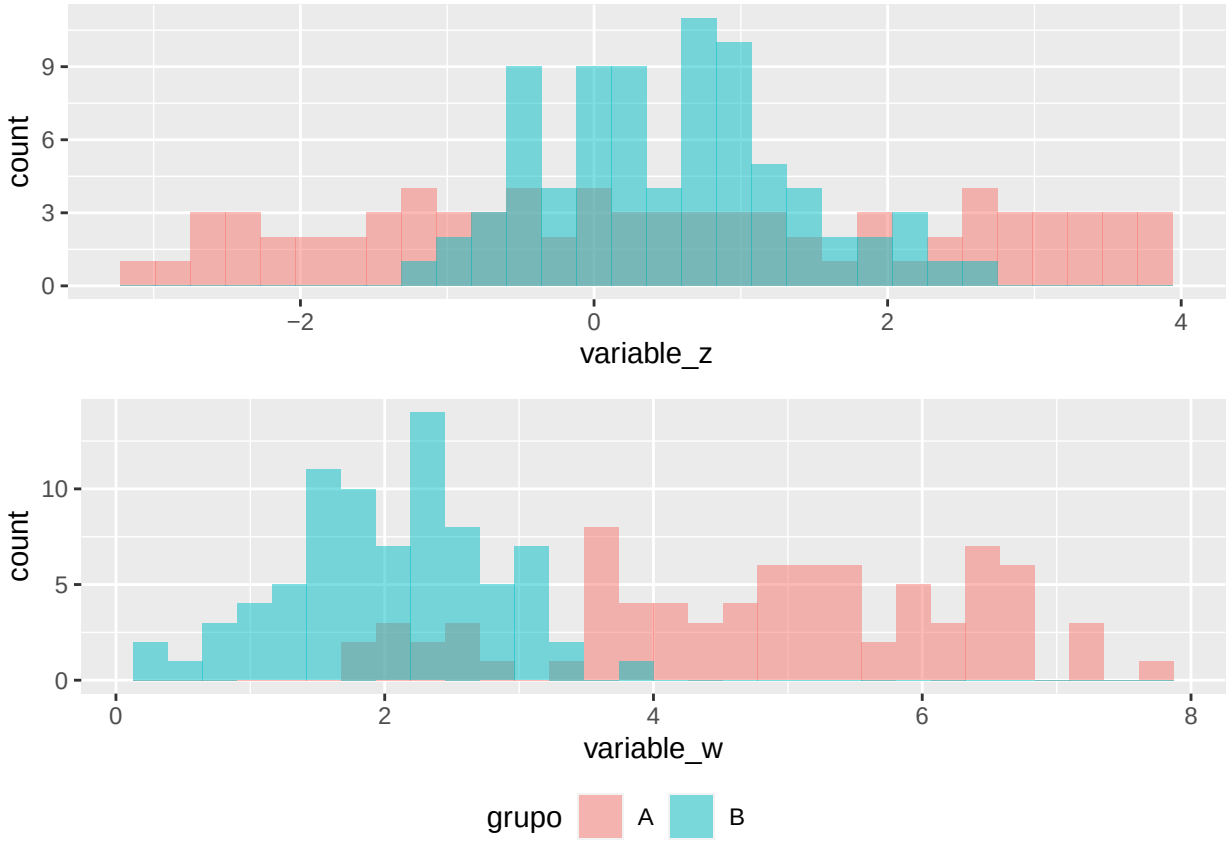

```r
plot(datos_test[, 1:2], col = datos$grupo, pch = 19, main="Test dataset")
```



Graphical exploration of data

First, we explore how well (or poorly) **each of the two explanatory variables** considered independently classifies for the groups.

```
p1 <- ggplot(data = datos_train, aes(x = variable_z, fill = grupo)) +
      geom_histogram(position = "identity", alpha = 0.5)
p2 <- ggplot(data = datos_train, aes(x = variable_w, fill = grupo)) +
      geom_histogram(position = "identity", alpha = 0.5)
ggarrange(p1, p2, nrow = 2, common.legend = TRUE, legend = "bottom")
```



In this sense, it seems that the **variable_w** is the one that best differentiates between groups (least overlapping).

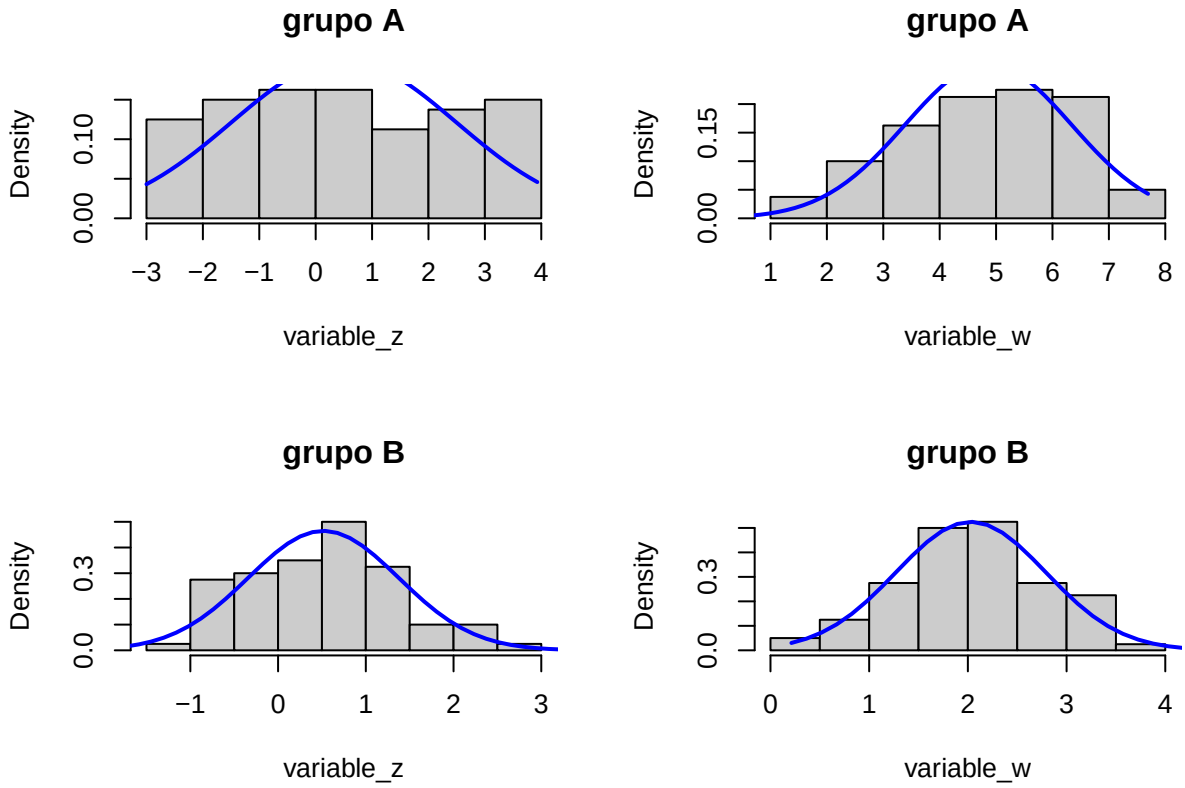
Univariate and multivariate normality

Univariate normality

Next we make a graphical exploration of the **normality** of the **univariate distributions** of our predictors by representing the **histograms** and the **qqplots**.

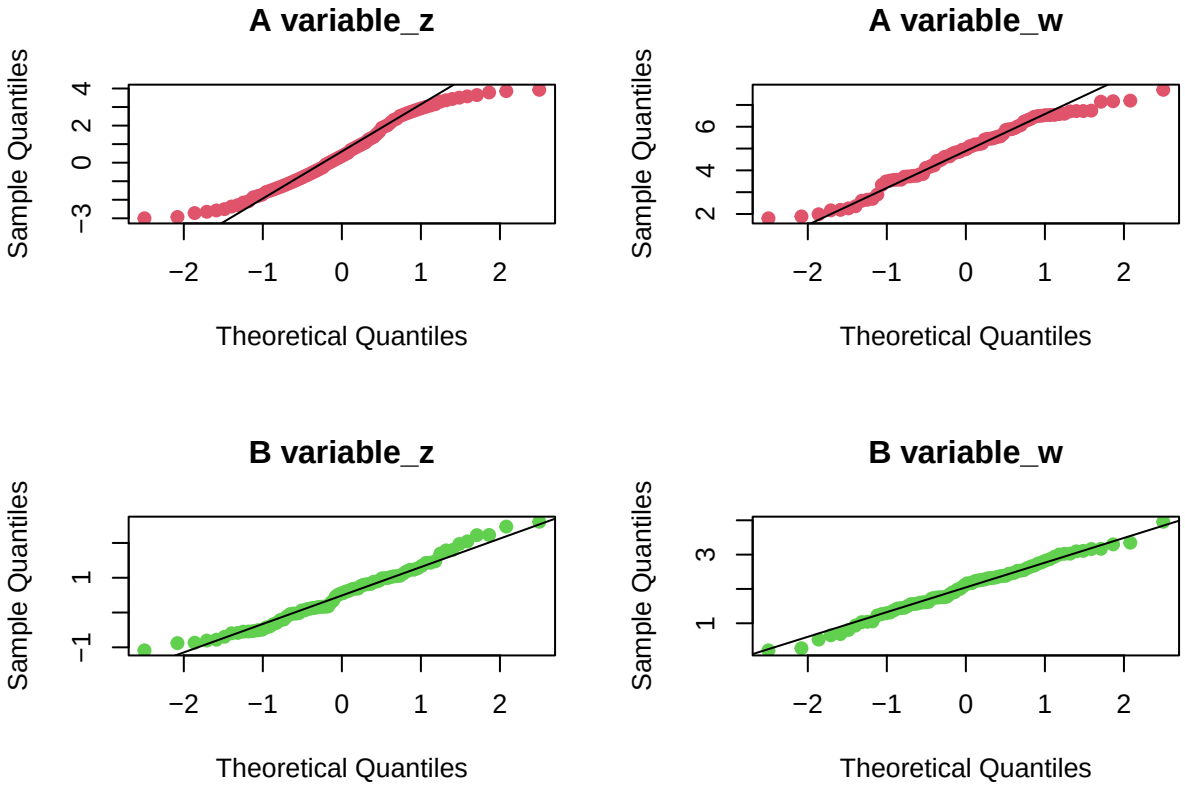
Univariate histograms

```
# Histogram representation of each variable for each group
par(mfcol = c(2, 2))
for (k in 1:2) {
  j0 <- names(datos_train)[k]
  x0 <- seq(min(datos_train[, k]), max(datos_train[, k]), le = 50)
  for (i in 1:2) {
    i0 <- levels(datos_train$grupo)[i]
    x <- datos_train[datos_train$grupo == i0, j0]
    hist(x, proba = T, col = grey(0.8), main = paste("grupo", i0),
    xlab = j0)
    lines(x0, dnorm(x0, mean(x), sd(x)), col = "blue", lwd = 2)
  }
}
```



Qqplots graphics

```
# Representation of normal quantiles of each variable for each group
par(mfcol = c(2, 2))
for (k in 1:2) {
  j0 <- names(datos_train)[k]
  x0 <- seq(min(datos_train[, k]), max(datos_train[, k]), le = 50)
  for (i in 1:2) {
    i0 <- levels(datos_train$grupo)[i]
    x <- datos_train[datos_train$grupo == i0, j0]
    qqnorm(x, main = paste(i0, j0), pch = 19, col = i + 1)
    qqline(x)
  }
}
```



This exploratory analysis can give us an idea of the possible normal distribution of the univariate variables, but it is always better to do the respective normality tests.

Univariate normality test (Shapiro-Wilk)

```
# Shapiro-Wilk normality test for each variable in each group
datos_tidy <- melt(datos_train, value.name = "valor")
datos_tidy %>%
  group_by(grupo, variable) %>%
  summarise(p_value_Shapiro.test = round(shapiro.test(valor)$p.value,5))
```

```
## # A tibble: 4 x 3
## # Groups:   grupo [2]
##   grupo variable   p_value_Shapiro.test
##   <fct> <fct>                     <dbl>
## 1 A     variable_z               0.0091
## 2 A     variable_w               0.0447
## 3 B     variable_z               0.206
## 4 B     variable_w               0.872
```

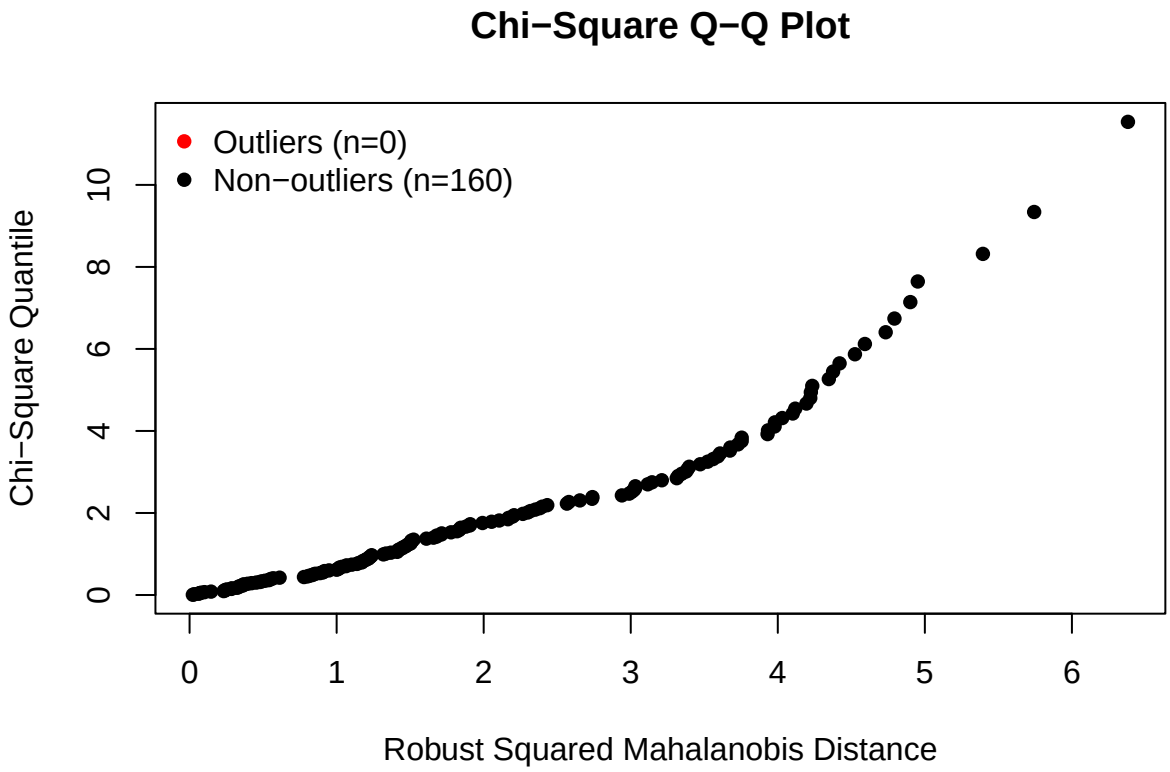
The variables **z** and **w** are not distributed according to a normal law for **group A** (p-value <0.05).

Multivariate normality test (Mardia, Henze-Zirkler and Royston)

The **MVN** package contains functions that allow you to perform the three tests that are commonly used to **test multivariate normality**.

This multivariate normality can be affected by the presence of outliers. In this package we also find **functions for outlier analysis**

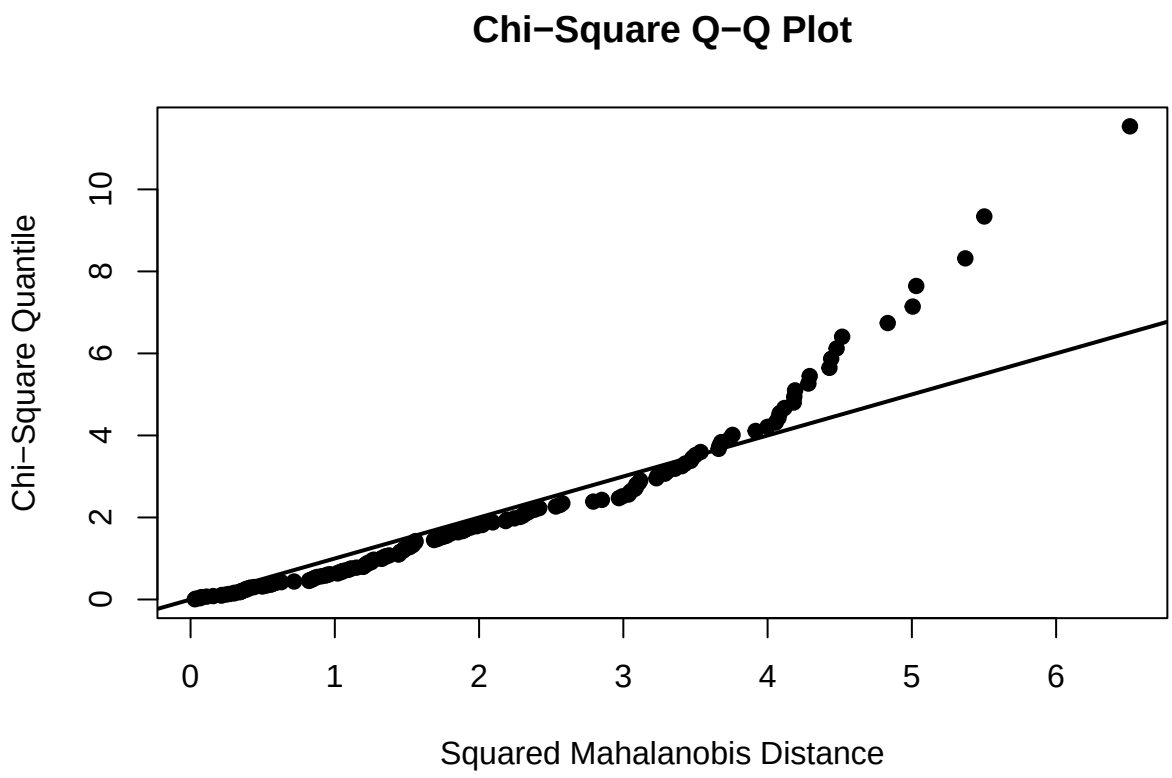
```
outliers <- mvn(data = datos_train[,-3], mvnTest = "hz", multivariateOutlierMethod = "quan")
```



In this case no outliers are detected. The two tests performed below find evidence, at 5% significance level, of lack of multivariate normality.

Although it is true that, as can be deduced from the following outputs, **the assumption of multivariate normality is not met**, the **quadratic discriminant analysis has a certain robustness** in this case, although it should be taken into account in the conclusions of the analysis.

```
# Royston multivariate normality test
royston_test <- mvn(data = datos_train[,-3], mvnTest = "royston", multivariatePlot = "qq")
```




```
royston_test$multivariateNormality
```

```
##      Test        H      p value MVN
## 1 Royston 23.06759 9.792823e-06  NO
```

```
# Henze-Zirkler multivariate normality test
hz_test <- mvn(data = datos_train[,-3], mvnTest = "hz")
hz_test$multivariateNormality
```

```
##            Test       HZ      p value MVN
## 1 Henze-Zirkler 4.951702 6.526013e-11  NO
```

Homogeneity of variance

We proceed as in *Section 1.3* above.

The **null hypothesis** to be tested is that of **equality of covariance matrices** in all groups.

```
boxM(data = datos_train[, -3], grouping = datos_train[, 3])
```

```
##
##  Box's M-test for Homogeneity of Covariance Matrices
##
## data:  datos_train[, -3]
## Chi-Sq (approx.) = 81.024, df = 3, p-value < 2.2e-16
```

In this case we reject the null hypothesis since **p-value $<$ 0.001** and therefore **we assume NO homogeneity of variances**.

It is important to remember that for this **conclusion to be reliable** the assumption of **multivariate**

normality must be met, which in this case is not the case. In fact, when there is no multivariate normal distribution, this test always comes out significant and therefore is not reliable.

Discriminant function

Although the assumption of multivariate normality is not verified, taking into account that the variances are not homogeneous, **a quadratic discriminant model is adjusted** because it is robust against the lack of this assumption, although it must be kept in mind given the possibility of obtaining unexpected results.

The **qda** function of the **MASS** package performs the sorting.

```
modelo_qda <- qda(grupo ~ variable_z + variable_w, data = datos_train)
modelo_qda
```

```
## Call:
## qda(grupo ~ variable_z + variable_w, data = datos_train)
##
## Prior probabilities of groups:
##   A   B
## 0.5 0.5
##
## Group means:
##   variable_z variable_w
## A  0.5000000   4.862263
## B  0.5170937   2.030475
```

The output of this object shows us the **prior probabilities** of each group, in this case 0.5 and the **means of each regressor per group**.

Once the classifier is built, we can classify new data based on its measurements by simply calling the **predict** function. For example, **we are going to classify all the observations in the test dataset**.

```
nuevas_observaciones <- datos_test
predict(object = modelo_qda, newdata = nuevas_observaciones)
```

```
## $class
##  [1] A A A A A A A A A A A A A A A A A A A A A B B A B B B B B B B A A B B B B B B
## [39] B A
## Levels: A B
##
## $posterior
##              A            B
## 3   0.99572804 4.271960e-03
## 4   0.88076443 1.192356e-01
## 9   0.87488971 1.251103e-01
## 12  0.91643185 8.356815e-02
## 15  0.99961804 3.819638e-04
## 16  0.99753642 2.463576e-03
## 20  0.99992181 7.819121e-05
## 41  0.99999997 3.024630e-08
## 52  0.99911537 8.846308e-04
## 60  0.99808348 1.916519e-03
## 64  0.99999270 7.302331e-06
## 66  0.99880428 1.195717e-03
## 68  0.99999951 4.908639e-07
## 69  0.93604603 6.395397e-02
## 73  0.99999816 1.842343e-06
```

```
## 75  0.99968839 3.116069e-04
## 78  0.99963898 3.610191e-04
## 89  0.99441459 5.585415e-03
## 96  0.96333973 3.666027e-02
## 100 0.97329074 2.670926e-02
## 105 0.03509319 9.649068e-01
## 107 0.01724841 9.827516e-01
## 108 0.57241519 4.275848e-01
## 115 0.08518801 9.148120e-01
## 118 0.45297454 5.470255e-01
## 136 0.11209239 8.879076e-01
## 137 0.04346586 9.565341e-01
## 138 0.01808141 9.819186e-01
## 143 0.05273681 9.472632e-01
## 144 0.18804503 8.119550e-01
## 150 0.86695494 1.330451e-01
## 155 0.60687367 3.931263e-01
## 161 0.44664763 5.533524e-01
## 166 0.02205248 9.779475e-01
## 167 0.04044205 9.595580e-01
## 169 0.04370428 9.562957e-01
## 173 0.02323925 9.767607e-01
## 187 0.48079614 5.192039e-01
## 192 0.01986393 9.801361e-01
## 195 0.79768574 2.023143e-01
```

For example, according to the discriminant function, the posterior probability that the first record of the test set is in **group A** is 99.6% while the probability that it is in **group B** is lower at 0.1%. Therefore this data would be classified in **group A**. The same reasoning is followed with the rest of the elements of the test set.

Model validation

The **confusionmatrix** function of the **biotools** package performs cross-validation of the classification model.

```
pred <- predict(object = modelo_qda, newdata = datos_test)
confusionmatrix(datos_test$grupo, pred$class)
```

```
##   new A new B
## A    20     0
## B     4    16
```

```
# Classification error percentage
trainig_error <- mean(datos_test$grupo != pred$class) * 100
paste("trainig_error=", trainig_error, "%")
```

```
## [1] "trainig_error= 10 %"
```

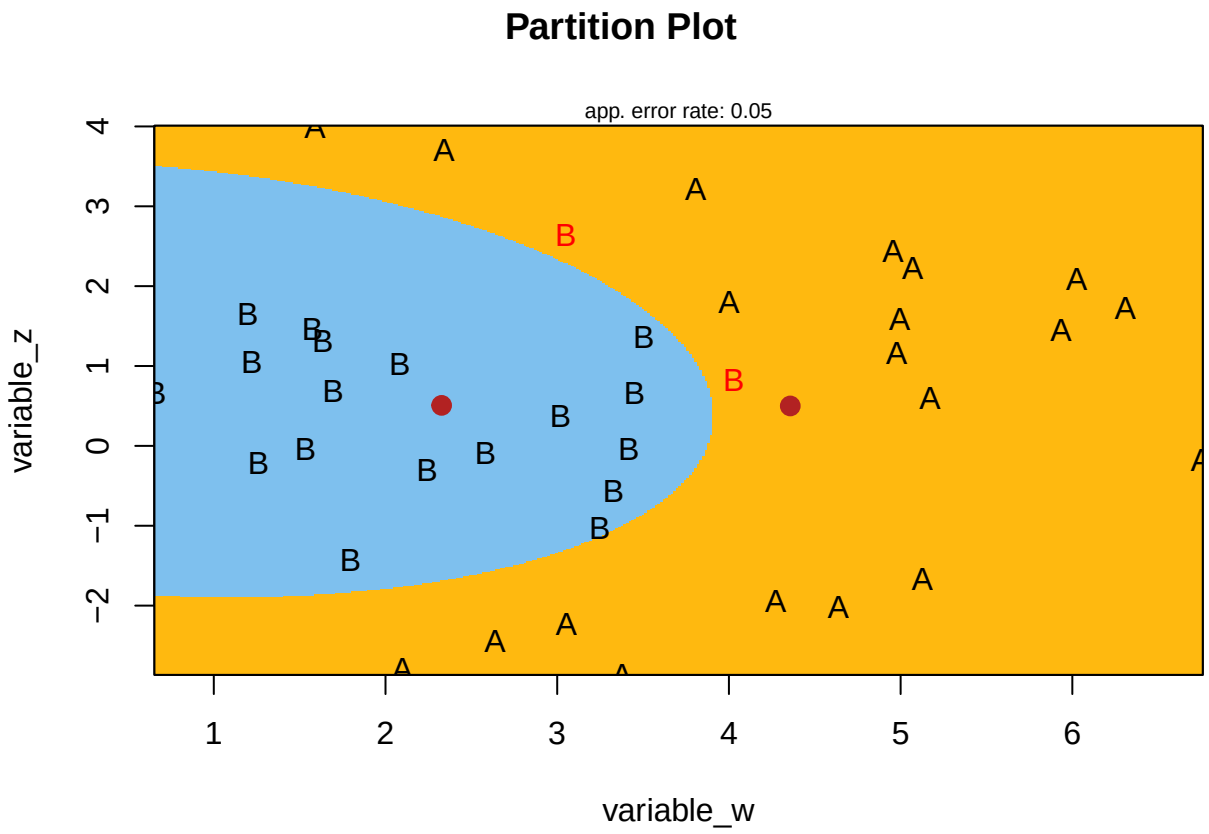
In this case the **correct classifications rate** is 90.0%.

Displaying rankings

The **partimat** function of the **klaR** package allows us to represent the **classification limits** of a linear or quadratic discriminant model for each pair of predictors. Each color represents a classification region according to the model, **the centroid** of each region and the **real value of the observations** are shown.

```
partimat(formula = grupo ~ variable_z + variable_w, data = datos_test,
         method = "qda", prec = 400,
```

```
        image.colors = c("darkgoldenrod1", "skyblue2"),
        col.mean = "firebrick")
```



LDA vs. QDA

The most appropriate classifier depends on the implications of assuming that all groups share a common covariance matrix since this assumption can produce a bias in the classifications or produce high variances.

- LDA produces **linear decision limits**, which translates into **less flexibility** and therefore less variance problem.
- QDA produces **quadratic limits** and therefore curves, which provides greater flexibility allowing a better fit to the data, less bias but greater risk of variance.
- In general terms, **LDA tends to achieve better rankings than QDA when there are few observations with which to train the model**, a scenario in which avoiding variance is crucial.
- If a **large number of training observations** are available or if it is not assumed that a common covariance matrix exists between classes, **QDA is more appropriate**.
- If **p predictors** are available, calculating a **common covariance matrix requires estimating p(p+1)/2 parameters**, while **calculating a different matrix for each group requires Kp(p +1)/2** (K is the number of groups of the response variable). For very high **p values**, the choice of method may be limited by **computational power**.