Análisis Discriminante Cuadrático

Idea intuitiva

El clasificador cuadrático o Quadratic Discriminat Analysis QDA se asemeja en gran medida al LDA, con la única diferencia de que el QDA considera que cada clase k tiene su propia matriz de covarianza (Σk) y, como consecuencia, la función discriminante toma forma cuadrática:

$$log(P(Y = k|X = x)) = -\frac{1}{2}\log|\Sigma_k| - \frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1}(x - \mu_k) + \log(\pi_k)$$

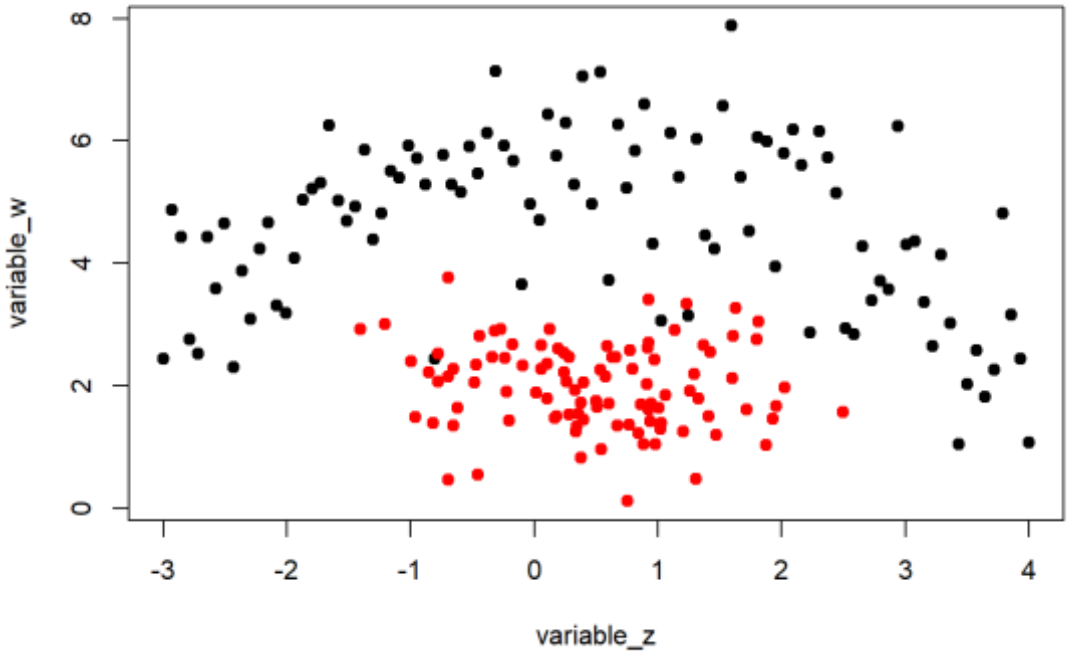
Para poder calcular la posterior probability a partir de esta ecuación discriminante es necesario estimar para cada clase, (Σk), μk y πk a partir de la muestra. Cada nueva observación se clasifica en aquella clase para la que el valor de la posterior probability sea mayor.

QDA genera límites de decisión curvos por lo que puede aplicarse a situaciones en las que la separación entre grupos no es lineal.

Ejemplo QDA 2 predictores

Se dispone de los siguientes datos simulados.

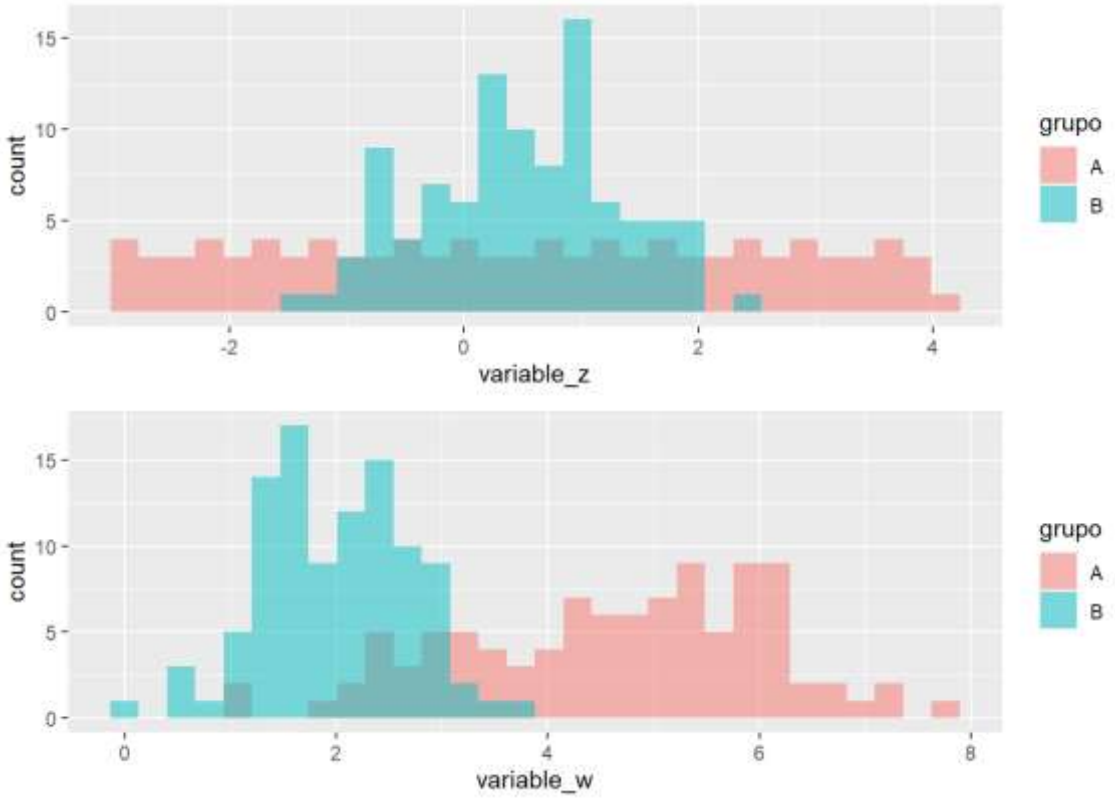
```
# datos simulados

set.seed(8558)

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, g
rupo = "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, g
rupo = "B")

datos <- rbind(grupoA, grupoB)

X11()

plot(datos[, 1:2], col = datos$grupo, pch = 19)


# Exploracion grafica de los datos

library(ggplot2)
```



La separación entre los grupos no es de tipo lineal, sino que muestra cierta curvatura. En este tipo de escenarios el método *QDA* es más adecuado que el *LDA.*

Exploración gráfica de los datos

```
library(gridExtra)
p1 <- ggplot(data = datos, aes(x = variable_z, fill = grupo)) +
  geom_histogram(position = "identity", alpha = 0.5)
p2 <- ggplot(data = datos, aes(x = variable_w, fill = grupo)) +
  geom_histogram(position = "identity", alpha = 0.5)
grid.arrange(p1, p2)
```



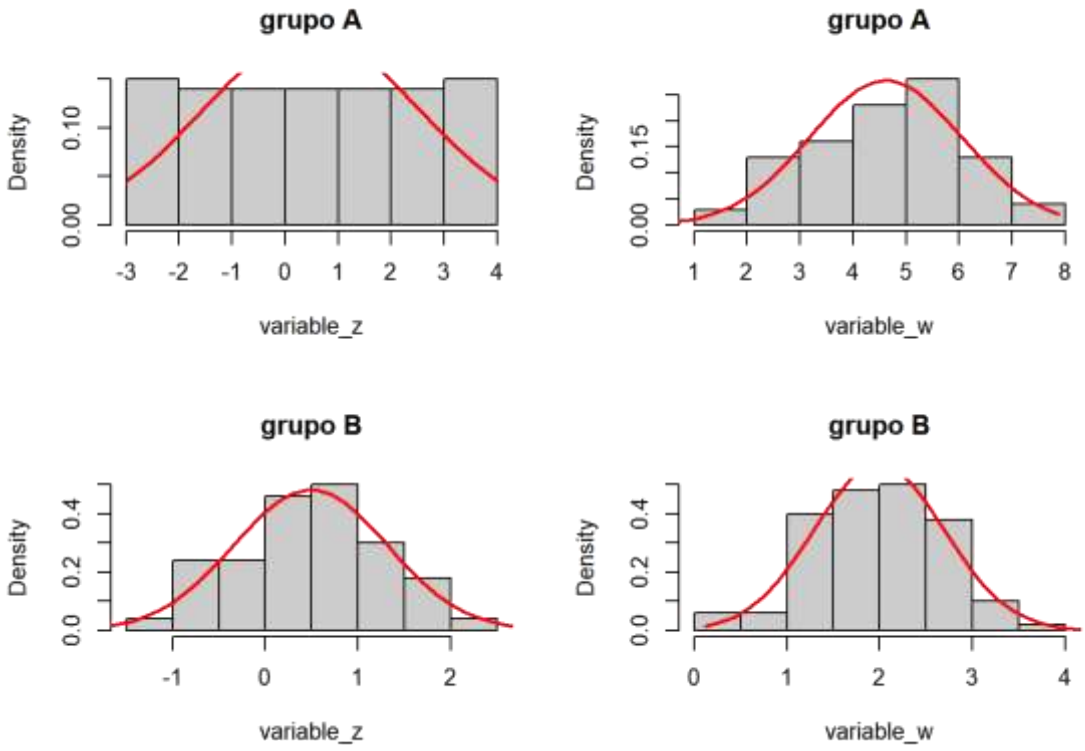
La variable *W* permite discriminar entre grupos mejor que la variable *z*.

Normalidad univariante, normalidad multivariante y homogeneidad de varianza

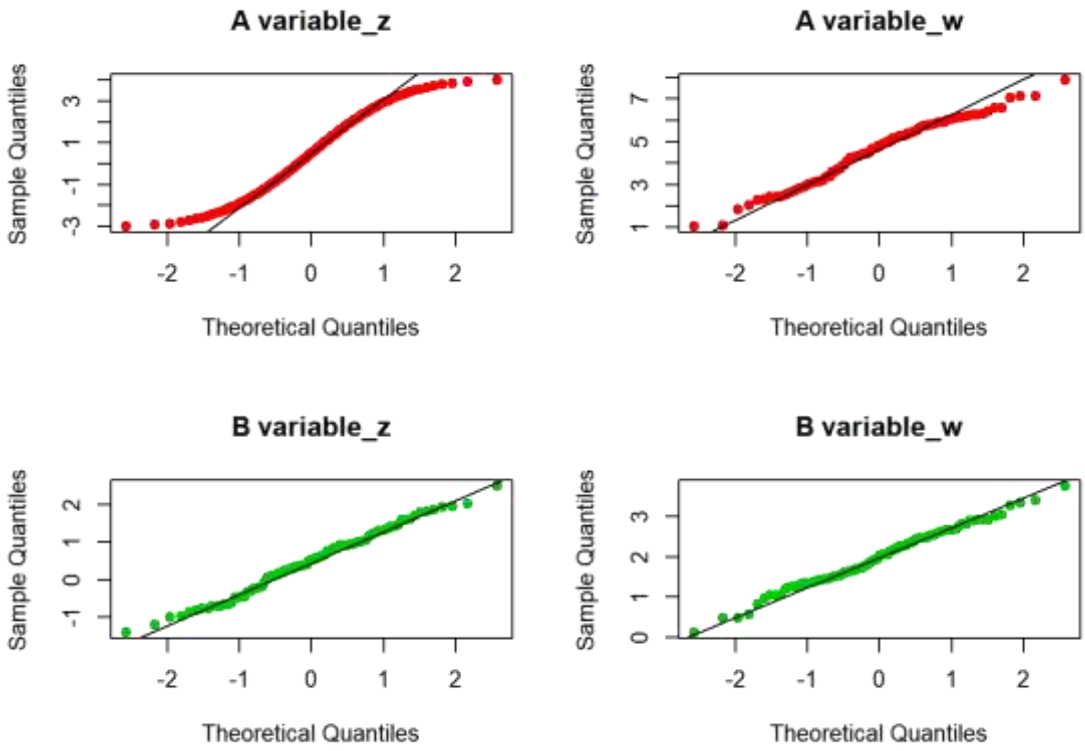
Distribución de los predictores de forma individual:

```
# Normalidad univariante, normalidad multivariante y homogeneidad de v
arianza


# representacion mediante histograma de cada variable para cada grupo
par(mfcol = c(2, 2))
for (k in 1:2) {
  j0 <- names(datos)[k]
  x0 <- seq(min(datos[, k]), max(datos[, k]), le = 50)
  for (i in 1:2) {
    i0 <- levels(datos$grupo)[i]
    x <- datos[datos$grupo == i0, j0]
    hist(x, proba = T, col = grey(0.8), main = paste("grupo", i0), xla
b = j0)
    lines(x0, dnorm(x0, mean(x), sd(x)), col = "red", lwd = 2)
  }
```

```
}
```




```
# representacion de cuantiles normales de cada variable para cada grupo
for (k in 1:2) {
  j0 <- names(datos)[k]
  x0 <- seq(min(datos[, k]), max(datos[, k]), le = 50)
  for (i in 1:2) {
    i0 <- levels(datos$grupo)[i]
    x <- datos[datos$grupo == i0, j0]
    qqnorm(x, main = paste(i0, j0), pch = 19, col = i + 1)
    qqline(x)
  }
}
```



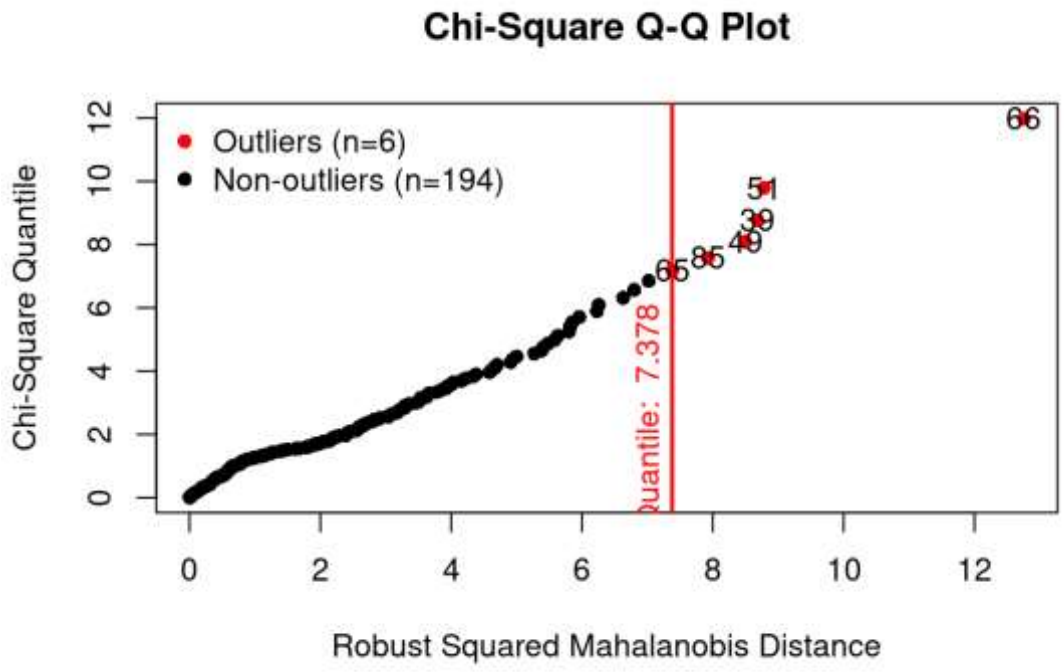

```
# Contraste de normalidad Shapiro-Wilk para cada variable en cada grup
o
library(reshape2)
library(knitr)
library(dplyr)
datos_tidy <- melt(datos, value.name = "valor")
kable(datos_tidy %>% group_by(grupo, variable) %>% summarise(p_value_S
hapiro.test =
        round(shapiro.test(valor)$p.value, 5)))
```

grupo	variable	p_value_Shapiro.test
A	variable_z	0.00172
A	variable_w	0.09319
B	variable_z	0.62479
B	variable_w	0.81022

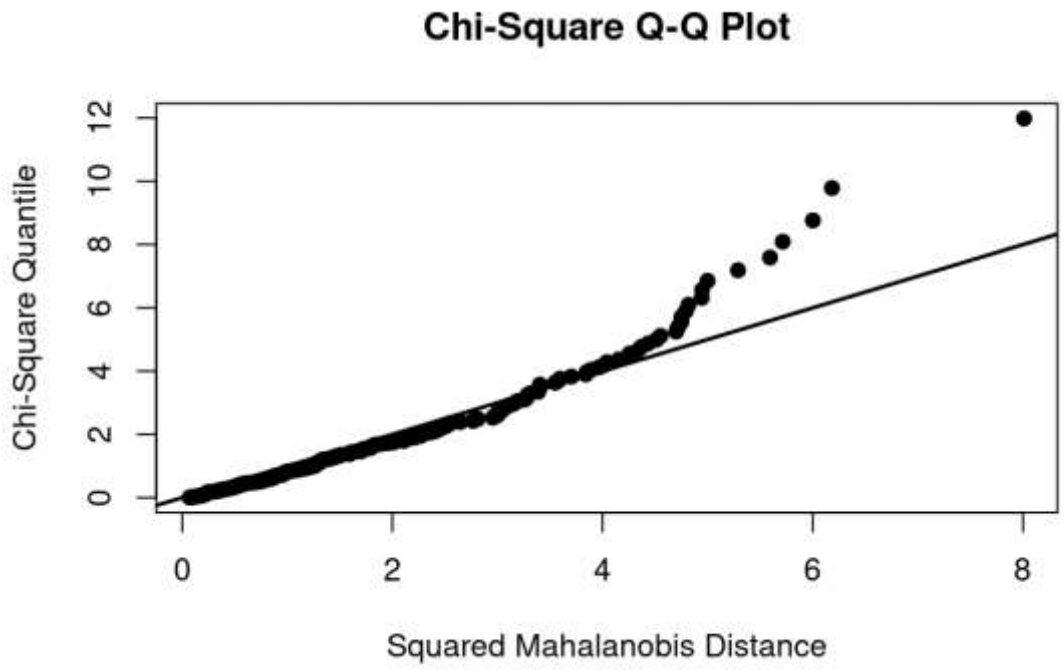
La variable Z no se distribuye de forma normal en el grupo A.

Normalidad multivariante:

```
library(MVN)
outliers <- mvn(data = datos[,-3], mvnTest = "hz", multivariateOutlier
Method = "quan")
```




```
royston_test <- mvn(data = datos[,-3], mvnTest = "royston", multivaria
tePlot = "qq")
```




```
# Normalidad multivariante:
library(rlang)
library(vctrs)
library(MVN)
```

```
# outliers <- mvOutlier(datos[,-3], qqplot = TRUE, method = "quan")
result <- mvn(data = datos[,-3], mvnTest = "mardia")
result$multivariateNormality
```

```
##              Test         Statistic             p value Result
## 1 Mardia Skewness  11.9047359065296  0.0180739132176235     NO
## 2 Mardia Kurtosis -2.98425282998472 0.00284271770498834     NO
## 3             MVN              <NA>                <NA>     NO
```

```
result <- mvn(data = datos[,-3], mvnTest = "royston")
result$multivariateNormality
```

```
##      Test        H     p value MVN
## 1 Royston 29.11024 4.77274e-07  NO
```

```
hz_test <- mvn(data = datos[,-3], mvnTest = "hz")
hz_test$multivariateNormality
```

Test <chr>	HZ <dbl>	p value <dbl>	MVN <chr>
Henze-Zirkler	6.739874	5.317968e-14	NO

Ambos test muestran evidencias significativas de falta de normalidad multivariante. El *QDA* tiene cierta robustez frente a la falta de normalidad multivariante, pero es importante tenerlo en cuenta en la conclusión del análisis.

Cálculo de la función discriminante

```
# Calculo de la funcion discriminante
library(MASS)
modelo_qda <- qda(grupo ~ variable_z + variable_w, data = datos)
modelo_qda
```

```
## Call:
## qda(grupo ~ variable_z + variable_w, data = datos)
##
## Prior probabilities of groups:
##   A   B
## 0.5 0.5
##
## Group means:
##   variable_z variable_w
## A  0.5000000   4.615307
## B  0.4864889   1.992911
```

Evaluación de los errores de clasificación

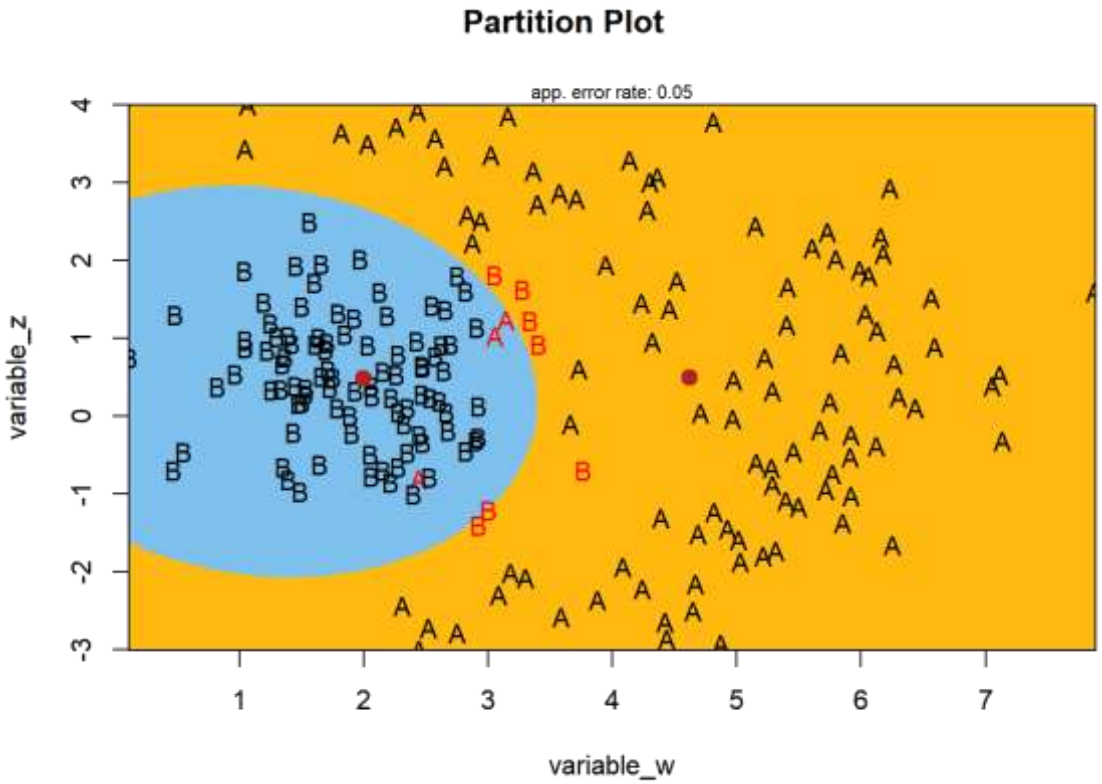
```
# Evaluacion de los errores de clasificacion
predicciones <- predict(object = modelo_qda, newdata = datos)
table(datos$grupo, predicciones$class, dnn = c("Clase real", "Clase predicha"))
```

```
##            Clase predicha
## Clase real  A  B
##          A 97  3
##          B  7 93
```

```
trainig_error <- mean(datos$grupo != predicciones$class) * 100
paste("trainig_error=", trainig_error, "%")
```

```
## [1] "trainig_error= 5 %"
```

```
library(klaR)
partimat(formula = grupo ~ variable_z + variable_w, data = datos, method = "qda",
         prec = 400, image.colors = c("darkgoldenrod1", "skyblue2"), col.mean =
           "firebrick")
```



Ejemplo QDA billetes falsos

Se pretende generar un modelo discriminante que permita diferenciar entre billetes verdaderos y falsos. Se han registrado múltiples variables para 100 billetes verdaderos y 100 billetes falsos:

- Status: si es verdadero (*genuine*) o falso (*counterfeit*).
- Length: longitud (mm)
- Left: Anchura del borde izquierdo (mm)
- Right: Anchura del borde derecho (mm)
- Bottom: Anchura del borde inferior (mm)
- Top: Anchura del borde superior (mm)
- Diagonal: longitud diagonal (mm)

```
library(mclust)
library(knitr)
data(banknote)
```

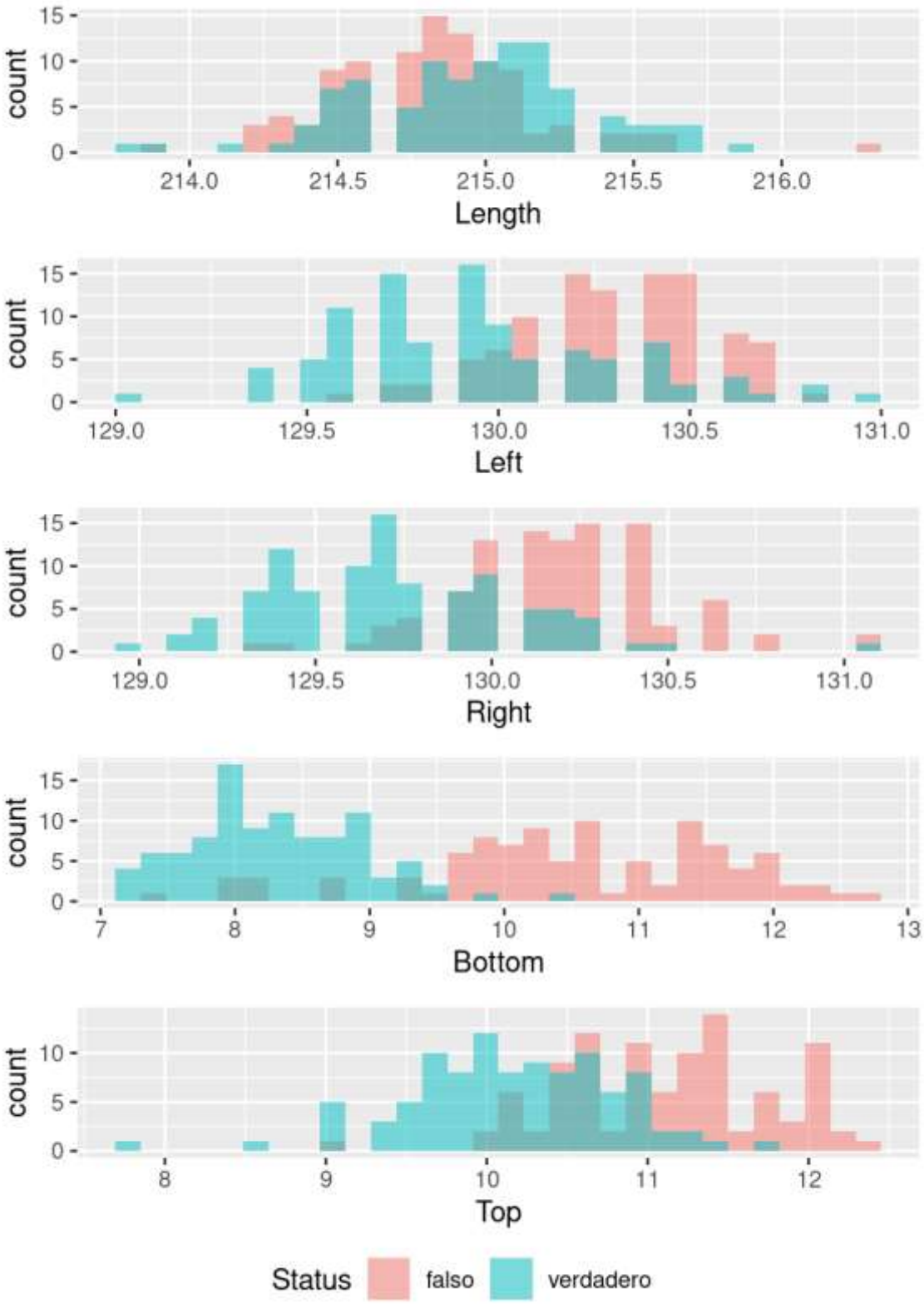
```
#se recodifican las clases de la variable Status: verdadero = 0, falso
= 1
levels(banknote$Status)
```

```
## [1] "counterfeit" "genuine"
```

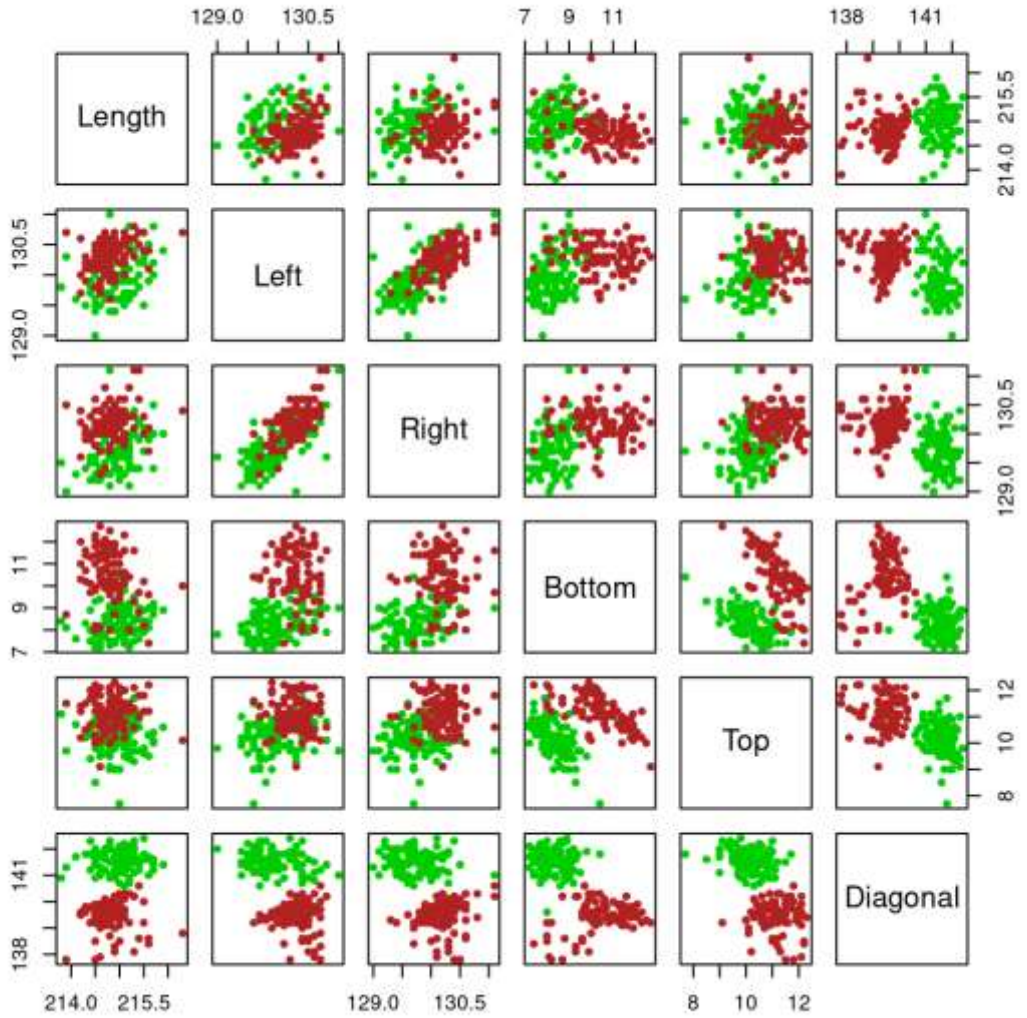
```
levels(banknote$Status) <- c("falso","verdadero")
kable(head(banknote))
```

StatusLengthLeftRightBottomTopDiagonalverdadero214.8131.0131.19.09.7141.0verdadero214.6129.7129.78.19.5141.7verdadero214.8129.7129.78.79.6142.2verdadero214.8129.7129.67.510.4142.0verdadero215.0129.6129.710.47.7141.8verdadero215.7130.8130.59.010.1141.4

```
library(ggplot2)
library(ggpubr)
p1 <- ggplot(data = banknote, aes(x = Length, fill = Status)) +
      geom_histogram(position = "identity", alpha = 0.5)
p2 <- ggplot(data = banknote, aes(x = Left, fill = Status)) +
      geom_histogram(position = "identity", alpha = 0.5)
p3 <- ggplot(data = banknote, aes(x = Right, fill = Status)) +
      geom_histogram(position = "identity", alpha = 0.5)
p4 <- ggplot(data = banknote, aes(x = Bottom, fill = Status)) +
      geom_histogram(position = "identity", alpha = 0.5)
p5 <- ggplot(data = banknote, aes(x = Top, fill = Status)) +
      geom_histogram(position = "identity", alpha = 0.5)
ggarrange(p1, p2, p3, p4, p5, nrow = 5, common.legend = TRUE, legend =
"bottom")
```




```
pairs(x = banknote[,-1], col = c("firebrick", "green3")[banknote$Statu
s],
      pch = 20)
```



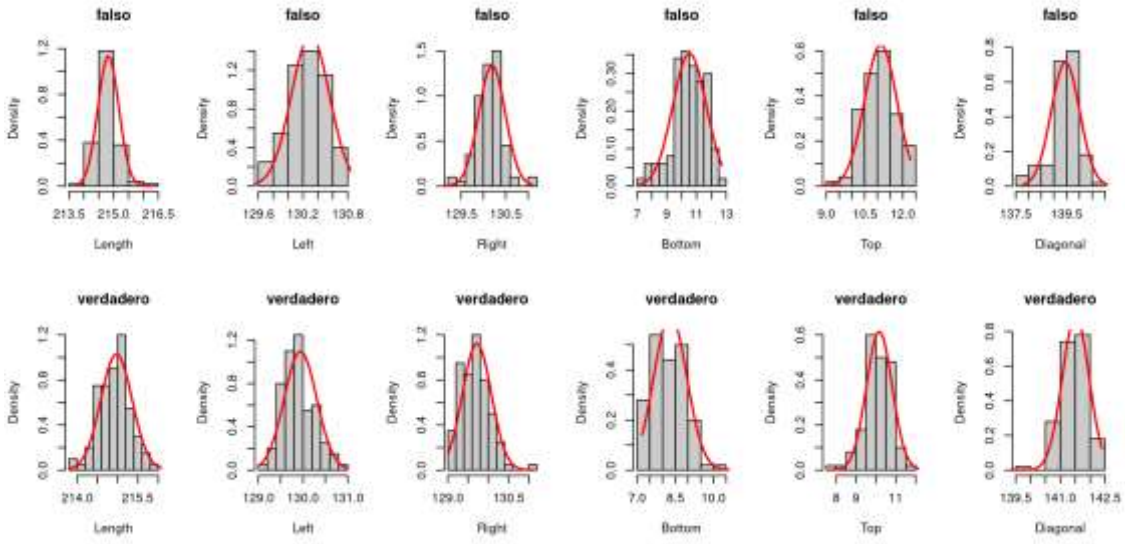
La proporción de billetes falsos en circulación es mucho menor que la de billetes verdaderos, por lo tanto, a pesar de que en la muestra se dispone de la misma cantidad de billetes de cada clase, no es conveniente considerar que las probabilidades previas son iguales. En este tipo de escenario se suele recurrir a estudios previos o muestras piloto que permitan estimar las proporciones poblacionales de cada clase. En nuestro caso se va a suponer que solo el 1% de los billetes en circulación son falsos.

$$\hat{\pi}_{verdadero} = 0.99$$

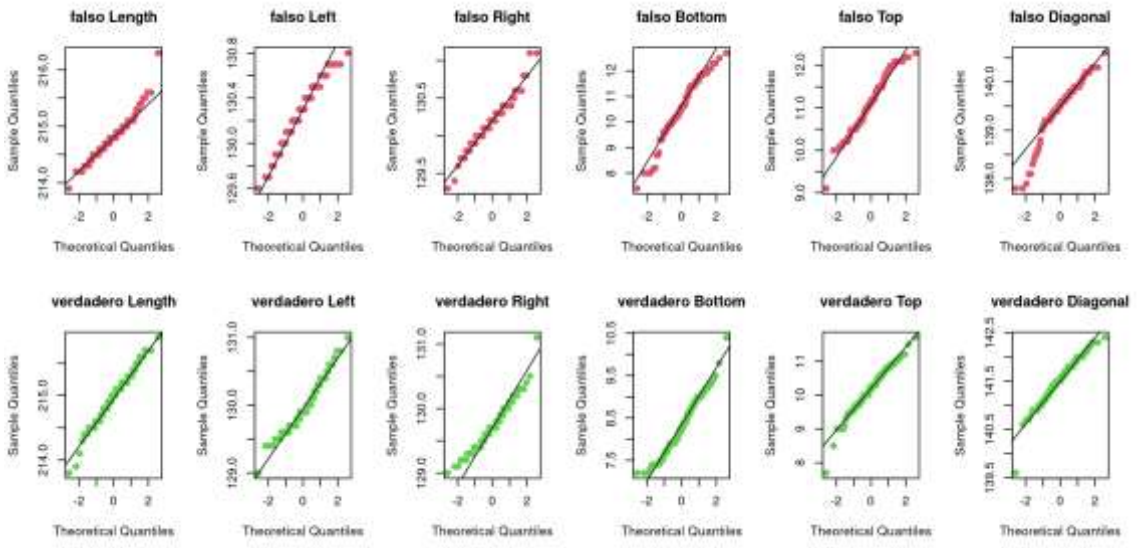
$$\hat{\pi}_{falso} = 0.01$$

Distribución de los predictores de forma individual:

```r
# Representación mediante Histograma de cada variable para cada tipo d
e billete
par(mfcol = c(2, 6))
for (k in 2:7) {
  j0 <- names(banknote)[k]
  x0 <- seq(min(banknote[, k]), max(banknote[, k]), le = 50)
  for (i in 1:2) {
    i0 <- levels(banknote$Status)[i]
    x <- banknote[banknote$Status == i0, j0]
    hist(x, proba = T, col = grey(0.8), main = paste( i0), xlab = j0)
    lines(x0, dnorm(x0, mean(x), sd(x)), col = "red", lwd = 2)
  }
}
```



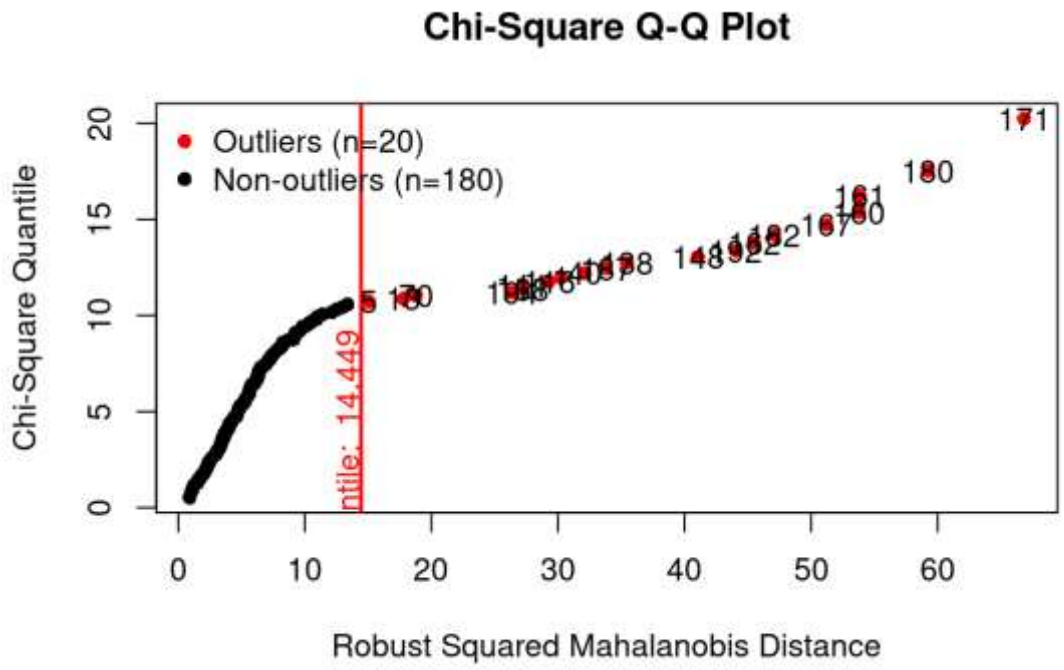

```r
# Representación de cuantiles normales de cada variable para cada tipo
de billete
for (k in 2:7) {
  j0 <- names(banknote)[k]
  x0 <- seq(min(banknote[, k]), max(banknote[, k]), le = 50)
  for (i in 1:2) {
    i0 <- levels(banknote$Status)[i]
    x <- banknote[banknote$Status == i0, j0]
    qqnorm(x, main = paste(i0, j0), pch = 19, col = i + 1)
    # los colores 2 y 3 son el rojo y verde
    qqline(x)
  }
}
```



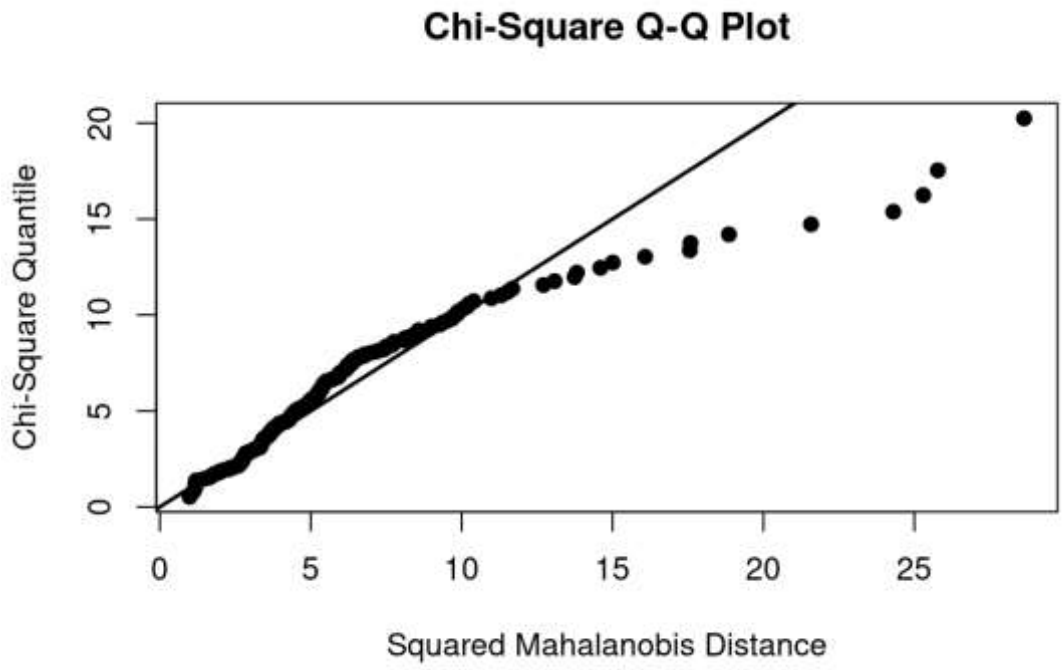

```
#Contraste de normalidad Shapiro-Wilk para cada variable en cada tipo
de billete
library(reshape2)
library(knitr)
library(dplyr)
datos_tidy <- melt(banknote, value.name = "valor")
datos_tidy %>% group_by(Status, variable) %>%
  summarise(p_value_Shapiro.test = round(shapiro.test(valor)$p.value,5
))
```

Status
<fctr>
variable
<fctr>
p_value_Shapiro.test
<dbl>
falsoLength0.00290falsoLeft0.02372falsoRight0.01683falsoBottom0.01115falsoTop0.049
09falsoDiagonal0.00003verdaderoLength0.25674verdaderoLeft0.00825verdaderoRight0.
01174verdaderoBottom0.04035

```
detach(package:mclust, unload = TRUE)
library(MVN)
outliers <- mvn(data = banknote[,-1], mvnTest = "hz", multivariateOutl
ierMethod = "quan")
```




```
royston_test <- mvn(data = banknote[,-1], mvnTest = "royston", multiva
riatePlot = "qq")
```



```
royston_test$multivariateNormality
```

Test <chr>	H <dbl>	p value <dbl>	MVN <chr>
Royston	67.03927	5.820549e-13	NO

```
hz_test <- mvn(data = banknote[,-1], mvnTest = "hz")
hz_test$multivariateNormality
```

Test <chr>	HZ <dbl>	p value <dbl>	MVN <chr>
Henze-Zirkler	1.780591	0	NO

Los datos no siguen una distribución normal multivariante, lo que tiene implicaciones directas en la precisión del *QDA*.

```
library(biotools)
boxM(data = banknote[, -1], grouping = banknote[, 1])
```

```
##  Box's M-test for Homogeneity of Covariance Matrices
##
## data:  banknote[, -1]
## Chi-Sq (approx.) = 121.9, df = 21, p-value = 3.198e-16
```

El test Box's M muestra mucha evidencia de que la matriz de covarianza no es constante en todos los grupos, esta condición hace que el QDA sea más adecuado.

```
library(MASS)
modelo_qda <- qda(formula = Status ~ ., data = banknote, prior = c(0.0
1, 0.99))
modelo_qda
```

```
## Call:
## qda(Status ~ ., data = banknote, prior = c(0.01, 0.99))
##
## Prior probabilities of groups:
##     falso verdadero
##      0.01      0.99
##
## Group means:
##             Length    Left   Right Bottom    Top Diagonal
## falso     214.823 130.300 130.193 10.530 11.133  139.450
## verdadero 214.969 129.943 129.720  8.305 10.168  141.517
```

```
predicciones <- predict(object = modelo_qda, newdata = banknote)
table(banknote$Status, predicciones$class,
      dnn = c("Clase real", "Clase predicha"))
```

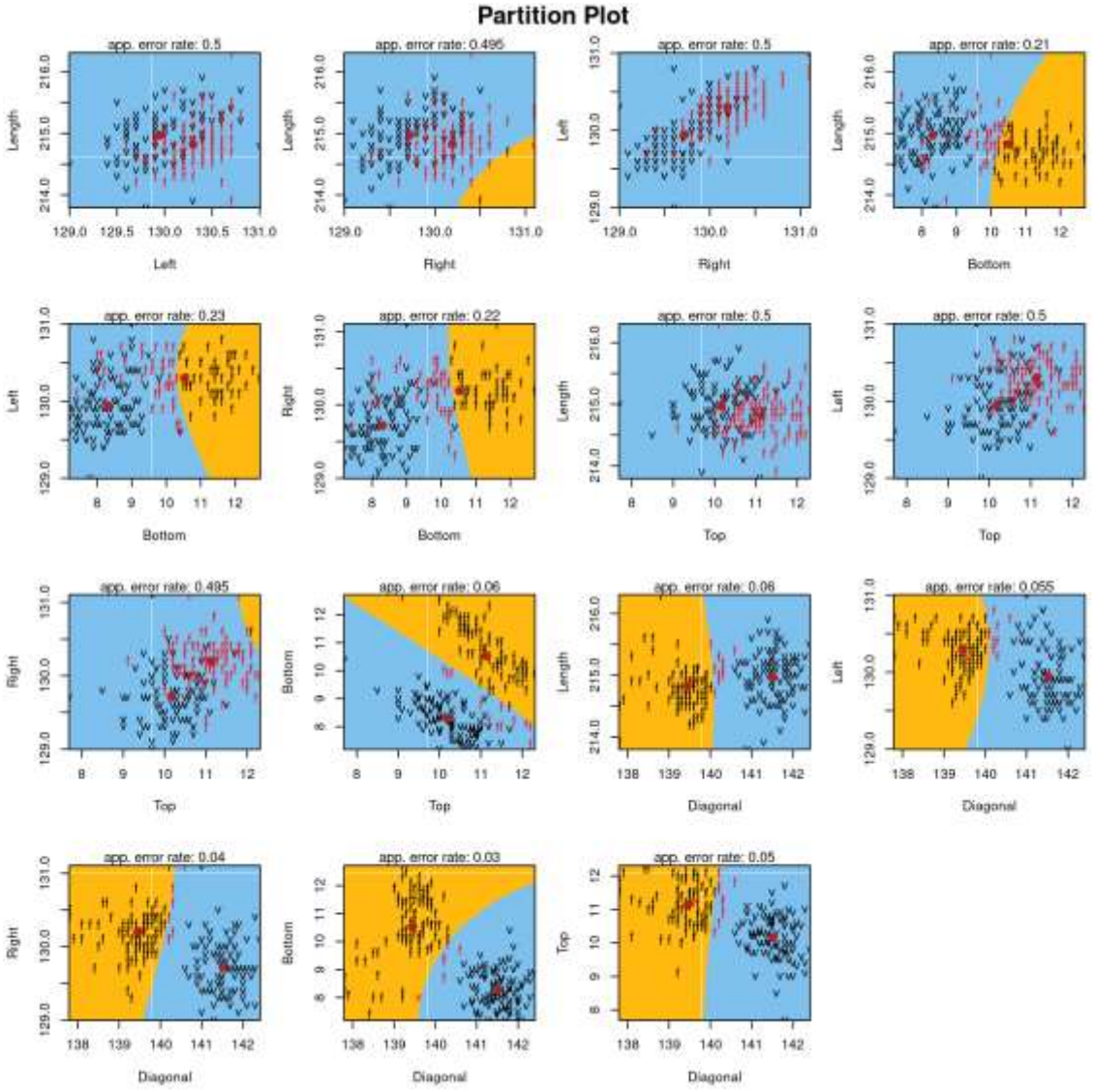
```
##             Clase predicha
## Clase real  falso verdadero
##   falso        99         1
##   verdadero     0       100
```

```
trainig_error <- mean(banknote$Status != predicciones$class) * 100
paste("trainig_error=", trainig_error, "%")
```

```
## [1] "trainig_error= 0.5 %"
```

```
library(klaR)
partimat(formula = Status ~ ., data = banknote, prior = c(0.01, 0.99),
         method = "qda", prec = 200,
         image.colors = c("darkgoldenrod1", "skyblue2"),
```

```
        col.mean = "firebrick", nplots.vert = 4)
```



Comparación entre QDA y LDA

Que clasificador es más adecuado depende de las implicaciones que tiene, en el balance *bias-varianza*, el asumir que todos los grupos comparten una matriz de covarianza común. *LDA* produce límites de decisión lineales, lo que se traduce en menor flexibilidad y por lo tanto menor problema de varianza. Sin embargo, si la separación de los grupos no es lineal, tendrá un bias grande. El método *QDA* produce límites cuadráticos y por lo tanto curvos, lo que aporta mayor

flexibilidad permitiendo ajustarse mejor a los datos, menor bias pero mayor riesgo de varianza.

En términos generales, *LDA* tiende a conseguir mejores clasificaciones que *QDA* cuando hay pocas observaciones con las que entrenar al modelo, escenario en el que evitar la varianza es crucial. Por contra, si se dispone de una gran cantidad de observaciones de entrenamiento o si no es asumible que existe una matriz de covarianza común entre clases, *QDA* es más adecuado.

Si se dispone de *p* predictores, calcular una matriz de covarianza común requiere estimar *p(p+1)/2* parámetros, mientras que calcular una matriz diferente para cada grupo requiere de *Kp(p+1)/2*. Para valores de *p* muy altos, la elección del método puede estar limitada por la capacidad computacional.

Bibliografía

An Introduction to Statistical Learning with Applications in R. James G., Witten D., Hastie T., Tibshirani R.

Using R With Multivariate Statistics Escrito por Randall E. Schumacker

http://halweb.uc3m.es/esp/Personal/personas/jmmarin/esp/DM/tema1dm.pdf

http://www2.stat.unibo.it/montanari/Didattica/Multivariate/Discriminant_analysis.pdf

Using discriminant analysis for multi-class classification: an experimental investigation (Tao Li, Shenghuo Zhu, Mitsunori Ogihara)

STAT 505 - Applied Multivariate Atatistical Analysis, PennState