LAB 08

TOPICS:

- Linear and Quadratic Discriminant Analysis

```
load("mcshapiro.test.RData")
```

1.

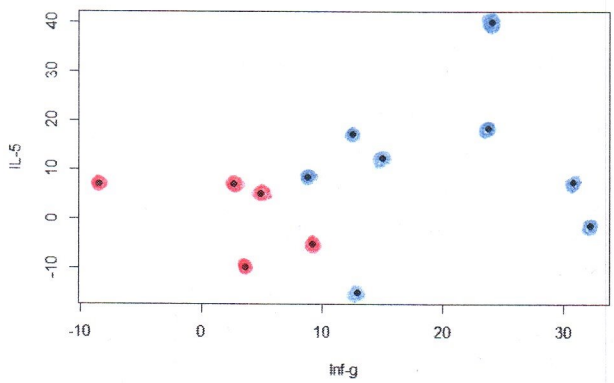
```
### ----------------------------------------------------------------------
### Example 1 (2 classes, univariate)
### ----------------------------------------------------------------------
cyto <- read.table('cytokines.txt', header=T)
head(cyto)
```

features labels

```
##        Infg       IL5 group
## 1 15.000000  12.312595     A
## 2 23.698101  18.422812     A
## 3  8.781799   8.501765     A
## 4 32.211799  -1.371131     A
## 5 30.838288   7.496666     A
## 6 12.876335 -15.159156     A
```

about a medical study: 13 patients, who have cancer, suggested to a treatment. Some of them are responding positively while others not.

```
attach(cyto)

A <- which(group=='A')   # Group A: favorable clinical outcome (the treatment has effect)
B <- which(group=='B')   # Group B: unfavorable clinical outcome (the treatment has no effect)

x11()
plot(cyto[,1], cyto[,2], pch=19, col=c(rep('blue',8),rep('red',5)), xlab='Inf-g', ylab='IL-5')
```



not responding to the treatment (B)
responding to the treatment (A)

→ idea: build a classifier to distinguish group A and group B by looking at the features

```
### Idea: we aim to find a "rule" to classify patients as Group A or Group B
### given the measurements of Inf-g and IL-5.
### In fact, we only consider the variable Infg since there is no statistical
### evidence to state that there is a difference in mean along the component IL5.

### -> Exercise: build a confidence region of level 95% for the difference
###    of the means between the two groups (independent populations)
###    and verify the previous statement.

### ----------------------------------------------------------------------
### LDA (univariate)                                   (1 dimensional LDA)
### ----------------------------------------------------------------------
# Assumptions:
# 1) if L=i, X.i ~ N(mu.i, sigma.i^2), i=A,B
# 2) sigma.A=sigma.B
# 3) c(A|B)=c(B|A) (equal misclassification costs)

# verify assumptions 1) e 2):
# 1) normality (univariate) within the groups
shapiro.test(cyto[A,1])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  cyto[A, 1]
## W = 0.90684, p-value = 0.3323
```

```
shapiro.test(cyto[B,1])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  cyto[B, 1]
## W = 0.87547, p-value = 0.2893
```

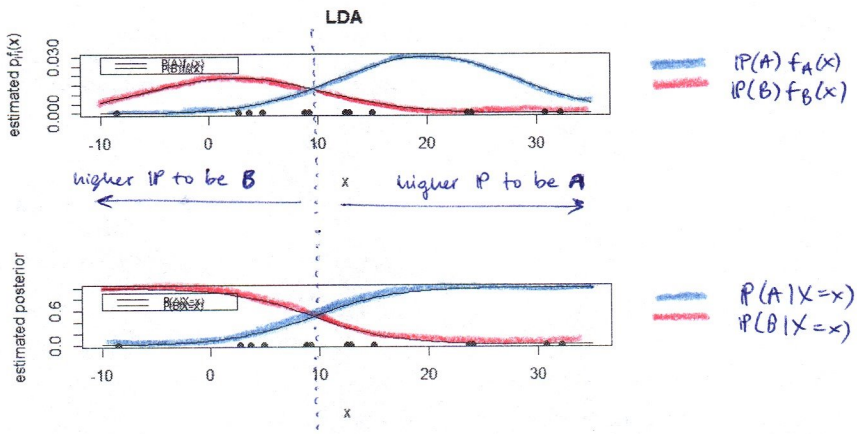
```
# 2) equal variance (univariate)
var.test(cyto[A,1],cyto[B,1])
```

H_0: the variance is the same
H_1: the variance is $\neq$

```
##
##  F test to compare two variances
##
## data:  cyto[A, 1] and cyto[B, 1]
## F = 1.8372, num df = 7, denom df = 4, p-value = 0.5812
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##   0.2024605 10.1458619
## sample estimates:
## ratio of variances
##           1.837155
```

```
nA <- length(A)
nB <- length(B)
n  <- nA + nB

# Prior probabilities (estimated from the data, no prior knowledge)
PA <- nA / n
PB <- nB / n

### ---------------------------------------------------------------------------
### Recall:
# the classification region is obtained by comparing pA*f.A and pB*f.B

MA <- mean(Infg[A])
MB <- mean(Infg[B])

SA <- var(Infg[A])
SB <- var(Infg[B])
S  <- ((nA-1) * SA + (nB-1) * SB) / (nA + nB - 2)  # pooled estimate

x <- seq(-10, 35, 0.5) # include the range of Infg

x11(width = 5)
par(mfrow=c(2,1))

plot(  x, PA * dnorm(x, MA, sqrt(S)), type='l', col='blue', ylab=expression(paste('estimated ', p[i] * f[i],'(x)')), main='L
DA')
points(x, PB * dnorm(x, MB, sqrt(S)), type='l', col='red')
points(Infg[A], rep(0, length(A)), pch=16, col='blue')
points(Infg[B], rep(0, length(B)), pch=16, col='red')
legend(-10, 0.03, legend=c(expression(paste('P(A)',f[A],'(x)')), expression(paste('P(B)',f[B],'(x)'))), col=c('blue','red'),
lty=1, cex = 0.7)

plot(  x, PA * dnorm(x, MA, sqrt(S)) / (PA * dnorm(x, MA, sqrt(S)) + PB * dnorm(x, MB, sqrt(S))), type='l', col='blue', ylab
='estimated posterior')
points(x, PB * dnorm(x, MB, sqrt(S)) / (PA * dnorm(x, MA, sqrt(S)) + PB * dnorm(x, MB, sqrt(S))), type='l', col='red')
points(Infg[A], rep(0, length(A)), pch=16, col='blue')
points(Infg[B], rep(0, length(B)), pch=16, col='red')
legend(-10, 0.9, legend=c('P(A|X=x)', 'P(B|X=x)'), col=c('blue','red'), lty=1, cex = 0.7)
```

graphical comparison



$\mathbb{P}(A) f_A(x)$
$\mathbb{P}(B) f_B(x)$

$\mathbb{P}(A|X=x)$
$\mathbb{P}(B|X=x)$

```
### end Recall
### ---------------------------------------------------------------------------

### LDA with R
library(MASS)
help(lda)
```

```
cyto.lda <- lda(group ~ Infg)
cyto.lda
```

```
## Call:
## lda(group ~ Infg)
##
## Prior probabilities of groups:
##         A         B
## 0.6153846 0.3846154
##
## Group means:
##        Infg
## A 19.997100
## B  2.357236
##
## Coefficients of linear discriminants:
##             LD1
## Infg 0.1230572
```

the default option are the prior probabilities assigned by the training dataset

```
# Note: if we don't specify the prior probabilities, they are estimated
# from the sample

# posterior probability and classification for x=0
x <- data.frame(Infg = 0)
# The command predict() returns a list containing (see the help of predict.lda):
# - the class associated with the highest posterior probability
predict(cyto.lda, x)$class
```

"lda(..)" can be used as classifier

```
## [1] B
## Levels: A B
```

```
# - the posterior probabilities for the classes
predict(cyto.lda, x)$posterior
```

```
##           A         B
## 1 0.07476761 0.9252324
```

```
# - in lda: the coordinates of the canonical analysis of Fisher
#           (Fisher's discriminant scores)
predict(cyto.lda, x)$x
```

```
##         LD1
## 1 -1.625897
```

```
x11()
plot(Infg[A], rep(0, length(A)), pch=16, col='blue', ylim=c(0,1),
     xlab='x', ylab='estimated posterior', main="LDA", xlim=range(Infg))
points(Infg[B], rep(0, length(B)), pch=16, col='red')
abline(v=0, col='grey')
points(c(0,0),c(predict(cyto.lda, data.frame(Infg = 0))$posterior),
       col=c('blue', 'red'), pch='*', cex=2.5)

# posterior probability for a grid of x's
x <- data.frame(Infg=seq(-10, 35, 0.5))

head(predict(cyto.lda, x)$posterior)
```

graphical representation

first we compute the posterior probabilities with the new datum "Infg = 0", and we obtain ● and ●

then we do the same but for a grid of points (finite grid of possible coordinates)

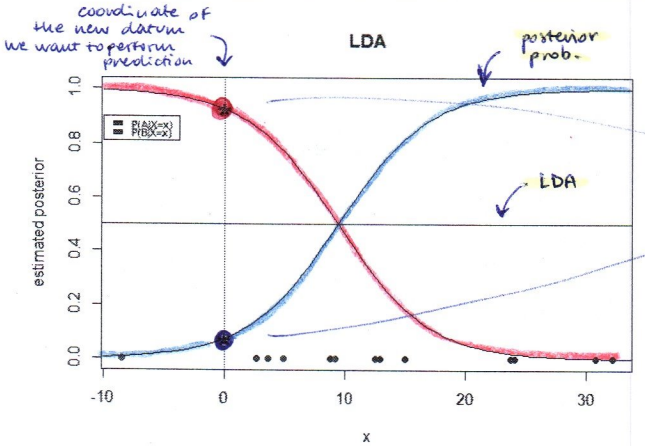
we obtain a representation of 2 continuous lines giving us the posterior probability computed for a finite grid on the x axes.

```
##              A         B
## 1 0.005558367 0.9944416
## 2 0.006347566 0.9936524
## 3 0.007248000 0.9927520
## 4 0.008275102 0.9917249
## 5 0.009446369 0.9905536
## 6 0.010781615 0.9892184
```

```
cyto.LDA.A <- predict(cyto.lda, x)$posterior[,1] # posterior probability for class A
cyto.LDA.B <- predict(cyto.lda, x)$posterior[,2] # posterior probability for class B

predict(cyto.lda, x)$class
```

```
##  [1] B B B B B B B B B B B B B B B B B B B B B B B B B B B B B B B B B B B B B
## [39] B A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A
## [77] A A A A A A A A A A A A A A A A A
## Levels: A B
```

```
lines(x[,1], cyto.LDA.A, type='l', col='blue', xlab='x', ylab='estimated posterior', main="LDA")
lines(x[,1], cyto.LDA.B, type='l', col='red')
abline(h = 0.5)
legend(-10, 0.9, legend=c('P(A|X=x)', 'P(B|X=x)'), fill=c('blue','red'), cex = 0.7)
```



P(A|X=x)
P(B|X=x)

posterior probabilities estimated with the function "predict(..)"

training data

```
dev.off()
```

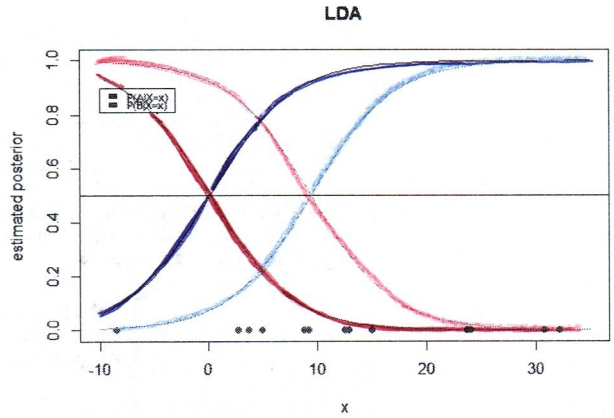
What if we want to set different prior probabilities? (suppose for example that we already know the proportion of patients which respond positively to the treatment (95%))

```
# set prior probabilities
cyto.lda.1 <- lda(group ~ Infg, prior=c(0.95,0.05))
cyto.lda.1
```

```
## Call:
## lda(group ~ Infg, prior = c(0.95, 0.05))
##
## Prior probabilities of groups:
##    A    B
## 0.95 0.05
##
## Group means:
##        Infg
## A 19.997100
## B  2.357236
##
## Coefficients of linear discriminants:
##            LD1
## Infg 0.1230572
```

What changes in the graphical representation?

```
x <- data.frame(Infg=seq(-10, 35, 0.5))

cyto.LDA.A.1 <- predict(cyto.lda.1, x)$posterior[,1] # posterior probability for class A
cyto.LDA.B.1 <- predict(cyto.lda.1, x)$posterior[,2] # posterior probability for class B

x11()
plot  (x[,1], cyto.LDA.A.1, type='l', col='blue', xlab='x', ylab='estimated posterior', main="LDA", ylim=c(0,1))
points(x[,1], cyto.LDA.B.1, type='l', col='red')
abline(h = 0.5)
legend(-10, 0.9, legend=c('P(A|X=x)', 'P(B|X=x)'), fill=c('blue','red'), cex = 0.7)
points(Infg[A], rep(0, length(A)), pch=16, col='blue')
points(Infg[B], rep(0, length(B)), pch=16, col='red')

points(x[,1], cyto.LDA.A, type='l', col='grey')
points(x[,1], cyto.LDA.B, type='l', col='grey')
```

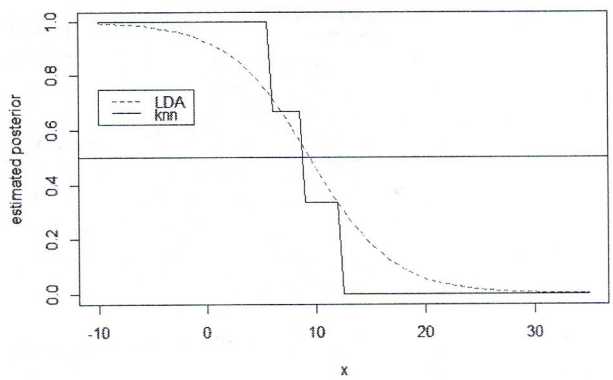


old $P(B|X=x)$
old $P(A|X=x)$
new $P(B|X=x)$
new $P(A|X=x)$

for comparison: (1 dimensional LDA vs. KNN)

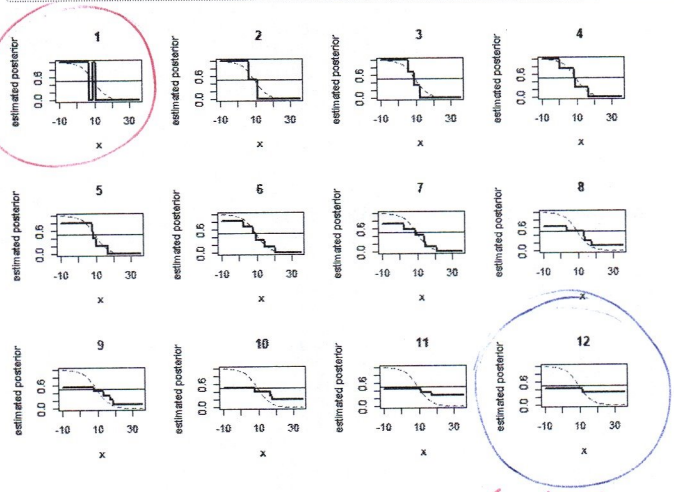
```
### ------------------------------------------------------------------------
### k-nearest neighbor classifier
### ------------------------------------------------------------------------
library(class)
help(knn)

cyto.knn <- knn(train = Infg, test = x, cl = group, k = 3, prob=T)
cyto.knn.class <- (cyto.knn == 'B')+0
cyto.knn.B <- ifelse(cyto.knn.class==1,
                     attributes(cyto.knn)$prob,
                     1 - attributes(cyto.knn)$prob)

x11()
plot(x[,1], cyto.LDA.B, type='l', col='red', lty=2, xlab='x', ylab='estimated posterior')
points(x[,1], cyto.knn.B, type='l', col='black', lty=1)
abline(h = 0.5)
legend(-10, 0.75, legend=c('LDA','knn'), lty=c(2,1), col=c('red','black'))
```

we have to provide:
- training set
- test set
- labels
- how many neighbours to take in account
- "prob = T" means that in the output we'll see the proportion of neighbors of the same label




```
# Let's change k
x11(width = 28, height = 21)
par(mfrow=c(3,4))
for(k in 1:12)
{
  cyto.knn <- knn(train = Infg, test = x, cl = group, k = k, prob=T)
  cyto.knn.class <- (cyto.knn == 'B')+0
  cyto.knn.B <- ifelse(cyto.knn.class==1, attributes(cyto.knn)$prob, 1 - attributes(cyto.knn)$prob)

  plot(x[,1], cyto.LDA.B, type='l', col='red', lty=2, xlab='x', ylab='estimated posterior', main=k)
  points(x[,1], cyto.knn.B, type='l', col='black', lty=1, lwd=2)
  abline(h = 0.5)
}
```

Overfitting (good bias, bad variance)



(good variance bad bias)

the higher the **k** the more we've considering to decide in which group the observations will be ⟹ the higher the **k** the closer to the overall mean

```
detach(cyto)
graphics.off()
```

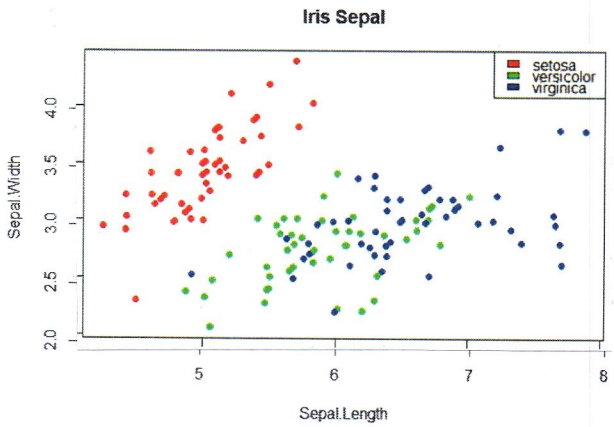
2.

```
### ------------------------------------------------------------------------
### Example 2 (3 classes, bivariate)
### ------------------------------------------------------------------------
help(iris)
### We consider only the first two variables, Sepal.Length and Sepal.Width
### (p=2, g=3); we aim to build a classifier based on the characteristic
### of the sepal that identifies the iris species.
```

```
attach(iris)
species.name <- factor(Species, labels=c('setosa','versicolor','virginica'))
g=3

i1 <- which(species.name=='setosa')
i2 <- which(species.name=='versicolor')
i3 <- which(species.name=='virginica')

n1 <- length(i1)
n2 <- length(i2)
n3 <- length(i3)
n <- n1+n2+n3

detach(iris)
```

```
iris2 <- iris[,1:2]

# Jittering
set.seed(1)
iris2 <- iris2 + cbind(rnorm(150, sd=0.025))    # jittering

# plot the data
x11()

plot(iris2, main='Iris Sepal', xlab='Sepal.Length', ylab='Sepal.Width', pch=19)
points(iris2[i1,], col='red', pch=19)
points(iris2[i2,], col='green', pch=19)
points(iris2[i3,], col='blue', pch=19)
legend("topright", legend=levels(species.name), fill=c('red','green','blue'))
```

→ we're adding some noise to the features (it's used to check if the data is robust w.r.t. small perturbations: if by jittering we obtain a whole new classifier then we're overfitting)




```
dev.off()
```

```
m <-  colMeans(iris2)
m1 <- colMeans(iris2[i1,])
m2 <- colMeans(iris2[i2,])
m3 <- colMeans(iris2[i3,])

S1 <- cov(iris2[i1,])
S2 <- cov(iris2[i2,])
S3 <- cov(iris2[i3,])
Sp  <- ((n1-1)*S1+(n2-1)*S2+(n3-1)*S3)/(n-g)
```

← we want to use LDA ⇒ we need S_{pooled}

```
# One-way MANOVA (See LAB 8)
fit <- manova(as.matrix(iris2) ~ species.name)
summary.manova(fit,test="Wilks")
```

```
##               Df   Wilks approx F num Df den Df    Pr(>F)
## species.name   2 0.16695   105.66      4    292 < 2.2e-16 ***
## Residuals    147
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We check that there is a significant effect of the species on the features (ANOVA can be the first step to check if it makes sense to perform a discriminant analysis using those features to predict the species' mean (obviously in SUPERVISED SETTING))

```
# Linear Discriminant Analysis (LDA)
lda.iris <- lda(iris2, species.name)
lda.iris
```

```
## Call:
## lda(iris2, species.name)
##
## Prior probabilities of groups:
##     setosa versicolor  virginica
##  0.3333333  0.3333333  0.3333333
##
## Group means:
##            Sepal.Length Sepal.Width
## setosa         5.008511    3.430511
## versicolor     5.938933    2.772933
## virginica      6.584188    2.970188
##
## Coefficients of linear discriminants:
##                    LD1        LD2
## Sepal.Length -2.137114 -0.8197481
## Sepal.Width   2.789529 -2.0844258
##
## Proportion of trace:
##    LD1    LD2
## 0.9644 0.0356
```

← computed from the Training data

this part concerns the Fisher interpretation of linear discriminant analysis, in particular
- gives us the directions of Fisher analysis (the first column is the first direction, --).

The proportional trace gives us the ratio between each eigenvalue and the trace of the matrix used to obtain this direction

```
# "coefficients of linear discriminants" and "proportion of trace":
# Fisher discriminant analysis.
# In particular:
# - coefficients of linear discriminants: versors of the canonical directions
#   [to be read column-wise]
# - proportion of trace: proportion of variance explained by the corresponding
#   canonical direction

Lda.iris <- predict(lda.iris, iris2)
names(Lda.iris)
```

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

```
# Estimate of AER:
# 1) APER
# 2) estimate of AER by cross-validation

# 1) Compute the APER
Lda.iris$class   # assigned classes
```

Estimate of the error (Actual Error Rate)

(Apparent Error rate

```
##   [1] setosa     setosa     setosa     setosa     setosa     setosa
##   [7] setosa     setosa     setosa     setosa     setosa     setosa
##  [13] setosa     setosa     setosa     setosa     setosa     setosa
##  [19] setosa     setosa     setosa     setosa     setosa     setosa
##  [25] setosa     setosa     setosa     setosa     setosa     setosa
##  [31] setosa     setosa     setosa     setosa     setosa     setosa
##  [37] setosa     setosa     setosa     setosa     setosa     versicolor
##  [43] setosa     setosa     setosa     setosa     setosa     setosa
##  [49] setosa     setosa     virginica  virginica  virginica  versicolor
##  [55] virginica  versicolor versicolor versicolor virginica  versicolor
##  [61] versicolor versicolor versicolor versicolor versicolor virginica
##  [67] versicolor versicolor virginica  versicolor versicolor versicolor
##  [73] virginica  versicolor virginica  virginica  virginica  virginica
##  [79] versicolor versicolor versicolor versicolor versicolor versicolor
##  [85] versicolor versicolor virginica  virginica  versicolor versicolor
##  [91] versicolor versicolor versicolor versicolor versicolor versicolor
##  [97] versicolor versicolor versicolor versicolor versicolor versicolor
## [103] virginica  virginica  virginica  virginica  versicolor virginica
## [109] virginica  virginica  virginica  virginica  virginica  versicolor
## [115] versicolor virginica  virginica  virginica  virginica  versicolor
## [121] virginica  versicolor virginica  virginica  virginica  virginica
## [127] versicolor versicolor virginica  virginica  virginica  virginica
## [133] virginica  virginica  versicolor virginica  versicolor virginica
## [139] versicolor virginica  virginica  virginica  versicolor virginica
## [145] virginica  virginica  virginica  virginica  versicolor versicolor
## Levels: setosa versicolor virginica
```

```
species.name     # true labels
```

```
##   [1] setosa     setosa     setosa     setosa     setosa     setosa
##   [7] setosa     setosa     setosa     setosa     setosa     setosa
##  [13] setosa     setosa     setosa     setosa     setosa     setosa
##  [19] setosa     setosa     setosa     setosa     setosa     setosa
##  [25] setosa     setosa     setosa     setosa     setosa     setosa
##  [31] setosa     setosa     setosa     setosa     setosa     setosa
##  [37] setosa     setosa     setosa     setosa     setosa     setosa
##  [43] setosa     setosa     setosa     setosa     setosa     setosa
##  [49] setosa     setosa     versicolor versicolor versicolor versicolor
##  [55] versicolor versicolor versicolor versicolor versicolor versicolor
##  [61] versicolor versicolor versicolor versicolor versicolor versicolor
##  [67] versicolor versicolor versicolor versicolor versicolor versicolor
##  [73] versicolor versicolor versicolor versicolor versicolor versicolor
##  [79] versicolor versicolor versicolor versicolor versicolor versicolor
##  [85] versicolor versicolor versicolor versicolor versicolor versicolor
##  [91] versicolor versicolor versicolor versicolor versicolor versicolor
##  [97] versicolor versicolor versicolor versicolor virginica  virginica
## [103] virginica  virginica  virginica  virginica  virginica  virginica
## [109] virginica  virginica  virginica  virginica  virginica  virginica
## [115] virginica  virginica  virginica  virginica  virginica  virginica
## [121] virginica  virginica  virginica  virginica  virginica  virginica
## [127] virginica  virginica  virginica  virginica  virginica  virginica
## [133] virginica  virginica  virginica  virginica  virginica  virginica
## [139] virginica  virginica  virginica  virginica  virginica  virginica
## [145] virginica  virginica  virginica  virginica  virginica  virginica
## Levels: setosa versicolor virginica
```

```
table(class.true=species.name, class.assigned=Lda.iris$class)
```

(misclassification table)

```
##             class.assigned
## class.true   setosa versicolor virginica
##   setosa         49          1         0
##   versicolor      0         36        14
##   virginica       0         15        35
```

(1) (14) (15) mistakes

```
errors <- (Lda.iris$class != species.name)
errors
```

```
##   [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [25] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [37] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
##  [49] FALSE FALSE  TRUE  TRUE  TRUE FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE
##  [61] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE  TRUE FALSE FALSE FALSE
##  [73]  TRUE FALSE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
##  [85] FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [97] FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE
## [109] FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE  TRUE
## [121] FALSE  TRUE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE
## [133] FALSE FALSE  TRUE FALSE  TRUE FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE
## [145] FALSE FALSE FALSE FALSE  TRUE  TRUE
```

```
sum(errors)
```

```
## [1] 30
```

```
length(species.name)
```

```
## [1] 150
```

```
APER   <- sum(errors)/length(species.name)
APER
```

$= \frac{\#\text{ mistakes}}{\#\text{ data}}$

```
## [1] 0.2
```

```
(1+14+15)/150
```

```
## [1] 0.2
```

```
# Remark: this is correct only if we estimate the prior with the empirical
#         frequences! Otherwise:
# prior <- c(1/3,1/3,1/3)
# G <- 3
# misc <- table(class.true=species.name, class.assigned=lda.iris$class)
# APER <- 0
# for(g in 1:G)
#   APER <- APER + sum(misc[g,-g])/sum(misc[g,]) * prior[g]

# 2) Compute the estimate of the AER by leave-one-out cross-validation
```

otherwise we have to correct the own taking to account the priors

```
### Recall:

errors_CV <- 0
for(i in 1:150){
  LdaCV.i <- lda(iris2[-i,], species.name[-i], prior=c(50,50,50)/150)
  errors_CV <- errors_CV + as.numeric(predict(LdaCV.i,iris2[i,])$class != species.name[i])
}
errors_CV
```

```
## [1] 30
```

```
AERCV   <- sum(errors_CV)/length(species.name)
AERCV
```

```
## [1] 0.2
```

```
####

# with R:
LdaCV.iris <- lda(iris2, species.name, CV=TRUE)
```

← specify CV = TRUE

```
LdaCV.iris$class
```

```
##   [1] setosa     setosa     setosa     setosa     setosa     setosa
##   [7] setosa     setosa     setosa     setosa     setosa     setosa
##  [13] setosa     setosa     setosa     setosa     setosa     setosa
##  [19] setosa     setosa     setosa     setosa     setosa     setosa
##  [25] setosa     setosa     setosa     setosa     setosa     setosa
##  [31] setosa     setosa     setosa     setosa     setosa     setosa
##  [37] setosa     setosa     setosa     setosa     setosa     versicolor
##  [43] setosa     setosa     setosa     setosa     setosa     setosa
##  [49] setosa     setosa     virginica  virginica  virginica  versicolor
##  [55] virginica  versicolor versicolor versicolor virginica  versicolor
##  [61] versicolor versicolor versicolor versicolor versicolor virginica
##  [67] versicolor versicolor virginica  versicolor versicolor versicolor
##  [73] virginica  versicolor virginica  virginica  virginica  virginica
##  [79] versicolor versicolor versicolor versicolor versicolor versicolor
##  [85] versicolor versicolor virginica  virginica  versicolor versicolor
##  [91] versicolor versicolor versicolor versicolor versicolor versicolor
##  [97] versicolor versicolor versicolor versicolor versicolor versicolor
## [103] virginica  virginica  virginica  virginica  versicolor virginica
## [109] virginica  virginica  virginica  virginica  virginica  versicolor
## [115] versicolor virginica  virginica  virginica  virginica  versicolor
## [121] virginica  versicolor virginica  virginica  virginica  virginica
## [127] versicolor versicolor virginica  virginica  virginica  virginica
## [133] virginica  virginica  versicolor virginica  versicolor virginica
## [139] versicolor virginica  virginica  virginica  versicolor virginica
## [145] virginica  virginica  virginica  virginica  versicolor versicolor
## Levels: setosa versicolor virginica
```

```
species.name
```

```
##   [1] setosa     setosa     setosa     setosa     setosa     setosa
##   [7] setosa     setosa     setosa     setosa     setosa     setosa
##  [13] setosa     setosa     setosa     setosa     setosa     setosa
##  [19] setosa     setosa     setosa     setosa     setosa     setosa
##  [25] setosa     setosa     setosa     setosa     setosa     setosa
##  [31] setosa     setosa     setosa     setosa     setosa     setosa
##  [37] setosa     setosa     setosa     setosa     setosa     setosa
##  [43] setosa     setosa     setosa     setosa     setosa     setosa
##  [49] setosa     setosa     versicolor versicolor versicolor versicolor
##  [55] versicolor versicolor versicolor versicolor versicolor versicolor
##  [61] versicolor versicolor versicolor versicolor versicolor versicolor
##  [67] versicolor versicolor versicolor versicolor versicolor versicolor
##  [73] versicolor versicolor versicolor versicolor versicolor versicolor
##  [79] versicolor versicolor versicolor versicolor versicolor versicolor
##  [85] versicolor versicolor versicolor versicolor versicolor versicolor
##  [91] versicolor versicolor versicolor versicolor versicolor versicolor
##  [97] versicolor versicolor versicolor versicolor virginica  virginica
## [103] virginica  virginica  virginica  virginica  virginica  virginica
## [109] virginica  virginica  virginica  virginica  virginica  virginica
## [115] virginica  virginica  virginica  virginica  virginica  virginica
## [121] virginica  virginica  virginica  virginica  virginica  virginica
## [127] virginica  virginica  virginica  virginica  virginica  virginica
## [133] virginica  virginica  virginica  virginica  virginica  virginica
## [139] virginica  virginica  virginica  virginica  virginica  virginica
## [145] virginica  virginica  virginica  virginica  virginica  virginica
## Levels: setosa versicolor virginica
```

```
table(class.true=species.name, class.assignedCV=LdaCV.iris$class)
```

```
##             class.assignedCV
## class.true   setosa versicolor virginica
##   setosa         49          1         0
##   versicolor      0         36        14
##   virginica       0         15        35
```

this is obtained with the leave-one-out cross validation

```
errorsCV <- (LdaCV.iris$class != species.name)
errorsCV
```

```
##   [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [25] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [37] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
##  [49] FALSE FALSE  TRUE  TRUE  TRUE FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE
##  [61] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE  TRUE FALSE FALSE FALSE
##  [73]  TRUE FALSE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
##  [85] FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [97] FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE
## [109] FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE  TRUE
## [121] FALSE  TRUE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE
## [133] FALSE FALSE  TRUE FALSE  TRUE FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE
## [145] FALSE FALSE FALSE FALSE  TRUE  TRUE
```

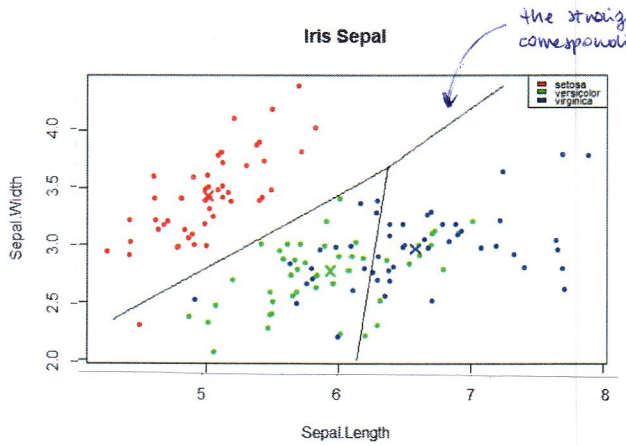
```
sum(errorsCV)
```

```
## [1] 30
```

```
AERCV   <- sum(errorsCV)/length(species.name)
AERCV
```

```
## [1] 0.2
```

```
# Remark: correct only if we estimate the priors through the sample frequencies!

# Plot the partition induced by LDA

x11()
plot(iris2, main='Iris Sepal', xlab='Sepal.Length', ylab='Sepal.Width', pch=20)
points(iris2[i1,], col='red', pch=20)
points(iris2[i2,], col='green', pch=20)
points(iris2[i3,], col='blue', pch=20)
legend("topright", legend=levels(species.name), fill=c('red','green','blue'), cex=.7)

points(lda.iris$means, pch=4,col=c('red','green','blue') , lwd=2, cex=1.5)

x  <- seq(min(iris[,1]), max(iris[,1]), length=200)
y  <- seq(min(iris[,2]), max(iris[,2]), length=200)
xy <- expand.grid(Sepal.Length=x, Sepal.Width=y)

z  <- predict(lda.iris, xy)$post  # these are P_i*f_i(x,y)
z1 <- z[,1] - pmax(z[,2], z[,3])  # P_1*f_1(x,y)-max{P_j*f_j(x,y)}
z2 <- z[,2] - pmax(z[,1], z[,3])  # P_2*f_2(x,y)-max{P_j*f_j(x,y)}
z3 <- z[,3] - pmax(z[,1], z[,2])  # P_3*f_3(x,y)-max{P_j*f_j(x,y)}

# Plot the contour line of Level (Levels=0) of z1, z2, z3:
# P_i*f_i(x,y)-max{P_j*f_j(x,y)}=0 i.e., boundary between R.i and R.j
# where j realizes the max.
contour(x, y, matrix(z1, 200), levels=0, drawlabels=F, add=T)
contour(x, y, matrix(z2, 200), levels=0, drawlabels=F, add=T)
contour(x, y, matrix(z3, 200), levels=0, drawlabels=F, add=T)
```



the straight lines are the ones corresponding with the LDA classifier (the partition is the one corresponding to the LDA classifier)

← from this representation we can see the errors

```
library(rgl)
library(mvtnorm)
open3d()
```

```
points3d(iris2[i1,1], iris2[i1,2], 0, col='red', pch=15)
points3d(iris2[i2,1], iris2[i3,2], 0, col='green', pch=15)
points3d(iris2[i3,1], iris2[i2,2], 0, col='blue', pch=15)
surface3d(x,y,matrix(dmvnorm(xy, m1, Sp) / 3, 50), alpha=0.4, color='red')
surface3d(x,y,matrix(dmvnorm(xy, m2, Sp) / 3, 50), alpha=0.4, color='green', add=T)
surface3d(x,y,matrix(dmvnorm(xy, m3, Sp) / 3, 50), alpha=0.4, color='blue', add=T)
box3d()
a = scene3d()
rgl.close()
x11()
rglwidget(a)
```

distributions of the posterior probabilities: the intersections of the surfaces of these distribution generate the straigth lines above (which divide (partition) the plane)

← remember that these are not the linear discriminant functions! these are posterior probabilities (the linear discriminant functions are in fact linear)

These two images are the corresponding to what we saw in **1** dimension:

(in this case the 3D is the posterior probabilities, the plot above represent the LDA: the intersections of the probabilities (as in the previous example) generates the LDA)

```
### ------------------------------------------------------------------------
### Quadratic Discriminand Analysis (QDA)
### ------------------------------------------------------------------------

help(qda)

qda.iris <- qda(iris2, species.name)
qda.iris
```

inputs: matrix of features + labels

```
## Call:
## qda(iris2, species.name)
##
## Prior probabilities of groups:
##     setosa versicolor  virginica
##  0.3333333  0.3333333  0.3333333
##
## Group means:
##            Sepal.Length Sepal.Width
## setosa        5.008511    3.430511
## versicolor    5.938933    2.772933
## virginica     6.584188    2.970188
```

```
Qda.iris <- predict(qda.iris, iris2)

# compute the APER
Qda.iris$class
```

```
##   [1] setosa     setosa     setosa     setosa     setosa     setosa
##   [7] setosa     setosa     setosa     setosa     setosa     setosa
##  [13] setosa     setosa     setosa     setosa     setosa     setosa
##  [19] setosa     setosa     setosa     setosa     setosa     setosa
##  [25] setosa     setosa     setosa     setosa     setosa     setosa
##  [31] setosa     setosa     setosa     setosa     setosa     setosa
##  [37] setosa     setosa     setosa     setosa     setosa     versicolor
##  [43] setosa     setosa     setosa     setosa     setosa     setosa
##  [49] setosa     setosa     virginica  virginica  virginica  versicolor
##  [55] virginica  versicolor virginica  versicolor virginica  versicolor
##  [61] versicolor versicolor versicolor versicolor versicolor virginica
##  [67] versicolor versicolor versicolor versicolor versicolor versicolor
##  [73] versicolor versicolor virginica  virginica  virginica  virginica
##  [79] versicolor versicolor versicolor versicolor versicolor versicolor
##  [85] versicolor versicolor virginica  virginica  versicolor versicolor
##  [91] versicolor versicolor versicolor versicolor versicolor versicolor
##  [97] versicolor versicolor versicolor versicolor virginica  versicolor
## [103] virginica  versicolor virginica  virginica  versicolor virginica
## [109] virginica  virginica  virginica  virginica  virginica  versicolor
## [115] versicolor virginica  virginica  virginica  virginica  versicolor
## [121] virginica  versicolor virginica  versicolor virginica  virginica
## [127] versicolor versicolor virginica  virginica  virginica  virginica
## [133] virginica  versicolor versicolor virginica  virginica  virginica
## [139] versicolor virginica  virginica  virginica  versicolor virginica
## [145] virginica  virginica  virginica  virginica  virginica  versicolor
## Levels: setosa versicolor virginica
```

```
species.name
```

```
##   [1] setosa     setosa     setosa     setosa     setosa     setosa
##   [7] setosa     setosa     setosa     setosa     setosa     setosa
##  [13] setosa     setosa     setosa     setosa     setosa     setosa
##  [19] setosa     setosa     setosa     setosa     setosa     setosa
##  [25] setosa     setosa     setosa     setosa     setosa     setosa
##  [31] setosa     setosa     setosa     setosa     setosa     setosa
##  [37] setosa     setosa     setosa     setosa     setosa     setosa
##  [43] setosa     setosa     setosa     setosa     setosa     setosa
##  [49] setosa     setosa     versicolor versicolor versicolor versicolor
##  [55] versicolor versicolor versicolor versicolor versicolor versicolor
##  [61] versicolor versicolor versicolor versicolor versicolor versicolor
##  [67] versicolor versicolor versicolor versicolor versicolor versicolor
##  [73] versicolor versicolor versicolor versicolor versicolor versicolor
##  [79] versicolor versicolor versicolor versicolor versicolor versicolor
##  [85] versicolor versicolor versicolor versicolor versicolor versicolor
##  [91] versicolor versicolor versicolor versicolor versicolor versicolor
##  [97] versicolor versicolor versicolor versicolor virginica  virginica
## [103] virginica  virginica  virginica  virginica  virginica  virginica
## [109] virginica  virginica  virginica  virginica  virginica  virginica
## [115] virginica  virginica  virginica  virginica  virginica  virginica
## [121] virginica  virginica  virginica  virginica  virginica  virginica
## [127] virginica  virginica  virginica  virginica  virginica  virginica
## [133] virginica  virginica  virginica  virginica  virginica  virginica
## [139] virginica  virginica  virginica  virginica  virginica  virginica
## [145] virginica  virginica  virginica  virginica  virginica  virginica
## Levels: setosa versicolor virginica
```

```
table(class.true=species.name, class.assigned=Qda.iris$class)
```

```
##             class.assigned
## class.true   setosa versicolor virginica
##   setosa         49          1         0
##   versicolor      0         37        13
##   virginica       0         15        35
```

```
errorsq <- (Qda.iris$class != species.name)
errorsq
```

```
##   [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [25] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [37] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
##  [49] FALSE FALSE  TRUE  TRUE  TRUE FALSE  TRUE FALSE  TRUE FALSE  TRUE FALSE
##  [61] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
##  [73] FALSE FALSE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
##  [85] FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [97] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE FALSE FALSE  TRUE FALSE
## [109] FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE  TRUE
## [121] FALSE  TRUE FALSE  TRUE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE
## [133] FALSE  TRUE  TRUE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE
## [145] FALSE FALSE FALSE FALSE FALSE  TRUE
```

```
APERq   <- sum(errorsq)/length(species.name)
APERq
```

```
## [1] 0.1933333
```

```
(15+13+1)/150
```

```
## [1] 0.1933333
```

```
# Remark: correct only if we estimate the priors through the sample frequencies!

# Compute the estimate of the AER by Leave-out-out cross-validation
QdaCV.iris <- qda(iris2, species.name, CV=T)
QdaCV.iris$class
```

```
##   [1] setosa     setosa     setosa     setosa     setosa     setosa
##   [7] setosa     setosa     setosa     setosa     setosa     setosa
##  [13] setosa     setosa     setosa     setosa     setosa     setosa
##  [19] setosa     setosa     setosa     setosa     setosa     setosa
##  [25] setosa     setosa     setosa     setosa     setosa     setosa
##  [31] setosa     setosa     setosa     setosa     setosa     setosa
##  [37] setosa     setosa     setosa     setosa     setosa     versicolor
##  [43] setosa     setosa     setosa     setosa     setosa     setosa
##  [49] setosa     setosa     virginica  virginica  virginica  versicolor
##  [55] virginica  versicolor virginica  versicolor virginica  versicolor
##  [61] versicolor versicolor versicolor versicolor versicolor virginica
##  [67] versicolor versicolor virginica  versicolor versicolor versicolor
##  [73] virginica  versicolor virginica  virginica  virginica  virginica
##  [79] versicolor versicolor versicolor versicolor versicolor versicolor
##  [85] versicolor virginica  virginica  virginica  versicolor versicolor
##  [91] versicolor versicolor versicolor versicolor versicolor versicolor
##  [97] versicolor versicolor versicolor versicolor versicolor versicolor
## [103] virginica  versicolor virginica  virginica  versicolor virginica
## [109] virginica  virginica  virginica  virginica  virginica  versicolor
## [115] versicolor virginica  virginica  virginica  virginica  versicolor
## [121] virginica  versicolor virginica  versicolor virginica  virginica
## [127] versicolor versicolor virginica  virginica  virginica  virginica
## [133] virginica  versicolor versicolor virginica  virginica  virginica
## [139] versicolor virginica  virginica  virginica  versicolor virginica
## [145] virginica  virginica  versicolor virginica  versicolor versicolor
## Levels: setosa versicolor virginica
```

```
species.name
```

```
##   [1] setosa     setosa     setosa     setosa     setosa     setosa
##   [7] setosa     setosa     setosa     setosa     setosa     setosa
##  [13] setosa     setosa     setosa     setosa     setosa     setosa
##  [19] setosa     setosa     setosa     setosa     setosa     setosa
##  [25] setosa     setosa     setosa     setosa     setosa     setosa
##  [31] setosa     setosa     setosa     setosa     setosa     setosa
##  [37] setosa     setosa     setosa     setosa     setosa     setosa
##  [43] setosa     setosa     setosa     setosa     setosa     setosa
##  [49] setosa     setosa     versicolor versicolor versicolor versicolor
##  [55] versicolor versicolor versicolor versicolor versicolor versicolor
##  [61] versicolor versicolor versicolor versicolor versicolor versicolor
##  [67] versicolor versicolor versicolor versicolor versicolor versicolor
##  [73] versicolor versicolor versicolor versicolor versicolor versicolor
##  [79] versicolor versicolor versicolor versicolor versicolor versicolor
##  [85] versicolor versicolor versicolor versicolor versicolor versicolor
##  [91] versicolor versicolor versicolor versicolor versicolor versicolor
##  [97] versicolor versicolor versicolor versicolor virginica  virginica
## [103] virginica  virginica  virginica  virginica  virginica  virginica
## [109] virginica  virginica  virginica  virginica  virginica  virginica
## [115] virginica  virginica  virginica  virginica  virginica  virginica
## [121] virginica  virginica  virginica  virginica  virginica  virginica
## [127] virginica  virginica  virginica  virginica  virginica  virginica
## [133] virginica  virginica  virginica  virginica  virginica  virginica
## [139] virginica  virginica  virginica  virginica  virginica  virginica
## [145] virginica  virginica  virginica  virginica  virginica  virginica
## Levels: setosa versicolor virginica
```

```
table(class.true=species.name, class.assignedCV=QdaCV.iris$class)
```

```
##             class.assignedCV
## class.true   setosa versicolor virginica
##   setosa         49          1         0
##   versicolor      0         34        16
##   virginica       0         18        32
```

```
errorsqCV <- (QdaCV.iris$class != species.name)
errorsqCV
```

```
##   [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [25] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [37] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [49] FALSE FALSE  TRUE  TRUE  TRUE FALSE  TRUE FALSE  TRUE FALSE  TRUE FALSE FALSE
##  [61] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE
##  [73]  TRUE FALSE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [85] FALSE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [97] FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE
## [109] FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE
## [121] FALSE  TRUE FALSE  TRUE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE
## [133] FALSE  TRUE  TRUE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE FALSE
## [145] FALSE FALSE  TRUE FALSE  TRUE  TRUE
```

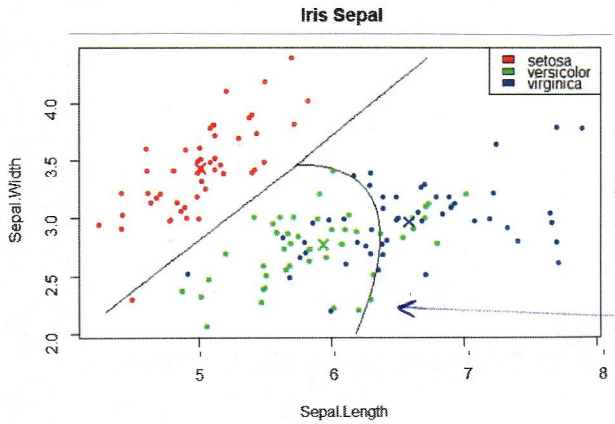
```
AERqCV  <- sum(errorsqCV)/length(species.name)
AERqCV
```

```
## [1] 0.2333333
```

QDA and LDA have comparable errors, in this case is more convenient to use LDA because it's more robust (LDA is based on the estimation of fewer parameters).

It's not that since QDA is more flexible then it's always convenient to choose it over LDA: QDA uses the data to estimate 3 covariances instead of 1 so we have more flexibility in fitting 3 covariances (so we're paying attention to bias) but these 3 estimates will be much more variable because we're using $\frac{1}{3}$ of the data for estimating each (instead the entire data for estimating just 1). It's always the bias-variance trade-off.

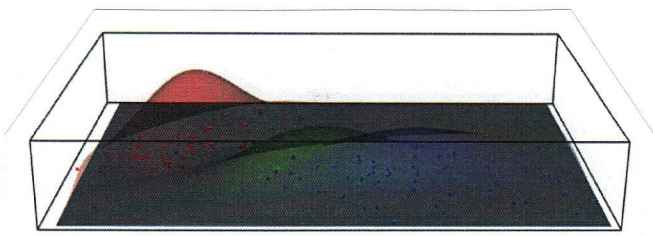
```
# Remark: correct only if we estimate the priors through the sample frequencies!

# Plot the partition induced by QDA
x11()
plot(iris2, main='Iris Sepal', xlab='Sepal.Length', ylab='Sepal.Width', pch=20)
points(iris2[i1,], col='red', pch=20)
points(iris2[i2,], col='green', pch=20)
points(iris2[i3,], col='blue', pch=20)
legend("topright", legend=levels(species.name), fill=c('red','green','blue'))

points(qda.iris$means, col=c('red','green','blue'), pch=4, lwd=2, cex=1.5)

x  <- seq(min(iris[,1]), max(iris[,1]), length=200)
y  <- seq(min(iris[,2]), max(iris[,2]), length=200)
xy <- expand.grid(Sepal.Length=x, Sepal.Width=y)

z  <- predict(qda.iris, xy)$post
z1 <- z[,1] - pmax(z[,2], z[,3])
z2 <- z[,2] - pmax(z[,1], z[,3])
z3 <- z[,3] - pmax(z[,1], z[,2])

contour(x, y, matrix(z1, 200), levels=0, drawlabels=F, add=T)
contour(x, y, matrix(z2, 200), levels=0, drawlabels=F, add=T)
contour(x, y, matrix(z3, 200), levels=0, drawlabels=F, add=T)
```



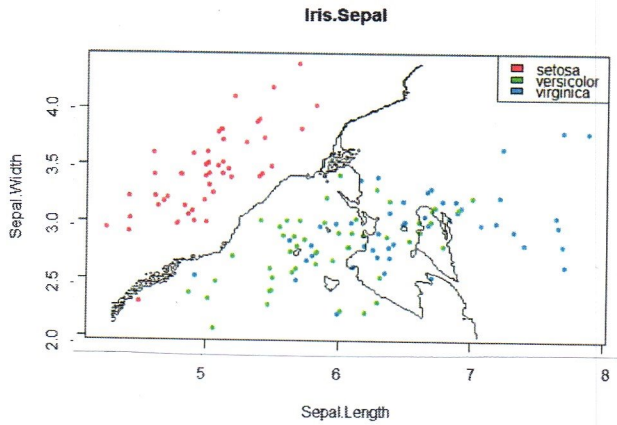
quadratic shape

```
open3d()
```

```
points3d(iris2[i1,1], iris2[i1,2], 0, col='red', pch=15)
points3d(iris2[i2,1], iris2[i3,2], 0, col='green', pch=15)
points3d(iris2[i3,1], iris2[i2,2], 0, col='blue', pch=15)
surface3d(x,y,matrix(dmvnorm(xy, m1, S1) / 3, 50), alpha=0.4, color='red')
surface3d(x,y,matrix(dmvnorm(xy, m2, S2) / 3, 50), alpha=0.4, color='green', add=T)
surface3d(x,y,matrix(dmvnorm(xy, m3, S3) / 3, 50), alpha=0.4, color='blue', add=T)
box3d()
a = scene3d()
rgl.close()
x11()
rglwidget(a)
```



```
### ------------------------------------------------------------------------
### knn-classifier
### ------------------------------------------------------------------------
# Plot the partition induced by knn

k <- 7

x11()
plot(iris2, main='Iris.Sepal', xlab='Sepal.Length', ylab='Sepal.Width', pch=20)
points(iris2[i1,], col=2, pch=20)
points(iris2[i3,], col=4, pch=20)
points(iris2[i2,], col=3, pch=20)
legend("topright", legend=levels(species.name), fill=c(2,3,4))

x  <- seq(min(iris[,1]), max(iris[,1]), length=200)
y  <- seq(min(iris[,2]), max(iris[,2]), length=200)
xy <- expand.grid(Sepal.Length=x, Sepal.Width=y)

iris.knn <- knn(train = iris2, test = xy, cl = iris$Species, k = k)

z  <- as.numeric(iris.knn)

contour(x, y, matrix(z, 200), levels=c(1.5, 2.5), drawlabels=F, add=T)
```



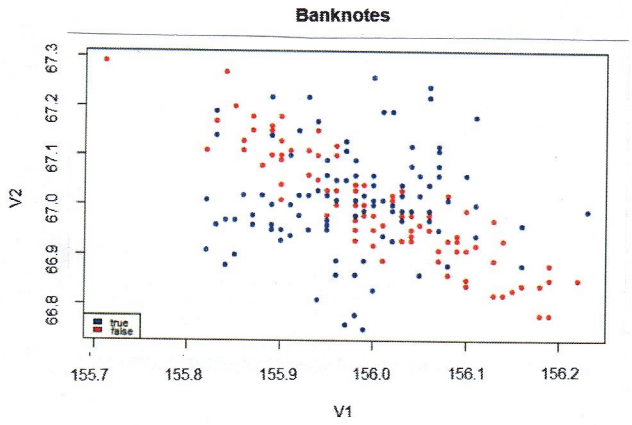
← much more flexibility but we're paying in terms of variance

(**Note**: with $k = 2$ we're adapting even more to the data: disaster)

```
graphics.off()
```

```
### ------------------------------------------------------------------------
### Problem 2 of 9/09/2009
### ------------------------------------------------------------------------
# The vending machines of the Exxon fuel contain optical detectors able
# to measure the size of the banknotes inserted. Knowing that 0.1% of the
# 10$ banknotes in circulation are counterfeit, Exxon would like to implement a
# software to identify false 10$ banknotes, as to minimize the economic losses.
# Assuming that:
#  - both the populations of real and false banknotes follow a normal
#    distribution (with different mean and covariance matrices);
#  - accepting a false banknote leads to an economic loss of 10$;
#  - rejecting a true banknote brings a economic loss quantifiable in 5 cents;
# satisfy the following requests of the Exxon:
# a) build an appropriate classifier, estimating the unknown parameters
#    starting from the two datasets moneytrue.txt and moneyfalse.txt, containing
#    data about 100 true banknotes and 100 counterfeit banknotes (in mm).
#    Qualitatively show the two classification regions in a graph;
# b) calculate the APER of the classifier and, based on the APER, estimate the
#    expected economic damage of the classifier;
# c) what is the estimated probability that the first 10$ banknote inserted in the
#    machine is rejected?

true <- read.table('moneytrue.txt',header=TRUE)
false <- read.table('moneyfalse.txt',header=TRUE)

banknotes <- rbind(true,false)
vf <- factor(rep(c('true','false'),each=100), levels=c('true','false'))

x11()
plot(banknotes[,1:2], main='Banknotes', xlab='V1', ylab='V2', pch=20)
points(false, col='red', pch=20)
points(true, col='blue', pch=20)
legend('bottomleft', legend=levels(vf), fill=c('blue','red'), cex=.7)
```

] we also have a cost function



this shows (more or less) that the difference between the two groups is expressed as difference in the covariance structure (the disperstion of the two population is characterized by 2 different covariate structure)

```
# question a)
mcshapiro.test(true)
```

```
## $Wmin
## [1] 0.9868194
##
## $pvalue
## [1] 0.5424
##
## $devst
## [1] 0.00996398
##
## $sim
## [1] 2500
```

```
mcshapiro.test(false)
```

```
## $Wmin
## [1] 0.9898126
##
## $pvalue
## [1] 0.7536
##
## $devst
## [1] 0.008618284
##
## $sim
## [1] 2500
```

```
# misclassification costs
c.vf <- 10
c.fv <- 0.05

#prior probabilities
pf <- 0.001
pt <- 1-0.001

# Prior modified to account for the misclassification costs
prior.c <- c(pt*c.fv/(c.vf*pf+c.fv*pt), pf*c.vf/(c.vf*pf+c.fv*pt))
prior.c
```

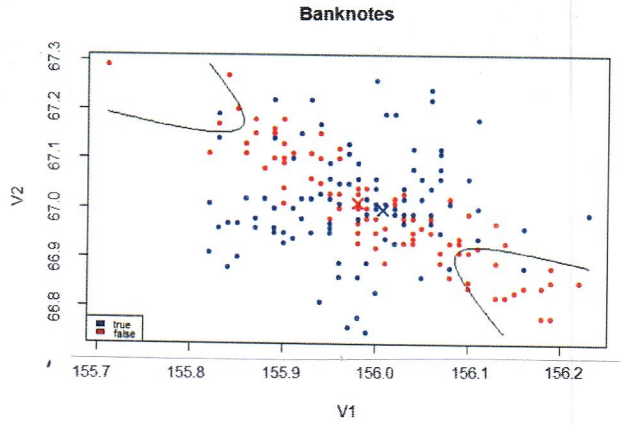
in order to take in account the misclassification cost we change the prior probabilities (LDA and QDA do not accept the misclassification cost as input)

```
## [1] 0.8331943 0.1668057
```

```
# QDA
qda.m <- qda(banknotes, vf, prior=prior.c)
qda.m
```

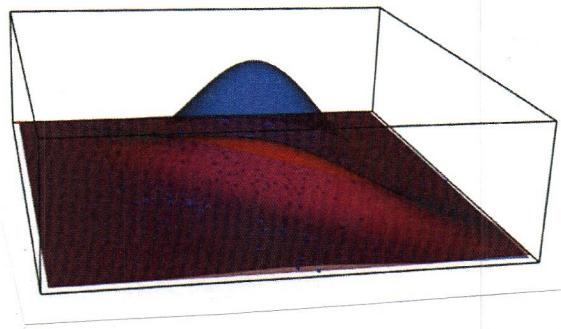
```
## Call:
## qda(banknotes, vf, prior = prior.c)
##
## Prior probabilities of groups:
##      true     false
## 0.8331943 0.1668057
##
## Group means:
##              V1      V2
## true  155.9796 67.0105
## false 156.0071 66.9974
```

```
x11()
plot(banknotes[,1:2], main='Banknotes', xlab='V1', ylab='V2', pch=20)
points(false, col='red', pch=20)
points(true, col='blue', pch=20)
legend('bottomleft', legend=levels(vf), fill=c('blue','red'), cex=.7)

points(qda.m$means, pch=4,col=c('red','blue') , lwd=2, cex=1.5)

x  <- seq(min(banknotes[,1]), max(banknotes[,1]), length=200)
y  <- seq(min(banknotes[,2]), max(banknotes[,2]), length=200)
xy <- expand.grid(V1=x, V2=y)

z  <- predict(qda.m, xy)$post
z1 <- z[,1] - z[,2]
z2 <- z[,2] - z[,1]

contour(x, y, matrix(z1, 200), levels=0, drawlabels=F, add=T)
contour(x, y, matrix(z2, 200), levels=0, drawlabels=F, add=T)
```




```
m1 <- colMeans(true)
m2 <- colMeans(false)

S1 <- cov(true)
S2 <- cov(false)

open3d()
```

```
points3d(true[,1], true[,2], 0, col='blue', pch=15)
points3d(false[,1], false[,2], 0, col='red', pch=15)
surface3d(x,y,matrix(dmvnorm(xy, m1, S1)*prior.c[1]/100 , 200), alpha=0.6, color='blue')
surface3d(x,y,matrix(dmvnorm(xy, m2, S2)*prior.c[2]/100 , 200), alpha=0.6, color='red', add=T)
box3d()
a = scene3d()
rgl.close()
x11()
rglwidget(a)
```



```
# question b)

# APER
Qda.m <- predict(qda.m)
table(class.true=vf, class.assigned=Qda.m$class)
```

remember: we have to take in account the different prior probabilities

```
##           class.assigned
## class.true true false
##      true    98     2
##      false   80    20
```

```
APER  <- (2*pt+80*pf)/100
APER
```

```
## [1] 0.02078
```

```
# Expected economic loss:
2/100*pt*c.fv+80/100*pf*c.vf
```

```
## [1] 0.008999
```

```
# question c)
# P[rejected] = P[rejected | true]P[true] + P[rejected | false]P[false]
2/100*pt + 20/100*pf
```

```
## [1] 0.02018
```

```
### ------------------------------------------------------------------------
### FISHER DISCRIMINANT ANALYSIS
### ------------------------------------------------------------------------
### Let's change viewpoint: we look for the directions that highlight
### the discrimination among groups
### -> we look for the canonical directions

# Remark. Assumptions: homogeneity of the covariance structure
# [we relax the normality assumption]

# Let's consider again the iris dataset
attach(iris)

species.name <- factor(Species, labels=c('setosa','versicolor','virginica'))

g <- 3

i1 <- which(species.name=='setosa')
i2 <- which(species.name=='versicolor')
i3 <- which(species.name=='virginica')

n1 <- length(i1)
n2 <- length(i2)
n3 <- length(i3)
n <- n1+n2+n3

detach(iris)

iris2 <- iris[,1:2]
head(iris2)
```

```
##   Sepal.Length Sepal.Width
## 1          5.1         3.5
## 2          4.9         3.0
## 3          4.7         3.2
## 4          4.6         3.1
## 5          5.0         3.6
## 6          5.4         3.9
```

```
set.seed(1)
iris2 <- iris2 + cbind(rnorm(150, sd=0.025))    # jittering

m <-  colMeans(iris2)
m1 <- colMeans(iris2[i1,])
m2 <- colMeans(iris2[i2,])
m3 <- colMeans(iris2[i3,])

S1 <- cov(iris2[i1,])
S2 <- cov(iris2[i2,])
S3 <- cov(iris2[i3,])
Sp  <- ((n1-1)*S1+(n2-1)*S2+(n3-1)*S3)/(n-g)

# covariance between groups (estimate)
B <- 1/g*(cbind(m1 - m) %*% rbind(m1 - m) +
          cbind(m2 - m) %*% rbind(m2 - m) +
          cbind(m3 - m) %*% rbind(m3 - m))
B
```

(B)

```
##              Sepal.Length Sepal.Width
## Sepal.Length    0.4183106 -0.13442959
## Sepal.Width    -0.1344296  0.07591288
```

```
# covariance within groups (estimate)
Sp
```

(Σ), here "Sp"

```
##              Sepal.Length Sepal.Width
## Sepal.Length   0.26682872  0.09357996
## Sepal.Width    0.09357996  0.11528495
```

```
# how many coordinates?
g <- 3
p <- 2
s <- min(g-1,p)
s
```

```
## [1] 2
```

```
# Matrix Sp^(-1/2)
val.Sp <- eigen(Sp)$val
vec.Sp <- eigen(Sp)$vec
invSp.2 <- 1/sqrt(val.Sp[1])*vec.Sp[,1]%*%t(vec.Sp[,1]) + 1/sqrt(val.Sp[2])*vec.Sp[,2]%*%t(vec.Sp[,2])
invSp.2
```

```
##            [,1]       [,2]
## [1,]  2.1570659 -0.7657082
## [2,] -0.7657082  3.3970568
```

```
# spectral decomposition of Sp^(-1/2) B Sp^(-1/2)
spec.dec <- eigen(invSp.2 %*% B %*% invSp.2)

# first canonical coordinate
a1 <- invSp.2 %*% spec.dec$vec[,1]
a1
```

```
##           [,1]
## [1,] -2.137114
## [2,]  2.789529
```

```
# second canonical coordinate
a2 <- invSp.2 %*% spec.dec$vec[,2]
a2
```

```
##            [,1]
## [1,] -0.8197481
## [2,] -2.0844258
```

```r
# compare with the output of lda():
lda.iris
```

```
## Call:
## lda(iris2, species.name)
##
## Prior probabilities of groups:
##     setosa versicolor  virginica
##  0.3333333  0.3333333  0.3333333
##
## Group means:
##            Sepal.Length Sepal.Width
## setosa         5.008511    3.430511
## versicolor     5.938933    2.772933
## virginica      6.584188    2.970188
##
## Coefficients of linear discriminants:
##                    LD1        LD2
## Sepal.Length -2.137114 -0.8197481
## Sepal.Width   2.789529 -2.0844258
##
## Proportion of trace:
##    LD1    LD2
## 0.9644 0.0356
```

```r
a1
```

```
##           [,1]
## [1,] -2.137114
## [2,]  2.789529
```

```r
a2
```

```
##            [,1]
## [1,] -0.8197481
## [2,] -2.0844258
```

```r
spec.dec$val/sum(spec.dec$val)
```

```
## [1] 0.96439337 0.03560663
```

Now that we have the canonical coordinates (directions of the) we can classify the data by looking at the euclidean dist.:

- we project the data in this new space
- we project the means in the new space
- we assign the data by euclidean distance

```r
### How are the data classified?
# Compute the canonical coordinates of the data

cc1.iris <- as.matrix(iris2)%*%a1
cc2.iris <- as.matrix(iris2)%*%a2

coord.cc <- cbind(cc1.iris,cc2.iris)

# Compute the coordinates of the mean within groups along the canonical directions
cc.m1 <- c(m1%*%a1, m1%*%a2)
cc.m2 <- c(m2%*%a1, m2%*%a2)
cc.m3 <- c(m3%*%a1, m3%*%a2)

# Assign data to groups
f.class=rep(0, n)
for(i in 1:n) # for each datum
{
  # Compute the Euclidean distance of the i-th datum from mean within the groups
  dist.m=c(d1=sqrt(sum((coord.cc[i,]-cc.m1)^2)),
           d2=sqrt(sum((coord.cc[i,]-cc.m2)^2)),
           d3=sqrt(sum((coord.cc[i,]-cc.m3)^2)))
  # Assign the datum to the group whose mean is the nearest
  f.class[i]=which.min(dist.m)
}
f.class
```

```
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [38] 1 1 1 1 2 1 1 1 1 1 1 1 1 3 3 3 2 3 2 2 2 3 2 2 2 2 2 2 3 2 2 3 2 2 2 3 2
##  [75] 3 3 3 3 2 2 2 2 2 2 2 2 3 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 2 3 3 3 3
## [112] 3 3 2 2 3 3 3 3 2 3 2 3 3 3 3 2 2 3 3 3 3 3 3 2 3 2 3 2 3 3 3 2 3 3 3 3 3
## [149] 2 2
```

```r
table(class.true=species.name, class.assigned=f.class)
```

```
##             class.assigned
## class.true    1  2  3
##   setosa     49  1  0
##   versicolor  0 36 14
##   virginica   0 15 35
```

```r
errors <- 150 - sum(diag(table(class.true=species.name, class.assigned=f.class)))
errors
```

```
## [1] 30
```

```r
length(species.name)
```

```
## [1] 150
```

```r
APERf   <- errors/length(species.name)
APERf
```

```
## [1] 0.2
```

we already knew, since this procedure leads to the same results as LDA (but in this case is obtained through the projection on the space that gives the directions of maximal discrimination in the groups) !

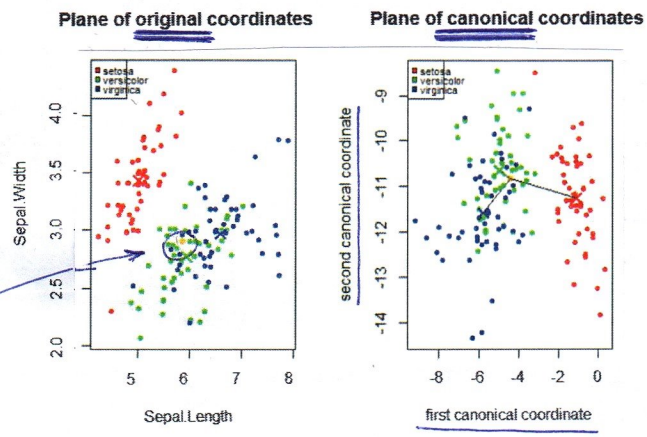
We can use the same procedure to classify new observations:

- project the datum in the new space
- compute the distance of the projected point from the means
- assign to the nearest mean

```
### How do I classify a new observation?
x.new <- c(5.85, 2.90)
# compute the canonical coordinates
cc.new <- c(x.new%*%a1, x.new%*%a2)
# compute the distance from the means
dist.m <- c(d1=sqrt(sum((cc.new-cc.m1)^2)),
            d2=sqrt(sum((cc.new-cc.m2)^2)),
            d3=sqrt(sum((cc.new-cc.m3)^2)))
# assign to the nearest mean
which.min(dist.m)
```

```
## d2
##  2
```

```
color.species=rep(c('red','green','blue'), each=50)

# visually
x11(width=14, height=7)
par(mfrow=c(1,2))
plot(iris2[,1], iris2[,2], main='Plane of original coordinates',
     xlab='Sepal.Length', ylab='Sepal.Width', pch=20, col=as.character(color.species))
legend("topleft", legend=levels(species.name), fill=c('red','green','blue'), cex=.7)
points(x.new[1], x.new[2], col='gold', pch=19)
points(m1[1], m1[2], pch=4,col='red' , lwd=2, cex=1.5)
points(m2[1], m2[2], pch=4,col='green' , lwd=2, cex=1.5)
points(m3[1], m3[2], pch=4,col='blue' , lwd=2, cex=1.5)

plot(cc1.iris, cc2.iris, main='Plane of canonical coordinates', xlab='first canonical coordinate', ylab='second canonical co
ordinate', pch=20, col=as.character(color.species))
legend("topleft", legend=levels(species.name), fill=c('red','green','blue'), cex=.7)

points(cc.m1[1], cc.m1[2], pch=4,col='red' , lwd=2, cex=1.5)
points(cc.m2[1], cc.m2[2], pch=4,col='green' , lwd=2, cex=1.5)
points(cc.m3[1], cc.m3[2], pch=4,col='blue' , lwd=2, cex=1.5)

points(cc.new[1], cc.new[2], col='gold', pch=19)

segments(cc.m1[1], cc.m1[2], cc.new[1], cc.new[2])
segments(cc.m2[1], cc.m2[2], cc.new[1], cc.new[2])
segments(cc.m3[1], cc.m3[2], cc.new[1], cc.new[2])
```

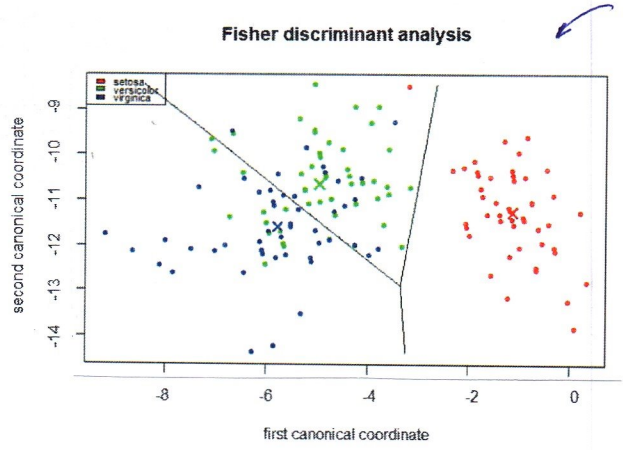


new datum

```
dev.off()
```

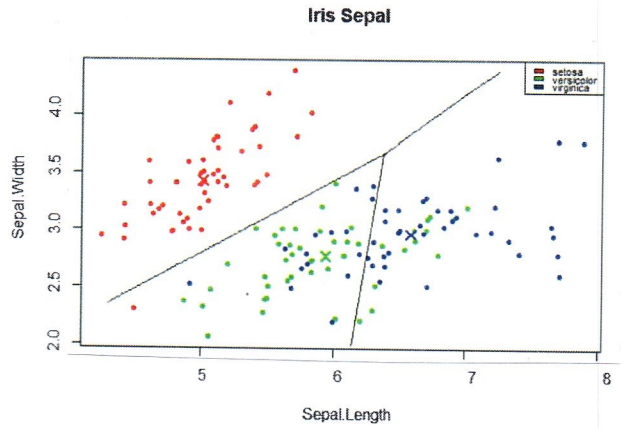
```
### We plot the partition generated by the canonical coordinates
color.species <- species.name
levels(color.species) <- c('red','green','blue')

x11()
plot(cc1.iris, cc2.iris, main='Fisher discriminant analysis', xlab='first canonical coordinate', ylab='second canonical coor
dinate', pch=20, col=as.character(color.species))
legend("topleft", legend=levels(species.name), fill=c('red','green','blue'), cex=.7)

points(cc.m1[1], cc.m1[2], pch=4,col='red' , lwd=2, cex=1.5)
points(cc.m2[1], cc.m2[2], pch=4,col='green' , lwd=2, cex=1.5)
points(cc.m3[1], cc.m3[2], pch=4,col='blue' , lwd=2, cex=1.5)

x.cc  <- seq(min(cc1.iris),max(cc1.iris),len=200)
y.cc  <- seq(min(cc2.iris),max(cc2.iris),len=200)
xy.cc <- expand.grid(cc1=x.cc, cc2=y.cc)

z  <- cbind( sqrt(rowSums(scale(xy.cc,cc.m1,scale=FALSE)^2)), sqrt(rowSums(scale(xy.cc,cc.m2,scale=FALSE)^2)), sqrt(rowSums
(scale(xy.cc,cc.m3,scale=FALSE)^2)))
z1.cc <- z[,1] - pmin(z[,2], z[,3])
z2.cc <- z[,2] - pmin(z[,1], z[,3])
z3.cc <- z[,3] - pmin(z[,1], z[,2])

contour(x.cc, y.cc, matrix(z1.cc, 200), levels=0, drawlabels=F, add=T)
contour(x.cc, y.cc, matrix(z2.cc, 200), levels=0, drawlabels=F, add=T)
contour(x.cc, y.cc, matrix(z3.cc, 200), levels=0, drawlabels=F, add=T)
```

boundaries of the classifier obtained with this procedure in the canonical coordinates space



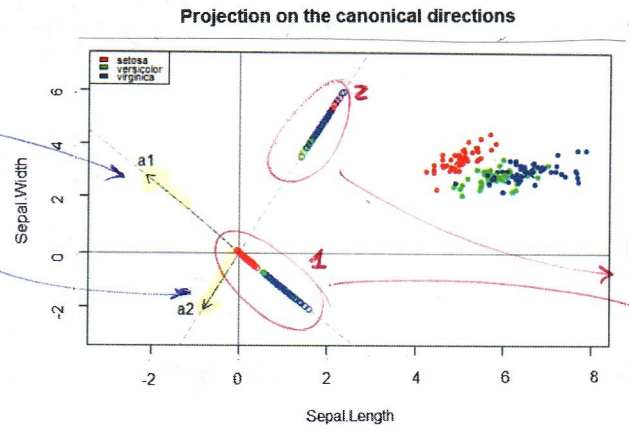

```
dev.off()
```

```
# Plot LDA
x11()
plot(iris2, main='Iris Sepal', xlab='Sepal.Length', ylab='Sepal.Width', pch=20)
points(iris2[i1,], col='red', pch=20)
points(iris2[i2,], col='green', pch=20)
points(iris2[i3,], col='blue', pch=20)
legend("topright", legend=levels(species.name), fill=c('red','green','blue'), cex=.7)

points(rbind(m1,m2,m3), pch=4,col=c('red','green','blue') , lwd=2, cex=1.5)

x  <- seq(min(iris[,1]), max(iris[,1]), length=200)
y  <- seq(min(iris[,2]), max(iris[,2]), length=200)
xy <- expand.grid(Sepal.Length=x, Sepal.Width=y)

z  <- predict(lda.iris, xy)$post  # these are P_i*f_i(x,y)
z1 <- z[,1] - pmax(z[,2], z[,3])  # P_1*f_1(x,y)-max{P_j*f_j(x,y)}
z2 <- z[,2] - pmax(z[,1], z[,3])  # P_2*f_2(x,y)-max{P_j*f_j(x,y)}
z3 <- z[,3] - pmax(z[,1], z[,2])  # P_3*f_3(x,y)-max{P_j*f_j(x,y)}

# Plot the contour line of level (Levels=0) of z1, z2, z3:
# P_i*f_i(x,y)-max{P_j*f_j(x,y)}=0 i.e., boundary between R.i and R.j
# where j realizes the max.
contour(x, y, matrix(z1, 200), levels=0, drawlabels=F, add=T)
contour(x, y, matrix(z2, 200), levels=0, drawlabels=F, add=T)
contour(x, y, matrix(z3, 200), levels=0, drawlabels=F, add=T)
```




```
dev.off()
```

```
#########################################################################
# Plot of the projections on the canonical directions (not orthogonal!)
x11()

plot(iris2, main='Projection on the canonical directions', xlab='Sepal.Length', ylab='Sepal.Width', pch=20, xlim=c(-3,8), yl
im=c(-3,7))
points(iris2[i1,], col='red', pch=20)
points(iris2[i2,], col='green', pch=20)
points(iris2[i3,], col='blue', pch=20)
legend('topleft', legend=levels(species.name), fill=c('red','green','blue'), cex=.7)

points(rbind(m1,m2,m3), pch=4,col=c('red','green','blue') , lwd=2, cex=1.5)

abline(h=0,v=0, col='grey35')

arrows(x0=0, y0=0, x1=a1[1], y1=a1[2], length=.1)
arrows(x0=0, y0=0, x1=a2[1], y1=a2[2], length=.1)

text(a1[1], a1[2], 'a1',pos=3)
text(a2[1], a2[2], 'a2',pos=2)

abline(coef=c(0,(a1[2]/a1[1])), col='grey55',lty=2)
abline(coef=c(0,(a2[2]/a2[1])), col='grey55',lty=2)

points(cc1.iris*a1[1]/(sum(a1^2)),cc1.iris*a1[2]/(sum(a1^2)),col=as.character(color.species))
points(cc2.iris*a2[1]/(sum(a2^2)),cc2.iris*a2[2]/(sum(a2^2)),col=as.character(color.species))
```



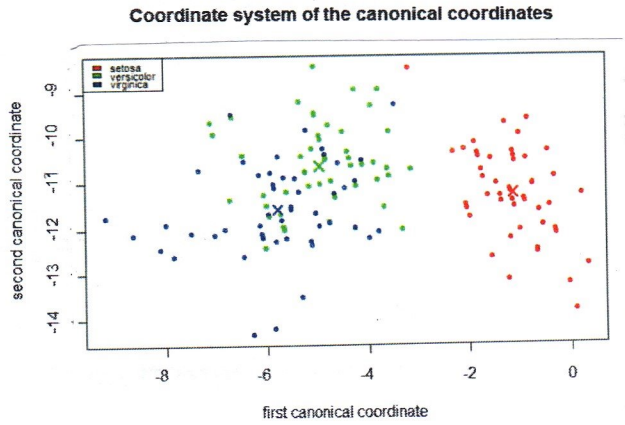
this is the original plane. We see where are the canonical directions and what is the effect of the projection of the original data on this new coordinates system

canonical directions (Fisher's directions)

they're not ⊥

we can see that the projection on 1 has a more definite separation of the groups

```
x11()
plot(cc1.iris, cc2.iris, main='Coordinate system of the canonical coordinates', xlab='first canonical coordinate', ylab='sec
ond canonical coordinate', pch=20, col=as.character(color.species))
legend('topleft', legend=levels(species.name), fill=c('red','green','blue'), cex=.7)

cc.m1 <- c(m1%*%a1, m1%*%a2)
cc.m2 <- c(m2%*%a1, m2%*%a2)
cc.m3 <- c(m3%*%a1, m3%*%a2)

points(cc.m1[1], cc.m1[2], pch=4,col='red' , lwd=2, cex=1.5)
points(cc.m2[1], cc.m2[2], pch=4,col='green' , lwd=2, cex=1.5)
points(cc.m3[1], cc.m3[2], pch=4,col='blue' , lwd=2, cex=1.5)
```




```
graphics.off()
```

LAB 08 - Additional Exercices

TOPICS:

- Linear and Quadratic Discriminant Analysis

```
load("mcshapiro.test.RData")
library(MASS)
```

```
### ------------------------------------------------------------------------------
### Problem 2 of 1/07/2009
### ------------------------------------------------------------------------------
# An art historian requires your help to identify a criterion of classification
# to discriminate the sculptures created by Gian Lorenzo Bernini from those of
# other contemporary sculptors, based on the weight [tons] and height [m]
# of 100 sculptures of undoubted attribution (sculptures.txt files). Taking into
# account that Bernini's sculptures are about 25% of the sculptures which have
# to be classified and that the purpose of the historian is to minimize the expected
# number of misclassifications:
# a) build two classifiers C1 and C2, respectively, assuming for C1 that the data
#    come from two normal populations with equal covariance matrix, and for C2 that
#    the data come from two normal populations with different covariance matrix;
# b) estimate by cross-validation the AER of the two classifiers and comment their
#    values;
# c) how will be classified by the two classifiers a 2 meter high and 4 tons heavy
#    statue?

sculpt <- read.table('sculptures.txt', header=T)
head(sculpt)
```

```
##   Altezza  Peso  Autore
## 1   1.810 3.906 Bernini
## 2   2.162 4.229 Bernini
## 3   2.516 4.073 Bernini
## 4   1.936 3.664 Bernini
## 5   2.468 4.715 Bernini
## 6   2.299 3.925 Bernini
```

```
autore <- factor(sculpt[,3], levels=c('Bernini', 'Altro'))
autore
```

```
##   [1] Bernini Bernini Bernini Bernini Bernini Bernini Bernini Bernini Bernini
##  [10] Bernini Bernini Bernini Bernini Bernini Bernini Bernini Bernini Bernini
##  [19] Bernini Bernini Bernini Bernini Bernini Bernini Bernini Bernini Bernini
##  [28] Bernini Bernini Bernini Bernini Bernini Bernini Bernini Bernini Bernini
##  [37] Bernini Bernini Bernini Bernini Bernini Bernini Bernini Bernini Bernini
##  [46] Bernini Bernini Bernini Bernini Bernini Altro   Altro   Altro   Altro
##  [55] Altro   Altro   Altro   Altro   Altro   Altro   Altro   Altro   Altro
##  [64] Altro   Altro   Altro   Altro   Altro   Altro   Altro   Altro   Altro
##  [73] Altro   Altro   Altro   Altro   Altro   Altro   Altro   Altro   Altro
##  [82] Altro   Altro   Altro   Altro   Altro   Altro   Altro   Altro   Altro
##  [91] Altro   Altro   Altro   Altro   Altro   Altro   Altro   Altro   Altro
## [100] Altro
## Levels: Bernini Altro
```

```
bernini <- sculpt[1:50,1:2]
altro <- sculpt[50:100,1:2]

# question a)
mcshapiro.test(bernini)
```

```
## $Wmin
## [1] 0.9651518
##
## $pvalue
## [1] 0.2728
##
## $devst
## [1] 0.008907978
##
## $sim
## [1] 2500
```

```
mcshapiro.test(altro)
```

```
## $Wmin
## [1] 0.9628303
##
## $pvalue
## [1] 0.2016
##
## $devst
## [1] 0.0080239
##
## $sim
## [1] 2500
```

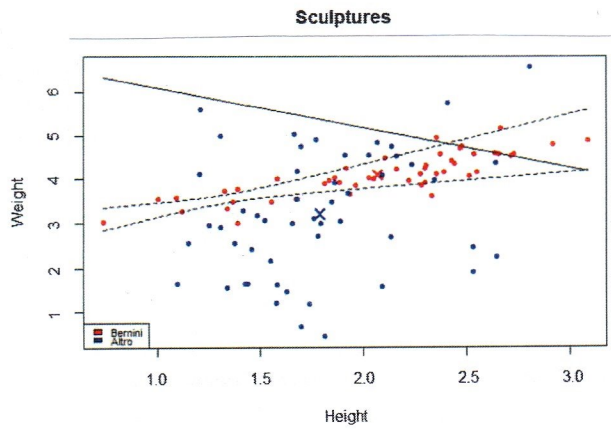
```
# LDA
lda.s <- lda(sculpt[,1:2], autore, prior=c(0.25, 0.75))
lda.s
```

```
## Call:
## lda(sculpt[, 1:2], autore, prior = c(0.25, 0.75))
##
## Prior probabilities of groups:
## Bernini   Altro
##    0.25    0.75
##
## Group means:
##         Altezza    Peso
## Bernini 2.06846 4.09668
## Altro   1.78840 3.20268
##
## Coefficients of linear discriminants:
##                LD1
## Altezza -0.7065278
## Peso    -0.7709165
```

```
# QDA
qda.s <- qda(sculpt[,1:2], autore, prior=c(0.25, 0.75))
qda.s
```

```
## Call:
## qda(sculpt[, 1:2], autore, prior = c(0.25, 0.75))
##
## Prior probabilities of groups:
## Bernini   Altro
##    0.25    0.75
##
## Group means:
##         Altezza    Peso
## Bernini 2.06846 4.09668
## Altro   1.78840 3.20268
```

```
x11()
plot(sculpt[,1:2], main='Sculptures', xlab='Height', ylab='Weight', pch=20)
points(bernini, col='red', pch=20)
points(altro, col='blue', pch=20)
legend('bottomleft', legend=levels(autore), fill=c('red','blue'), cex=.7)

points(lda.s$means, pch=4,col=c('red','blue') , lwd=2, cex=1.5)

x  <- seq(min(sculpt[,1]), max(sculpt[,1]), length=200)
y  <- seq(min(sculpt[,2]), max(sculpt[,2]), length=200)
xy <- expand.grid(Altezza=x, Peso=y)

z  <- predict(lda.s, xy)$post
z1 <- z[,1] - z[,2]
z2 <- z[,2] - z[,1]

z.q  <- predict(qda.s, xy)$post
z1.q <- z.q[,1] - z.q[,2]
z2.q <- z.q[,2] - z.q[,1]

contour(x, y, matrix(z1, 200), levels=0, drawlabels=F, add=T)
contour(x, y, matrix(z2, 200), levels=0, drawlabels=F, add=T)

contour(x, y, matrix(z1.q, 200), levels=0, drawlabels=F, add=T, lty=2)
contour(x, y, matrix(z2.q, 200), levels=0, drawlabels=F, add=T, lty=2)
```




```
dev.off()
```

```
# question b)
# LDA
LdaCV.s <- lda(sculpt[,1:2], autore, prior=c(0.25, 0.75), CV=T)
table(class.true=autore, class.assignedCV=LdaCV.s$class)
```

```
##           class.assignedCV
## class.true Bernini Altro
##    Bernini       5    45
##    Altro         3    47
```

```
AER.CV.l <- 45/50*0.25+3/50*0.75
AER.CV.l
```

```
## [1] 0.27
```

```
# QDA
QdaCV.s <- qda(sculpt[,1:2], autore, prior=c(0.25, 0.75), CV=T)
table(class.true=autore, class.assignedCV=QdaCV.s$class)
```

```
##           class.assignedCV
## class.true Bernini Altro
##    Bernini      30    20
##    Altro         6    44
```

```
AER.CV.q <- 20/50*0.25+6/50*0.75
AER.CV.q
```

```
## [1] 0.19
```

```
# question c)
predict(lda.s, c(Altezza=2, Peso=4))
```

```
## $class
## [1] Altro
## Levels: Bernini Altro
##
## $posterior
##        Bernini     Altro
## [1,] 0.3069988 0.6930012
##
## $x
##           LD1
## [1,] -0.542401
```

```
predict(qda.s, c(Altezza=2, Peso=4))
```

```
## $class
## [1] Bernini
## Levels: Bernini Altro
##
## $posterior
##        Bernini     Altro
## [1,] 0.6162577 0.3837423
```

```
# (?) points(2,4, pch=3, col='springgreen', lwd=2)

graphics.off()

Qda.m <- predict(qda.s)
table(class.true=autore, class.assigned=Qda.m$class)
```

```
##           class.assigned
## class.true Bernini Altro
##    Bernini      32    18
##    Altro         5    45
```

```
### ------------------------------------------------------------------------
### Problem 2 of 16/07/2010
### ------------------------------------------------------------------------
# A young Portuguese engineer wants to build a machine able to automatically
# distinguish between two species of sardines (the Atlantic sardines and the
# Iberian sardine) on the basis of the length [cm] of the sardine. To
# do this, it aims to build a classifier based on the measurements of 500
# Atlantic sardines and 500 Iberian sardine (sardine.txt file).
# a) Perform two tests to verify the hypothesis of normality of the two
#    populations, a test for equality of means, a test for equality of
#    variances.
# b) On the basis of the previous tests and knowing that 75% of fished sardines
#    belong to the Atlantic species while 25% to the Iberian species, build a
#    classifier that minimizes the number of misclassified sardines and report
#    the parameters.
# c) Estimate the AER of the classifier analytically using the estimated
#    probability densities.

sardine <- read.table('sardine.txt', header=T)
head(sardine)
```

```
##   Atlantica Iberica
## 1     10.44    9.06
## 2      9.05    8.81
## 3     10.41    8.89
## 4     10.68    8.75
## 5      8.91    8.06
## 6     10.05    8.37
```

```
# question a)
sardinaa <- sardine[[1]]
sardinai <- sardine[[2]]

shapiro.test(sardinaa)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  sardinaa
## W = 0.99556, p-value = 0.1677
```

```
shapiro.test(sardinai)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  sardinai
## W = 0.99786, p-value = 0.7853
```

```
var.test(sardinaa, sardinai)
```

```
##
##  F test to compare two variances
##
## data:  sardinaa and sardinai
## F = 1.0258, num df = 499, denom df = 499, p-value = 0.7757
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  0.8605683 1.2228683
## sample estimates:
## ratio of variances
##           1.025847
```

```
t.test(sardinaa, sardinai, var.eq=T)
```

```
##
##  Two Sample t-test
##
## data:  sardinaa and sardinai
## t = 48.525, df = 998, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  1.487279 1.612641
## sample estimates:
## mean of x mean of y
##  10.02956   8.47960
```

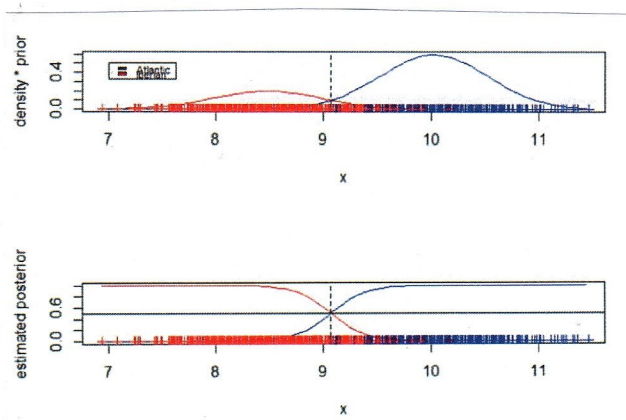
```
# question b) and c)

MA <- mean(sardinaa)
MI <- mean(sardinai)
SD <- sqrt((var(sardinaa) + var(sardinai))/2)

# Analitically
misclass <- function(x){
  0.25*(1 - pnorm(x, MI, SD)) + 0.75*pnorm(x, MA, SD)
  }
optimize(f=misclass, lower=min(sardine), upper=max(sardine))
```

```
## $minimum
## [1] 9.073782
##
## $objective
## [1] 0.05183513
```

```
R   <- optimize(f=misclass, lower=min(sardine), upper=max(sardine))$minimum
AER <- optimize(f=misclass, lower=min(sardine), upper=max(sardine))$objective

# check
AER == 0.25*(1 - pnorm(R, mean(sardinai), SD)) + 0.75*pnorm(R, mean(sardinaa), SD)
```

```
## [1] TRUE
```

```
# some plots
specie <- factor(rep(c('Atlantica', 'Iberica'), each=500))
lunghezza <- c(sardine[[1]], sardine[[2]])

fit <- lda(specie ~ lunghezza, prior=c(0.75, 0.25))
x <- data.frame(lunghezza=seq(min(sardine), max(sardine), 0.05))

LDA <- predict(fit, x)$posterior[,2] # classe iberica

x11()
par(mfrow=c(2,1))
plot(x[,1], 0.25*(dnorm(x[,1], MI, SD)), type='l', col='red', lty=1, xlab='x', ylab='density * prior', ylim=c(0,0.6))
lines(x[,1], 0.75*(dnorm(x[,1], MA, SD)), type='l', col='blue', lty=1, xlab='x', ylab='density * prior')
abline(v=R, lty=2)
points(sardinaa, rep(0, 500), pch=3, col='blue')
points(sardinai, rep(0, 500), pch=3, col='red')
legend(7,.5,legend=c('Atlantic','Iberian'),fill=c('blue','red'),cex=.7)

plot(x[,1], LDA, type='l', col='red', lty=1, xlab='x', ylab='estimated posterior')# rosso = iberica
lines(x[,1], 1 - LDA, type='l', col='blue', lty=1, xlab='x', ylab='estimated posterior')# blu = atlantica
abline(h = 0.5)
abline(v=R, lty=2)

points(sardinaa, rep(0, 500), pch=3, col='blue')
points(sardinai, rep(0, 500), pch=3, col='red')
```




```
dev.off()
```

```
# Compute the APER
prior <- c(0.75,0.25)
G <- 2
misc <- table(classe.vera=specie, classe.allocata=predict(fit)$class)
misc
```

```
##             classe.allocata
## classe.vera Atlantica Iberica
##   Atlantica       481      19
##   Iberica          56     444
```

```
APER <- 0
for(g in 1:G)
  APER <- APER + sum(misc[g,-g])/sum(sum(misc[g,])) * prior[g]
APER
```

```
## [1] 0.0565
```

```
AER
```

```
## [1] 0.05183513
```

```
### ------------------------------------------------------------------------
### Problem 2 of 14/02/2011
### ------------------------------------------------------------------------
# The ATM is considering the possibility of including in the turnstiles optical
# readers able to measure the size [mm] of tickets.
# In order to build a suitable software, it asks you to build a classification
# rule which minimizes the expected number of errors.
# Starting with the measures relating to 100 regular tickets (true.txt file)
# and 100 counterfeit tickets (false.txt files) and knowing that the
# 0.5% of the banknotes in circulation is counterfeit:
# a) construct an appropriate classification rule (in particular, verify the
#    assumptions when possibile, and provide a qualitative graph of the
#    classification regions);
# b) compute the APER and discuss its value;
# c) on the basis of the rule identified at point (a), how will be classified
#    a ticket long 85.5 mm and wide 55.0 mm? What is the probability that the
#    ticket will be false?

true <- read.table('true.txt', header=T)
false <- read.table('false.txt', header=T)
head(true)
```

```
##   Lunghezza Larghezza
## 1    84.603    54.650
## 2    84.976    55.219
## 3    84.850    55.031
## 4    85.120    54.838
## 5    85.008    54.789
## 6    85.197    54.947
```

```
head(false)
```

```
##   Lunghezza Larghezza
## 1    85.224    56.320
## 2    84.978    54.256
## 3    85.600    54.034
## 4    86.371    55.243
## 5    86.720    55.164
## 6    83.946    53.410
```

```
fv <- as.factor(rep(c('vero','falso'), c(100,100)))
head(fv)
```

```
## [1] vero vero vero vero vero vero
## Levels: falso vero
```

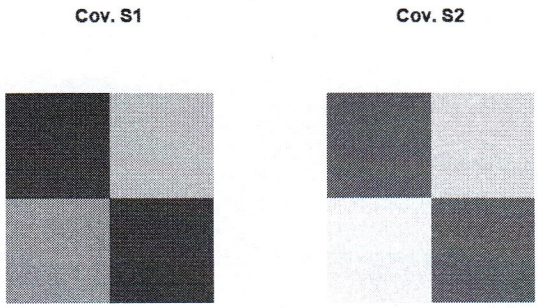
```
biglietti <- rbind(true, false)

### question a)
# normality
mcshapiro.test(true)
```

```
## $Wmin
## [1] 0.9930443
##
## $pvalue
## [1] 0.9288
##
## $devst
## [1] 0.005143173
##
## $sim
## [1] 2500
```

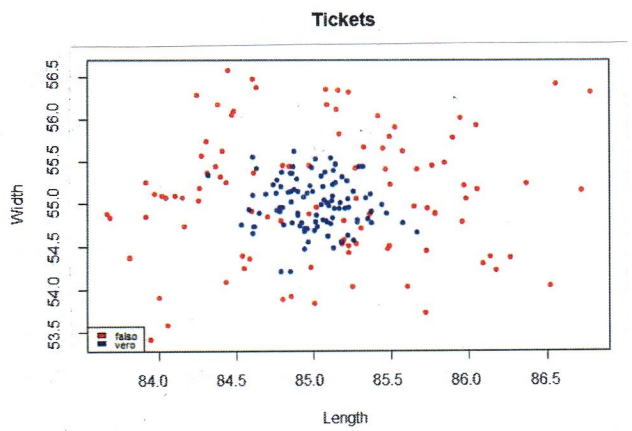
```
mcshapiro.test(false)
```

```
## $Wmin
## [1] 0.9888231
##
## $pvalue
## [1] 0.678
##
## $devst
## [1] 0.00934486
##
## $sim
## [1] 2500
```

```
# homogeneity of covariances
S1<-cov(true)
S2<-cov(false)
x11()
par(mfrow=c(1,2))
image(S1, col=heat.colors(100),main='Cov. S1', asp=1, axes = FALSE, breaks = quantile(rbind(S1,S2), (0:100)/100, na.rm=TRUE
))
image(S2, col=heat.colors(100),main='Cov. S2', asp=1, axes = FALSE, breaks = quantile(rbind(S1,S2), (0:100)/100, na.rm=TRUE
))
```

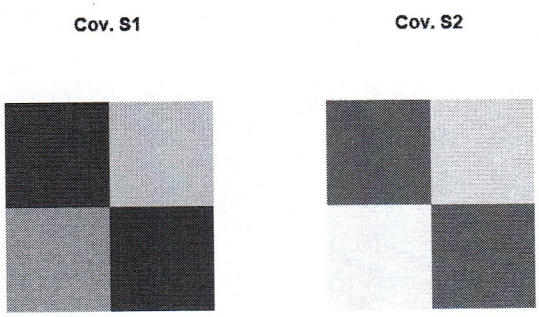



```
x11()
plot(biglietti, main='Tickets', xlab='Length', ylab='Width', pch=20)
points(false, col='red', pch=20)
points(true, col='blue', pch=20)
legend('bottomleft', legend=levels(fv), fill=c('red','blue'), cex=.7)
```




```
# The assumption of homogeneity of covariances doesn't seem to be satisfied
dev.off()
```

```
# QDA
qda.bigl <- qda(biglietti, fv, prior = c(0.005, 0.995))
```




```
qda.bigl # Look at the order in Levels to set the priors!!
```

```
## Call:
## qda(biglietti, fv, prior = c(0.005, 0.995))
##
## Prior probabilities of groups:
## falso  vero
## 0.005 0.995
##
## Group means:
##       Lunghezza Larghezza
## falso  85.06597  55.10586
## vero   85.00095  55.00439
```

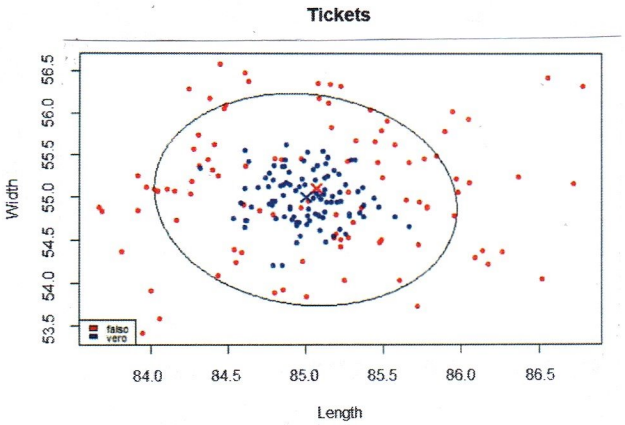
```
Qda.bigl <- predict(qda.bigl, biglietti)
head(Qda.bigl)
```

```
## $class
##   [1] vero  vero  vero  vero  vero  vero  vero  vero  vero  vero  vero  vero
##  [13] vero  vero  vero  vero  vero  vero  vero  vero  vero  vero  vero  vero
##  [25] vero  vero  vero  vero  vero  vero  vero  vero  vero  vero  vero  vero
##  [37] vero  vero  vero  vero  vero  vero  vero  vero  vero  vero  vero  vero
##  [49] vero  vero  vero  vero  vero  vero  vero  vero  vero  vero  vero  vero
##  [61] vero  vero  vero  vero  vero  vero  vero  vero  vero  vero  vero  vero
##  [73] vero  vero  vero  vero  vero  vero  vero  vero  vero  vero  vero  vero
##  [85] vero  vero  vero  vero  vero  vero  vero  vero  vero  vero  vero  vero
##  [97] vero  vero  vero  vero  falso vero  vero  falso falso falso falso falso
## [109] vero  falso falso falso vero  falso falso vero  vero  vero  vero  falso
## [121] vero  vero  vero  falso vero  vero  falso vero  vero  vero  falso vero
## [133] vero  vero  vero  vero  vero  vero  vero  vero  vero  vero  vero  falso
## [145] vero  vero  vero  falso falso vero  vero  vero  vero  falso vero  falso
## [157] vero  vero  falso falso vero  falso vero  falso vero  vero  vero  vero
## [169] vero  falso vero  falso vero  vero  vero  vero  vero  falso vero  vero
## [181] vero  falso falso vero  falso vero  vero  falso vero  vero  falso vero
## [193] vero  vero  vero  falso vero  vero  falso vero
## Levels: falso vero
##
## $posterior
##             falso         vero
## 1    0.0042179401 9.957821e-01
## 2    0.0008181501 9.991818e-01
## 3    0.0007493768 9.992506e-01
## 4    0.0007772127 9.992228e-01
## 5    0.0007618846 9.992381e-01
## 6    0.0008682556 9.991317e-01
## 7    0.0007842171 9.992158e-01
## 8    0.0010510093 9.989490e-01
## 9    0.0010833223 9.989167e-01
## 10   0.0009585909 9.990414e-01
## 11   0.0312264208 9.687736e-01
## 12   0.0018986701 9.981013e-01
## 13   0.0025864312 9.974136e-01
## 14   0.0007241383 9.992759e-01
## ..
## 921  0.4627574812 5.372425e-01
## 931  0.0018740709 9.981259e-01
## 941  0.2534045331 7.465955e-01
## 951  0.3941759103 6.058241e-01
## 961  0.8633572342 1.366428e-01
## 971  0.1334234584 8.665765e-01
## 981  0.0031785819 9.968214e-01
## 991  1.0000000000 2.770420e-13
## 1001 0.0012109654 9.987890e-01
```

```
summary(Qda.bigl)
```

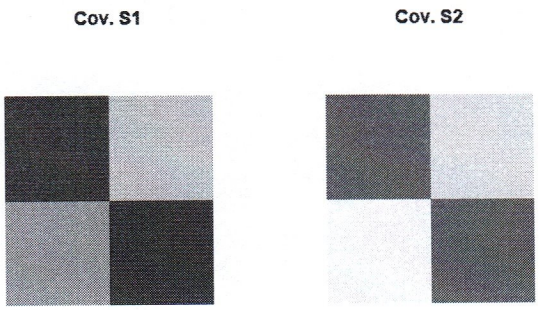
```
##           Length Class  Mode
## class     200    factor numeric
## posterior 400    -none- numeric
```

```
x11()
plot(biglietti, main='Tickets', xlab='Length', ylab='Width', pch=20)
points(false, col='red', pch=20)
points(true, col='blue', pch=20)
legend('bottomleft', legend=levels(fv), fill=c('red','blue'), cex=.7)
points(qda.bigl$means, pch=4,col=c('red','blue') , lwd=2, cex=1.5)

x  <- seq(min(biglietti[,1]), max(biglietti[,1]), length=200)
y  <- seq(min(biglietti[,2]), max(biglietti[,2]), length=200)
xy <- expand.grid(Lunghezza=x, Larghezza=y)

z  <- predict(qda.bigl, xy)$post
z1 <- z[,1] - z[,2]
z2 <- z[,2] - z[,1]

contour(x, y, matrix(z1, 200), levels=0, drawlabels=F, add=T)
contour(x, y, matrix(z2, 200), levels=0, drawlabels=F, add=T)
```




```
dev.off()
```

```
### question b)

MC <- table(classe.vera=fv, classe.allocata=Qda.bigl$class)
```




```
MC
```

```
##             classe.allocata
## classe.vera falso vero
##       falso    34   66
##       vero      0  100
```

```
APER <- 66/100 * 0.005
APER
```

```
## [1] 0.0033
```

```
### domanda c)

predict(qda.bigl, cbind(Lunghezza = 85.5, Larghezza = 55))
```

```
## $class
## [1] vero
## Levels: falso vero
##
## $posterior
##              falso      vero
## [1,] 0.004642767 0.9953572
```

```
### ------------------------------------------------------------------------
### Problem 3 of 12/02/2014
### ------------------------------------------------------------------------
# To cope with the economic crisis, the Hotel Mont Blanc in Courmayeur has
# decided to apply special discounts on their Carnival rates in case the
# snow at skiing facilities is predicted to be of poor quality. The file neve.txt
# reports, for the last 60 years, the data on total snowfall [cm] and the medium
# temperature [°C] recorded in the two months from December to January, together
# with the judgment on the quality of the Carnival snow provided by the ALpine Guides
# Society of Courmayeur.
# a) Build a classifier for the quality of the Carnival snow that minimizes the expected
#    cost of misclassification (display a qualitative graph of the classification regions)
#    when assuming that:
#    - There is no economic loss in the case in which the snow is good and no discount
#      is applied; there is no economic loss in case the snow is bad and the Hotel
#      operates the discount; there is a loss of 3000??? in the case the snow is bad
#      and no discounts are applied; there is a loss of 2000??? in the case the snow is
#      good and discounts are applied;
#    - A season characterized by good snow is associated with a higher variability in
#      temperatures and in the amount of snow.
# b) Compute the APER of the classifier.
# c) Based on the estimates at point (b), estimate the expected economic loss of the
#    classifier.
# d) Based on the classifier build at point (a) and knowing that the last two months
#    December-January a total of 200 cm of snow have fallen and the average temperature
#    was -4 °C, would you recommend to the hotel to apply the special discount of Carnival?

# question a)
neve <- read.table('neve.txt', header=T)
good<-neve[neve[,3]=='good',1:2]
bad<-neve[neve[,3]=='bad',1:2]

mcshapiro.test(good)$pvalue
```

```
## [1] 0.756
```

```
mcshapiro.test(bad)$pvalue
```

```
## [1] 0.448
```

```
prior <- c(dim(bad)[1]/(sum(dim(bad)[1],dim(good)[1])),dim(good)[1]/(sum(dim(bad)[1],dim(good)[1])))
pb <- prior[1]
pg <- prior[2]

c.bg <- 2000
c.gb <- 3000

# Modified prior to account for the misclassification costs
prior.c <- c(bad=pb*c.gb/(c.bg*pg+c.gb*pb),good=pg*c.bg/(c.bg*pg+c.gb*pb))
prior.c
```

```
##       bad      good
## 0.3134328 0.6865672
```

```
qda.m <- qda(giudizio ~ quantita + temperatura, data=neve, prior=prior.c)
qda.m
```

```
## Call:
## qda(giudizio ~ quantita + temperatura, data = neve, prior = prior.c)
##
## Prior probabilities of groups:
##       bad      good
## 0.3134328 0.6865672
##
## Group means:
##      quantita temperatura
## bad  139.3286   -2.157143
## good 180.5674   -4.600000
```

```
x11()
plot(neve[,1:2], main='Snow', xlab='V1', ylab='V2', pch=20)
points(bad, col='red', pch=20)
points(good, col='blue', pch=20)
legend('bottomleft', legend=levels(as.factor(neve[,3])), fill=c('red','blue'), cex=.7)

points(qda.m$means, pch=4,col=c('red','blue') , lwd=2, cex=1.5)

x  <- seq(min(neve[,1]), max(neve[,1]), length=200)
y  <- seq(min(neve[,2]), max(neve[,2]), length=200)
xy <- expand.grid(quantita=x, temperatura=y)

z  <- predict(qda.m, xy)$post
z1 <- z[,1] - z[,2]
z2 <- z[,2] - z[,1]

contour(x, y, matrix(z1, 200), levels=0, drawlabels=F, add=T)
contour(x, y, matrix(z2, 200), levels=0, drawlabels=F, add=T)

# question b) [APER]
Qda.m <- predict(qda.m)
table(classe.vera=neve[,3], classe.allocata=Qda.m$class)
```

```
##             classe.allocata
## classe.vera bad good
##        bad   12    2
##        good   3   43
```

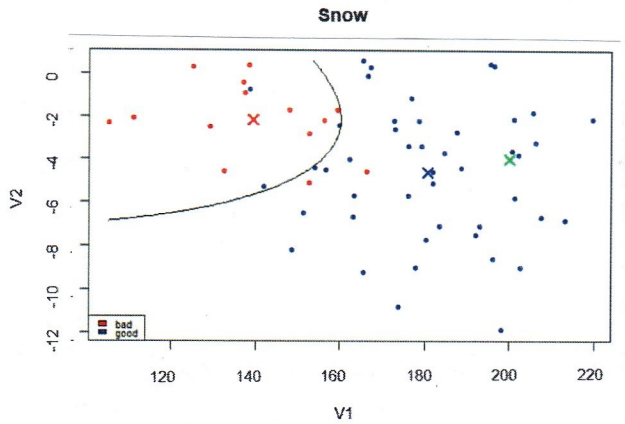
```
APER  <- (2+3)/(46+14)
APER
```

```
## [1] 0.08333333
```

```
# question c) [Expected economic loss]
(3*c.bg+2*c.gb)/60
```

```
## [1] 200
```

```
# question d) [Classification of 2012-2013]
z0.new <- data.frame(quantita=200, temperatura=-4)
points(z0.new, pch=4, col='springgreen', lwd=2, cex=1.5)
```




```
predict(qda.m,z0.new)$class
```

```
## [1] good
## Levels: bad good
```

```
graphics.off()
```

```
### ------------------------------------------------------------------------
### Problem 2 of 9/09/2009
### ------------------------------------------------------------------------
# A young statistician from Tromso wants to build a classifier able to
# predict the presence or absence of snowfall on the day D + 1 using
# the average temperature [°C] and humidity of day G. Using
# the data of the last 30 days (file snow.txt) and knowing that in the
# last 40 years in Tromso it has snowed an average of 207 days a year:
# a) build a classifier [report the model assumptions and provide a
#    qualitative graph of the classification regions].
# b) Estimate the APER of classifier (a), and compare it with that of the
#    trivial classifier.
# c) The temperature and the humidity measured yesterday in Tromso are
#    -10 Â° C and 0.75, respectively. Estimate the likelihood of snow
#    for today by using both the classifier (a) that the trivial classifier.


snow <- read.table('snow.txt', header=TRUE)
head(snow)
```

```
##    Temperature.G Humidity.G Snow.G.1
## D1         -2.25       0.67  no-snow
## D2        -15.14       1.01     snow
## D3        -14.00       1.07     snow
## D4        -10.47       0.59  no-snow
## D5         -8.57       0.83  no-snow
## D6        -14.31       0.93     snow
```

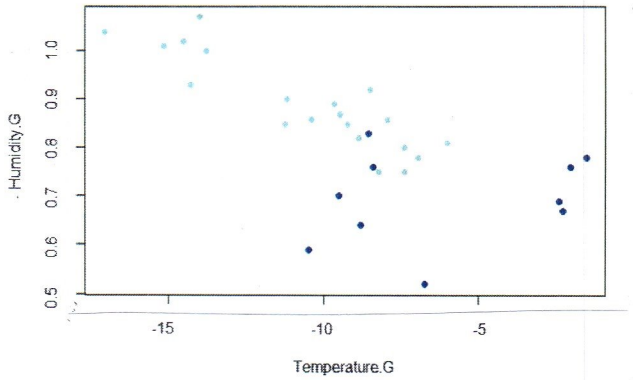
```
attach(snow)

i1 <- which(Snow.G.1=='no-snow')
i2 <- which(Snow.G.1=='snow')

# question a)
mcshapiro.test(snow[i1,1:2])$p
```

```
## [1] 0.9864
```

```
mcshapiro.test(snow[i2,1:2])$p
```

```
## [1] 0.9956
```

```
x11()
plot(Temperature.G,Humidity.G, pch=19, col=ifelse(Snow.G.1=='no-snow','blue','lightblue'))
```

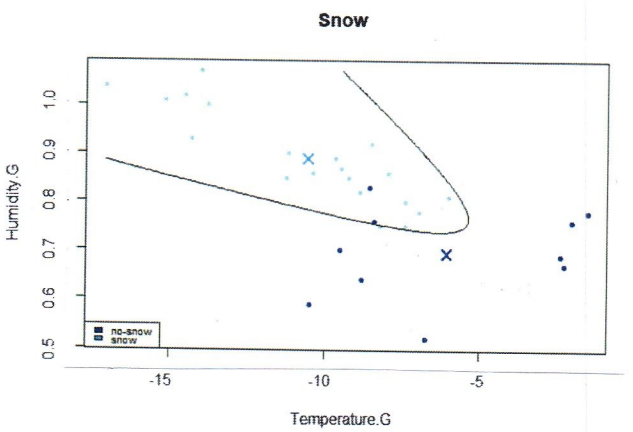



```
dev.off()
```

```
# QDA
library(MASS)
qda.s <- qda(snow[,1:2], snow[,3], prior=c(1-207/365,207/365))
qda.s
```

```
## Call:
## qda(snow[, 1:2], snow[, 3], prior = c(1 - 207/365, 207/365))
##
## Prior probabilities of groups:
##   no-snow      snow
## 0.4328767 0.5671233
##
## Group means:
##         Temperature.G Humidity.G
## no-snow        -6.065      0.694
## snow          -10.559      0.889
```

```
x11()
plot(snow[,1:2], main='Snow', pch=20, col=ifelse(Snow.G.1=='no-snow','blue','lightblue'))
legend('bottomleft', legend=levels(as.factor(snow[,3])), fill=c('blue','steelblue2'), cex=.7)

points(qda.s$means, pch=4,col=c('blue','steelblue2') , lwd=2, cex=1.5)

x  <- seq(min(snow[,1]), max(snow[,1]), length=200)
y  <- seq(min(snow[,2]), max(snow[,2]), length=200)
xy <- expand.grid(Temperature.G=x, Humidity.G=y)

z  <- predict(qda.s, xy)$post
z1 <- z[,1] - z[,2]
z2 <- z[,2] - z[,1]

contour(x, y, matrix(z1, 200), levels=0, drawlabels=F, add=T)
contour(x, y, matrix(z2, 200), levels=0, drawlabels=F, add=T)
```




```
dev.off()
```

```
# question b)
Qda.s <- predict(qda.s)
table(classe.vera=snow[,3], classe.allocata=Qda.s$class)
```

```
##            classe.allocata
## classe.vera no-snow snow
##     no-snow       8    2
##     snow          1   19
```

```
prior <- c(1-207/365,207/365)

APER  <- 2/10*prior[1]+1/20*prior[2]
APER
```

```
## [1] 0.1149315
```

```
# Trivial classifier; classifies always as the most likely class a priori
APER.banale <- prior[1]
APER.banale
```

```
## [1] 0.4328767
```

```
# question c)
new.day <- c(Temperature.G=-10, Humidity.G=0.75)
predict(qda.s, new.day)$posterior[2]
```

```
## [1] 0.1497035
```

```
prior.snow=207/365
prior.snow
```

```
## [1] 0.5671233
```

```
graphics.off()
```