Final project: LDA

Jose Antonio Alvarez Ocete

For fractions:

```r
library(MASS)
options(digits=4)
```

```r
#import data
data <- read.table("data/anaconda.dat")

n1 <- 28
n2 <- 28

#set up X
X_1 <- as.matrix(data[1:n1,1:2])
X_2 <- as.matrix(data[(n1+1):(n1+n2),1:2])

names <- c('Snout Vent Length', 'Weight')
colnames(X_1)<- names
colnames(X_2) <- names

r <- length(names)

sampleMean<-function(X, n) {
  Ones <- rep(1,n)
  return (1/n * t(X)%*%Ones)
}

sampleCovariance<-function(X, n, sample_mean) {
  Ones <- rep(1,n)
  return (1/(n-1) * t(X - Ones%*%t(sample_mean))%*%(X - Ones%*%t(sample_mean)))
}
```

Conduct a two sample test to make sure the populations are different form each other.

```r
# Returns true if we reject the Ho: the populations are the same. Returns false if we fail to reject.
hotelling_test<-function(X_1, X_2, alpha=0.05, print=FALSE) {
  n1 <- length(X_1[,1])
  n2 <- length(X_2[,1])
  p <- length(X_1[1,])

  # Compute sample mean and S
  x1_sample_mean <- sampleMean(X_1, n1)
  x2_sample_mean <- sampleMean(X_2, n2)
  S1 <- sampleCovariance(X_1, n1, x1_sample_mean)
  S2 <- sampleCovariance(X_2, n2, x2_sample_mean)
  Spooled <- (n1-1)/(n1+n2-2) * S1 + (n2-1)/(n1+n2-2) * S2
  # Compute Hotelling's T^2 statistic
  diff <- x1_sample_mean - x2_sample_mean
  T_2 <- t(diff)%*%solve((1/n1 + 1/n2)*Spooled)%*%diff
  F <- (n1+n2-2)*p / (n1+n2-1-p) * qf(1-alpha, df1=p, df2=n1+n2-1-p)

  # Print the results if specified
  if (print) {
    cat('Reject if T_2 =', T_2, ' > ', F, '= F', fill=TRUE)
  }

  return (T_2 > F)
}

reject <- hotelling_test(X_1, X_2, alpha=0.05, print=TRUE)
```

```
## Reject if T_2 = 76.92  >  6.463 = F
```

```
cat('H_0: u_1 = u_2. Hypothesis rejected? --->', reject)
```

```
## H_0: u_1 = u_2. Hypothesis rejected? ---> TRUE
```

Classifier function

```
# If x_0 is NULL, the classifier is printed and nothing else is done
fisher<-function(X_1, X_2, x_0=NULL) {
  n1 <- length(X_1[,1])
  n2 <- length(X_2[,1])

  # Compute sample mean and S
  x1_sample_mean <- sampleMean(X_1, n1)
  x2_sample_mean <- sampleMean(X_2, n2)
  S1 <- sampleCovariance(X_1, n1, x1_sample_mean)
  S2 <- sampleCovariance(X_2, n2, x2_sample_mean)
  Spooled <- (n1-1)/(n1+n2-2) * S1 + (n2-1)/(n1+n2-2) * S2

  # Compute the classification
  w <- t(x1_sample_mean - x2_sample_mean) %*% solve(Spooled)
  mid <- 1/2*(x1_sample_mean + x2_sample_mean)

  # Print our classifier or classify the value
  if (is.null(x_0))
    cat(w, '*(x_0 -', mid, ') >= 0')
  else {
    result <- w%*%(x_0 - mid)
    if (result >= 0) {
      return (1)
    }
    return (2)
  }
}
```

Print the classiffier obtained

```
# Use the classifier with our whole population, without predicting anything
fisher(X_1, X_2)
```

```
## 0.05095 -0.01986 *(x_0 - 288.5 22.28 ) >= 0
```

Compute the apparent error rate (AER):

```r
# Returns 0 if then prediction is correct, 1 otherwise.
classify<-function(X_1, X_2, x_0, expeted_class) {
  if ( fisher(X_1, X_2, x_0) != expeted_class ) {
    return (1)
  }
  return (0)
}

# Use the classifier with our whole population, without predicting anything
errors <- 0
for (i in 1:n1) {
  errors <- errors + classify(X_1, X_2, X_1[i,], 1)
}
for (i in 1:n2) {
  errors <- errors + classify(X_1, X_2, X_2[i,], 2)
}
AER <- errors / (n1+n2)
cat( fractions(AER), '=', errors, '/', (n1+n2))
```

```
## 0.08929 = 5 / 56
```

Compute the expected actual error rate (EAER):

```r
# Use the classifier with our whole population, without predicting anything
errors <- 0
for (i in 1:n1) {
  errors <- errors + classify(X_1[-i,], X_2, X_1[i,], 1)
}
for (i in 1:n2) {
  errors <- errors + classify(X_1, X_2[-i,], X_2[i,], 2)
}
EAER <- errors / (n1+n2)
cat( fractions(EAER), '=', errors, '/', (n1+n2))
```

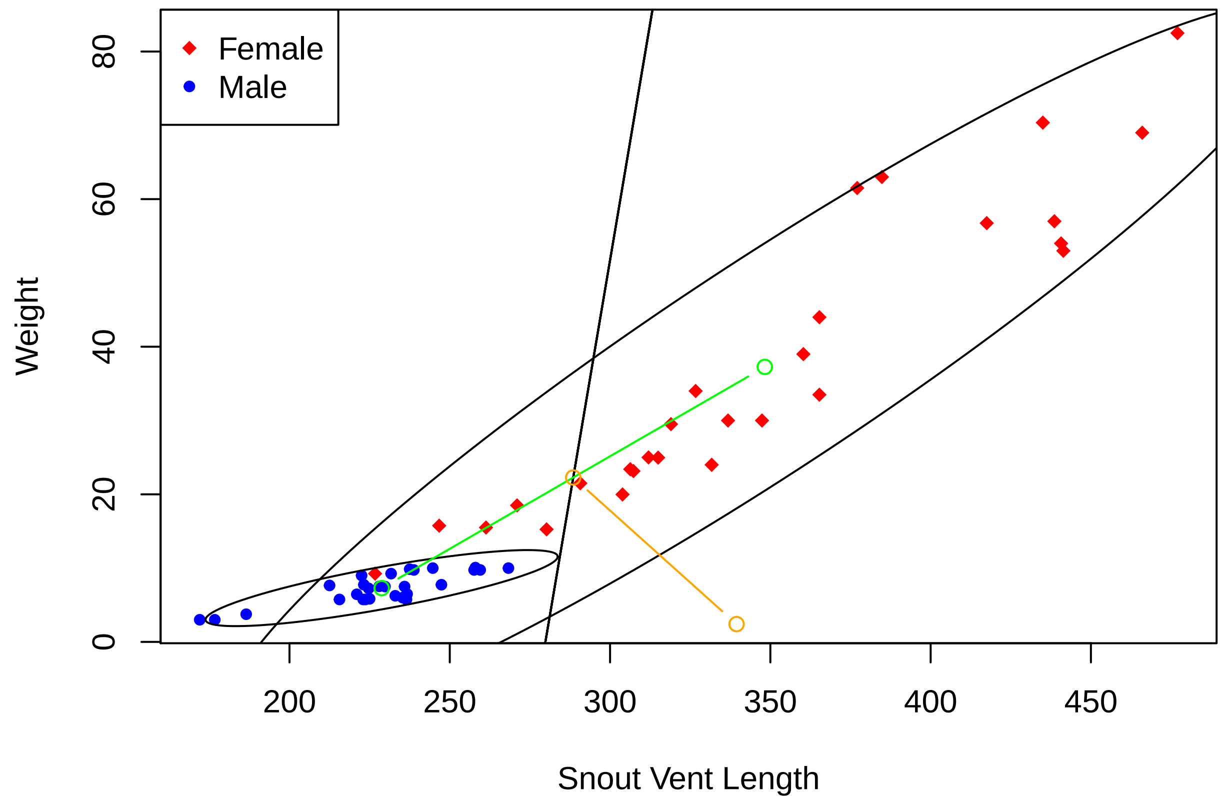
```
## 0.08929 = 5 / 56
```

```r
#library(rrcov)
library(mixtools)
```

```
## mixtools package, version 1.2.0, Released 2020-02-05
## This package is based upon work supported by the National Science Foundation under Grant No. SES-0518
```

```r
par(mfrow=c(1,1), mar=c(4,4,2,1))
plot(data$V1,data$V2, xlab=names[1] ,ylab=names[2],
     pch=rep(c(18,20),each=28), col=rep(c(2,4), each=28), main="")
legend("topleft", legend=c("Female","Male"), pch=c(18,20), col=c(2,4), cex=1)

# Method 1
x1 <- X_1
x2 <- X_2

# compute sample mean vectors:
x1.mean <- colMeans(x1)
x2.mean <- colMeans(x2)
```

```r
# compute pooled estimate for the covariance matrix:
S.u <- (n1-1)/(n1+n2-2) * var(x1) + (n2-1)/(n1+n2-2) * var(x2)
w <- solve(S.u)%*%(x1.mean-x2.mean)
w0 <- -(x1.mean+x2.mean)%*%w/2

lines(data[,1],-(w[1]*data[,1]+c(w0))/w[2])
lines(cbind(x1.mean, x2.mean))

alpha <- .05
ellipse(x1.mean, var(x1), alpha=alpha, npoints=250, newplot=FALSE)
ellipse(x2.mean, var(x2), alpha=alpha, npoints=250, newplot=FALSE)

# Compute line between mean points and the line w
points(c(x2.mean[1],x1.mean[1]), c(x2.mean[2],x1.mean[2]), type='b', col='green')
mid <- (x1.mean + x2.mean)/2

new_w <- 1000*w + mid

points(c(new_w[1], mid[1]), c(new_w[2], mid[2]), type='b', col='orange')
```


Female
Male
Weight
Snout Vent Length