Análisis de Datos 3

MANADINE

10 de enero de 2024

Teoría

1. (1 punto) Define los siguientes conceptos:

   a) Especificidad.

   b) Sensibilidad.

   c) Curva ROC.

   d) Área bajo la curva.

2. (1 punto) Explica cómo funcionan los bosques aleatorios.

3. (1 punto) Explica el método de los k vecinos más cercanos (kNN). Ventajas e inconvenientes principales.

Problemas

1. (6 puntos) Considera los datos morfométricos (en milímetros) `andariques.rda` relativos a dos tipos de crustáceos marinos.

```
load("~/Descargas/andariques.rda")
```

   a) (1 punto) Describe cada variable, incluyendo una estimación de la densidad de la variable `estatura`.

```
ls()
```

```
## [1] "andariques"
```

```
## los datos se llaman "andariques"
```

```
summary(andariques)
```

```
##      tipo         sexo     longitud.cefalotOrax anchura.posterior
##  verde :100   hembra:100   Min.   : 0.00        Min.   : 0.00
##  marrOn:100   macho :100   1st Qu.:12.80        1st Qu.:11.00
##                            Median :15.55        Median :12.75
##                            Mean   :15.38        Mean   :12.65
##                            3rd Qu.:18.05        3rd Qu.:14.30
##                            Max.   :23.10        Max.   :20.20
##  longitud.caparazOn anchura.caparazOn    estatura
##  Min.   :14.70      Min.   :17.10     Min.   : 0.00
##  1st Qu.:27.27      1st Qu.:31.50     1st Qu.:11.40
##  Median :32.10      Median :36.80     Median :13.90
##  Mean   :32.11      Mean   :36.41     Mean   :13.95
##  3rd Qu.:37.23      3rd Qu.:42.00     3rd Qu.:16.60
##  Max.   :47.60      Max.   :54.60     Max.   :21.60
```

Las dos primeras variables son cualitativas (describibles gráficamente mediante gráficas de barras) y por los recuentos se observa que se trata de muestras diseñadas, es decir, que el número de individuos que presentan cada característica se fijó de antemano. Para comprobarlo, veamos la distribución conjunta:

```
table(andariques[c("tipo","sexo")])
```

```
##         sexo
## tipo     hembra macho
##   verde      50    50
##   marrOn     50    50
```

En efecto, se trata de un diseño experimental. Las frecuencias no sirven para estimar proporciones poblacionales. Las variables cualitativas no tienen valores anómalos.

El resto de variables son cuantitativas. En tres de ellas hay observaciones nulas, lo que es imposible y han de indicar datos perdidos. Las sustituiremos por `NA`:

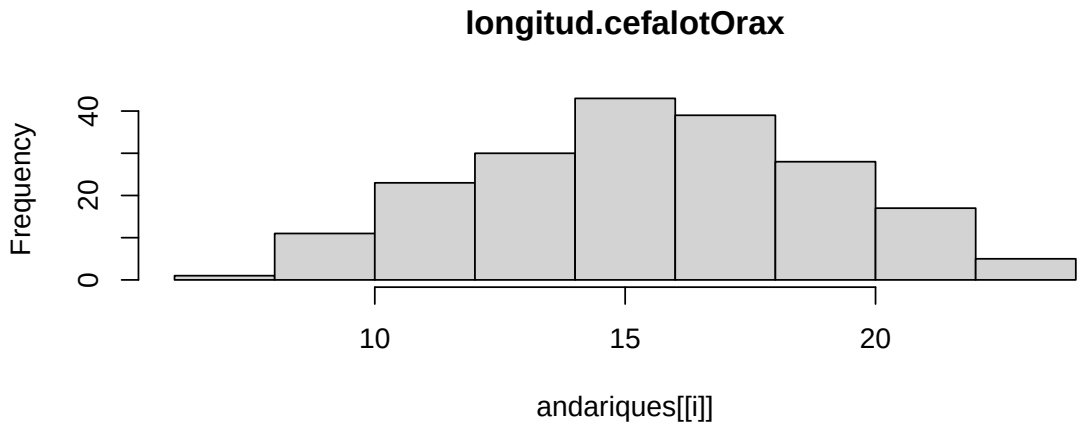
```
for(j in 3:ncol(andariques)) andariques[andariques[,j]==0,j] <- NA
```

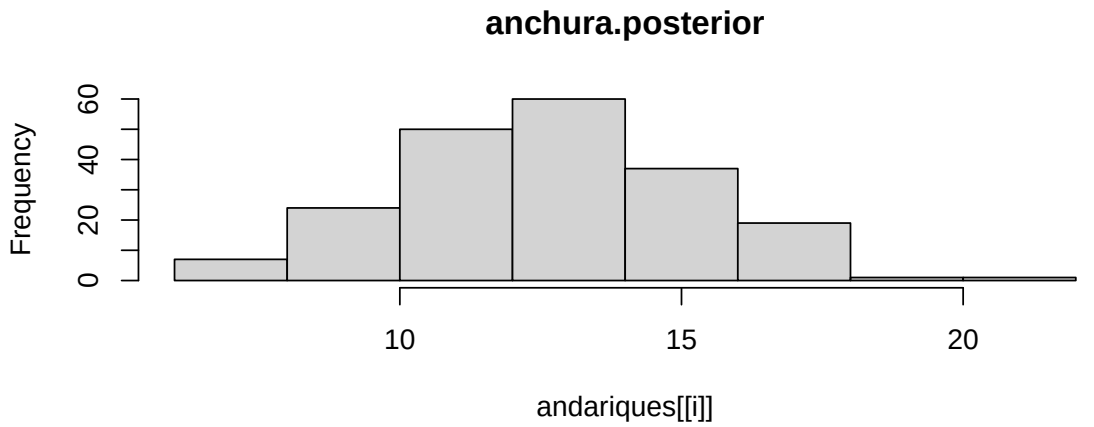
```
summary(andariques[-(1:2)])
```

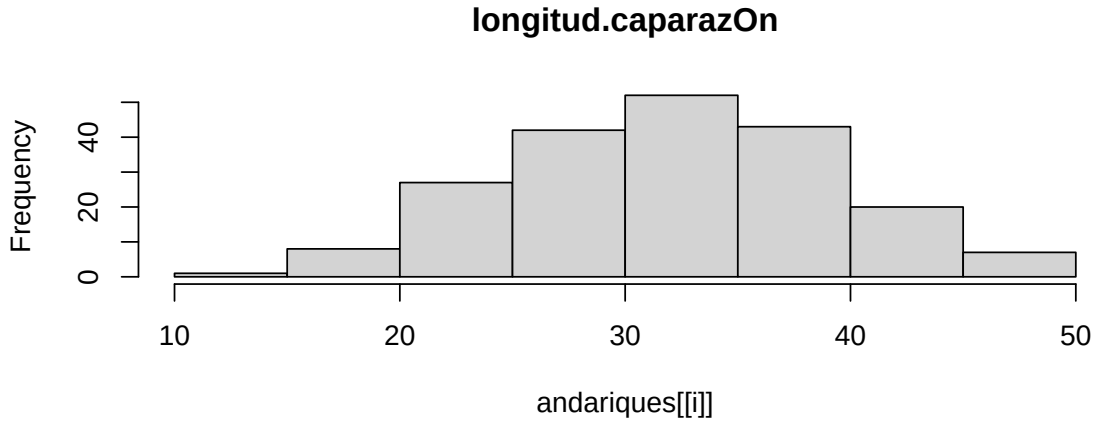
```
##  longitud.cefalotOrax anchura.posterior longitud.caparazOn
##  Min.   : 7.20        Min.   : 6.50     Min.   :14.70
##  1st Qu.:12.90        1st Qu.:11.00     1st Qu.:27.27
##  Median :15.60        Median :12.80     Median :32.10
##  Mean   :15.61        Mean   :12.72     Mean   :32.11
##  3rd Qu.:18.20        3rd Qu.:14.30     3rd Qu.:37.23
##  Max.   :23.10        Max.   :20.20     Max.   :47.60
##  NA's   :3            NA's   :1
##  anchura.caparazOn    estatura
##  Min.   :17.10     Min.   : 6.10
##  1st Qu.:31.50     1st Qu.:11.40
##  Median :36.80     Median :13.90
##  Mean   :36.41     Mean   :14.02
##  3rd Qu.:42.00     3rd Qu.:16.60
##  Max.   :54.60     Max.   :21.60
##                    NA's   :1
```

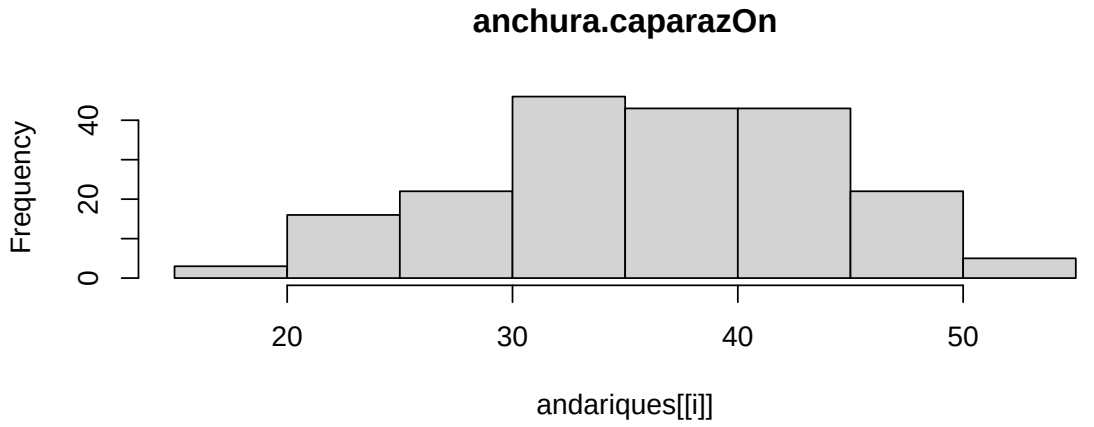
No se aprecia nada raro en los números que quedan. Veamos representaciones gráficas de cada distribución (histograma y densidad).

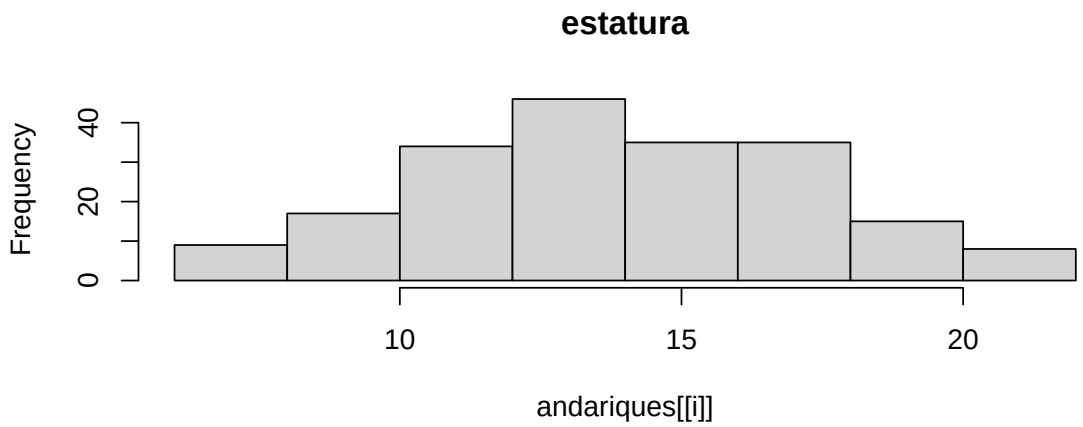
```
for (i in 3:ncol(andariques))
    hist(andariques[[i]],main=names(andariques)[i])
```



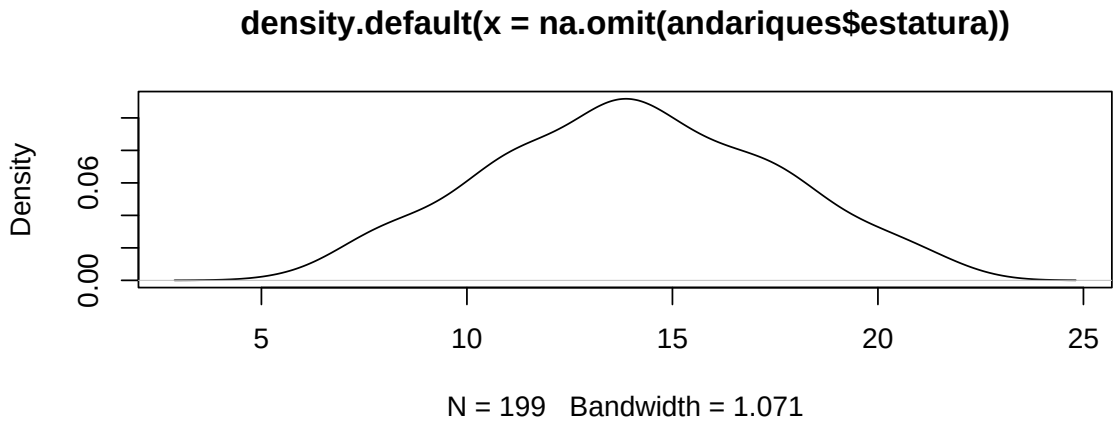

anchura.posterior
Frequency
0
20
40
60
10
15
20
andariques[[i]]


longitud.caparazOn
Frequency
0
20
40
10
20
30
40
50
andariques[[i]]


anchura.caparazOn
Frequency
0
20
40
20
30
40
50
andariques[[i]]




```
plot(density(na.omit(andariques$estatura))) # requerida por enunciado
```



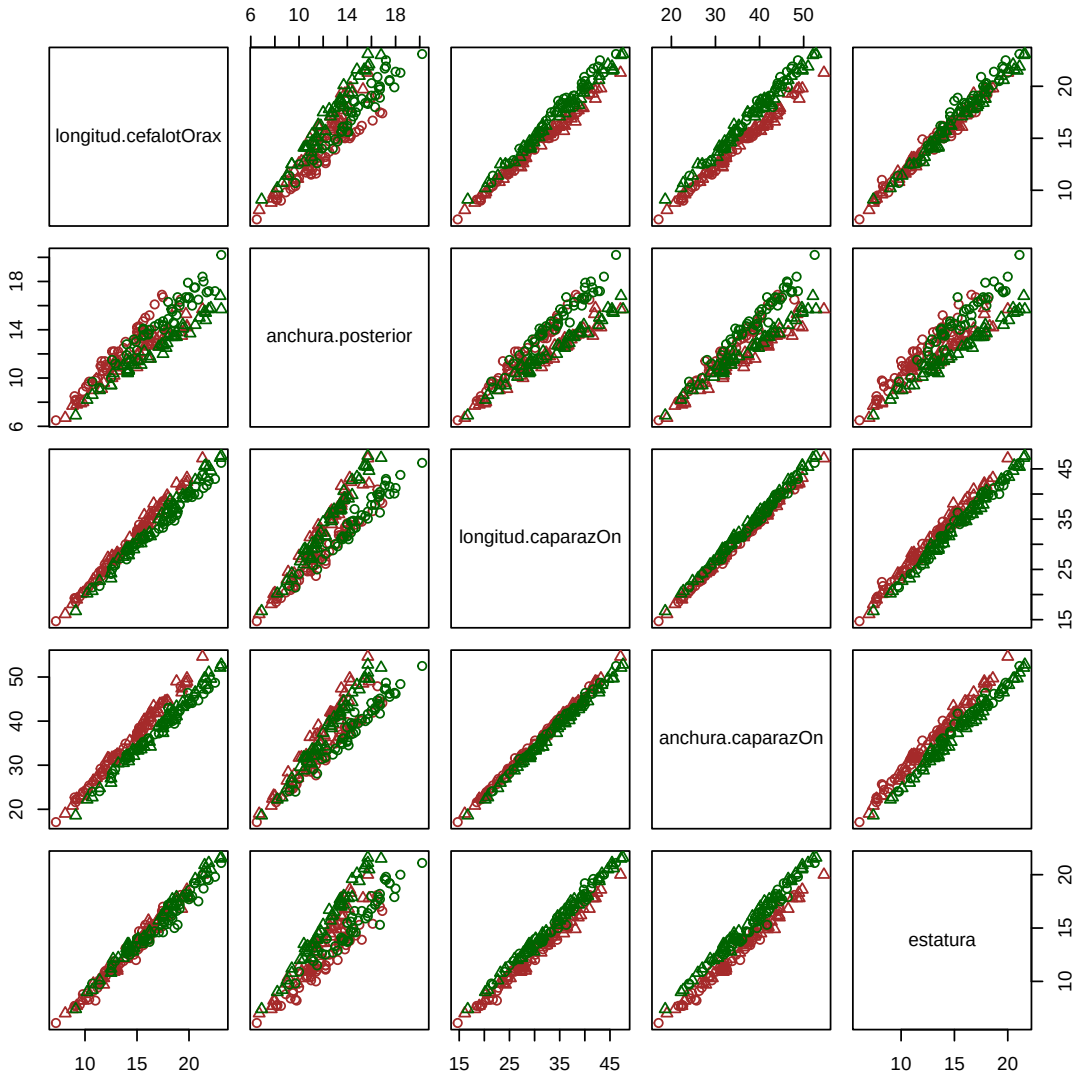
Todas las distribuciones son unimodales y acampanadas, lo que aparenta un buen comportamiento estadístico. Echemos un vistazo a las relaciones entre variables cuantitativas:

```
round(100*cor(andariques[-(1:2)], use="pair"))

##                     longitud.cefalotOrax anchura.posterior
## longitud.cefalotOrax                  100                91
## anchura.posterior                      91               100
## longitud.caparazOn                     98                89
## anchura.caparazOn                      97                90
## estatura                               99                89
##                     longitud.caparazOn anchura.caparazOn estatura
## longitud.cefalotOrax                 98                97       99
## anchura.posterior                    89                90       89
## longitud.caparazOn                  100               100       98
## anchura.caparazOn                   100               100       97
```

```
## estatura                                98                 97       100
```

```
pairs(andariques[-(1:2)],
      col = c("brown","darkgreen")[as.numeric(andariques$tipo)],
      pch = c(1,2)[as.numeric(andariques$sexo)])
```



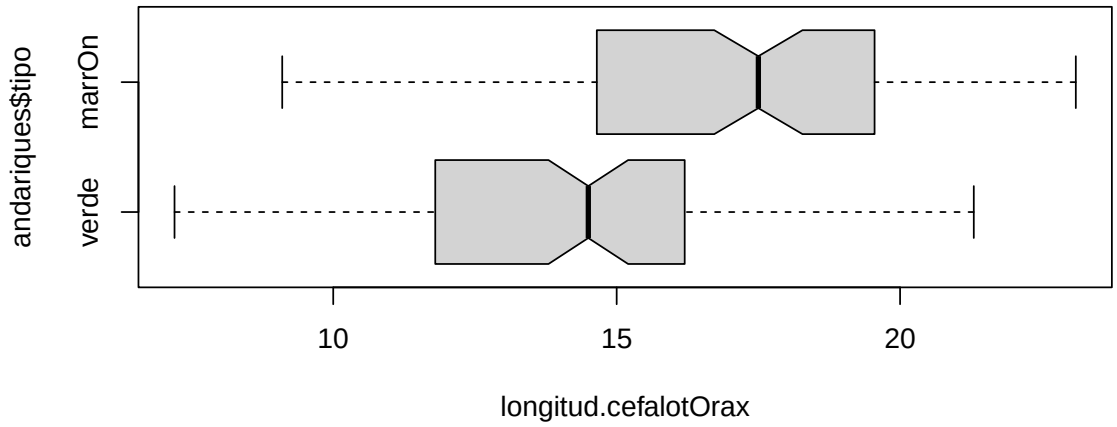
Se aprecia que las relaciones son fundamentalmente lineales (o sea, rectas) con cierta influencia de `tipo` (por ejemplo, entre `longitud.cefalotOrax` y `anchura.caparazOn`) o `sexo` (sobre todo cuando interviene `anchura.posterior`.

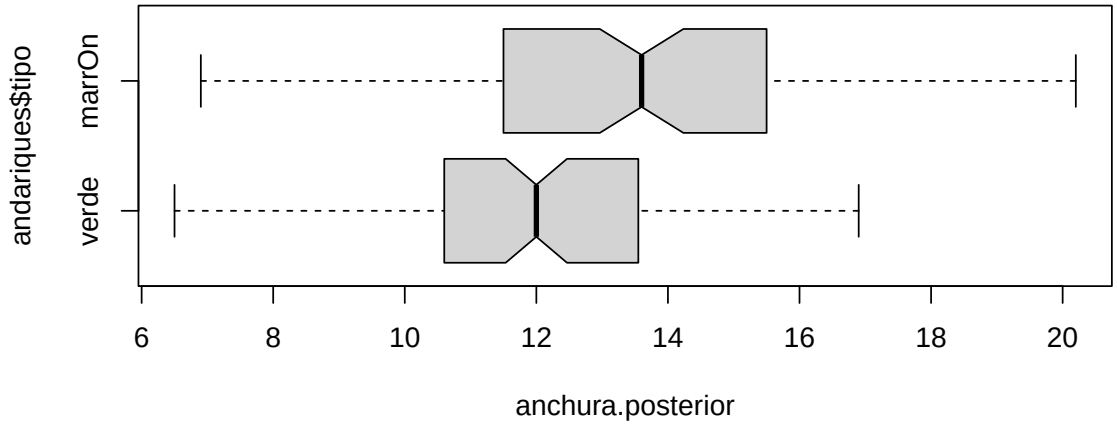
b) (1 punto) Comenta la relación de `tipo` con cada variable restante.

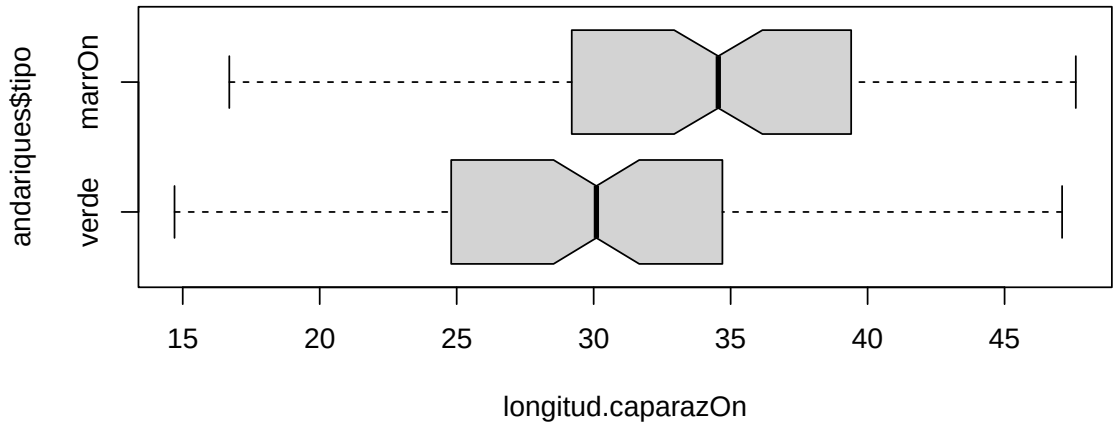
Ya habíamos visto que `tipo` y `sexo` procedían de un diseño experimental de 50 individuos en cada combinación. Por otro lado, es natural suponer que se trata de variables independientes.

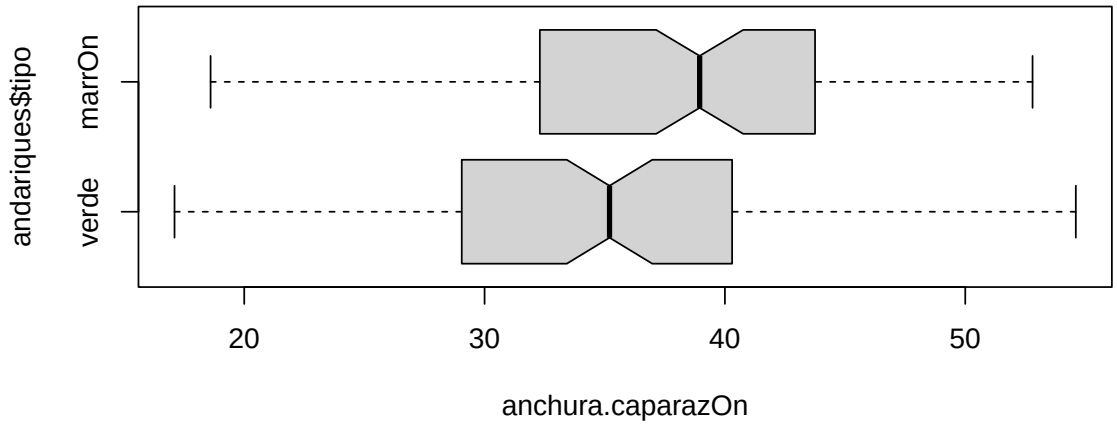
Para visualizar la relación entre tipo y las variables cuantitativas, usaremos diagramas de cajas:

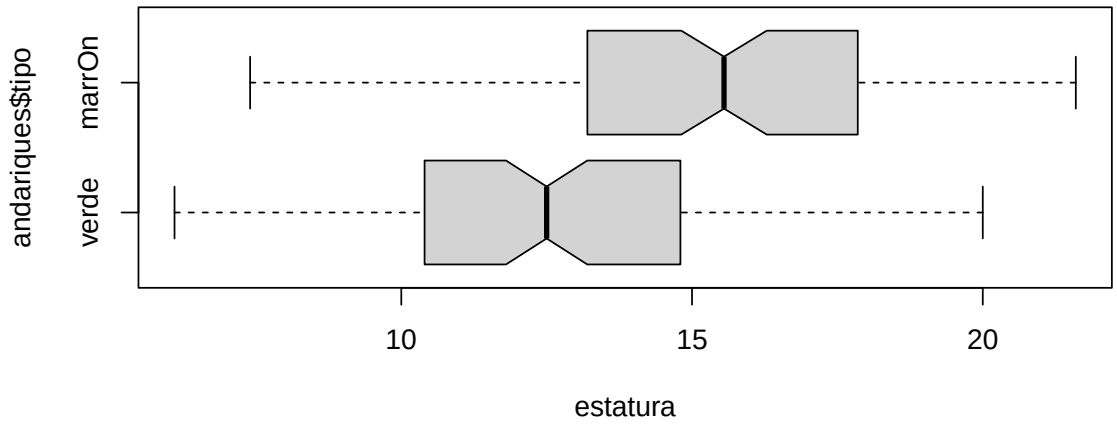
```r
for (i in 3:ncol(andariques))
    boxplot(andariques[[i]]~andariques$tipo, xlab=names(andariques)[i],
            horizontal=TRUE, notch=TRUE)
```






andariques$tipo
marrOn
verde
15
20
25
30
35
40
45
longitud.caparazOn


andariques$tipo
marrOn
verde
20
30
40
50
anchura.caparazOn


andariques$tipo
marrOn
verde
10
15
20
estatura

Hay diferencias significativas en todas las variables, y siempre el tipo marrón toma valores mayores:

```
sapply(names(andariques)[-(1:2)],
       function (variable)
           wilcox.test(as.formula(paste(variable, "~ tipo")),
                       andariques, alternative="less") $ p.value)
```

```
## longitud.cefalotOrax    anchura.posterior   longitud.caparazOn
##         9.621316e-10         1.162128e-05         3.734323e-05
##    anchura.caparazOn              estatura
##         1.169878e-03         2.937616e-09
```

c) (3 puntos) Propón dos modelos para predecir el `tipo` a partir del resto de variables,

- uno basado en una máquina de vector soporte,

```
library(e1071)
mvs <- svm(tipo ~ ., andariques)
summary(mvs)
```

```
##
## Call:
## svm(formula = tipo ~ ., data = andariques)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##
## Number of Support Vectors:  134
##
##  ( 67 67 )
##
##
## Number of Classes:  2
##
## Levels:
##  verde marrOn
```

```
## para usar tune hay que eliminar NA:
tune (svm, tipo~., data = na.omit(andariques),
      ranges = list(cost=c(1000,10000,1e5),
                    gamma=c(.001,1e-4,1e-5)),
      tunecontrol=tune.control(cross=nrow(na.omit(andariques))))
```

```
##
## Parameter tuning of 'svm':
##
```

```
## - sampling method: leave-one-out
##
## - best parameters:
##  cost gamma
##  1000 0.001
##
## - best performance: 0
## con tales parámetros se consigue 100% aciertos
```

Veamos cuántos vectores soporte usa el modelo:

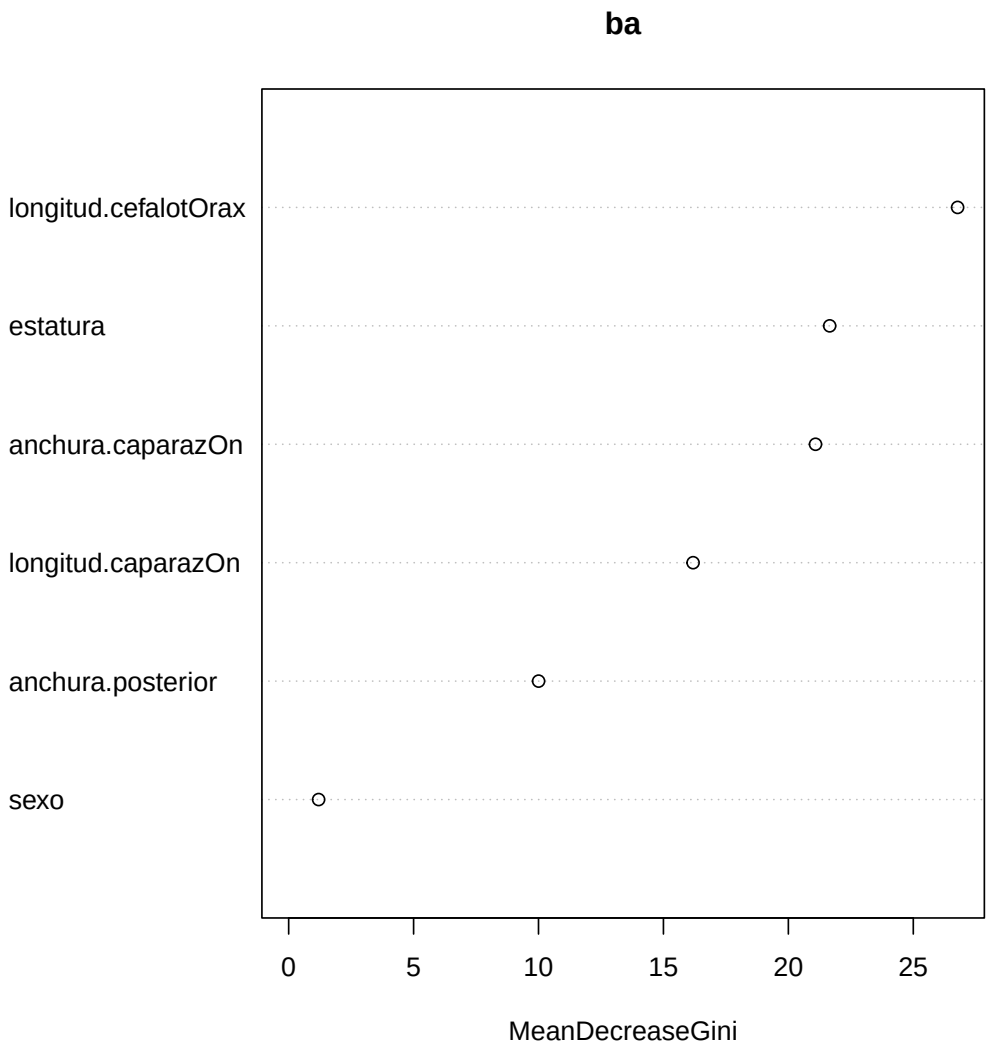
```
length(mvs$index) # número de vectores soporte
## [1] 134
nrow(andariques)  # tamaño muestral
## [1] 200
```

Más de la mitad de los puntos son vectores soporte, por lo que no sería posible una interpretación sencilla.

- y otro basado en un método distinto de clasificación supervisada impartido en la asignatura.
  Se usa aquí el bosque aleatorio por ejemplo. En el apartado siguiente se mencionarán también redes neuronales y *bagging*.

```
library(randomForest)
## randomForest 4.7-1
## Type rfNews() to see new features/changes/bug fixes.
ba <- randomForest(tipo ~ ., na.omit(andariques))
ba
##
## Call:
##  randomForest(formula = tipo ~ ., data = na.omit(andariques))
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 10.77%
## Confusion matrix:
##        verde marrOn class.error
## verde     86     11   0.1134021
## marrOn    10     88   0.1020408
importance(ba)
##                       MeanDecreaseGini
## sexo                          1.199333
## longitud.cefalotOrax         26.773587
## anchura.posterior            10.003530
## longitud.caparazOn           16.186780
```

```
## anchura.caparazOn            21.087143
## estatura                     21.654114
varImpPlot(ba)
```



Por lo visto en la matriz de confusión, la tasa de error es similar en ambas categorías,[1] por lo que no haremos distinción entre los dos tipos de errores en lo sucesivo.

d) (1 punto) Compara ambos modelos.

Emplearemos validación cruzada dejando uno fuera. Se omiten los casos con valores perdidos, ya que `randomForest` da error y `svm` no los clasifica.

Proporción de aciertos de la máquina de vector soporte:

```
aciertos <- sapply (1:nrow(andariques),
                    function (i)
```

[1] También la máquina de vector soporte tenía la misma tasa, cero.

```
              {
                  modelo <- svm(tipo ~ ., andariques[-i,],
                                cost = 1e4, gamma = 1e-4)
                  res <- predict(modelo, andariques[i,])
                  if (length(res)==1) res==andariques[i,"tipo"] else NA
              })
mean(aciertos, na.rm=TRUE)

## [1] 1
```

Proporción de aciertos del bosque aleatorio:

```
aciertos <- sapply (1:nrow(na.omit(andariques)),
              function (i)
              {
                  modelo <- randomForest(tipo ~ ., na.omit(andariques)[-i,])
                  predict(modelo, na.omit(andariques)[i,]) ==
                      na.omit(andariques)[i,"tipo"]
              })
mean(aciertos)

## [1] 0.9025641
```

Proporción de aciertos de un perceptrón con 5 neuronas ocultas:

```
library(nnet)
aciertos <- sapply (1:nrow(na.omit(andariques)),
              function (i)
              {
                  modelo <- nnet(tipo ~ ., na.omit(andariques)[-i,],
                                 size=5, decay=0.01, trace=FALSE)
                  na.omit(andariques)[i,"tipo"] ==
                    predict(modelo,na.omit(andariques)[i,],type="class")
              })
mean(aciertos)

## [1] 1
```

Proporción de aciertos del *bagging*:

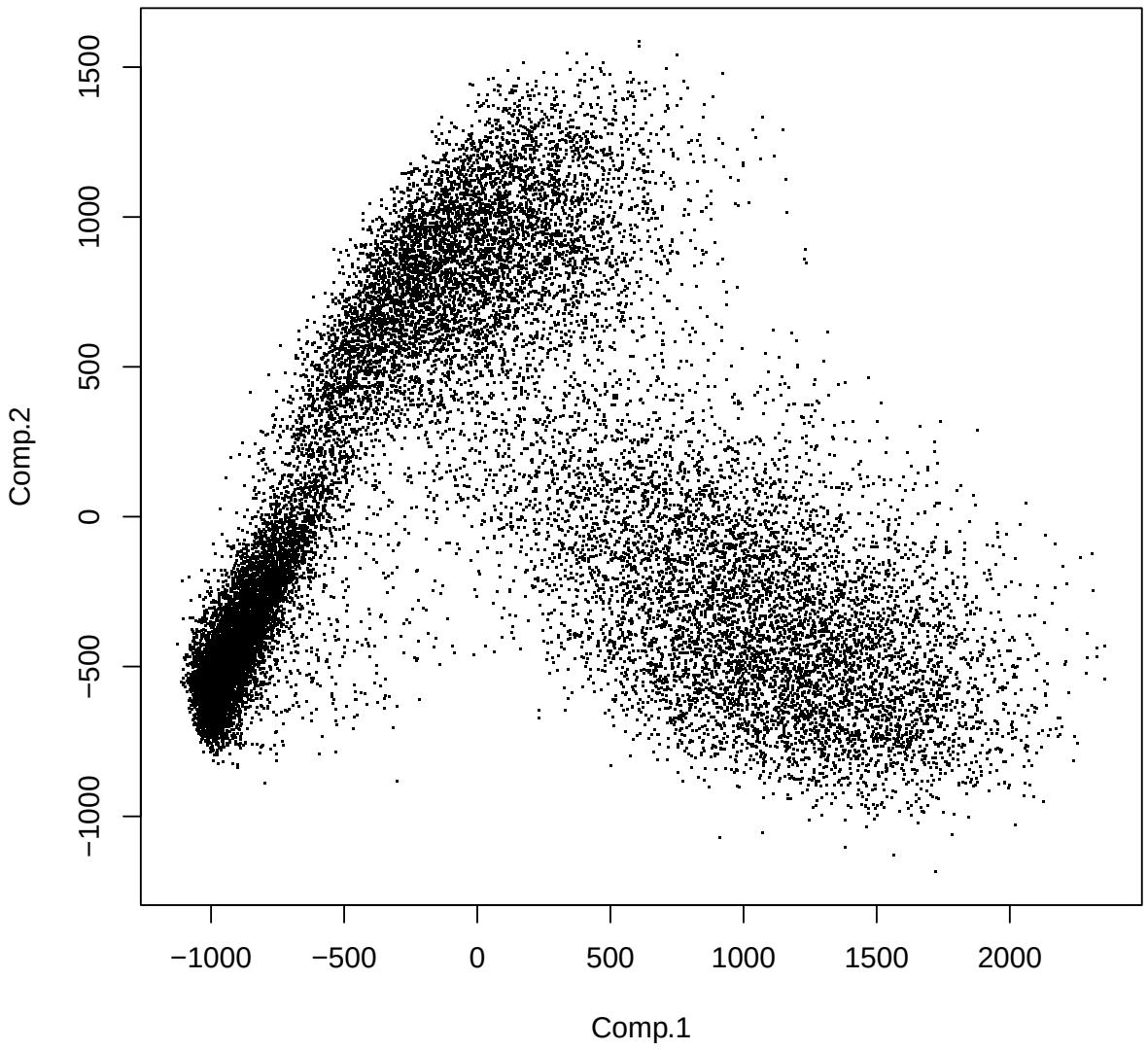
```
library(ipred)
aciertos <- sapply (1:nrow(na.omit(andariques)),
              function (i)
              {
                  modelo <- bagging(tipo ~ ., na.omit(andariques)[-i,])
                  na.omit(andariques)[i,"tipo"] ==
                    predict(modelo,na.omit(andariques)[i,])
              })
mean(aciertos)
```

```
## [1] 0.8974359
```

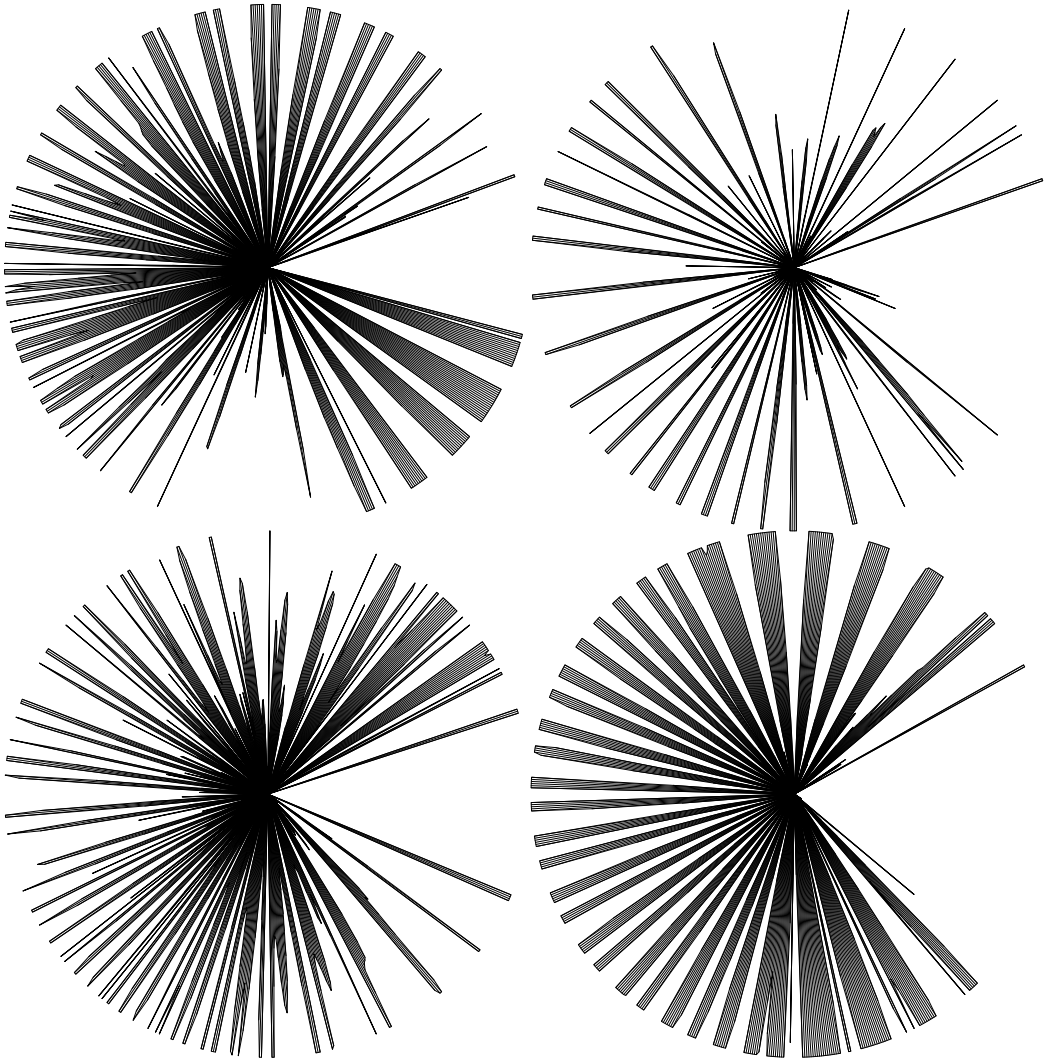
- La máquina de vector soporte y la red neuronal consiguen una predicción perfecta con estos datos.
- El bosque aleatorio y el *bagging* tienen un 10 % de error.

2. (1 punto) Considera los datos `temenis.rda`, correspondientes a niveles de gris en imágenes de 28×28 pixeles. Explica cómo crear en R un mapa autoorganizado 2×2 con dichos datos.

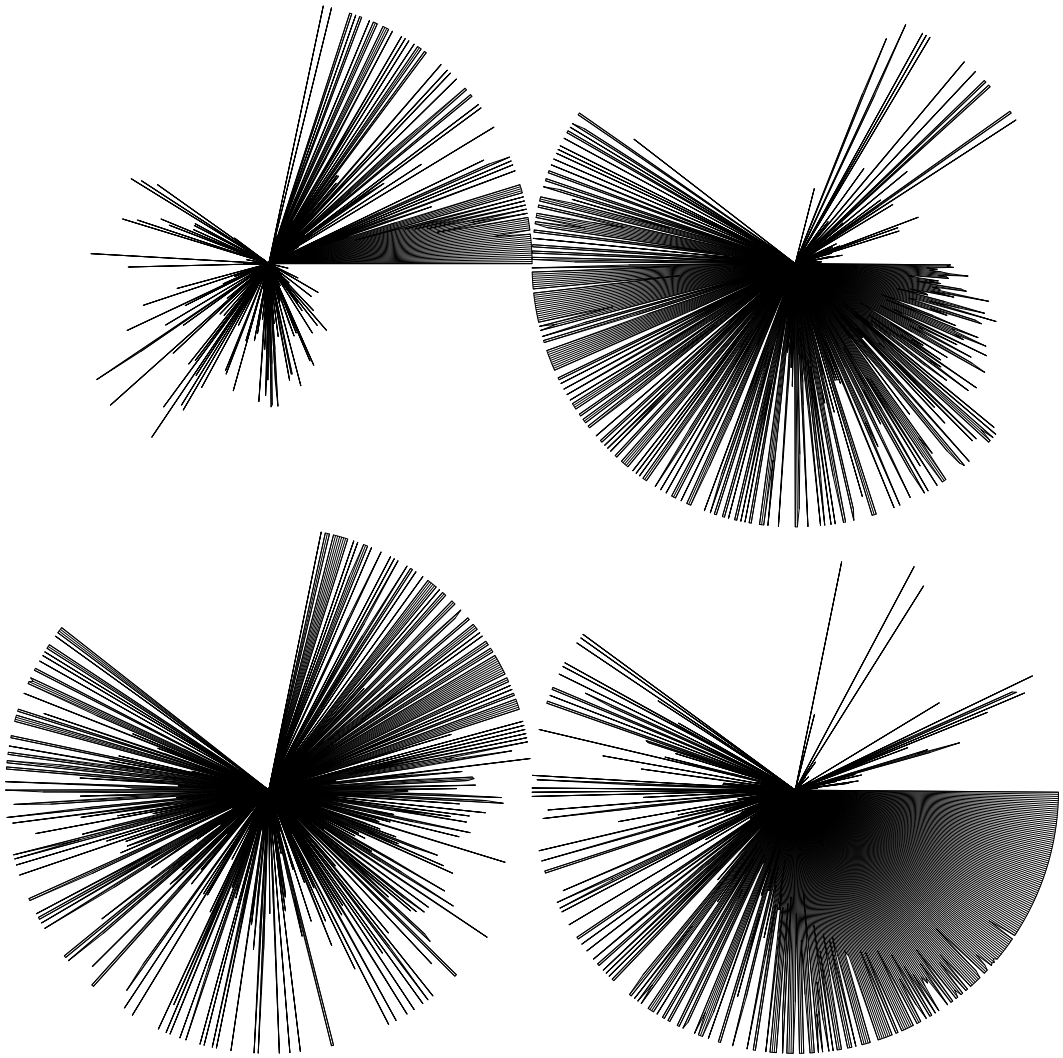
```
load("temenis.rda")
a <- princomp(temenis)
plot(a$scores[,1:2], pch=".") # tres cúmulos
```



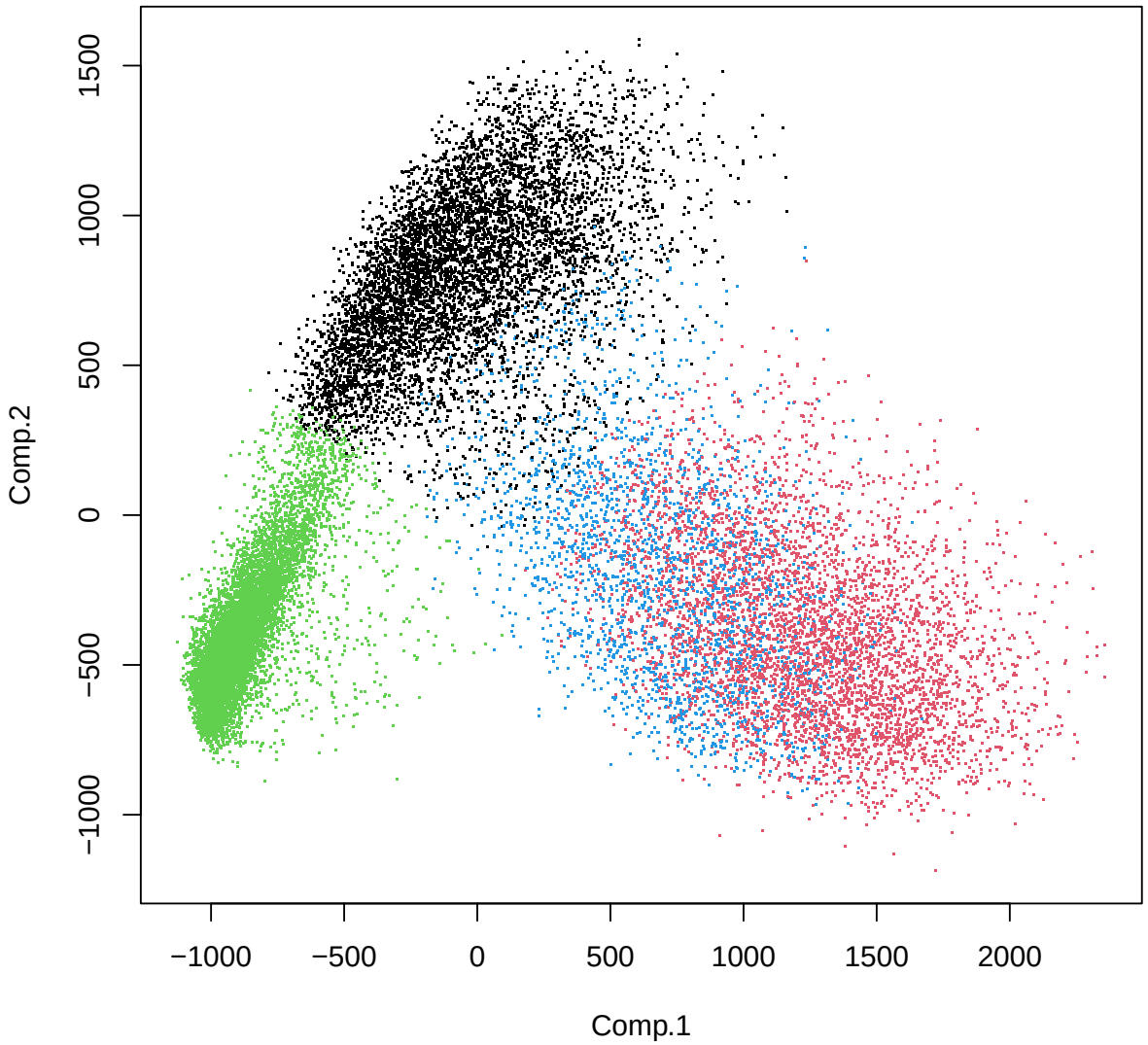
```r
library(class)
m <- batchSOM(temenis, somgrid(2,2), c(2,2,2,1,1,1,0,0,0))
plot(m) # tres categorías densas, una rala
```



```
## ordeno variables según importancia en la primera componente
temeniso <- temenis[,order(a$loadings[,1])]
mo <- batchSOM(temeniso, somgrid(2,2), c(2,2,2,1,1,1,0,0,0))
plot(mo)
```



```
g <- as.numeric(knn1(mo$codes,temeniso,1:4))
plot(a$scores[,1:2], pch=".", col=g)
```



Negros y azules constituyen el mismo cúmulo.