Caso KcHospital: Analysis en r

Lectura de los datos del caso

La lectura de datos en el lenguage **R** es muy flexible. Aunque existen funciones para la lectura de dtos procedentes d eotros programas de análisi estadístico, una formaestándar es la lectura de dtos en formato ASCII separados con tabulador. Si tenemos los datos en otro programa de análisis de datos o en una hoja electrónica Excel, sólo tendremos que grabarlos en formato texto con separción de tabulaciones El fichero se grabará con la extensión .dat o .txt, como es el caso del ficheor de datos del hopsital KC ("hospital/hospital.txt") y las preferencias de los consumidores entrevistados ("hospital/hospital.prefs.txt"). Para su lectura en el entorno **R** utilizamos la función `read.table`. Como argumentos utilizamos el nombre y ubicación del fichero de datos, entrcomillado, y la indicación de que la primera fila del ficheor contiene los nombres de las varaibles (`header=T`). En este ejemplo os muestro también como seleccionar la carpeta donde tengo los datos y los guiones que utilizo en el análisis (el camino (en inglés, path) hacia los documentos). La función `setwd()` utiliza como argumento la localización la carpeta en mi disco ("/Users/jlopezsi/Documents/github/mktg-models-projects"), localización que cambiará en vuetro caso. También he especificado la carpareta `hospital` donde está el fichero de datos `"hospital.txt"`. Podeís utilizar esta estrategia o bien utilizar la función `file.choose()`.

```
#utilizar file.choose() si no conocéis dónde se encuentra vuestro fichero
setwd("/Users/jlopezsi/Documents/github/mktg-models-projects")
kchospital <- read.table("hospital/hospital.txt", header = TRUE)
kcprefs <- read.table("hospital/hospital.prefs.txt", header = TRUE)
```

Ahora que ya los tenemos en le espacio de trabajo podemos consultar nombre de las variables contenidas en la tabla de datos kchospital que acabamos de crear. Para ello utilizamos la función `names` y como argumento el nombre de la tabal de datos, kchospital o kcprefs, sin comillas. Aquí tenemos el nombre de las variables contenidas en ambos ficheros.

```
names(kchospital)
```

```
##  [1] "CASO"     "KCMATER"  "HB"       "HC"       "HD"       "HE"
##  [7] "HF"       "HG"       "HH"       "HI"       "FIREP"    "FICCUID"
## [13] "FIDIST"   "FICDOC"   "FICAMIG"  "FIPUBLI"  "FIAMAB"   "FIIMOD"
## [19] "FIEMOD"   "FISEG"    "FAMIL"    "ELECHOSP" "SERPERF"  "ALEALTAD"
## [25] "AMSERV"   "ASCOM"    "ACALIDAD" "ACOMOD"   "VKC"      "VB"
## [31] "VC"       "VD"       "VE"       "VF"       "VG"       "VH"
## [37] "VI"       "INTERES"  "COMPROM"  "ESFUER"   "MOTIV"    "DIFICULT"
## [43] "SEXO"     "ESTADOMA" "EDAD"     "NIVELEDU" "INGRESOS"
```

```
names(kcprefs)
```

```
## [1] "HK" "B"  "C"  "D"  "E"  "F"  "G"  "H"  "I"
```

Si queremos ver el contenido de cualquier variables debemos hacer referencia a la tabla de datos que lo contiene. Por ejemplo, `kchospital$FIREP` nos permite acceder al contenido de la variable `reputación` que está en la tabla de datos `kchospital`. Cuidado, en **R** se diferencia entre las mayúsculas y las minúsculas. Por ello deberemos introducir los nombre de los ficheros y variables en minúsculas o mayúsculas según sea el caso. Aquí el nombre de la tabla de datos está en minúsculas y las variables en mayúsculas; así debemos hacer referencia a ellas.

Bases de segmentación

para facilitar el análisis de segmentación primero vamos a seleccionar las bases de segmentación, las variables que los consumidores consideran importantes a la hora de seleccionar un hospital materno-infantil. Concretamente,

```
bases <- kchospital[,c(11:20)]
names(bases)
```

```
##  [1] "FIREP"   "FICCUID" "FIDIST"  "FICDOC"  "FICAMIG" "FIPUBLI" "FIAMAB"
##  [8] "FIIMOD"  "FIEMOD"  "FISEG"
```

Proceso de segmentación en dos etapas: aglomeración jerárquica y partición

Vamos a utilizar, en primer lugar, los algoritmos clásicos implementados en los programas de análisis de datos más habituales: el jerárquico aglomerativo, hclust, y el de partición de una muestra, kmeans. Si queremos saber algo más sobre el algoritmo implementado podemos utilizar la ayuda, help, y como argumento en nombre de la función de la que queremos saber algo, hclust,como argumento. Por ejemplo, `help(hclust)`. Con la función `scale(bases)`podemos normalizar los datos si es preciso.

clasificación con los datos originales

Para ello seguimos el proceso usual. Primero, analizamos visualmente la heterogeneidad que existe en la muestra de clientes o posibles clientes –para ello utilizamos un procedimiento jerárquico de aglomeración, `hclust`–, segundo, decidimos cuántos segmentos formar y aplicamos en procedimiento de partición del mercado, `kmeans`.

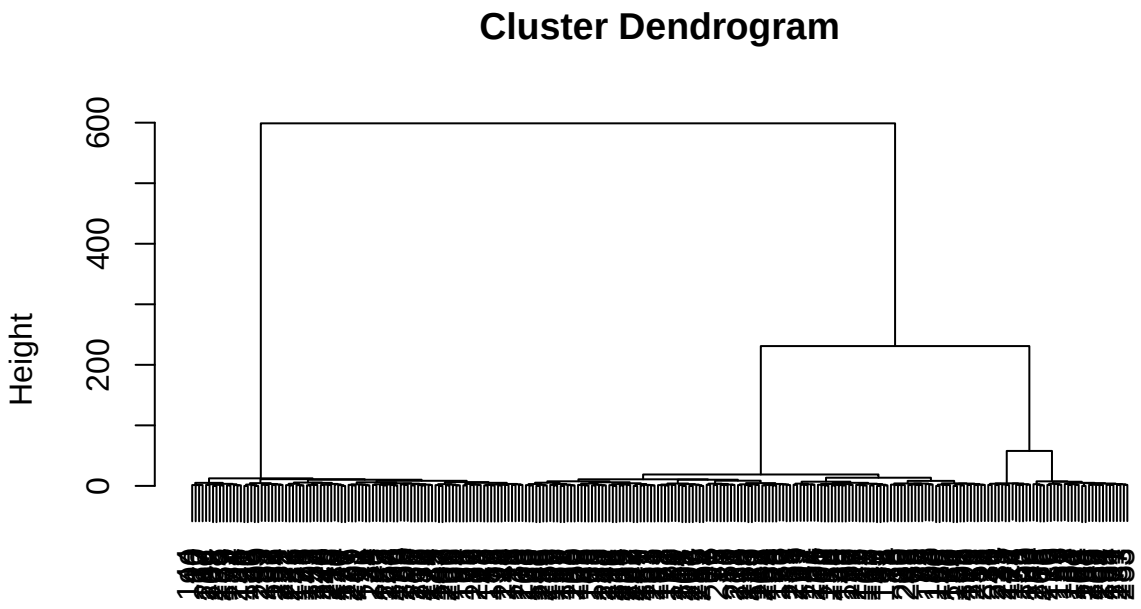
Exploración de la heterogeneidad en los datos

Veamos el resultado de la clasificación jerárquica.

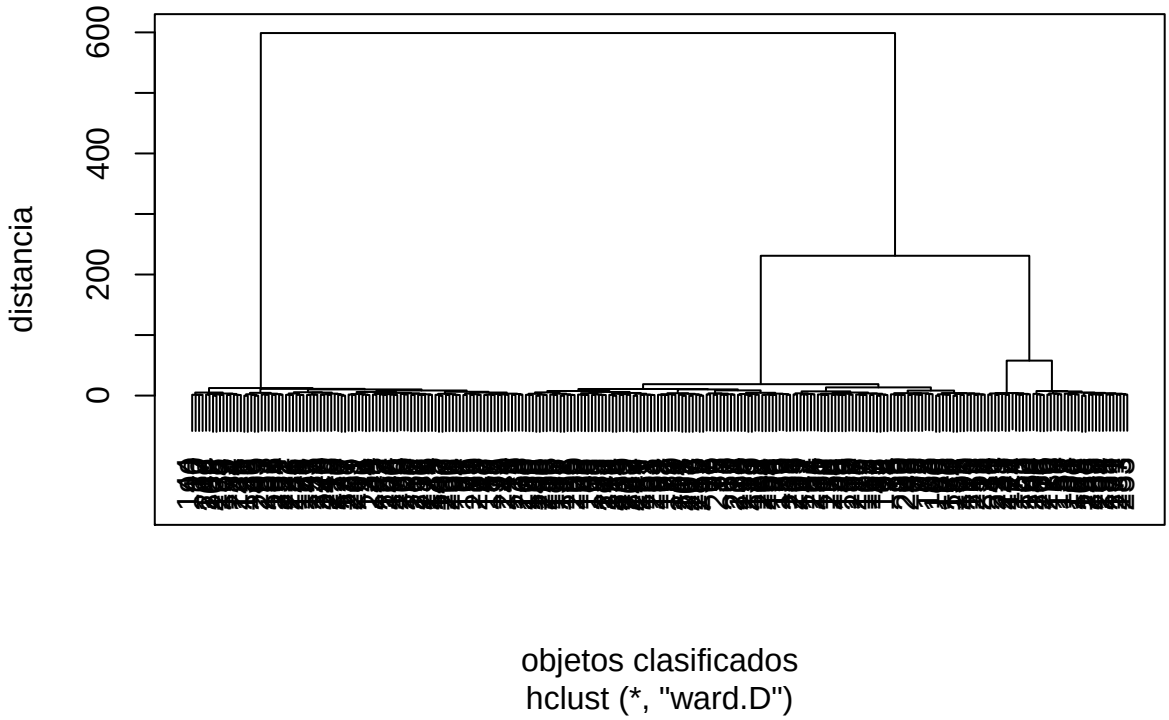
```
bases.hclust<-hclust(dist(bases), method="ward")
```

```
## The "ward" method has been renamed to "ward.D"; note new "ward.D2"
```

```
plot(bases.hclust)
```

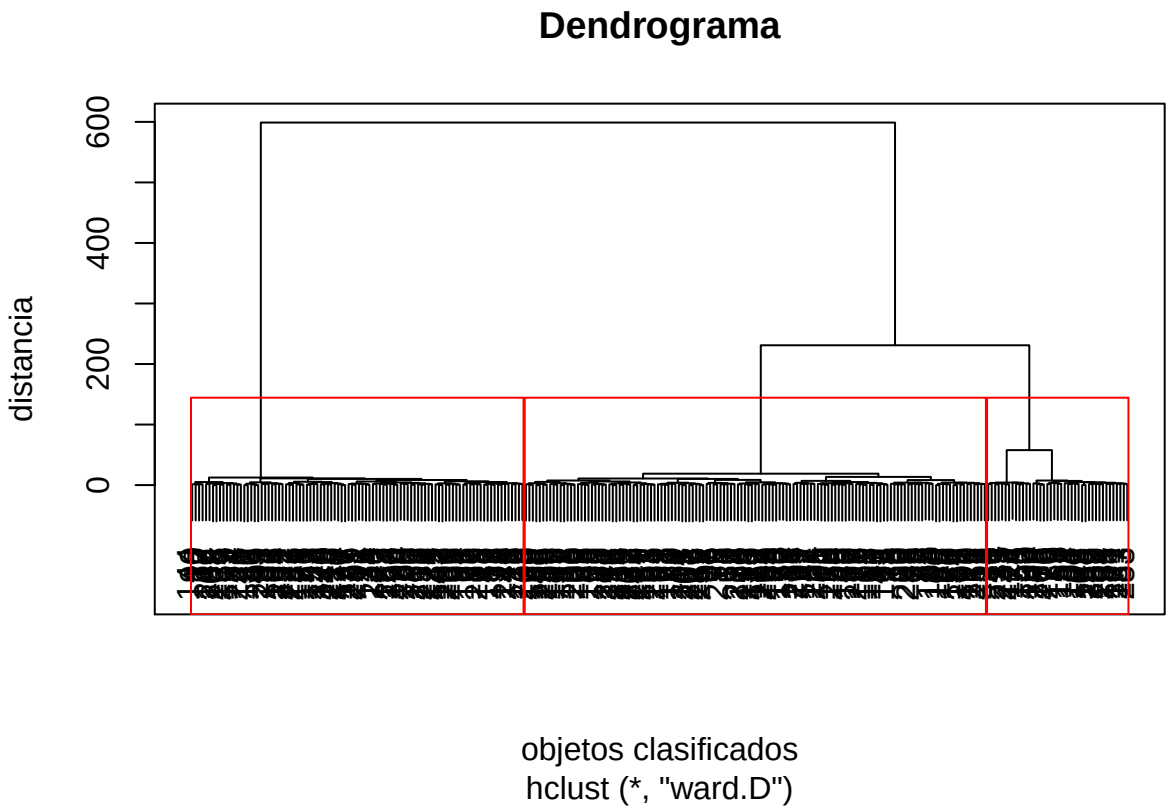



```
plot(bases.hclust, main="Dendrograma",ylab="distancia",xlab="objetos clasificados",frame.plot=TRUE)
```



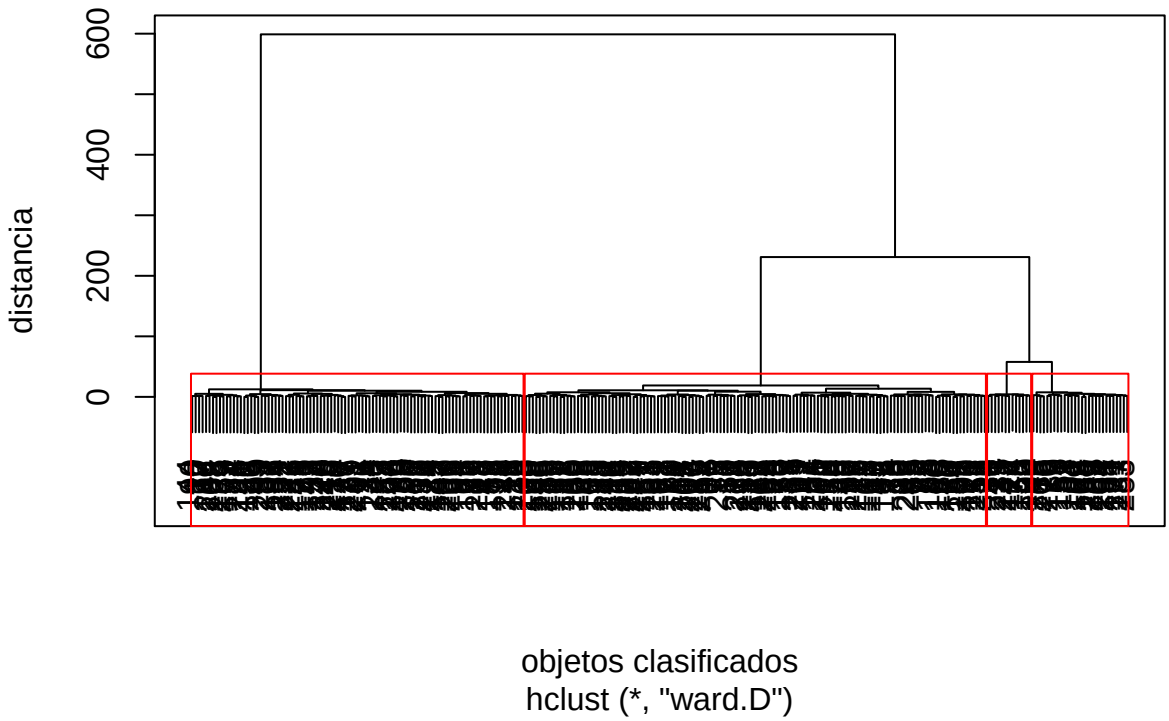
Aunque el dendrograma (que es como se denomina al gráfico en forma de árbol invertido que muestra el proceso de clasificación) ya es lo suficientemente elegante, podemos añadirle un título y modificar el nombre de los ejes del gráfico. Si queremos resaltar el número de grupos que formaríamos, podemos utilizar la siguiente función. Si fueran tres los segmentos a formar, obtendríamos:

```
plot(bases.hclust, main="Dendrograma",ylab="distancia",xlab="objetos clasificados",frame.plot=TRUE)
rect.hclust(bases.hclust, k=3, border=2)
```



Si formaramos cuatro segmentos, entonces utilizaríamos:

```
plot(bases.hclust, main="Dendrograma",ylab="distancia",xlab="objetos clasificados",frame.plot=TRUE)
rect.hclust(bases.hclust, k=4, border=2)
```



Partición con kmeans

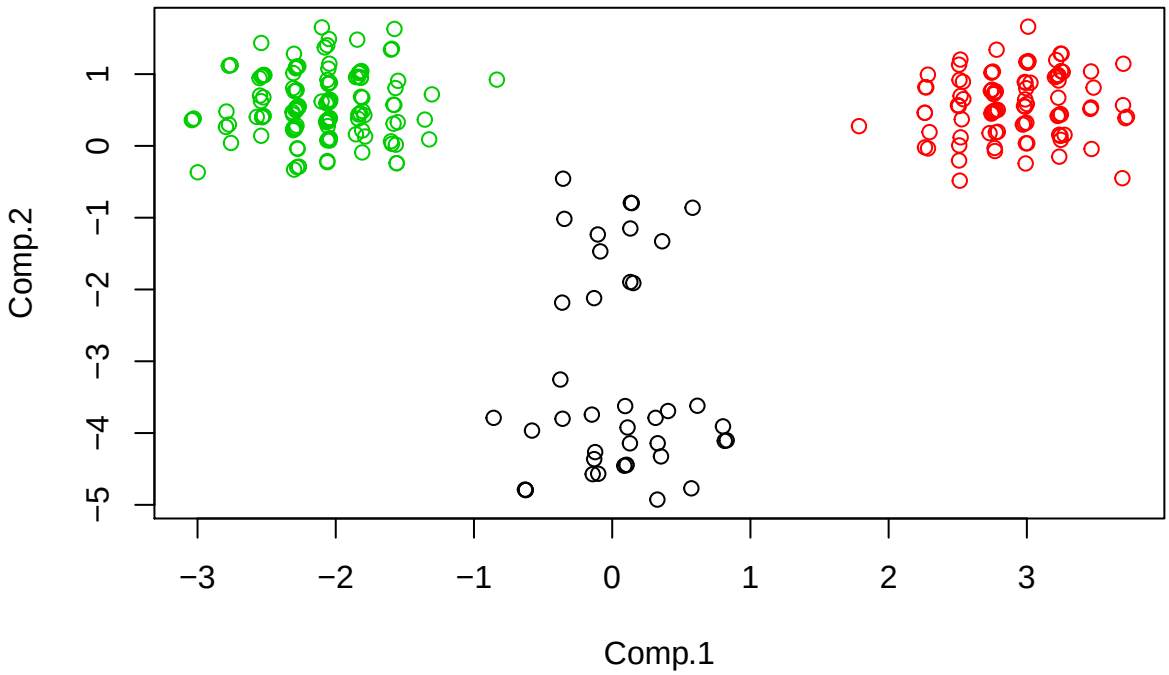
Durante la segunda fase, después de decidir cuántos grupos formaremos, utilizaremos un procedimiento de partición, concretamente `Kmeans`.

```
bases.kmeans3<-kmeans(bases, 3)
```

Ayudas visuales para analizar la estructura de los datos. Sólo si las bases de segmentación están muy correlacionadas. En caso contrario tendremos que basarnos en el dendograma que ya hemos obtenido.

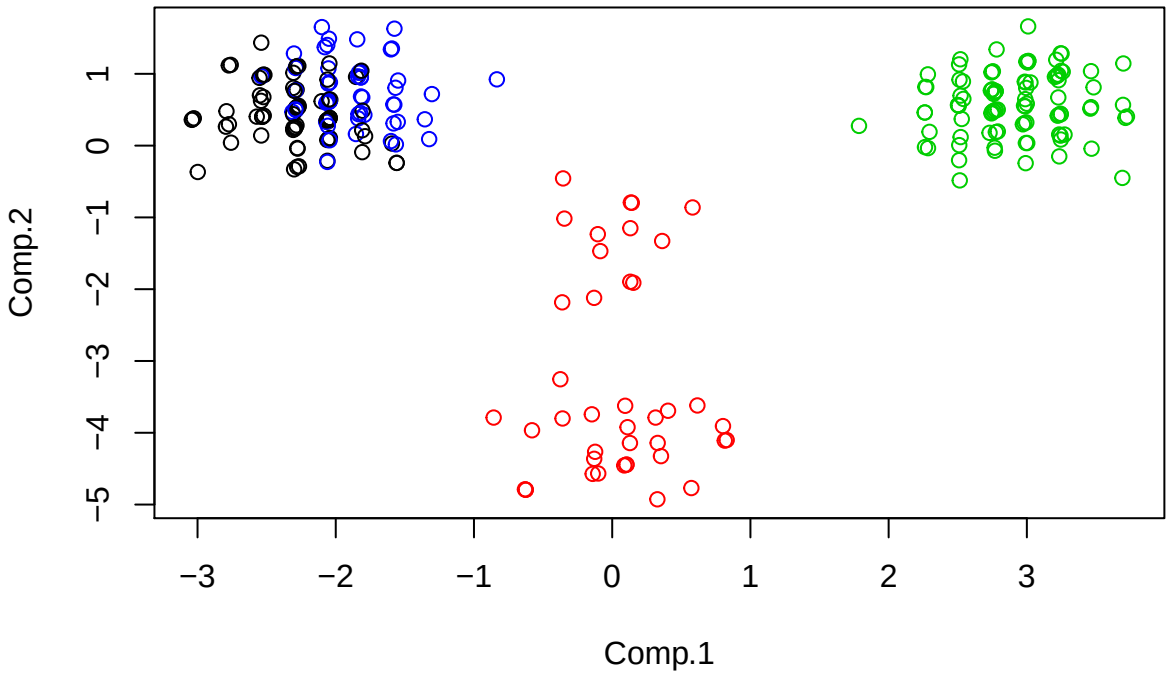
Si, además, hemos obtenido los componentes principales, podemos visualizar el resultado de la clasificación. Primero obtenemos los componentes principales y después utilizamos éstos para visualizar la segmentación realizada

```
library(MASS)
bases.pca<-princomp(bases, cor=T)
bases.puntos<-predict(bases.pca)
plot(bases.puntos[,1:2], col=bases.kmeans3$cluster)
```



Repetimos el proceso con cuatro grupos:

```
bases.kmeans4<-kmeans(bases, 4)
plot(bases.puntos[,1:2], col=bases.kmeans4$cluster)
```



Visualmente podemos ver que el algoritmo nos ha dividido un grupo que parece homogéneo, mientras que el grupo que parece formado por dos segmentos lo ha dejado igual. Esto es debido a que el algoritmo `kmeans` forma grupos esféricos.

¿Podemos mejorar la clasificación realizada?

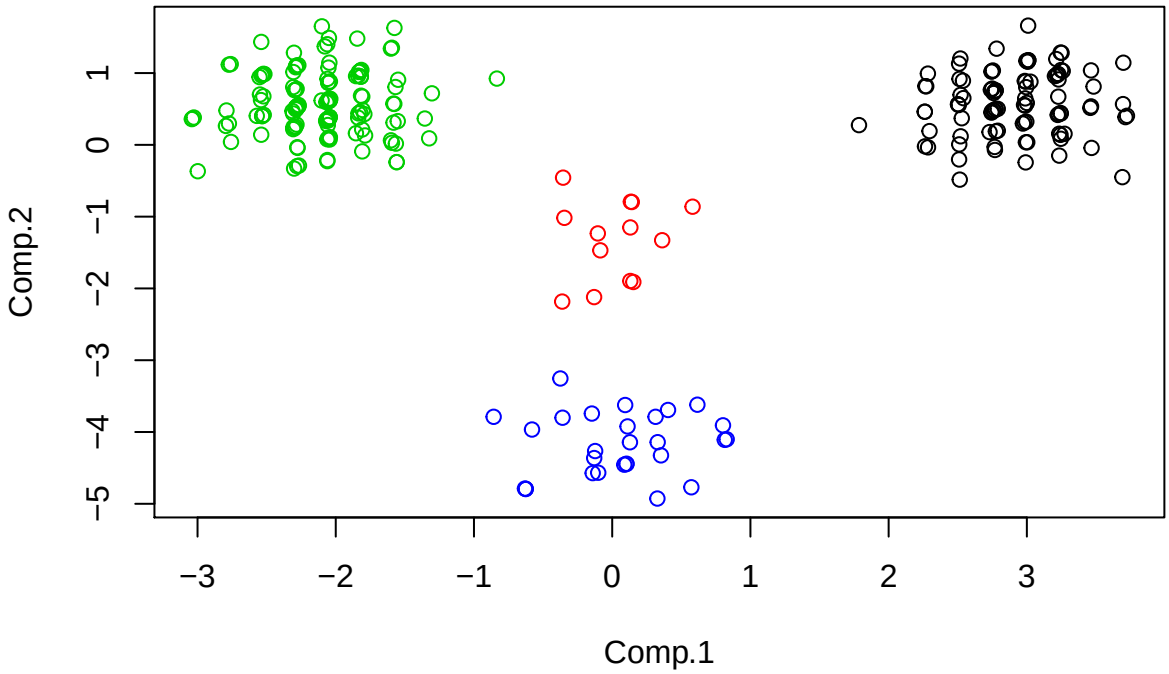
En ciertas ocasiones podemos mejorar la clasificación facilitando al algoritmo `kmeans` los centros iniciales a partir de los cuales iniciar la partición. Para ello tendremos que calcular la media de cada variable en cada uno de los grupos formados con el procedimiento jerárquico de aglomeración. Veamos cómo obtener las medias:

```
centros4<-tapply(as.matrix(bases),
                 list(rep(cutree(bases.hclust, 4),
                          ncol(as.matrix(bases))),
                      col(as.matrix(bases))
                      ),
                 mean)
options(digits=3)
centros4
```

```
##      1    2    3    4    5    6    7    8    9   10
## 1 5.50 2.74 2.81 5.28 2.10 2.20 2.80 5.34 5.16 2.62
## 2 2.46 2.77 5.77 2.23 2.08 2.08 2.85 2.23 2.31 2.69
## 3 2.47 5.51 2.77 2.13 5.08 2.17 5.80 2.41 2.16 2.65
## 4 2.50 2.36 2.89 2.18 2.32 5.21 2.86 2.29 2.32 5.57
```

Ahora repetimos el proceso y visualizamos el resultado.

```
bases.kmeans4<-kmeans(bases, centros4)
plot(bases.puntos[,1:2], col=bases.kmeans4$cluster)
```



¿Existe correlación entre las variables? Cambio de variables o construcción de las nuevas bases de segmentación

La existencia de correlación entre las variables podría ser uno de los motivos por los cuales el procedimiento no funcione correctamente. Averigüemos si las correlaciones son elevadas.

```
cor(bases)
```

```
##          FIREP FICCUID   FIDIST FICDOC FICAMIG FIPUBLI FIAMAB FIIMOD
## FIREP    1.000  -0.564 -0.12817  0.857  -0.664  -0.182 -0.675  0.858
## FICCUID -0.564   1.000 -0.20524 -0.572   0.839  -0.329  0.872 -0.567
## FIDIST  -0.128  -0.205  1.00000 -0.128  -0.218  -0.036 -0.193 -0.132
## FICDOC   0.857  -0.572 -0.12817  1.000  -0.664  -0.188 -0.671  0.843
## FICAMIG -0.664   0.839 -0.21779 -0.664   1.000  -0.257  0.917 -0.629
## FIPUBLI -0.182  -0.329 -0.03603 -0.188  -0.257   1.000 -0.279 -0.195
## FIAMAB  -0.675   0.872 -0.19262 -0.671   0.917  -0.279  1.000 -0.642
## FIIMOD   0.858  -0.567 -0.13187  0.843  -0.629  -0.195 -0.642  1.000
## FIEMOD   0.834  -0.563 -0.09700  0.840  -0.653  -0.163 -0.665  0.820
## FISEG   -0.214  -0.302 -0.00376 -0.205  -0.241   0.797 -0.279 -0.232
##         FIEMOD    FISEG
## FIREP    0.834 -0.21421
## FICCUID -0.563 -0.30159
## FIDIST  -0.097 -0.00376
## FICDOC   0.840 -0.20477
## FICAMIG -0.653 -0.24133
## FIPUBLI -0.163  0.79734
## FIAMAB  -0.665 -0.27925
## FIIMOD   0.820 -0.23185
## FIEMOD   1.000 -0.19085
## FISEG   -0.191  1.00000
```

Si existe correlación entre las variables que los consumidores consideran importantes a la hora de escoger un hospital, puede que la clasificación obtenida no sea la correcta. En este caso debemos eliminar la correlación existente entre las variables utilizadas. Para ello podemos utilizar una técnica ampliamente utilizada para la reducción de datos, el análisis de los componentes principales. Esta técnica crea unas nuevas variables a partir de una combinación lineal de las originales. Estas nuevas variables, componentes principales, poseen dos características muy útiles en el análisis de datos multivariantes: no están correlacionadas entre sí y nos explican la correlación que se observa entre las variables originales.

Para la extracción de los componentes principales que existen en la matriz de datos de factores importantes, bases, utilizaremos la biblioteca de programas de Venables y Ripley (1999), MASS.

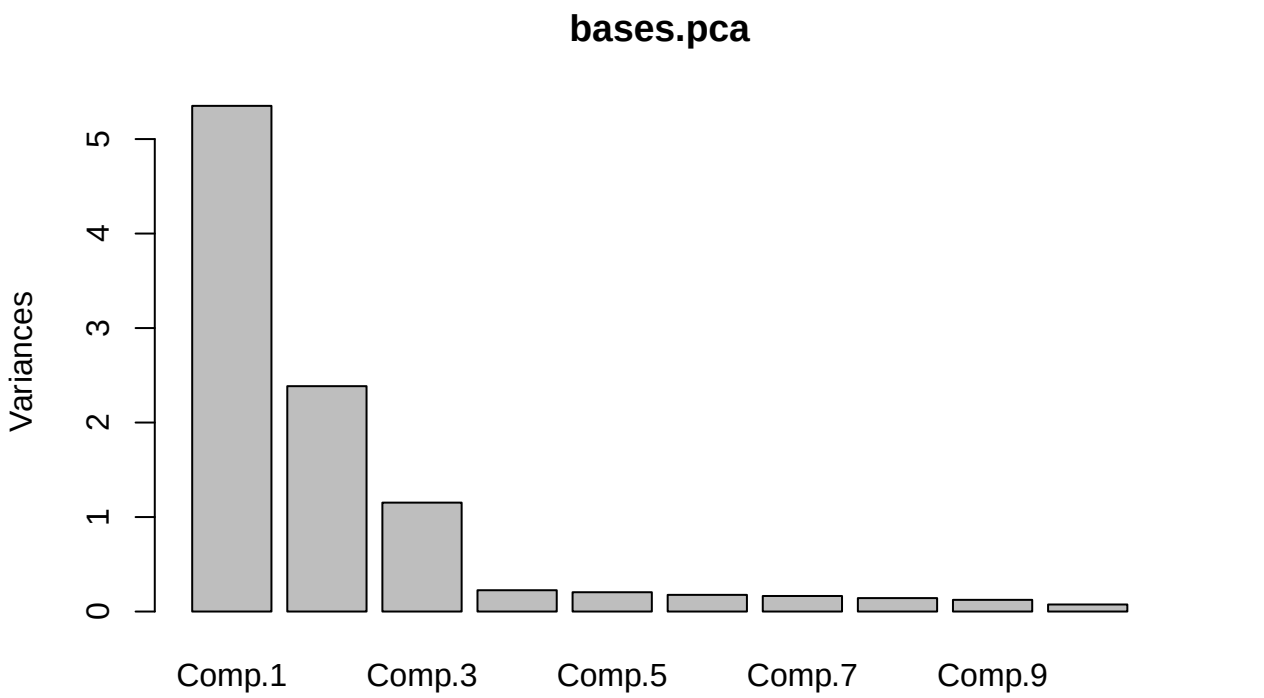
```
library(MASS)
bases.pca<-princomp(bases, cor=T)
summary(bases.pca)
```

```
## Importance of components:
##                          Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7
## Standard deviation      2.313  1.544  1.074 0.4750 0.4520 0.4198 0.4057
## Proportion of Variance  0.535  0.238  0.115 0.0226 0.0204 0.0176 0.0165
## Cumulative Proportion   0.535  0.774  0.889 0.9114 0.9319 0.9495 0.9659
##                          Comp.8 Comp.9 Comp.10
## Standard deviation     0.3773 0.3515 0.27316
## Proportion of Variance 0.0142 0.0124 0.00746
## Cumulative Proportion  0.9802 0.9925 1.00000
```

```
bases.puntos<-predict(bases.pca)
```

Para continuar con el análisis primero decidiremos con cuántos componentes principales trabajaremos. Para ello pediremos que nos muestre gráficamente la cantidad de varianza que las nuevas variables son capaces de explicar (función `screepot(fipca)` con el objeto creado como argumento). Podemos ser que con tres es suficiente pues a partir de cuarto componente la varianza explicada es inferior a la unidad (varianza de una variable original estandarizada) y se observa una inflexión en el gráfico.

```
screeplot(bases.pca)
```

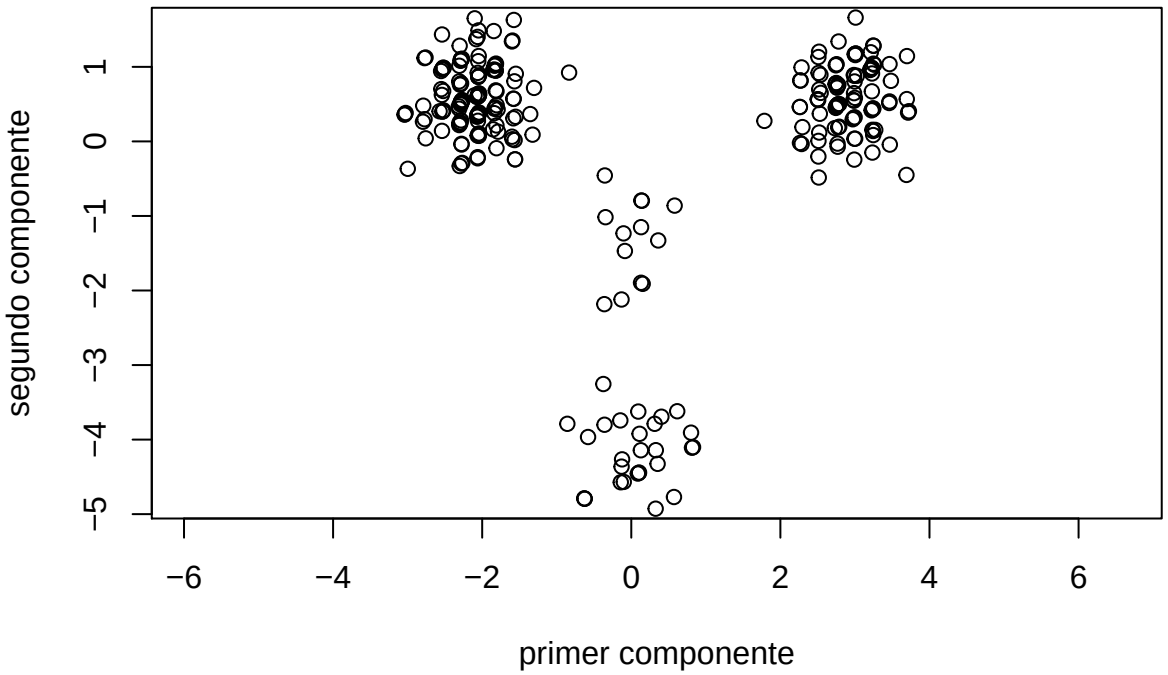


Ahora vamos a crear una nueva variable (una matriz de datos) con los puntos asignados a los clientes de la muestra en las nuevas variables o componentes principales (podemos utilizar predict(fipca) o bien fipca$scores, en ambos casos obtenemos el mismo resultado).

```
bases.puntos<-predict(bases.pca)
```

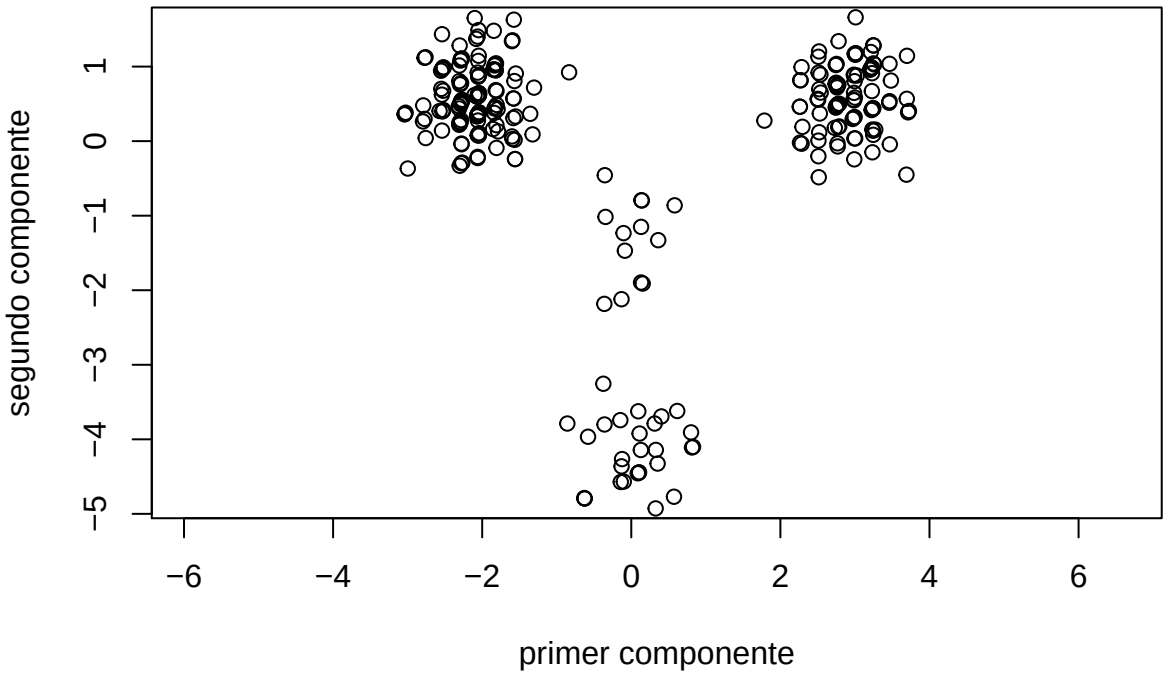
Una ventaja de trabajar con un espacio reducido de variables es poder visualizar cómo se distribuyen los clientes que queremos clasificar. Si formamos un gráfico con los dos primeros componentes (representan casi el 78% de la variabilidad de la muestra) podremos visualmente ver la formación de los grupos. La función `eqscplot` nos los permite hacer fácilmente. El gráfico nos muestra que existen 4 grupos, dos de ellos grandes con un valor reducido en el segundo componente, y dos, reducidos ambos, con un valor elevador en el segundo componente y medio en el primero.

```
eqscplot(bases.puntos, xlab="primer componente", ylab="segundo componente")
```



Si queremos guardar estos tres componentes para futuros análisis, podemos asignarlos a tres nuevas variables de la hoja de datos.

```
eqscplot(bases.puntos, xlab="primer componente", ylab="segundo componente")
```



Si queremos guardar estos tres componentes para futuros análisis, podemos asignarlos a tres nuevas variables de la hoja de datos.

Repetimos el proceso de clasificación con las nuevas bases de segmentacion

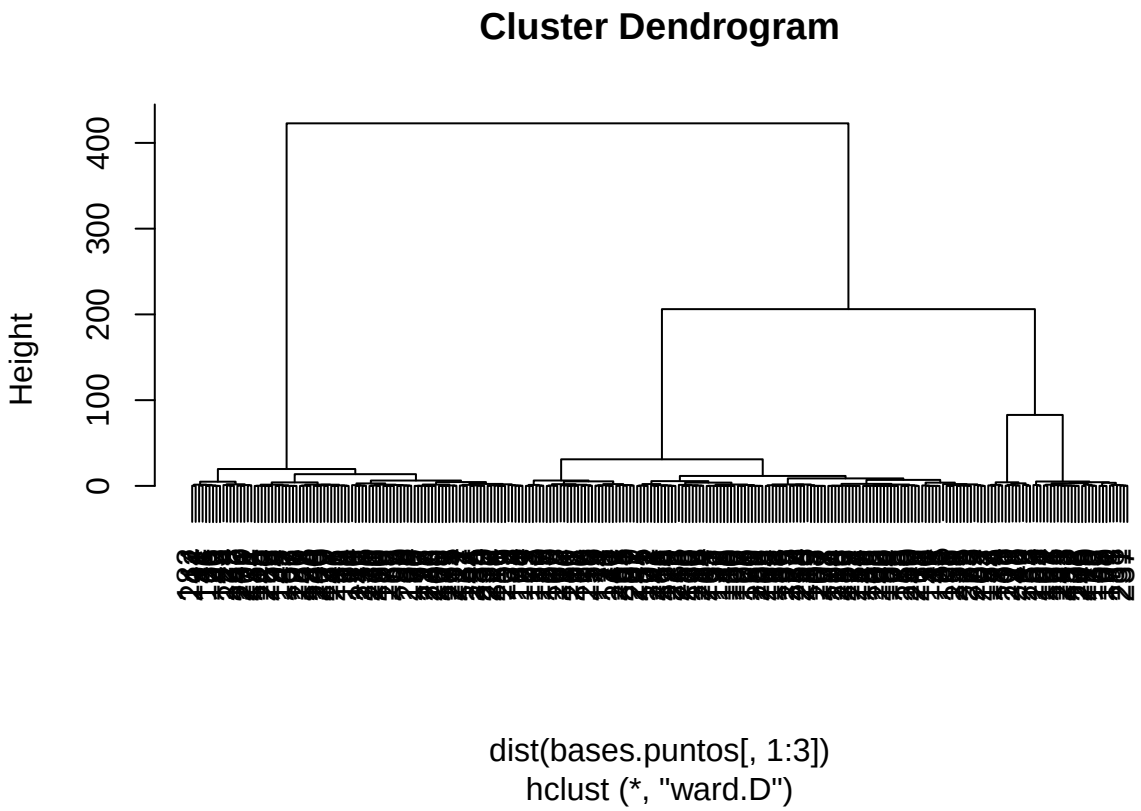
De nuevo vamos a utilizar los algoritmos clásicos implementados en los programas de análisis de datos más habituales: el jerárquico, `hclust`, y el de partición `kmeans`.

Obtendremos un nuevo objeto de la clase 'hclust, bases.puntos.hclust, cuyo proceso de aglomeración podemos observar gráficamente gracias a la función `plot` que precisa como argumento el nombre del objeto que acabamos de crear, bases.puntos.hclust. Las distancias verticales nos dan una medida visual de la heterogeneneidad que existe en los grupos. Un examen visual nos indica que pueden haber 3 o 4 grupos, dos grandes y dos pequeños que podrían agruparse en uno solo, aunque la varianza dentro de los grupos sería bastante mayor que la correspondiente a una agrupación con 4.

```
bases.puntos.hclust<-hclust(dist(bases.puntos[,1:3]), method="ward")
```

```
## The "ward" method has been renamed to "ward.D"; note new "ward.D2"
```

```
plot(bases.puntos.hclust)
```



Si queremos saber a qué grupos pertenecen cada uno de los clientes podemos utilizar la función cutree, nos asigna cada caso a un grupo.

```
cutree(bases.puntos.hclust, 4)
```

```
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##   1   2   1   1   1   1   3   3   4   1   3   1   3   3   3   1   3   1
##  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
##   3   3   1   1   3   2   3   1   3   2   3   1   1   4   1   1   1   3
```

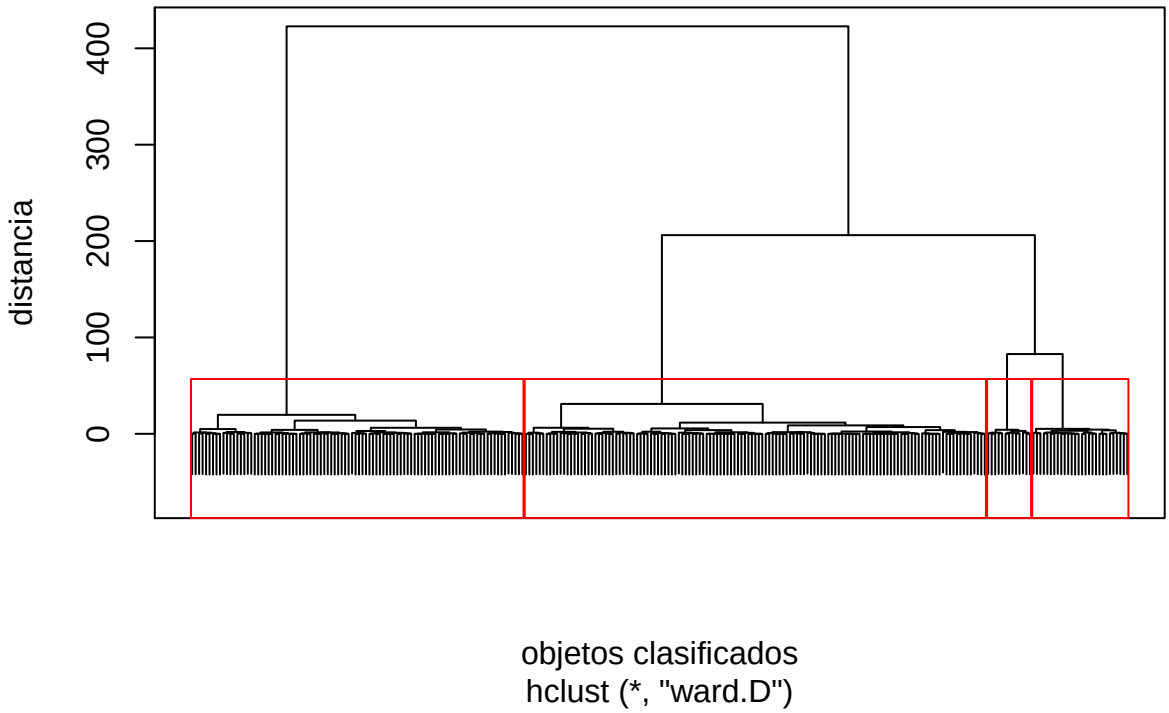
```
##  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
##   3   2   2   2   1   1   1   1   3   3   1   3   3   3   3   4   3   3
##  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
##   3   3   3   3   3   3   3   1   3   4   3   3   1   3   4   1   3   3
##  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
##   3   3   1   4   1   3   3   1   4   3   3   1   3   3   3   4   1   1
##  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108
##   1   1   3   1   1   3   3   3   3   2   4   3   3   4   3   3   3   3
## 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126
##   1   3   1   3   3   3   1   3   3   3   3   3   3   4   3   1   1   1
## 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144
##   2   3   3   3   4   3   4   1   3   1   1   4   3   1   1   3   1   1
## 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162
##   1   1   1   3   3   3   3   4   2   3   4   4   1   3   1   3   3   3
## 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
##   3   1   1   1   3   4   3   3   4   3   3   2   1   1   1   1   1   3
## 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198
##   1   1   4   3   1   3   1   3   3   3   3   3   3   1   3   1   1   3
## 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216
##   3   1   1   3   3   3   3   4   3   1   1   4   2   1   3   3   3   3
## 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234
##   3   3   1   1   1   3   1   1   1   1   3   4   1   1   4   1   2   1
## 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252
##   3   1   4   3   1   1   3   2   4   1   3   1   3   1   4   3   3   3
## 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270
##   4   3   3   3   1   3   3   1   3   3   3   1   3   1   1   3   3   1
```

Para tener que escribir menos podemos almacenar los tres componentes en una nueva tabla de datos, bases.puntos3:

```
bases.puntos3<-bases.puntos[,1:3]
head(bases.puntos3)
```

```
##   Comp.1 Comp.2  Comp.3
## 1  2.971  0.297  0.1311
## 2  0.362 -1.329  3.1436
## 3  2.999  0.584 -0.0365
## 4  2.529  0.371  0.1078
## 5  3.244  1.286  0.1529
## 6  3.006  1.158  0.2285
```

```
plot(bases.puntos.hclust,
     main="Dendrograma PCA",
     labels=FALSE,
     ylab="distancia",
     xlab="objetos clasificados",
     frame.plot=TRUE
     )
rect.hclust(bases.puntos.hclust, k=4, border=2)
```



¿Cuántos grupos formar? Vemos que al utilizar los componentes principales el dendograma nos sugiere que es mayor la heterogeneidad entre los dos segmentos más pequeños (la longitud de la línea que los une es mayor). Por ello tentativamente formaremos 4 grupos.

Ahora realizaremos la clasificación con el algoritmo `kmeans`. El resultado de este algoritmo es muy sensible a los centros iniciales a partir de los cuales inicia la partición. Veámoslo, en este caso es así.

Primero realizaremos la clasificación con unos centros obtenidos aleatoriamente y el resultado lo asignaremos al objeto `bases.puntos.kmeans`. La asignación de cada individuo a las particiones obtenidas se encuentra almacenado en el atributo `cluster` del objeto `bases.puntos.kmeans`, y podemos acceder a ella de la forma habitual, `bases.puntos.kmeans$cluster`.

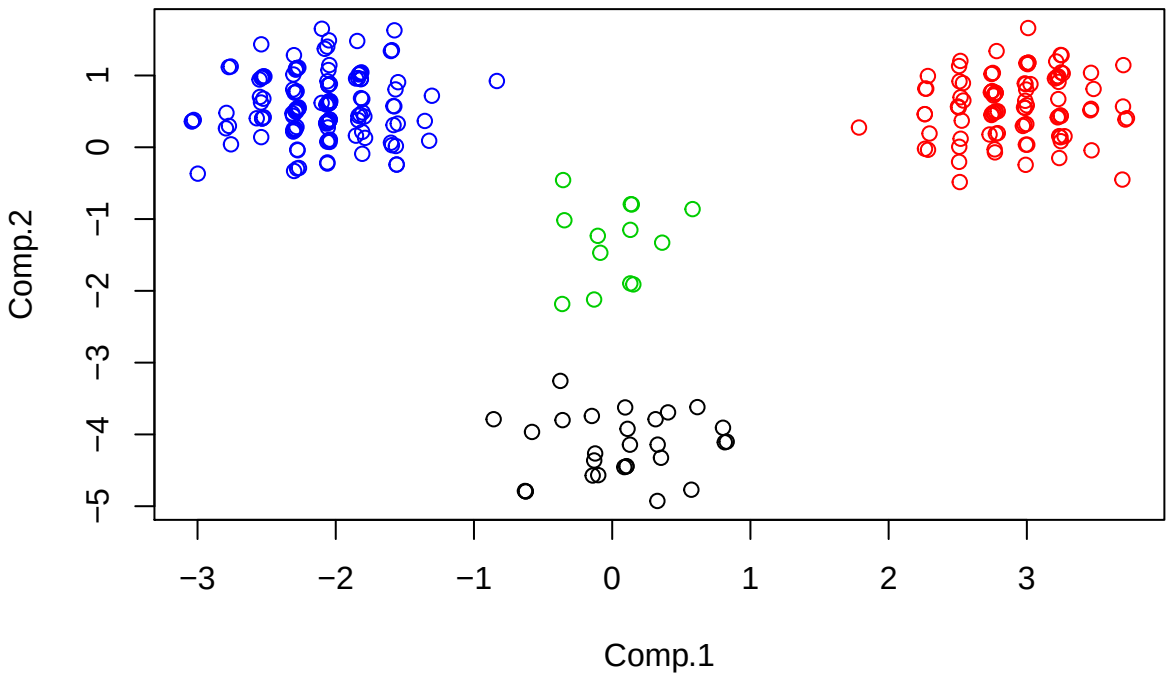
```
bases.puntos.kmeans<-kmeans(bases.puntos3, 4)
bases.puntos.kmeans$cluster
```

```
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##   2   3   2   2   2   2   4   4   1   2   4   2   4   4   4   2   4   2
##  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
##   4   4   2   2   4   3   4   2   4   3   4   2   2   1   2   2   2   4
##  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
##   4   3   3   3   2   2   2   2   4   4   2   4   4   4   4   1   4   4
##  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
##   4   4   4   4   4   4   4   2   4   1   4   4   2   4   1   2   4   4
##  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
##   4   4   2   1   2   4   4   2   1   4   4   2   4   4   4   1   2   2
##  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108
##   2   2   4   2   2   4   4   4   4   3   1   4   4   1   4   4   4   4
## 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126
##   2   4   2   4   4   4   2   4   4   4   4   4   4   1   4   2   2   2
```

```
## 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144
##   3   4   4   4   1   4   1   2   4   2   2   1   4   2   2   4   2   2
## 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162
##   2   2   2   4   4   4   4   1   3   4   1   1   2   4   2   4   4   4
## 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
##   4   2   2   2   4   1   4   4   1   4   4   3   2   2   2   2   2   4
## 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198
##   2   2   1   4   2   4   2   4   4   4   4   4   4   2   4   2   2   4
## 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216
##   4   2   2   4   4   4   4   1   4   2   2   1   3   2   4   4   4   4
## 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234
##   4   4   2   2   2   4   2   2   2   2   4   1   2   2   1   2   3   2
## 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252
##   4   2   1   4   2   2   4   3   1   2   4   2   4   2   1   4   4   4
## 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270
##   1   4   4   4   2   4   4   2   4   4   4   2   4   2   2   4   4   2
```

Representación gráfica del resultado con el algoritmo kmeans con centros aleatorios, incluidos los centros finales de los 4 grupos o particiones realizadas. Utilizaremos al función plot y como argumento el nombre del objeto

```
plot(bases.puntos[,1:2], col=bases.puntos.kmeans$cluster)
```



Además, podemos añadir al gráfico los centros de cada segmento

```
#points(bases.puntos.kmeans$centers, col=bases.puntos.kmeans$cluster, pch=8)
```

¿Podemos mejorar la partición utilizando los centros iniciales proporcionados por el algoritmo de aglomeración jerárquica?

Veamos, ahora, el resultado de la partición si les damos como centros los obtenidos en la clasificación con el método de Ward. Para su cálculo utilizamos el procedimiento propuesto por Venables y Ripley (1999: 338).

```
centros.puntos4<-tapply(bases.puntos3,
                        list(rep(cutree(bases.puntos.hclust, 4),
                                 ncol(bases.puntos3)),
                             col(bases.puntos3)),
                        mean)
centros.puntos4
```

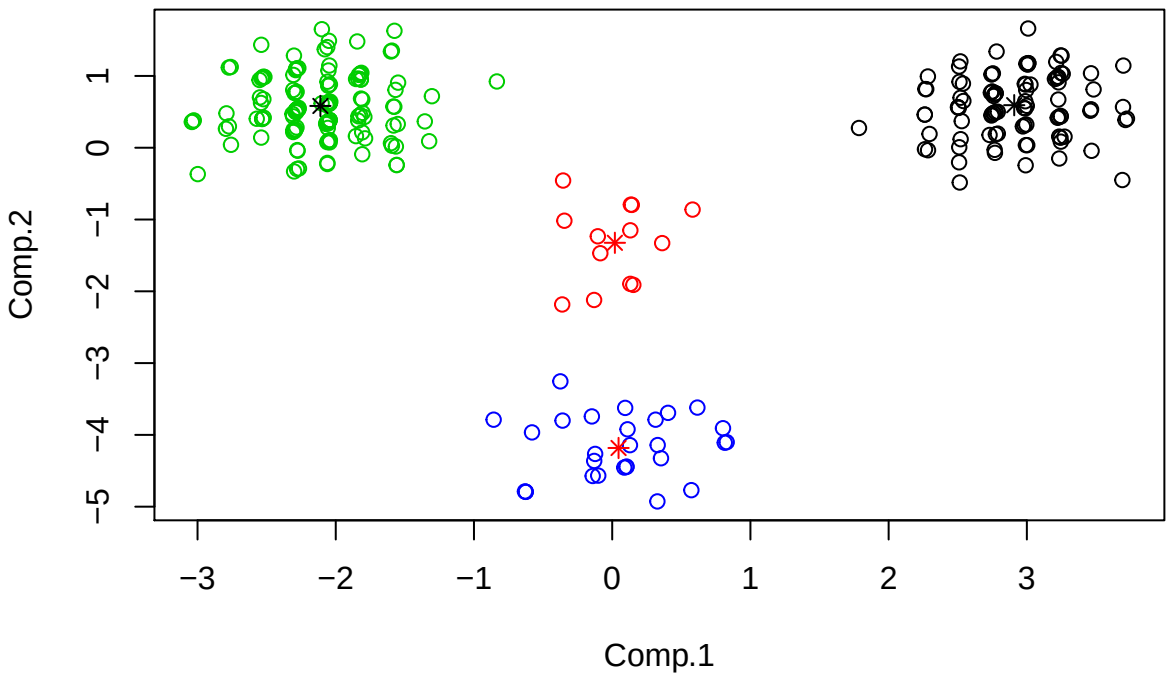
```
##         1      2      3
## 1  2.9086  0.593 -0.128
## 2  0.0193 -1.324  4.087
## 3 -2.1110  0.582 -0.145
## 4  0.0460 -4.183 -0.772
```

Volvemos a clasificar los individuos con el procedimiento kmeans y asignamos el resultado al objeto fikmeanspca2. Fíjate, ahora, que no es necesario el argumento correspondiente al número de grupos a formar, pues este dato lo obtiene del número de filas de la matriz de medias, el objeto identificado con el nombre de centros.

```
bases.puntos.kmeans4.c<-kmeans(bases.puntos3, centros.puntos4)
```

Volvamos a representar gráficamente el resultado de la clasificación contenida en el objeto fikmeanspca3.

```
plot(bases.puntos[,1:2], col=bases.puntos.kmeans4.c$cluster)
points(bases.puntos.kmeans4.c$centers,col=1:2, pch=8)
```



Procedimientos más robustos: el paquete `cluster`

Ahora debemos cargar en la memoria del ordenador la bibliteca de los paquetes que necesitamos para realizar el análisis de segmentación. Concretamente, necesitamos la bilbioteca `cluster`. Las otras funciones utilizadas

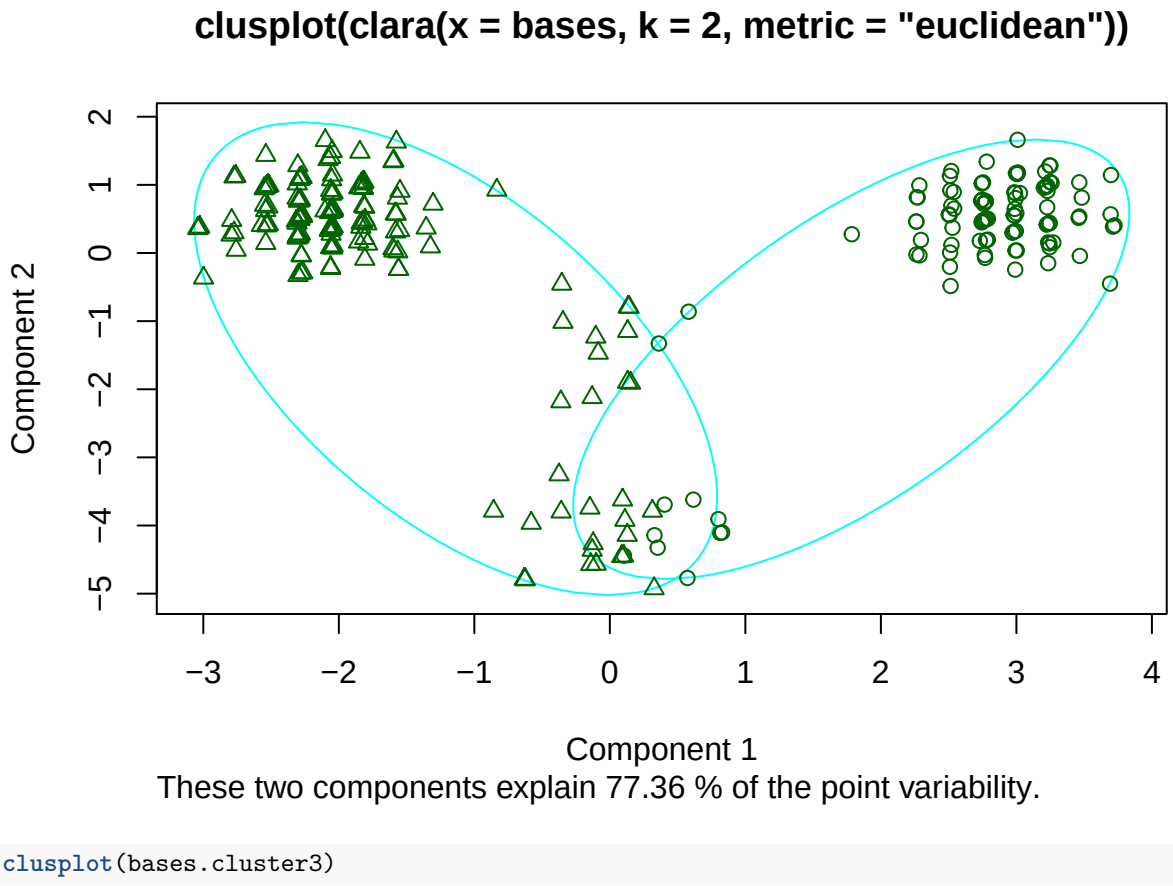
en la segmentación están disponibles en el programa base de **R**. Para cargar los paquetes utilizamos la función `library()` y como argumento el nombre del paquete. El paque `cluster` contiene un conjunto de algoritmos de clasificación robustos propuestos pro Kauffman y Rouseeuw (1990). Los podemos agrupar en tres tipos: algoritmos para partir un conjunto de observaciones (`pam`, `clara`, y `fanny`), algoritmos de aglomeración jerárquica (`agnes`) y, finalmente, algoritmos para la división jerárquica de un conjunto de aobseraciones (`diana` y `mona`).

```
library(cluster)
```

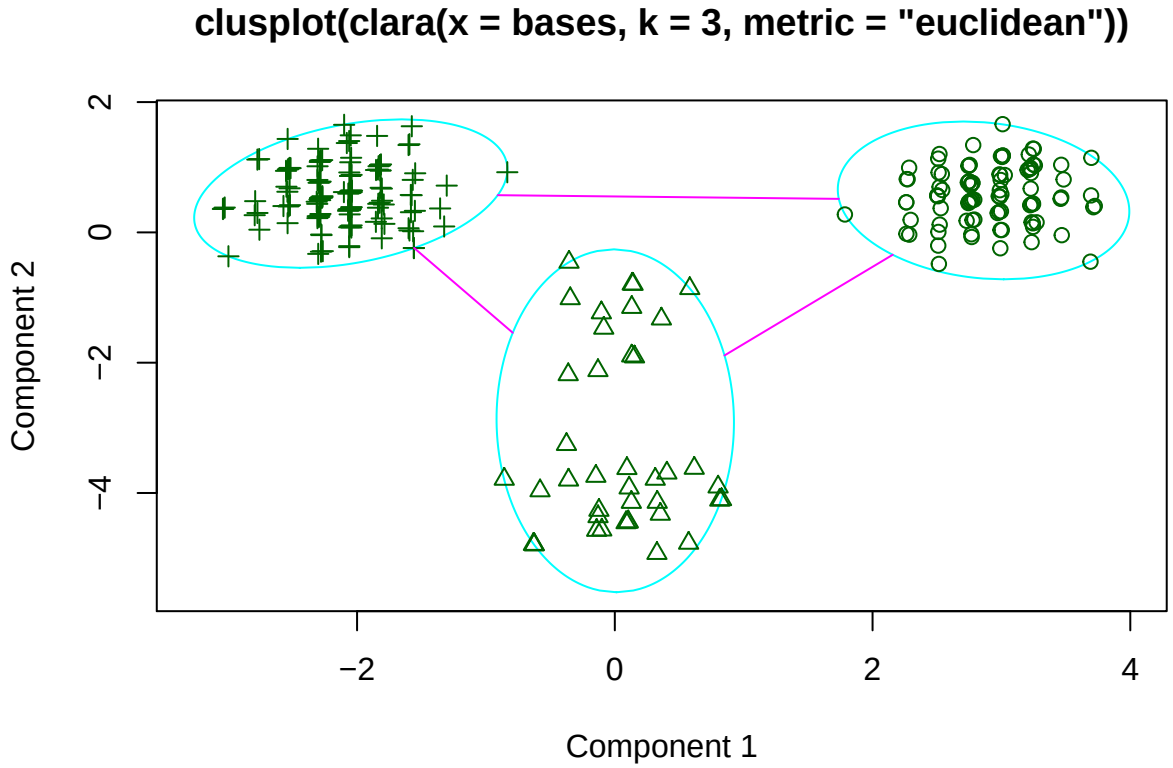
Partición con las variables originales

Realizamos la primera segmentación con los datos originales. Veamos ahora los resultados del programa `cluster`. Para ello empleamos el procedimiento de partición denominado `clara`: cuyo significado es *clustering large applications*. La ventaja del paquete de procedimiento resarrollado por Kaufman y Rouseeuw (1990) reside en que es menos sensible que *Kmeans* cuando los segmentos no tienen una forma esférica y, además, nos proporciona un índice de la calidad de la partición obtenida. Veámoslo:

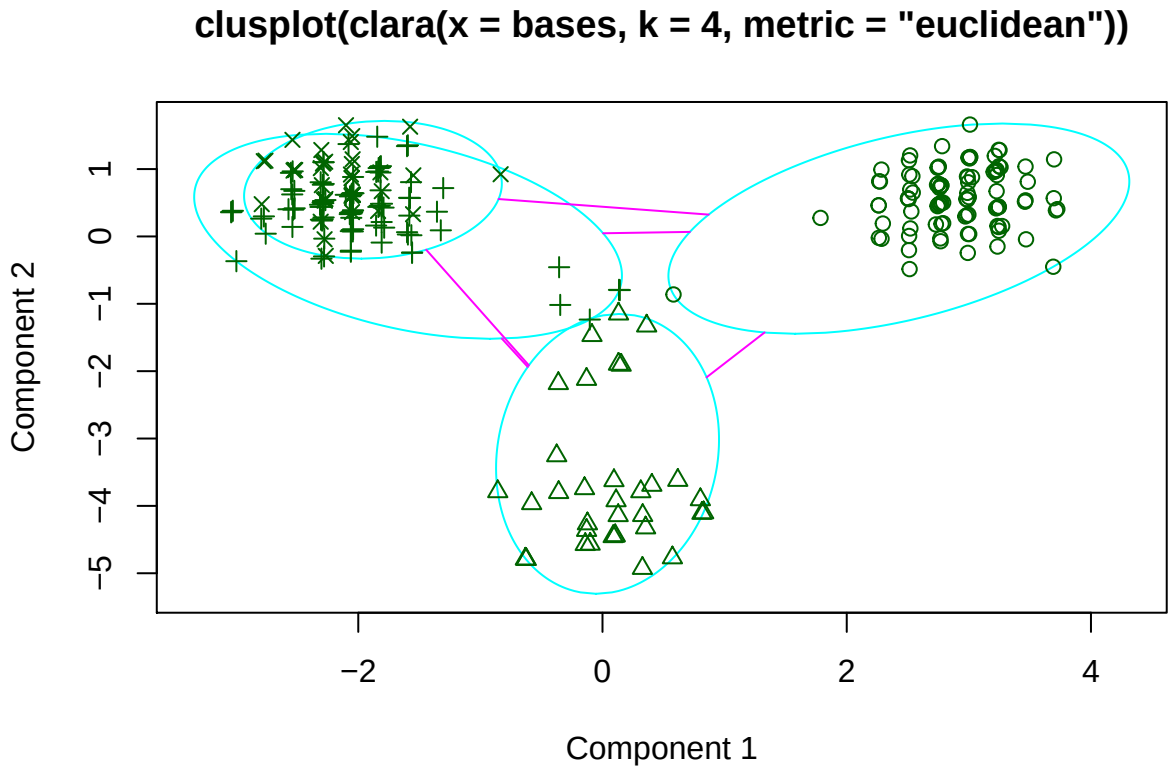
```
bases.cluster<-clara(bases, 2, metric="euclidean")
bases.cluster3<-clara(bases, 3, metric="euclidean")
bases.cluster4<-clara(bases, 4, metric="euclidean")
bases.cluster5<-clara(bases, 5, metric="euclidean")
clusplot(bases.cluster)
```



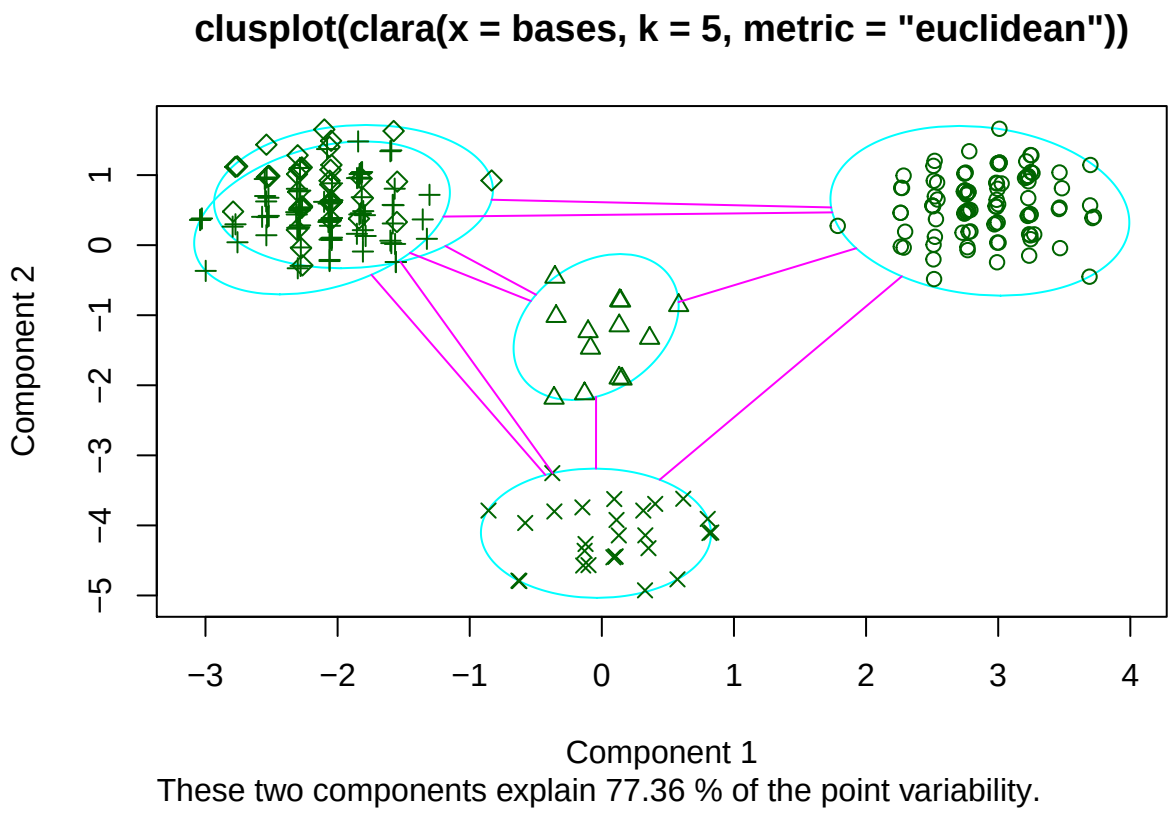

```
clusplot(bases.cluster3)
```




```
clusplot(bases.cluster4)
```



```
clusplot(bases.cluster5)
```



Repetimos la operación con 3, 4 y 5 particiones obteniendo los siguientes índices de calidad de: 0,53, 0,63, y 0,42, respectivamente para 2, 3, 4 y 5 particiones. Los resultados nos indican que la mejor clasificación se obtiene con 3 particiones como se observa también la representación gráfica en un espacio reducido mostrada por la función `clusplot`. `Clara` con 3 clusters sólo tiene un error al fusionar los clusters 3 y 4.

```
names(bases.cluster)
```

```
##  [1] "sample"     "medoids"    "i.med"      "clustering" "objective"
##  [6] "clusinfo"   "diss"       "call"       "silinfo"    "data"
```

```
bases.cluster$silinfo$avg.width
```

```
## [1] 0.533
```

```
bases.cluster3$silinfo$avg.width
```

```
## [1] 0.635
```

```
bases.cluster4$silinfo$avg.width
```

```
## [1] 0.419
```

```
bases.cluster5$silinfo$avg.width
```

```
## [1] 0.418
```

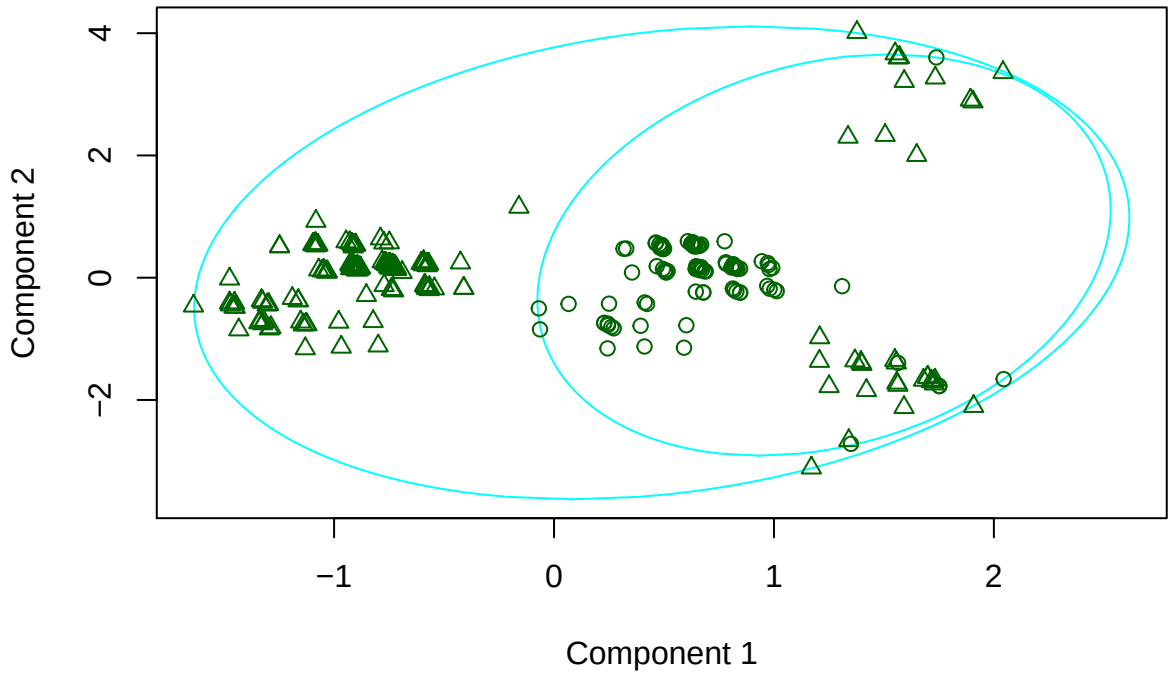
Si le padimos los atributos del objeto de la clase cluster que acabamos de obtener (con la función `names(bases.cluster)`) nos indica cómo podemos acceder de forma individual a cada uno de los apartador que contiene el resumen, los argumentos disponibles son los siguientes: *sample*, *medoids*, *clustering*, *objective*, *clusinfo*, *silinfo*, *diss* y *data*. Para acceder directamente al resultado de la clasificación podemos utilizar la misma estructura que utilizamos en una tabla de datos, el nombre de objeto, `bases.cluster`, seguido del nombre del argumento al cual deseamos acceder, `clustering`, ambos conectados con el símbolo del dólar, *bases.cluster$clustering*.

Si queremos obtener ayuda sobre alguna función o biblioteca de programas incorporada a nuestra lenguaje **R**, podemos utilizar la función `help.start()` y sin argumento, paréntesis vacíos.

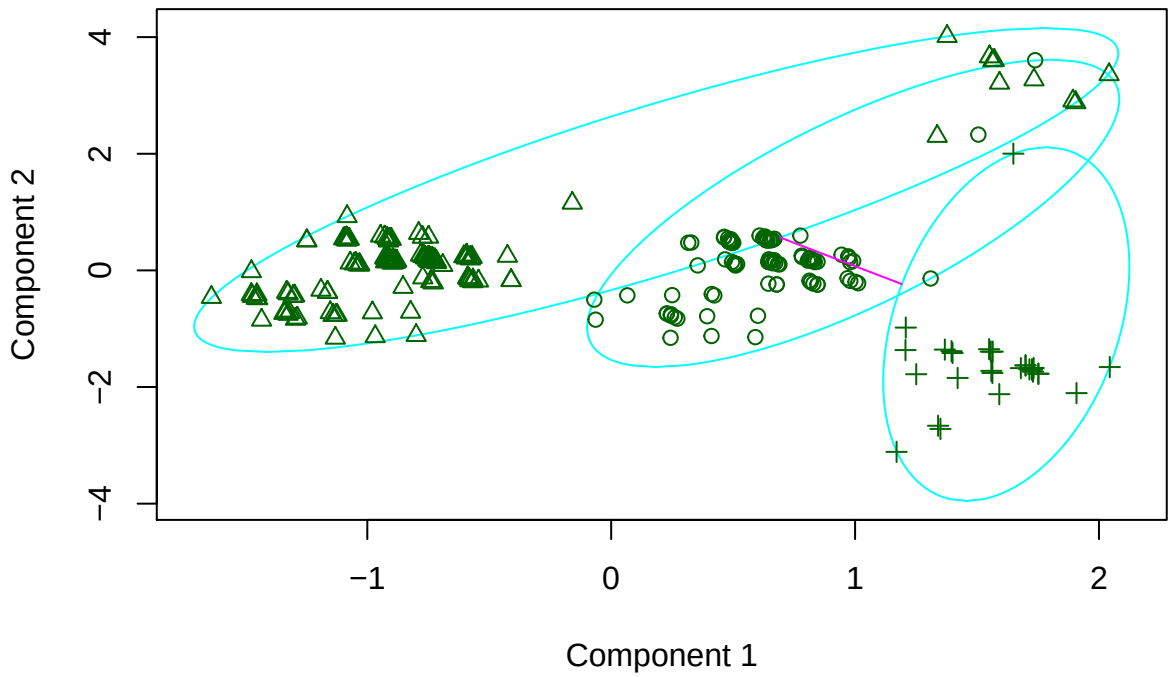
¿Podemos confiar en el resultado obtenido? ¿Existe correlación entre las variables?

Veamos ahora el resultado de la clasificación utilizando el algoritmo `clara` con los tres componente principales estimados. Creamos un nuevo objeto, bases.puntos.cluste, con 2, 3, 4 y 5 grupos, de forma sucesiva. El índice de calidad obtenido es de: 0,69, 0,77, 0,81 y 0,68; con lo que los resultados sugieren que cuatro grupos es la mejor solución. Evidentemente, la representación gráfica de los grupos obtenidos (función `clusplot(bases.puntos.cluster)`) únicamente en la partición en cuatro grupos éstos se muestran totalmente separados sin solapamiento alguno. En cambio, en el resto de las soluciones obtenidas algunos grupo se encuentran solapados, reduciendo el índice de calidad de la estructura obtenida.

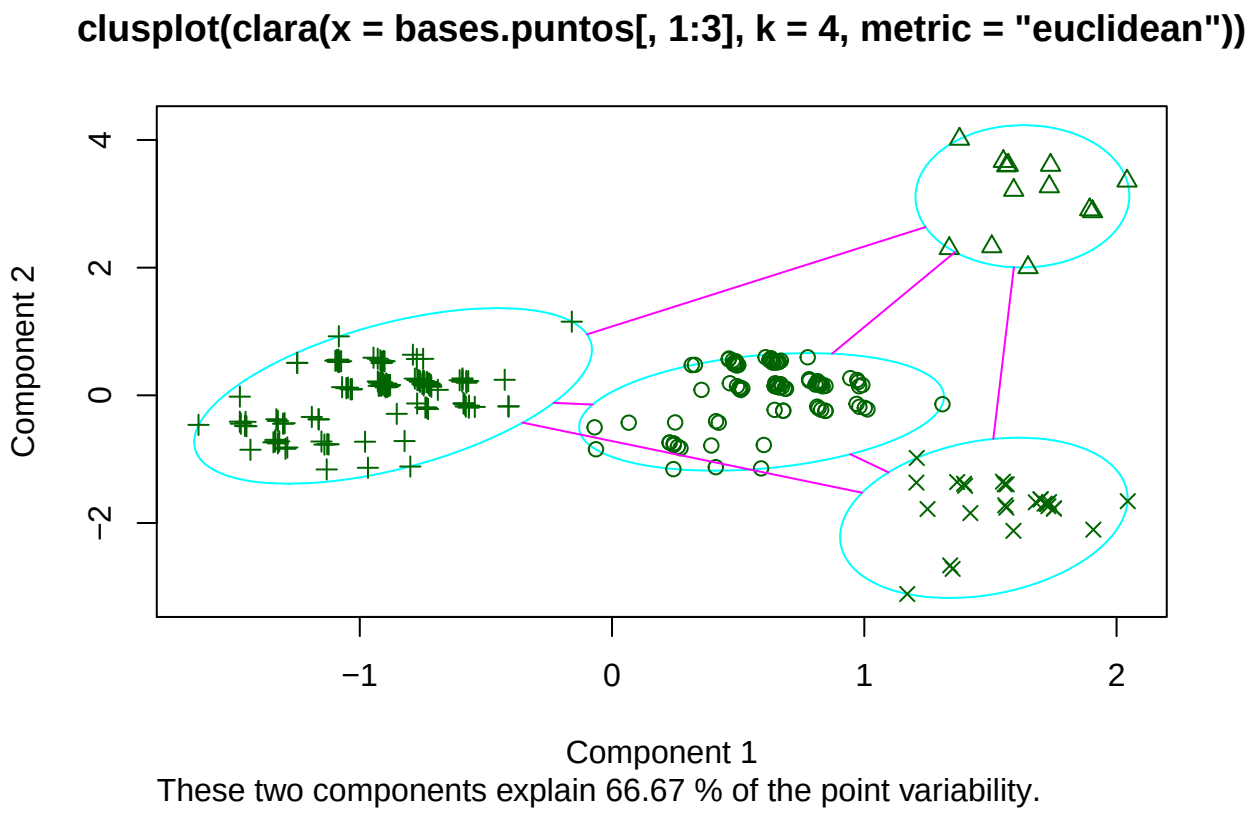
```
#con componentes principales

bases.puntos.cluster<-clara(bases.puntos[,1:3],2,metric="euclidean")
bases.puntos.cluster3<-clara(bases.puntos[,1:3],3,metric="euclidean")
bases.puntos.cluster4<-clara(bases.puntos[,1:3],4,metric="euclidean")
bases.puntos.cluster5<-clara(bases.puntos[,1:3],5,metric="euclidean")

clusplot(bases.puntos.cluster)
```



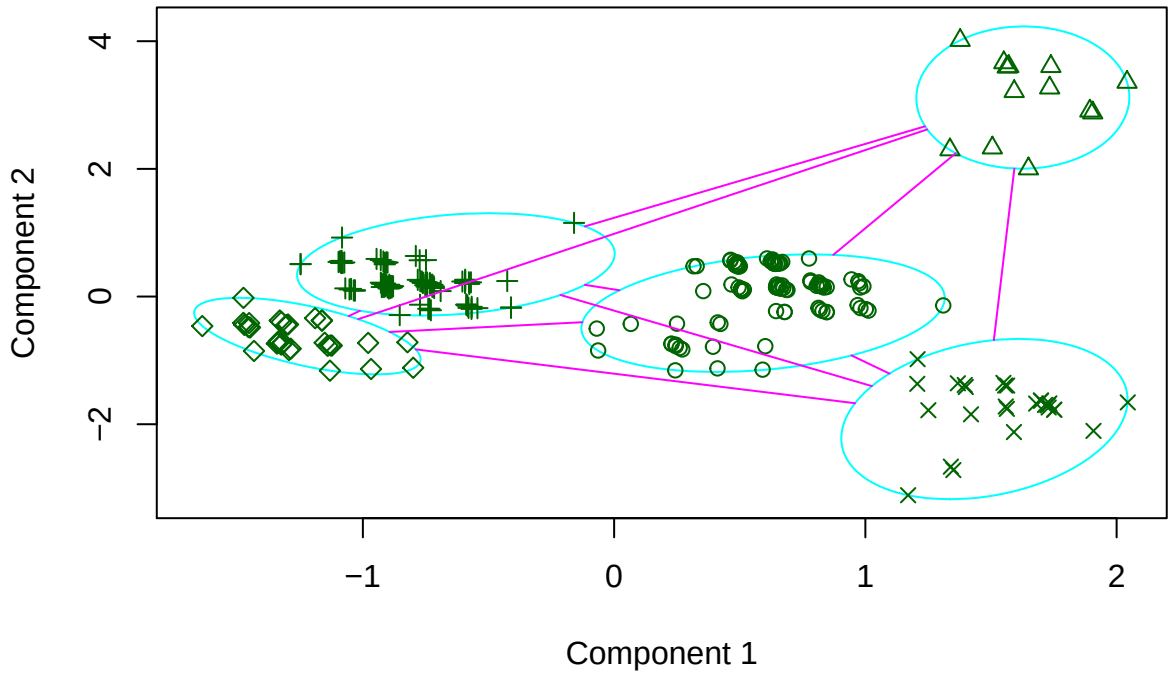

```
clusplot(bases.puntos.cluster3)
```



```
clusplot(bases.puntos.cluster4)
```




```
clusplot(bases.puntos.cluster5)
```




```
bases.puntos.cluster$silinfo$avg.width
```

```
## [1] 0.58
```

```
bases.puntos.cluster3$silinfo$avg.width
```

```
## [1] 0.791
```

```
bases.puntos.cluster4$silinfo$avg.width
```

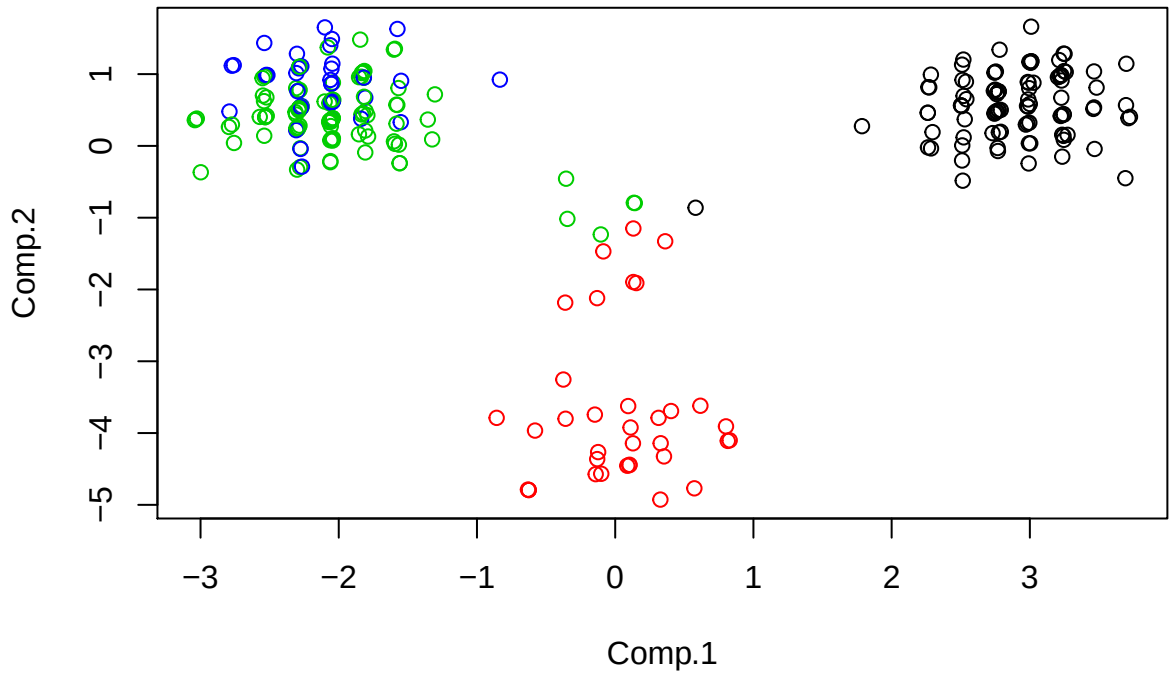
```
## [1] 0.796
```

```
bases.puntos.cluster5$silinfo$avg.width
```

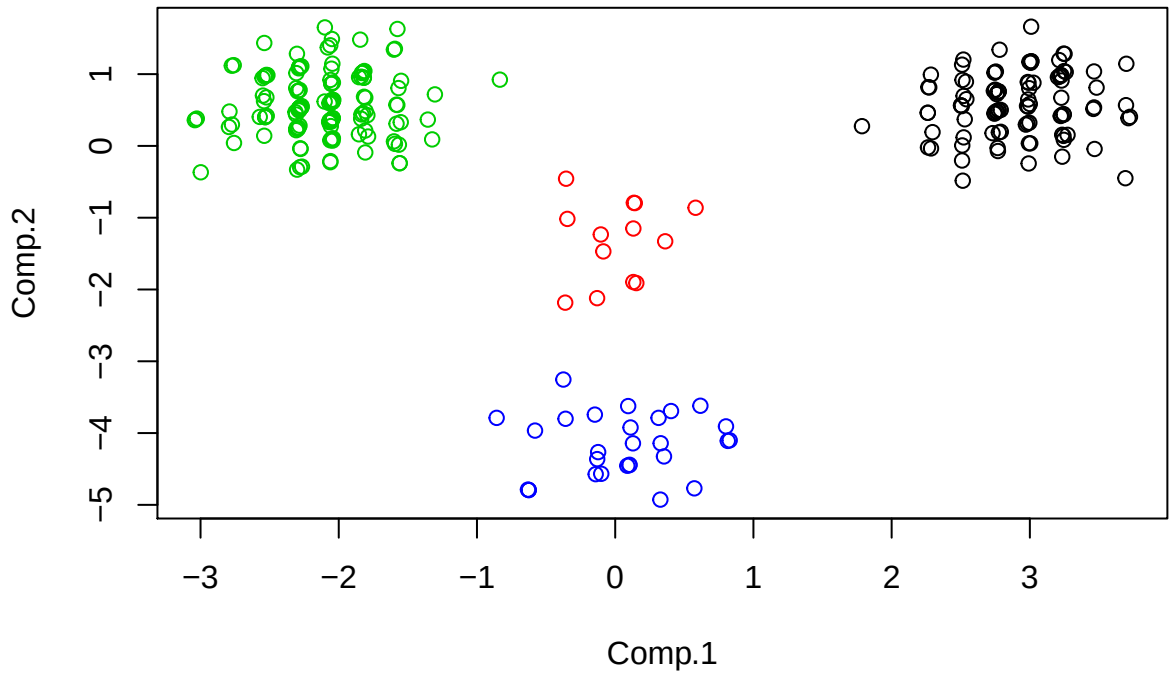
```
## [1] 0.673
```

Guardamos el resultado de la clasificación con 4 segmentos y realizamos una representación gráfica en el espacio de los primeros componentes principales del resultado obtenido con las variabels originales y después con los componentes principales (comparar con el resultado obtenido con 3 segmentos.

```
kchospital$clusterpca<-bases.puntos.cluster4$clustering
plot(bases.puntos[,1:2], col=bases.cluster4$clustering)
```




```
plot(bases.puntos[,1:2], col=bases.puntos.cluster4$clustering)
```



¿Hay diferencias? Veamos, ahora, el resultado de la clasificación realizada por Kmeans con las variables originales (sin sustituirlas por los componentes principales) y el obtenido con clara. ¿Existen diferencias?

Si queremos exportar los resultados, podemos utilizar las siguientes funciones:

```
#write.table(kchospital, "kcclas.dat", quote = FALSE, sep = ",", row.names=F)
```

Identificación de los consumidores asignados a cada segmento

Las variables despriptoras suelen ser variables nomnales con diferentes niveles. Si la base de datos no contiene las etiquetas de los niveles, primero deberemos preparar la base de datos como sigue, por ejemplo.

```
names(kchospital)
```

```
##  [1] "CASO"       "KCMATER"    "HB"         "HC"         "HD"
##  [6] "HE"         "HF"         "HG"         "HH"         "HI"
## [11] "FIREP"      "FICCUID"    "FIDIST"     "FICDOC"     "FICAMIG"
## [16] "FIPUBLI"    "FIAMAB"     "FIIMOD"     "FIEMOD"     "FISEG"
## [21] "FAMIL"      "ELECHOSP"   "SERPERF"    "ALEALTAD"   "AMSERV"
## [26] "ASCOM"      "ACALIDAD"   "ACOMOD"     "VKC"        "VB"
## [31] "VC"         "VD"         "VE"         "VF"         "VG"
## [36] "VH"         "VI"         "INTERES"    "COMPROM"    "ESFUER"
## [41] "MOTIV"      "DIFICULT"   "SEXO"       "ESTADOMA"   "EDAD"
## [46] "NIVELEDU"   "INGRESOS"   "clusterpca"
```

```
library(plyr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
##
## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
##
## The following object is masked from 'package:MASS':
##
##     select
##
## The following object is masked from 'package:stats':
##
##     filter
##
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
descriptores<-kchospital[,c(43:47)]
head(descriptores)
```

```
##   SEXO ESTADOMA EDAD NIVELEDU INGRESOS
## 1    0        0  100        1        1
## 2    1       10    1        0        0
## 3    0        0  100        1       10
## 4    0        0    0        1        1
## 5    0        0  100        1       10
## 6    0        0  100        1        1
```

```r
class(descriptores$SEXO)
```

```
## [1] "integer"
```

```r
descriptores$SEXO<-as.factor(descriptores$SEXO)
summary(descriptores$SEXO)
```

```
##   0   1
## 244  26
```

```r
class(descriptores$ESTADOMA)
```

```
## [1] "integer"
```

```r
descriptores$ESTADOMA<-as.factor(descriptores$ESTADOMA)
summary(descriptores$ESTADOMA)
```

```
##   0   1  10
## 231  28  11
```

```r
class(descriptores$EDAD)
```

```
## [1] "integer"
```

```r
descriptores$EDAD<-as.factor(descriptores$EDAD)
summary(descriptores$EDAD)
```

```
##   0   1  10 100
##  66  35  23 146
```

```r
class(descriptores$NIVELEDU)
```

```
## [1] "integer"
```

```r
descriptores$NIVELEDU<-as.factor(descriptores$NIVELEDU)
summary(descriptores$NIVELEDU)
```

```
##   0   1  10 100
##  45  86  96  43
```

```r
class(descriptores$INGRESOS)
```

```
## [1] "integer"
```

```
descriptores$INGRESOS<-as.factor(descriptores$INGRESOS)
summary(descriptores$INGRESOS)
```

```
##     0     1    10   100  1000 10000
##     9    39    60   105    30    27
```

```
descriptores$SEXO <- revalue(descriptores$SEXO, c("0" ="Mujer", "1" = "Hombre"))
descriptores$ESTADOMA <- revalue(descriptores$ESTADOMA,
                                 c("0" ="Casado",
                                   "10"="Soltero",
                                   "1" = "Otro")
                                 )
descriptores$EDAD <- revalue(descriptores$EDAD,
                             c("0" ="<=25 años",
                               "1"="55+",
                               "10"="41-55",
                               "100" = "26-40")
                             )
descriptores$NIVELEDU <- revalue(descriptores$NIVELEDU,
                                 c("0" ="ESO",
                                   "1"="Graduado Univ.",
                                   "10"="Bachillerato +",
                                   "100" = "Bachillerato")
                                 )
descriptores$INGRESOS <- revalue(descriptores$INGRESOS,
                                 c("0" ="<=$100000",
                                   "1"="$60000+",
                                   "10"="$40001-60000",
                                   "100" = "$30001-40000",
                                   "1000"="$20001-30000",
                                   "10000"= "$10001-20000")
                                 )
```

Despues elaboramos las tablas que nos describen los segmentos

```
options(digits=3)
descriptores.tabla<- list(
t(prop.table(table(bases.puntos.cluster4$clustering, descriptores$SEXO))),
t(prop.table(table(bases.puntos.cluster4$clustering, descriptores$ESTADOMA))),
t(prop.table(table(bases.puntos.cluster4$clustering, descriptores$EDAD))),
t(prop.table(table(bases.puntos.cluster4$clustering, descriptores$NIVELEDU))),
t(prop.table(table(bases.puntos.cluster4$clustering, descriptores$INGRESOS)))
)
do.call(rbind, descriptores.tabla)
```

```
##                    1       2       3       4
## Mujer          0.3556 0.02593 0.49259 0.02963
## Hombre         0.0000 0.02222 0.00000 0.07407
## Casado         0.3556 0.00741 0.49259 0.00000
## Otro           0.0000 0.00000 0.00000 0.10370
## Soltero        0.0000 0.04074 0.00000 0.00000
## <=25 años      0.2370 0.00000 0.00741 0.00000
```

```
## 55+            0.0000 0.03333 0.01852 0.07778
## 41-55          0.0000 0.00370 0.07778 0.00370
## 26-40          0.1185 0.01111 0.38889 0.02222
## ESO            0.0000 0.03333 0.05556 0.07778
## Graduado Univ. 0.3074 0.00000 0.01111 0.00000
## Bachillerato + 0.0481 0.01481 0.27407 0.01852
## Bachillerato   0.0000 0.00000 0.15185 0.00741
## <=$100000      0.0000 0.01111 0.00000 0.02222
## $60000+        0.1444 0.00000 0.00000 0.00000
## $40001-60000   0.1889 0.00741 0.01111 0.01481
## $30001-40000   0.0222 0.00370 0.34815 0.01481
## $20001-30000   0.0000 0.01481 0.06296 0.03333
## $10001-20000   0.0000 0.01111 0.07037 0.01852
```

Informe

Problema de marketing: ¿Debería el hopital comercializar un servici especializado en maternidad para incrementar la tasa de ocupación?

En términos comerciales: Determinar la facilidad con la que los consumidores identifican los hospitales, sus preferencias e intenciones de compra para el servicio de maternidad.

Componentes específicos del problema comercial

1. ¿Qué criterios utilizan los consumidores cuando seleccionan un hospital?
2. ¿Cuál es la posicón relativa del hospital Kc en comparación con la competencia?
3. ¿Qué competidores tienen el mejor servicio de maternidad según sus percepciones?
4. ¿Cuál es el mercado potencial del servicio de maternidad? ¿Quienes son los pacientes en términos demográficos y psicológicos?
5. ¿Y el mercado potencial del hospital KC?
6. ¿Podemos incrementar la tasa de reconocimienot del hospital con la introducción del servicio de maternidad u otro servicio especializado?

```
#library(DiagrammeR)
#diagrama.causalidad <-DiagrammeR("
#  graph BT;
#    Cuidado.especializado --> Ventaja.competitiva;
#    Ventaja.competitiva --> Reconocimiento;
#    Reconocimiento -->Preferencia;
#    Preferencia --> Tasa.de.ocupacion;
#    Tasa.de.ocupacion--> Beneficio;
#")
#diagrama.causalidad
```

Componentes principales (Q2A-J) Reducir componentes principales y ver si existe heterogeneidad respecto a los últimos hospitales visitados (Q1). Explicar solución PCA

PCA 1 “fundamentos del cuidado”. En un extremo variables relacionadas con la reputación, y en el otro, los aspectos sociales del cuidado.

PCA 2 “aspectos comerciales”, correlaciones positcias para “advertising,” y “acceptance of maternity insurance.”

PCA 3 “proximidad” dimensión correlacionada con “distance from home.”

Cluster: Una solución con 4 segmentos:

Cluster 1 96 2 13 3 133 4 28

Según la dimensión de los segmentos se podrían agrupar en tres segmentos.

Describir segmentos según su importancia en los factores de elección del hospital.

Cluster 1 precoupado por los fundamentos del cuidado

Cluster 2 preocupado por la proximidad.

Cluster 3 este grupo se deja aconsejar por los conocidos y no le importan los aspectos comerciales.

cluster 4 is high on factor 2, meaning this group is responsive to business decisions related to maternity care.

Si tabulamos la clasificación en segmentos con los resultados de la primera pregunta, tenemos que una correspondencia clara entre el uso previo y la asginación a los clusters: cluster 1 corresponde a los hospitales B, C, F; cluster 2 a KC hospital; cluster 3 ...

EXECUTIVE SUMMARY

Los resultados muestran que la experiencia previa en hosptales es el criterio más importante para los consumidors cuando deciden a qué hospital ir. Otros criterios importantes son: la calidad del servicio, las promociones que incrementan el valor, y la proximidad geográfica. Por ello ofrecer un servicio de matenidad para conseguir que los consumidores repitan en su elección de hospital parece una bena idea.

Las características que el servicio debe son varias. Primero la atención personal fue un criterio importante para los consumidores con mayor nivel educativo e ingresos superiores. Segundo estos consumidores tambien perciben que deben tener libertad para escoger hospital, mientras que los consumidores con bajos ingresos y educación inferior creen que no tienen elección. Adicionalmente, los grupos más privilegiados socialmente tambien ven la publiciadad como algo positivio, indicando que la publicidad puede influir en su elección.

Además, los consumidores familiares con los servicios de matenidad creen que la comodidad es una caracteritica deseable un un servicio de matenidad. Por ello, la atención personal junto con los fundamentos del cuidado profesional y la comodidad deberían resaltar en la publicidad que intenta ganar clientes para el servicio de maternidad de KC. Esa publicia, junto con el servicio de maternidad, incrementaría el nivel de reconocimiento de la marca e incrementaria el valor de su imagen (en ambos está por detras de sus competidores)