Segmentación de mercados

Jordi López Sintas

9 de enero de 2015

# Fundamentos de la segmentación de mercados

La segmentación de los mercados (*market segmentation analysis*) se refiere al análisis de la estructura de la demanda de un mercado (los beneficios buscados por los consumidores, sus preferencias, etc.) y a la estimación de la intensidad de la demanda para un potencial producto que aspira a una determinada posición en la estructura estimada del mercado (Johnson 1971). La segmentación de los mercados es una decisión estratégica puesto que que la elección de las bases para realizarla determinará las características que suponemos importan a los consumidores.

Una selección incorrecta de las bases de segmentación supondrá partir de unas hipótesis erróneas acerca del comportamiento de los consumidors. Por ello, su reacción frente a nuestra decisiones comerciales será inferior, o contraria, a la prevista, con el consiguiente incremento en costes debido a la ineficacia de las decisi0nes comerciales realizadas. Los costes de manufactura y financieros asociados a la estrategia comercial también se resentirán como ocurrió en 1984 cuando Apple comercializó su primer Macintosh (REF). También debemos tener presente que los problemas de consumo de los individuos cambian a lo largo del tiempo, por lo que las bases de segmentación también deberán cambiar para poder describir y entender adecuadamente la estructura de las preferencias de los consumidores.

Suponiendo que las características que hemos medido de los consumidores, desciben adecuadamente la estructura de las preferencias de los consumidores en un determiando mercado, el proceso de segmentación está interesado en proporcionar respuestas a dos tipos de problemas de la dirección comercial: 1) cómo debería segmentarse un mercado en función de las propiedades estructurales de las preferencias de los consumidores, y 2) cómo debería asignarse los recursos comerciales entre los segmentos (Mahajan & Jain 1978).

Supongamos que una estrategia comercial está compuesta de los elementos tradicionales del mix comercial, calidad del producto ofrecido, precio, distribución y promoción, E(q,p,d,c). El problema de segmentación del mercado reside en entender su estructura: identificar qué es lo que los consumidores consideran importante, y medir cómo difieren en sus valoraciones de lo que es importante. Si conocemos esos dos grupos de datos podremos inferir la preferencia de los consumidors hacia los productos que la empresa ofrecerá al mercado (y hacia los de la competencia).

Nuestro problema de segmentación, entonces, empieza identificando los beneficios importantes para los consumidores, midiendo cómo difieren en la importancia asignada a esos beneficios y finaliza seleccionando un número *s* de segmentos, , y definiendo las estrategias diseñadas para cada segmento, y, simultáneamente, asginando los recursos comerciales a las estrategias (Winter 1979). Encontrar *s* segmentos de consumidores y analziar las correspondientes estrategias comerciales que proporcionan el máxico valor a la empresa. El número óptimo de estrategias, $e \\* <s$, y se obtendrá de la resolución del problema de maximización de benficios:

Con la estrategia sujeta a que los recursos aplicados no excedan a los dispones, .

Muchos investigadores han levantado dudas acerca de la validez de los resultados obtenidos en los análisis de segmentación de mercados. Concretamente han señalado:

1. diferentes algoritmos de clasificación producen diferentes resultados e incluso e con diferentes condiciones iniciales o distintos niveles de precisión pueden llegar a producir diferentes resultados (Hoek, Gendall and Esslemont 1996); Funkhouser 1983),
2. los criteris para seleccionar el número de segmentos difierente en sus conclusiones (Hoek, Gendall and Esslemont 1996; Wedel & Kamakura 1998; Helsen and Green 1991; Weingessel, Dimitriadou, & Dolnical 1999).

Afortunadamente en la actualidad disponemos, además del tradicinal algoritmo de partición convexa, *kmeans* (Hartigan & Wong 1978), de algoritmos robustos frente a la violación de la hipótesis de convexidad en la que se basa la solución proporcinada por *kmeans*, como son los aloritmos propuestos por Kauffman & Rouseeuw (1990), y de los procedmientos de clasificación basados en clases latentes, o clasificación basada en modelos estadísticos para segmentar poblaciones mixtas (Wedel & Kamakura 1998; Bartholomew & Knott 1999: Vermunt & Magidson 2000).

En la actualidad la tarea de seleccionar le número adecuado de segmentos que muestran la estructura de la demanda de los consumidores es una tarea relativamente más fácil gracias a la difusión que han tenido los algoritmos robustos de Kauffman y Rouseeuw (1990) y los basados en clases latentes, junto con las facilidades gráficas que ofrecen los computadores. Los procedimientos propuestos por Kauffman y Rouseeuw (1990) ya nos ofrecen un indicador de la calidad de la partición realizada del mercado, como veremos más adelante, pero también los métodos basados en clases latentes nos ofrecen diferentes estadísticos en los cuales basar nuestra decisión.

# Segmentación en dos etapas: El procedimiento estándar

## Fase exploratoria: la clasificación jerárquica

La clasificación jerárquica con el procedimiento de Ward minimiza la suma de las distancias euclidianas al cuadrado entre el individuo i y el centro del segmento al que se asigna,. El primer sumatorio calcula el error de la observación *i* en el segmento *k* para todas las variables, el segundo realiza el cálculo para todos los individuos en el segmento *k* y, finalmente, el último sumatorio calcula la suma del error para todos los segmentos (Wishart 1998).

**Funciona de la siguiente manera**

1. Computa la matriz de las distancias euclidianas al cuadrado para todo par de observaciones, *i*, *i’*, (*i*,*i’*)=.
2. Agrupa las dos observaciones o grupos *i* y *i’* más cercanos, es decir, cuya agrupación minimiza el incremento en el error, *E*. Inicialmente serán aquellos casos cuya distancia, , sea mínima.
3. Transforma la matriz inicial de distancias euclidianas al cuadrado en , la cual contendrá el error al cuadrado de la unión del nuevo segmento .
4. Repite los pasos 2 y 3 y cada vez forma un nuevo grupo con las observaciones o grupos cuya unión resulte en un incremento mínimo en el error .
5. Finaliza cuando todos los casos se hayan agrupado en un solo segmento

El procedimiento minimiza el cuadrado de la suma de las desviaciones entre los individuos y el centro del grupo al que se le ha asignado. Realiza un proceso de aglomeración con tantas fases o etapas como individuos ha de clasificar, *n*, de manera que una vez clasificados en su totalidad se minimiza una medida de la heterogeneidad, . En la etapa inicial existen tantos grupos como individuos debe agrupar. A partir de ese momento en cada etapa se formará un nuevo segmento agrupando a dos de los segmentos ya formados en etapas anteriores o un grupo y un individuo aun no clasificado, aquéllos más parecidos entre sí, de forma que se minimice en cada agrupación el incremento en la suma de las diferencias entre el individuo agrupado y la media del grupo al que se asigna, . En la etapa inicial la suma del cuadrado de las distancias es cero y se va incrementando a medida que se realizan las agrupaciones.

Suponogamos que tenemos cinco individuos medidos en una única variable .

Esto es: d(A, B)=4+25-(1/2)(2+5)2=29-24,5=4,5

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E |
| A | 0 | 4,5 | 24,5 | 32 | 98 |
| B |  | 0 | 8 | 12,5 | 60,5 |
| C |  |  | 0 | **0,5** | 24,5 |
| D |  |  |  | 0 | 18 |
| E |  |  |  |  | 0 |

La agrupación C y D es la que minimiza el SQD.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | A | B | CD | E |
| A | 0 | **4,5** | 38 | 98 |
| B |  | 0 | 14 | 60,5 |
| CD |  |  | 0 | 28,66 |
| E |  |  |  | 0 |

La agrupación A y B es la que minimiza la SQD

|  |  |  |  |
| --- | --- | --- | --- |
|  | AB | CD | E |
| AB | 0 | 41 | 108,66 |
| CD |  | 0 | **28,66** |
| E |  |  | 0 |

La agrupación *CDE* es la que minimiza la *SQD*. Finalmente *d*(*AB*, *CDE*)= 113,2.

La agrupación se muestra visualmente con un gráfico denominado dendrograma, donde las letras identifican a los individuos.

ejemplo<-c(2,5,9,10,15)  
ejemplo

## [1] 2 5 9 10 15

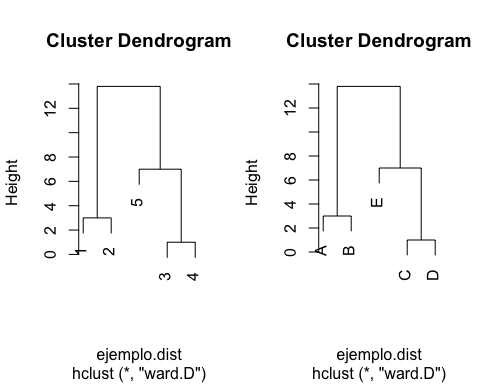
ejemplo.dist<-dist(ejemplo)  
ejemplo.hclust<-hclust(ejemplo.dist, method="ward")

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

ejemplo.hclust

##   
## Call:  
## hclust(d = ejemplo.dist, method = "ward")  
##   
## Cluster method : ward.D   
## Distance : euclidean   
## Number of objects: 5

par(mfrow=c(1,2))  
plot(ejemplo.hclust)  
labels<-c("A", "B", "C", "D", "E")  
ejemplo.hclust$labels<-labels  
plot(ejemplo.hclust)



par(mfrow=c(1,1))

Veamos cómo lo haríamos en el entorno R de análisis de datos.

## Partición de la muestra en k segmentos: el procedimiento k-means

Los algoritmos de clasificación del tipo *k-means* inician el proceso de selección especificando el número de clases que desea el analista, *k*, así como los centros iniciales para cada uno de los *k* clusters a formar. Después asigna cada individuo al grupo cuyo centro tenga más próximo. La solución inicial se mejora de manera iterativa hasta que se alcanza alguna forma de estabilidad, por ejemplo, hasta que las reasignaciones no reducen la suma del cuadrado de las distancias de los consumidores a los centros de los segmentos a los que han sido asignados (Helsen y Green 1991). Concretamente:

1. Empieza el proceso con un conjunto de centros iniciales o coordenadas de los *k* segmentos. Estos centros pueden obtenerse de manera aleatoria o por algún procedimiento como el propuesto por Hartigan y Wong (1979).
2. Después asignamos al consumidor *i* al cluster cuyo centro está más próximo. Los centros permanecen inalterados durante todo el proceso de asignación.
3. Seguidamente calculamos un nuevo conjunto de centros de los segmentos, como la media de los consumidores asignados a cada uno. Estos nuevos centros serán la base de un nuevo ciclo de reasignaciones.
4. Repetimos los procesos 2 y 3 hasta que ningún individuo cambie de grupo en el proceso de asignación especificado en el paso 2.

En este algoritmo el centro de cada segmento se define como la media de todos los individuos asignados al segmento en cada una de las variables en las que se han medido sus propiedades. Por ello necesita una matriz de datos *n*x*p* en lugar de una matriz de distancias entre los individuos (a diferencia de lo que ocurre en los algoritmos de clasificación jerárquica). El propósito del algoritmo es minimizar la suma de las distancias euclidianas al cuadrado de la partición realizada en *k* segmentos. Implícitamente asume que los grupos muestran una distribución normal esférica. Vemos un ejemplo.

*n*= número de individuos a clasificar

*p*= número de variables en las que se han medido las propiedades de los individuos

*x*(*i,j*)= valor que muestra el individuo *i* en la variable *j*

*x*(*l,j*)= valor medio de la variable *j* en el segmento *l*.

*n*(*l*)=número de individuos en el grupo *l*.

*d*(*i,l*)= distancia entre el individuo *i* y el centro del segmento *l*.

*d*(*i,l*)=raíz cuadra de la suma para toda *p* de las diferencias al cuadrado

*e*(*p*(*n, k*))=error de la partición

*p*(*n, k*)= resultado de partir la muestra en *k* segmentos y asignar a los *n* individuos a cada uno de los *k* segmentos.

*min* *e*(*p*(*n, k*))=*d2*(*i,l*(*i*))=*min* *SQDI*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tipo de pescado | Energía | Calorías | Calcio | Sum(i) |
| Caballa | 5 | 9 | 20 | 34 |
| Perca | 6 | 11 | 2 | 19 |
| Salmón | 4 | 5 | 20 | 29 |
| Sardina | 6 | 9 | 46 | 61 |
| Atún | 5 | 7 | 1 | 13 |
| Camarones | 3 | 1 | 12 | 16 |

Supongamos que queremos formar tres segmentos. El procedimiento nos daría como resultado los siguientes segmentos:

Segmento 1: perca, atún y caballa

Segmento 2: caballa y salmón

Segmento 3: sardina

Seguidamente calcularemos la media de las propiedades de los objetos clasificados en cada uno de los segmentos.

|  |  |  |  |
| --- | --- | --- | --- |
| Segmento | Energía | Calorías | Calcio |
| 1 | 14/3 | 19/3 | 5 |
| 2 | 9/2 | 7 | 20 |
| 3 | 6 | 9 | 46 |

Y calculamos la distancia euclidiana entre los individuos y las medias de cada grupo:

E(p(n=6,k=3))=SQDI=^(1,1)+(2,1)+…+(6,1)+…+(1,3)+…+(6,3)=137,805

Seguidamente probamos si cualquier cambio en la asignación de individuos a los segmentos reduce el error de la clasificación o suma del cuadrado de las distancias entre individuos y centros. Siendo *n*(*l*) el número de individuos asignados al segmento *l*, y *l*(*i*) el segmento que contiene al individuo *i*, primero calculamos las distancias al cuadrado entre el primer individuos y los centros de cada uno de los grupos:

(1,1)=++=232,22

A la hora de decidir donde clasificar al individuo i, calcularemos la variación en la *SQDI*. En este caso, cambiar la caballa de segmento incrementaría el error de la partición realizada.

Realizamos el proceso para todos los objetos y encontramos, en este caso, que el objeto 6, el camarón, puede ser clasificado en el segmento 2, en lugar del *1* inicial, y reducir el error de la clasificación. Quedando así la clasificación siguiente:

Segmento 1: perca y atún

Segmento 2: caballa, salmón y camarón

Segmento 3: sardinas.

peces<-read.csv("peces.csv", row.names=1, header=T)  
peces

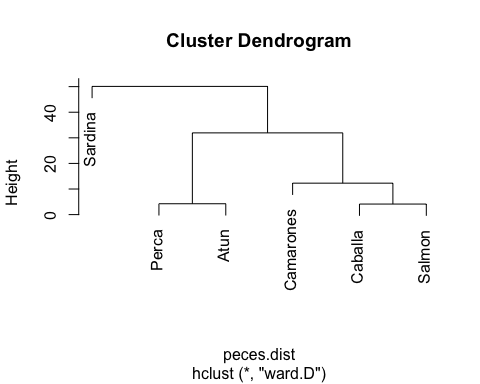
## Energia Calorias Calcio  
## Caballa 5 9 20  
## Perca 6 11 2  
## Salmon 4 5 20  
## Sardina 6 9 46  
## Atun 5 7 1  
## Camarones 3 1 12

peces.dist<-dist(peces)  
  
peces.dist

## Caballa Perca Salmon Sardina Atun  
## Perca 18.138357   
## Salmon 4.123106 19.078784   
## Sardina 26.019224 44.045431 26.381812   
## Atun 19.104973 4.242641 19.131126 45.055521   
## Camarones 11.489125 14.456832 9.000000 35.057096 12.688578

Después de calcular las distancias, podemos iniciar el proceso de aglomeración con el procedimiento de ward.

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



Después partimos la muestra en tres segmentos:

## KMNS(\*, k=3): iter= 1, indx=0  
## QTRAN(): istep=6, icoun=1  
## KMNS(\*, k=3): iter= 2, indx=3  
## KMNS(\*, k=3): iter= 3, indx=6

Si queremos saber las opciones de lal función k-means, podemos utilizar la función ? seguida del nombre de la función

?kmeans

Para saber los objetos incluidos en el resultado de clasificar la muestra, podemos uitlizar la función names().

names(peces.kmeans)

## [1] "cluster" "centers" "totss" "withinss"   
## [5] "tot.withinss" "betweenss" "size" "iter"   
## [9] "ifault"

Los centros de los segmentos se encuentran en el objeto centers. La classificación se encuentra en el objeto cluster, y en el objeto iter el número de iteraciones realizadas.

peces.kmeans$centers

## Energia Calorias Calcio  
## 1 5.5 9 1.50000  
## 2 6.0 9 46.00000  
## 3 4.0 5 17.33333

peces.kmeans$cluster

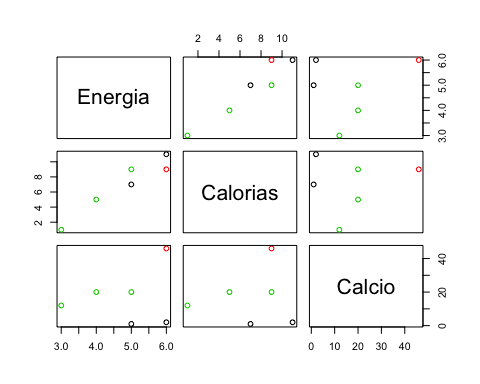
## Caballa Perca Salmon Sardina Atun Camarones   
## 3 1 3 2 1 3

peces.kmeans$iter

## [1] 3

El resultado lo podemos visualizar en el espacio de las bases de segmentación

plot(peces[1:3], col=peces.kmeans$cluster)  
points(peces[1:3])  
points(peces.kmeans$centers, col=1:3, pch=8,cex=2)



# Procedimientos más robustos

## PAM

El algoritmo pam tiene por objetivo encontrar los *k* consumidores representativos de la heterogeneidad del mercado alrededor de los cuales se clasifican los *n* consumidores medidos en *p* variables (una matriz de datos de tamaño *n*x*p*). Estos *k* consumidores representativos formarán los *k* segmentos en los que se dividirá una muestra de un mercado. Los *k* consumidores representativos se obtienen de minimizar la disimilitud entre todos los consumidores y su consumidor representativo más cercano. Es decir, el procedimiento busca encontrar conjunto de *k* segmentos de individuos pertenecientes a la muestra que minimiza la siguiente función objetivo:

De tal manera que cada consumidor se asigna al segmento al que pertenece el consumidor representativo más cercano. Es decir, el consumidor *i* se asigna al segmento *k* cuando el consumidor representativo *m(k)* está más cerca de *i* que de cualquier otro consumidor representativo, , para todo consumidor representativo *w* diferente de *k*.

Dado que el algoritmo pam sólo depende de las disimilitudes o distancias entre consumidores, la función sólo precisa como argumento una matriz de disimilitudes. Si el argumento que damos a la función es una matriz de datos *n*x*p*, entonces la función pam la convertirá en una matriz de disimilitudes.

El algoritmo pam es parecido al conocido método *k-means* (MacQueen 1967) ampliamente conocido en marketing (REF), implementado en el entorno *R* por medio de la función kmeans que utiliza el algoritmo de Hartigan y Wong (1979). El algoritmo k-means tiene por objetivo minimizar la suma del cuadrado de las distancias euclidianas e implícitamente asume que los grupos se distribuyen de forma esférica. La función pam, según Kaufman y Rouseeuw (1990), es más robusta que *k-means* porque el algoritmo implementado en la función pam minimiza la suma de las distancias sin elevarlas al cuadrado. Incluso los centros provisinales de los segmentos no son necesarios para iniciar el proceso de división de la muestra.

Adicionalmente la función pam del paquete cluster nos ofrece una gráfica denominada silueta (Rouseeuw 1987) y el correspondiente índice de calidad del proceso de segmentación con el objeto de ayudarnos a seleccionar el número de grupos que formaremos. La construcción del gráfico procede de la siguiente manera. Para cada consumidor *i* simbolizaremos con *A* el grupo de pertenencia y computaremos la disimilitud media del consumidor *i* frente al resto de consumidores del segmento *A*.

Después consideraremos cualquier grupo *C* diferente del anterior, *A*, y calcularemos la similitud media de *i* frente a todos los elementos de *C*, . Después realizaremos el mismo cálculo para el resto de los segmentos formados y tomaremos el valor menor, . Llamaremos *B* al segmento cuya disimilitud media sea menor frente al consumidor *i*, d(i,B)=b(i), que llamaremos el segmento vecino del consumidor *i*. Esta es la segunda mejor la clasificación del consumidor *i*. La longitud de la silueta se calculará de la siguiente manera:

. El valor de *s*(*i*) está acotado entre -1 y 1. Si el valor está cerca de 1, nos indica que el consumidor i está bien clasificado; si está cerca de 0, está mal clasificado (está más cerca de *B* que de *A*).

Para finalizar calcula un índice de calidad global que es la media de todos los índices de calidad de la clasificación de todos los consumidores, *s*(*i*), correspondientes a los *n* consumidores de la muestra de datos.

|  |  |
| --- | --- |
| Índice de calidad | Interpretación |
| 0,71-1,00 | Hemos encontrado una estructura fuerte |
| 0,51-0,70 | Hemos encontrado una estructura razonable |
| 0,26-0,50 | Hemos encontrado una estructura débil que podría ser artificial |
| 0,25 | No hemos encontrado una estructura en los datos |

## CLARA

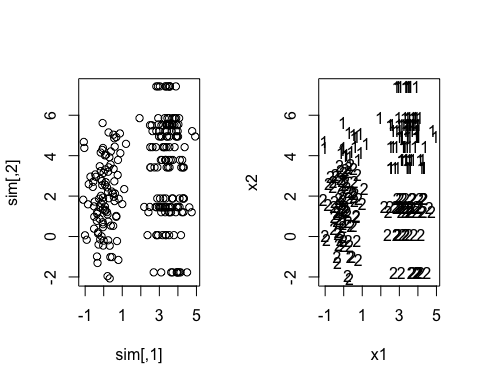
La función pam almacena en la memoria del ordenador toda la matriz de disimilitudes cuyo tamaño es . Este procedimiento es ineficiente con matrices datos que contienen más de 250 consumidores, pues tanto la memoria ocupada como el tiempo de cálculo crecen de manera cuadrática en función del tamaño de la muestra, ). Para resolver este problema Kaufman y Rouseeuw (1986) han propuesto la función clara. Esta función sólo guarda en la memoria del ordenador un subconjunto de la matriz de disimilitudes. El procedimiento de la función clara toma muestras de la matriz original de datos, para cada una de ellas realiza una segmentación y obtiene *k* segmentos que minimizan la función objetivo de pam. Después compara la suma de las distancias y se queda con la partición que la minimiza. Finalmente asigna los *n* consumidores a la partición anterior. De esta manera el almacenamiento en memoria y el tiempo de computación crece sólo de manera lineal con el tamaño de la muestra, *n*.

Vamos a generar una base de datos artificial y a realiar una partición con kmeans, después visuallizaremos los datos generados y la identificación que a realizado el procedimiento de partición.

#datos simulados  
sim<-rbind(cbind(rnorm(100,0,0.5),  
rnorm(100,2,1.5)),cbind(rnorm(150,3.5,0.5), rnorm(15,4,2.5)))  
head(sim)

## [,1] [,2]  
## [1,] -0.2527250 1.867373  
## [2,] -0.5677982 2.708102  
## [3,] 0.3697388 3.567280  
## [4,] 0.2675124 3.423510  
## [5,] -0.0589612 5.614422  
## [6,] -0.1578450 2.839106

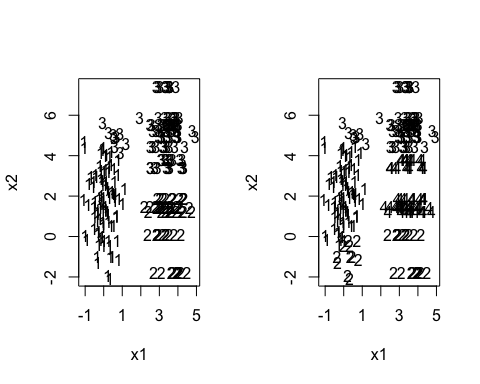
par(mfrow=c(1,2))  
plot(sim)  
#Partición de lal muestra con kmeans  
#library(MASS)  
sim.kmeans<-kmeans(sim, 2)  
#eqscplot(sim, type="n", xlab="x1", ylab="x2")  
plot(sim, type="n", xlab="x1", ylab="x2")  
text(sim, labels=sim.kmeans$cluster)



par(mfrow=c(1,1))

repetimos el proceso para 3 y 4 segmentos.

par(mfrow=c(1,2))  
sim.kmeans3<-kmeans(sim, 3)  
plot(sim, type="n", xlab="x1", ylab="x2")  
text(sim, labels=sim.kmeans3$cluster)  
  
sim.kmeans4<-kmeans(sim, 4)  
plot(sim, type="n", xlab="x1", ylab="x2")  
text(sim, labels=sim.kmeans4$cluster)



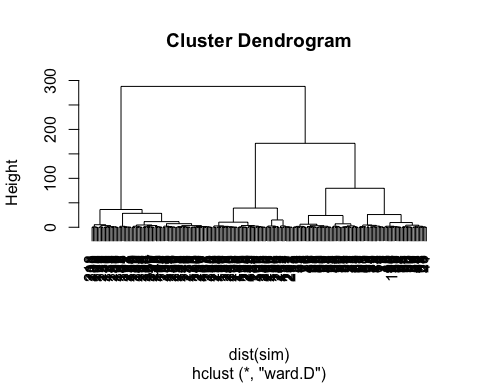
par(mfrow=c(1,1))

Ahora calcularemos los centros iniciales para ver si mejoramos la clasificación

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

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

plot(sim.hclust)



centros.h<-tapply(sim, list(rep(cutree(sim.hclust, 2),ncol(sim)), col(sim)),mean)  
centros.h

## 1 2  
## 1 1.352996 1.489807  
## 2 3.441462 5.125233

sim.kmeans.centros<-kmeans(sim, centros.h)

Repetimos la partición pero para 3 y 4 segmentos

centros.h3<-tapply(sim, list(rep(cutree(sim.hclust, 3), ncol(sim)), col(sim)), mean)  
centros.h3

## 1 2  
## 1 0.06233665 1.9548443  
## 2 3.50409574 0.7147458  
## 3 3.44146152 5.1252330

sim.kmeans.centros3<-kmeans(sim, centros.h3)  
centros.h4<-tapply(sim, list(rep(cutree(sim.hclust, 4), ncol(sim)), col(sim)), mean)  
centros.h4

## 1 2  
## 1 -0.07294034 0.5025624  
## 2 0.19230866 3.3501739  
## 3 3.50409574 0.7147458  
## 4 3.44146152 5.1252330

sim.kmeans.centros4<-kmeans(sim, centros.h4)  
table(sim.kmeans$cluster, sim.kmeans.centros$cluster)

##   
## 1 2  
## 1 0 109  
## 2 141 0

table(sim.kmeans3$cluster, sim.kmeans.centros3$cluster)

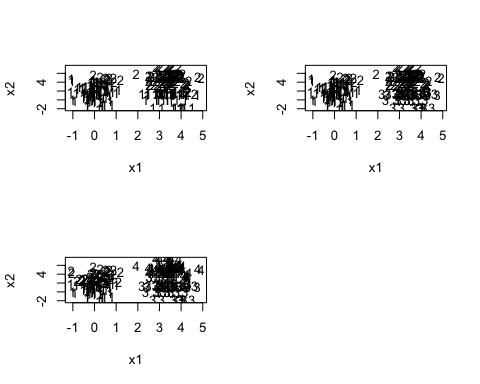
##   
## 1 2 3  
## 1 91 0 0  
## 2 0 60 0  
## 3 0 0 99

table(sim.kmeans4$cluster, sim.kmeans.centros4$cluster)

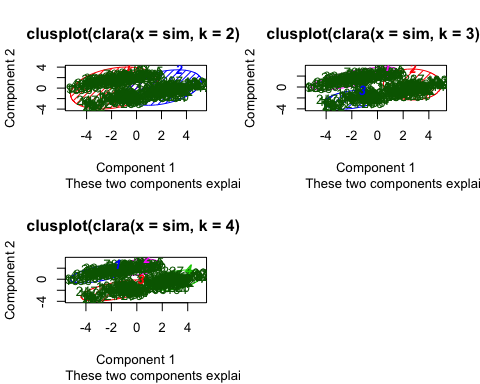
##   
## 1 2 3 4  
## 1 43 39 0 0  
## 2 11 0 20 0  
## 3 0 7 0 70  
## 4 0 0 40 20

Ahora vamos a utilizar el paquete cluster y concretamente la función clara. Estimaremos una partición con 2, 3 y 4 segmentos y visualizaremos el resultado en el espacio de las bases de segmentación.

library(cluster)  
sim.clara<-clara(sim, 2)  
sim.clara3<-clara(sim, 3)  
sim.clara4<-clara(sim, 4)  
par(mfrow=c(2,2))  
plot(sim, type="n", xlab="x1", ylab="x2")  
text(sim, labels=sim.clara$clustering)  
plot(sim, type="n", xlab="x1", ylab="x2")  
text(sim, labels=sim.clara3$clustering)  
plot(sim, type="n", xlab="x1", ylab="x2")  
text(sim, labels=sim.clara4$clustering)  
par(mfrow=c(1,1))

 También podemos utilitzar la función clusplot para visualizar el resultado:

par(mfrow=c(2,2))  
clusplot(sim.clara, color=TRUE, shade=TRUE, labels=2, lines=0)  
#plot(sim.clara)  
clusplot(sim.clara3, color=TRUE, shade=TRUE,labels=2, lines=0)  
#plot(sim.clara3)  
clusplot(sim.clara4, color=TRUE, shade=TRUE,labels=2, lines=0)  
par(mfrow=c(1,1))



#plot(sim.clara4)

Ahora comparamos los resultados con kmeans utilizando la función table. Y podemos ver que los resultados difieren aunque ligeramente.

table(clara=sim.clara$clustering, kmeans=sim.kmeans.centros$cluster)

## kmeans  
## clara 1 2  
## 1 136 9  
## 2 5 100

table(clara=sim.clara3$clustering, kmeans=sim.kmeans.centros3$cluster)

## kmeans  
## clara 1 2 3  
## 1 91 0 0  
## 2 0 0 99  
## 3 0 60 0

table(clara=sim.clara4$clustering, Kmeans=sim.kmeans.centros4$cluster)

## Kmeans  
## clara 1 2 3 4  
## 1 54 0 0 0  
## 2 0 46 0 0  
## 3 0 0 60 7  
## 4 0 0 0 83

# Un ejemplo adicional

Vamos a clasificar a las ciudades de los EEUU en función de los críemens violentos perpetrados según los arrestos efectudos por la policía en 1973. La base de datos se componen de 50 observaciones (estados) y 4 variables (tipos de arrestos--Murder, Assault y Rape, número de casos por cada 100.000 habitantes-- y población urbana--UrbanPop--en porcentaje).

Como las variables no están medidas en la misma escala, procederemos a estandarizarlas.

mydata = USArrests  
mydata<- na.omit(mydata) # listwise deletion of missing  
mydata.orig = mydata #save original data copy  
mydata <- scale(mydata) # standardize variables

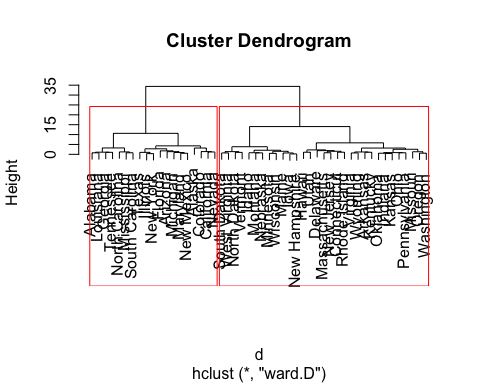
Después exploraremos los datos utilizando el procedimienot de Ward para realizar una agrupación jerárquica. Utilizaremos la distancia euclidiana para medir la similitud entre los estados.

d <- dist(mydata, method = "euclidean") # distance matrix  
fit<- hclust(d, method="ward")

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

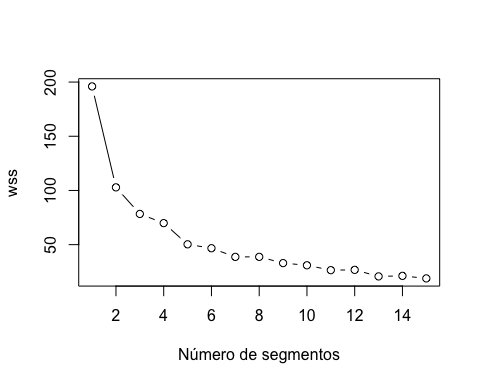
Visualizar el gráfico y mostrar el dendrograma con bordes rojos alrededor de *k* clusters

plot(fit) # display dendogram  
k1 = 2   
# eyeball the no. of clusters  
# cut tree into k1 clusters  
groups = cutree(fit, k=k1)  
rect.hclust(fit, k=2, border="red")



De nuevo la pregunta aparece, ¿cómo conocer el número óptimo de segmentos? Observando el dendrograma en ciertas ocasiones puede ser de ayuda para tomar la decisión. Pero en otras ocasiones, ¿qué deberíamos hacer? La mayoría de los paquetes de partición de muestras, si no todos, esperan que conozcas el número correcto de segmentos a formar, como es el caso de la función kmeans. Con R podemos obtener un *scree plot* de la familia de los gráficos que te muestran cómo la varianza dentro de los segmentos (un idicador de la calidad de la solución) se reduce con el número de segmentos formados. Por ello, con **R** puedes tomar una decisión informada. Aquí mostramos un ejemplo de cómo obtener un *scree plot*.

# Determina el número de segmentos  
#Calcula la variación dentro de los segmentos  
wss<- (nrow(mydata)-1)\*sum(apply(mydata,2,var))  
#Calcula la variación dentro de los segmentos para 14 particiones,   
#de 2 hasta 15  
for (i in 2:15) wss[i] <- sum(kmeans(mydata,centers=i) $withinss)   
plot(1:15, wss, type="b", xlab="Número de segmentos", ylab="wss")



# Busca un 'codo' en la distribución de la wss  
# Ese 'codo' nos indicará el número adecuado de segmentos a formar, en este caso  
k1=2

Después vamos arealizar la partición (kmeans) como hemos visto anteriormente.

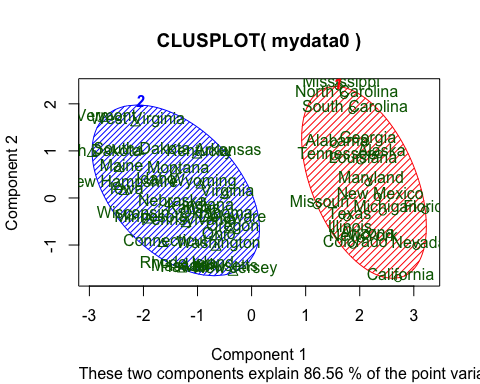
fit <- kmeans(mydata, k1) # k1 es el número de segmentos a formar  
fit$centers

## Murder Assault UrbanPop Rape  
## 1 1.004934 1.0138274 0.1975853 0.8469650  
## 2 -0.669956 -0.6758849 -0.1317235 -0.5646433

#Obtenemos los centros  
aggregate(mydata.orig,by=list(fit$cluster),FUN=mean)

## Group.1 Murder Assault UrbanPop Rape  
## 1 1 12.165 255.2500 68.40000 29.16500  
## 2 2 4.870 114.4333 63.63333 15.94333

# Añadimos el resultado de la partición a la base de datos  
mydata0<- data.frame(mydata.orig, fit$cluster)  
# Mostramso el resultado  
library(cluster)  
clusplot(mydata0, fit$cluster, color=TRUE, shade=TRUE,labels=2,  
lines=0)



# Preparación de los datos: Análisis componentes principals

El análisis de los componentes principales es una técnica para la reducción de datos multivariantes. Su propósito es encontrar un conjunto de nuevas variables, igual en número a las variables originales, aunque con ciertas propiedades: 1) las nuevas variables, llamadas componentes principales, están incorrelacionadas entre ellas (son independientes en el espacio formado por los componente principales), 2) los componentes principales están formados a partir de las variables originales, 3) los componentes principales explican la varianza de las variables originales, de manera que el primer componente explica la mayor parte posible de variación, el segundo la mayor parte de la varianza residual, y así sucesivamente.

Vamos a ver un ejemplo. Tomemos los datos, USAarrests correspondientes a los delitos cometidos en las ciudades de los EEUU en 1973 que proporciona uno de los paquetes de **R**. Los datos corresponden a arrestos por cada 100.000 residentes. Si los delitos están correlacionados entonces es posible construir una nueva base de datos a partir los datos originales sobre los delitos en los que las nuevas variables no estén correlacionadas. Estas propiedades de los componentes principales es de utilidad cuando utilizamos los medos de segmentación de mercados.

# Examinando la correlación entre los delitos cometidos en las ciudades de los EEUU  
cor(mydata)

## Murder Assault UrbanPop Rape  
## Murder 1.00000000 0.8018733 0.06957262 0.5635788  
## Assault 0.80187331 1.0000000 0.25887170 0.6652412  
## UrbanPop 0.06957262 0.2588717 1.00000000 0.4113412  
## Rape 0.56357883 0.6652412 0.41134124 1.0000000

Podemos ver que las variables Murder, Assault y Rape están muy correlacionadas. En cambio UrbanPop, porcentaje de población urbana, no está correlacionada con los asesinatos, asaltos o violaciones. Vamos a ver qué nos dicen los componentes principales de estos datos

# Principal Components Analysis  
# from the correlation matrix  
#fit.princomp<- princomp(mydata, cor=TRUE)  
fit.prcomp <-prcomp(mydata, scale=T, retx=T)  
summary(fit.prcomp)

## Importance of components:  
## PC1 PC2 PC3 PC4  
## Standard deviation 1.5749 0.9949 0.59713 0.41645  
## Proportion of Variance 0.6201 0.2474 0.08914 0.04336  
## Cumulative Proportion 0.6201 0.8675 0.95664 1.00000

Como dijimos anteriormente, cada componente principal explica la mayor parte posible de la variación residual. Así vemos que el primer componentes explica un 62%, el segundo un 25%, el tercero un 9 % y el cuarto un 4%.

Podemos saber cómo se forman los componentes principales a partir de las variables originales examinando las cargas o rotaciones. Estos son los vectores singulares que multiplican las variables originales para producir los components principales.

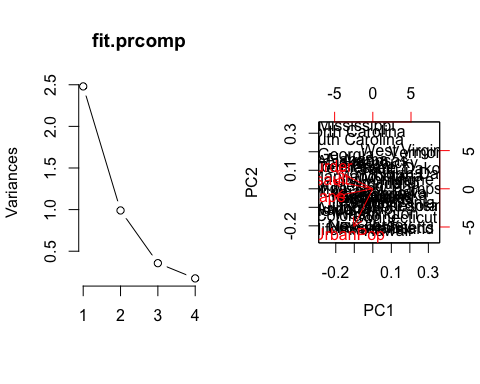
# print variance accounted for  
# pc loadings  
#loadings(fit.princomp)   
fit.prcomp$rotation

## PC1 PC2 PC3 PC4  
## Murder -0.5358995 0.4181809 -0.3412327 0.64922780  
## Assault -0.5831836 0.1879856 -0.2681484 -0.74340748  
## UrbanPop -0.2781909 -0.8728062 -0.3780158 0.13387773  
## Rape -0.5434321 -0.1673186 0.8177779 0.08902432

Estos coeficientes que multiplican a las variables origianles para producir los components principales nos indica la contribución de cada variable original (estandarizada) a la construcción de los components principales. Así el primer componente es una combinación lineal de asesinatos, asaltos y violaciones (PC1=-0,536Murder-0,583Assault-0,278UrbanPop-0,543Rape), mientras que el segundo es una construcción de la población urbana principalmente. Podríamos decir que la estructura de los delitos cometidos en las poblaciones se pueden ordenar en dos característias, los delitos y la población urbana.

Podemos observarlo si graficamos la contribución de los componentes a la variación de los datos originales estandarizados y situamos los datos originales en el plano de los componentes principales.

# scree plot  
par(mfrow=c(1,2))  
plot(fit.prcomp,type="lines")   
biplot(fit.prcomp)



par(mfrow=c(1,1))

Puntuación en las ciudades en los componentes principales (presentamos las seis primeras ciudades y las 6 últimas de la base de datos), la correlación entre los componentes principales contruidos (que debería ser cero) y la correlación de las variables originales con los factores

head(fit.prcomp$x[, c(1,2)])

## PC1 PC2  
## Alabama -0.9756604 1.1220012  
## Alaska -1.9305379 1.0624269  
## Arizona -1.7454429 -0.7384595  
## Arkansas 0.1399989 1.1085423  
## California -2.4986128 -1.5274267  
## Colorado -1.4993407 -0.9776297

tail(fit.prcomp$x[, c(1,2)])

## PC1 PC2  
## Vermont 2.7732561 1.3881944  
## Virginia 0.0953667 0.1977278  
## Washington 0.2147234 -0.9603739  
## West Virginia 2.0873931 1.4105263  
## Wisconsin 2.0588120 -0.6051251  
## Wyoming 0.6231006 0.3177866

cor(fit.prcomp$x[, c(1,2)])

## PC1 PC2  
## PC1 1.00000e+00 -4.56204e-17  
## PC2 -4.56204e-17 1.00000e+00

cor(scale(mydata), fit.prcomp$x[, c(1,2)])

## PC1 PC2  
## Murder -0.8439764 0.4160354  
## Assault -0.9184432 0.1870211  
## UrbanPop -0.4381168 -0.8683282  
## Rape -0.8558394 -0.1664602

Un proceso de segmentación ahora tendríamos que decidir si utilizar los componentes principales o bien reducir las bases de segmentación con la información proporcionada con los components principales. En el segundo caso podríamos decidir utilizar únicamente dos variables, los asantos y la población urbana.

Cuando los componentes principales no se pueden interpretar bien, es posible utilizar otros paquetes que nos proporcionan la posibilidad de girar los componentes sobre su eje con el objeto de encontrar una matriz de correlaciones más interpretable. En el entorno **R** podemos utilizar los paquetes psych y GPArotation. El primer paquete realiza la extracción de los componentes principales y utitliza al segundo para encontrar una matriz de correlaciones más interpretable. Vamos a utilizar los mismos datos sobre arrestos para ilustrar su uso. En este ejemplo utilizamos una rotación habitual, varimax, pero también es posible utilizar otros procedimientos para rotar la matriz de estructura inicial: quatimax, promax, oblimin, simplimax, o cluster.

library(psych)  
library(GPArotation)  
fit.principal<- principal(mydata, nfactors=2, rotate="varimax")  
fit.principal

## Principal Components Analysis  
## Call: principal(r = mydata, nfactors = 2, rotate = "varimax")  
## Standardized loadings (pattern matrix) based upon correlation matrix  
## PC1 PC2 h2 u2 com  
## Murder 0.94 -0.06 0.89 0.115 1.0  
## Assault 0.92 0.18 0.88 0.121 1.1  
## UrbanPop 0.07 0.97 0.95 0.054 1.0  
## Rape 0.73 0.48 0.76 0.240 1.7  
##   
## PC1 PC2  
## SS loadings 2.26 1.21  
## Proportion Var 0.57 0.30  
## Cumulative Var 0.57 0.87  
## Proportion Explained 0.65 0.35  
## Cumulative Proportion 0.65 1.00  
##   
## Mean item complexity = 1.2  
## Test of the hypothesis that 2 components are sufficient.  
##   
## The root mean square of the residuals (RMSR) is 0.08   
## with the empirical chi square 3.44 with prob < NA   
##   
## Fit based upon off diagonal values = 0.98

# Segmentación basada en un modelo estadístico

Los modelos de clases latentes suponen que la muestra que estamos analizando es representativa de una población mixta compuesta por *k* segmentos de tamaño . Por ello la probabilidad de que las medidas obtenidas del individuos *i* en las *p* variables o propiedades, , hayan sido generadas por un modelo M con parámetros es igual a la suma, para las *k* clases, de la probabilidad de que la observación del individuo *i* pertenezca a cada una de las clases ponderada por la proporción en que se han mezclado, . Concretamente,

Para estimar los parámetros del modelo M, , la media y la matrixz de varianzas de cada clase latente, , tenemos que formar la función de verosimilitud y maximizar su logaritmo mediante el procedimiento EM, para un determinado número de clases latentes, *k*:

## Clasificación a posteriori

Los modelos de clases latentes utilizados en los problemas de segmentación no solamente están interesados en estima rlos parámetros del modelo que genera la muestra mixta sino también en la clasificación de los individuos a los *k* grupos. Para ello utiliza las probabilidades calculadas después de estimar el modelo. Así una observación **y**i se asigna al grupo cuya probabilidad de pertenencia es mayor, es decir, la probabilidad de pertenecer al grupo *k* condicionada a las características medidas del sujeto *i*, **y***i*,

## Selección del modelo

Los modelos estimados con un número diferente de clases latentes o grupos no son modelos anidados, el modelo con k-1 segmentos o grupos no es un modelo reducido del modelo con k segmentos. Por ello tenemos que basarnos en criterios heurísticos nos aproximan la cantidad de información que incorpora y al mismo tiempo tienen en cuenta el número de parámetros que los definen. Concretamente el Bayesian Information Criteria, BIC, y el Consistent Akaike’s Information Criteria, CAIC. Estos criterios favorecen a los modelos que muestran un menor valor del estadístico.

library(mclust)

## Package 'mclust' version 5.1  
## Type 'citation("mclust")' for citing this R package in publications.  
##   
## Attaching package: 'mclust'  
##   
## The following object is masked from 'package:psych':  
##   
## sim

fit.Mclust<- Mclust(mydata)  
fit.Mclust

## 'Mclust' model object:  
## best model: diagonal, equal shape (VEI) with 3 components

names(fit.Mclust)

## [1] "call" "data" "modelName" "n"   
## [5] "d" "G" "BIC" "bic"   
## [9] "loglik" "df" "hypvol" "parameters"   
## [13] "z" "classification" "uncertainty"

# view solution summary

El mejor modelo tiene varianzas iguales diagonales (VEI) con tres segmentos

fit.Mclust$BIC

## Bayesian Information Criterion (BIC):  
## EII VII EEI VEI EVI VVI EEE  
## 1 -583.0950 -583.0950 -594.8311 -594.8311 -594.8311 -594.8311 -524.0454  
## 2 -538.8430 -540.8779 -527.0645 -528.6274 -535.4765 -537.2752 -526.5856  
## 3 -537.5739 -525.7362 -526.2295 -512.9696 -543.7186 -527.7408 -532.9479  
## 4 -521.7731 -526.6936 -517.8151 -519.4226 -543.1230 -543.7168 -534.2090  
## 5 -533.2088 -538.0439 -531.1501 -534.8807 -561.6269 -564.2794 -548.5223  
## 6 -548.2697 -545.5376 -549.8833 -545.3595 -591.5334 -583.3005 -567.0752  
## 7 -554.7524 -564.2459 -555.9070 -562.4990 -596.0957 -601.8503 -567.1581  
## 8 -564.6494 -571.8128 -568.0429 -580.0928 -612.3113 -617.3733 -576.3987  
## 9 -576.0047 -586.1872 -577.0350 -585.9141 -643.0224 -648.6003 -578.1330  
## EVE VEE VVE EEV VEV EVV VVV  
## 1 -524.0454 -524.0454 -524.0454 -524.0454 -524.0454 -524.0454 -524.0454  
## 2 -526.9234 -523.2596 -528.7465 -525.2101 -527.3191 -535.5699 -537.8664  
## 3 -524.2621 -523.1587 -543.5679 -539.6804 -550.5477 -563.9978 -567.0451  
## 4 -533.5558 -532.9792 -551.4833 -562.0097 -575.3697 -580.1516 -588.1424  
## 5 -543.1353 -550.5246 -563.5430 -576.7656 -594.4986 -610.2251 -628.3149  
## 6 NA -564.6998 NA -625.1663 -622.9173 NA NA  
## 7 NA -577.0973 NA -623.9653 -643.6280 NA NA  
## 8 NA NA NA -642.8520 -656.7630 NA NA  
## 9 NA NA NA -679.3634 -679.5256 NA NA  
## attr(,"criterion")  
## [1] "BIC"  
##   
## Top 3 models based on the BIC criterion:  
## VEI,3 EEI,4 VEI,4   
## -512.9696 -517.8151 -519.4226

# lookup all the options attempted  
classif <- fit.Mclust$classification   
# classifiation vector  
mydata1 <- cbind(mydata.orig, classif)   
# append to dataset  
# draw dendrogram with red borders around the k1 clusters  
mydata1[1:10,]

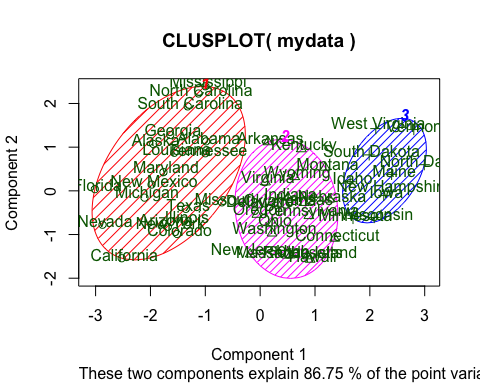
## Murder Assault UrbanPop Rape classif  
## Alabama 13.2 236 58 21.2 1  
## Alaska 10.0 263 48 44.5 1  
## Arizona 8.1 294 80 31.0 1  
## Arkansas 8.8 190 50 19.5 2  
## California 9.0 276 91 40.6 1  
## Colorado 7.9 204 78 38.7 1  
## Connecticut 3.3 110 77 11.1 2  
## Delaware 5.9 238 72 15.8 2  
## Florida 15.4 335 80 31.9 1  
## Georgia 17.4 211 60 25.8 1

#view top 10 rows  
table(fit$cluster, classif)

## classif  
## 1 2 3  
## 1 20 0 0  
## 2 0 20 10

Si queremos guardar el resultado

#write.table(mydata1,file.choose())   
#save output  
fit1=cbind(classif)  
rownames(fit1)=rownames(mydata)  
library(cluster)  
clusplot(mydata, fit1, color=TRUE, shade=TRUE,labels=2, lines=0)

 Paara obtener las medias en los segmenotos

cmeans=aggregate(mydata.orig,by=list(classif),FUN=mean); cmeans

## Group.1 Murder Assault UrbanPop Rape  
## 1 1 12.165 255.25 68.40 29.165  
## 2 2 5.965 136.60 69.95 18.460  
## 3 3 2.680 70.10 51.00 10.910

# Identificación de los individuos que forman los segmentos

Supongamos que la división de teléfonos celulares de Conglomerate Inc. ha llegado a un acuerdo con un fabricantes de microordenadores para desarrollar, producir y comercializar un producto híbrido entre un ayudante personal digital (PDA) y un teléfono ‘inteligente’. Provisionalmente lo han llamado ConneCtor que transmite y recibe tanto datos como voz (a diferencia de los PDAs actuales que únicamente transmiten datos.)

ConneCtor es ligero y adopta la forma de un teléfono móvil con una pequeña pantalla de cristal líquido (LCD) sensible al tacto, a todo lo largo del teléfono. Su sistema operativo (abierto) desarrolla las funciones estándares de un teléfono móvil y las tareas habituales de una agenda electrónica (calendario, calculadora, libro de direcciones, etc.) Puede enviar y recibir faxes, mensajes orales y e-mail.

Supongamos que han identificados unas bases de segmentación (Innovador, usuadio de busca, usuario de teléfono, usuario de agenda, accede a información pasiva, envía información, tiempo que trabaja fuera de la oficina, importancia de los móviles, importancia compartir información) así como un conjunto de variables descriptivas de los individuos (edad, eduación, ingresos, trabaja en construcción, en emergencias, en ventas, en servicios, es un profesional, tiene un pda, lee la revista Business Week, pc Magazine, Field & Stream, Modern Gourmet).

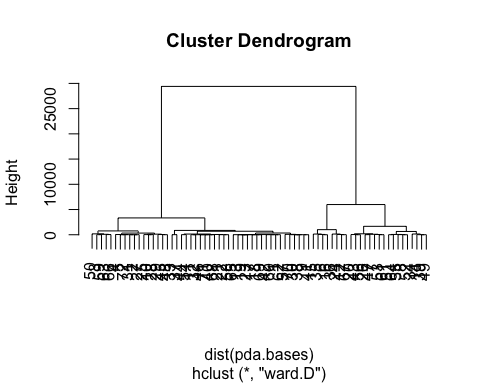
Con las de segmentación han explorado la estructura de las preferencias y beneficios buscados por los consumidores y han identificado 4 segmentos.

## Innovador UsePage UsePhone UseSchdl InfoPaisve InfoActive RemoteLoc  
## 1 1 2 4 4 1 2 5  
## 2 3 5 3 3 5 5 5  
## 3 4 3 3 4 2 3 4  
## 4 1 1 6 5 2 2 3  
## 5 3 4 2 6 5 4 4  
## 6 6 4 5 5 4 5 5  
## Wireless ShareInfo Monthly Invoice Age Eduaction Income Construction  
## 1 1 5 20 500 32 3 20 0  
## 2 7 2 40 650 42 3 47 0  
## 3 2 5 30 800 22 3 28 0  
## 4 2 3 10 650 46 2 45 0  
## 5 1 3 35 650 54 2 51 0  
## 6 2 2 0 1400 25 4 81 0  
## Emergency Sales Service Professnl PDA Bus\_Week PC\_Mag Fiel.Stream  
## 1 0 0 0 0 0 0 1 0  
## 2 0 0 1 0 0 0 0 0  
## 3 0 0 1 0 0 1 1 0  
## 4 0 1 0 0 0 1 0 0  
## 5 0 0 1 0 0 0 0 1  
## 6 0 0 0 1 1 0 1 0  
## M\_Gourmet  
## 1 0  
## 2 0  
## 3 0  
## 4 0  
## 5 0  
## 6 1

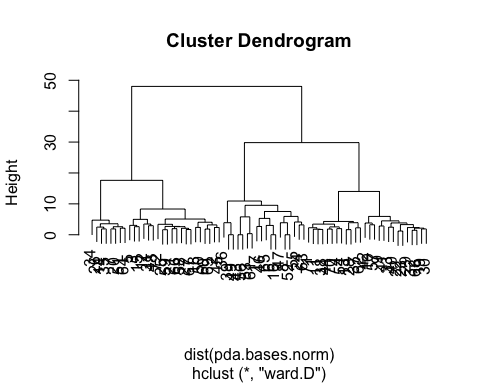
## [1] "Innovador" "UsePage" "UsePhone" "UseSchdl"   
## [5] "InfoPaisve" "InfoActive" "RemoteLoc" "Wireless"   
## [9] "ShareInfo" "Monthly" "Invoice" "Age"   
## [13] "Eduaction" "Income" "Construction" "Emergency"   
## [17] "Sales" "Service" "Professnl" "PDA"   
## [21] "Bus\_Week" "PC\_Mag" "Fiel.Stream" "M\_Gourmet"

## Innovador UsePage UsePhone UseSchdl InfoPaisve InfoActive RemoteLoc  
## 1 1 2 4 4 1 2 5  
## 2 3 5 3 3 5 5 5  
## 3 4 3 3 4 2 3 4  
## 4 1 1 6 5 2 2 3  
## 5 3 4 2 6 5 4 4  
## 6 6 4 5 5 4 5 5  
## Wireless ShareInfo Monthly Invoice  
## 1 1 5 20 500  
## 2 7 2 40 650  
## 3 2 5 30 800  
## 4 2 3 10 650  
## 5 1 3 35 650  
## 6 2 2 0 1400

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



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



## 1 2 3 4 5 6 7 8 9 10 11  
## 1 -0.747 -1.05 0.67 0.23 -0.28 -0.598 0.03 -0.72 0.16 -0.54 -0.76  
## 2 -0.014 0.83 -0.88 -0.49 0.17 -0.054 0.23 1.04 -0.23 0.74 -0.38  
## 3 1.189 0.18 0.41 0.62 -0.33 0.506 -0.48 -0.61 -0.78 -0.67 0.87  
## 4 -0.213 0.94 -0.91 -0.91 1.22 0.941 0.39 1.11 1.80 1.40 1.47

## [1] "cluster" "centers" "totss" "withinss"   
## [5] "tot.withinss" "betweenss" "size" "iter"   
## [9] "ifault"

## [1] 28 20 15 9

Una vez identificados los segmentos que estructuran el mercado, debemos identificar a los individuos que forman cada segmento. Esta información nos permite: 1) cuantificar el volumen de los segmentos y así analizar su rentabilidad, y 2) Conocer a qué medios de comunicación estan expuestos.

Para ello podemos utilizar la función discriminante lineal, lda implementada en el paquete MASS.

## [1] "Innovador" "UsePage" "UsePhone" "UseSchdl"   
## [5] "InfoPaisve" "InfoActive" "RemoteLoc" "Wireless"   
## [9] "ShareInfo" "Monthly" "Invoice" "Age"   
## [13] "Eduaction" "Income" "Construction" "Emergency"   
## [17] "Sales" "Service" "Professnl" "PDA"   
## [21] "Bus\_Week" "PC\_Mag" "Fiel.Stream" "M\_Gourmet"

## [1] "Age" "Eduaction" "Income" "Construction"  
## [5] "Emergency" "Sales" "Service" "Professnl"   
## [9] "PDA" "Bus\_Week" "PC\_Mag" "Fiel.Stream"   
## [13] "M\_Gourmet"

## Call:  
## lda(pda.des, pda.bases.norm.kmeans4$cluster)  
##   
## Prior probabilities of groups:  
## 1 2 3 4   
## 0.39 0.28 0.21 0.12   
##   
## Group means:  
## Age Eduaction Income Construction Emergency Sales Service Professnl PDA  
## 1 34 2.4 39 0.071 0.036 0.54 0.11 0.14 0.21  
## 2 39 1.9 42 0.050 0.100 0.20 0.50 0.10 0.15  
## 3 36 3.3 69 0.067 0.000 0.13 0.00 0.67 0.87  
## 4 37 1.9 36 0.556 0.333 0.00 0.00 0.11 0.11  
## Bus\_Week PC\_Mag Fiel.Stream M\_Gourmet  
## 1 0.29 0.36 0.071 0.036  
## 2 0.10 0.30 0.150 0.050  
## 3 0.73 0.67 0.133 0.333  
## 4 0.00 0.22 0.667 0.000  
##   
## Coefficients of linear discriminants:  
## LD1 LD2 LD3  
## Age -0.0068 -0.0048 -0.0321  
## Eduaction -0.6758 -0.2229 0.3940  
## Income -0.0196 0.0102 -0.0245  
## Construction 1.3648 2.3666 -0.1541  
## Emergency 1.3794 1.9141 -0.7648  
## Sales 0.5361 -0.6424 0.1606  
## Service 1.7271 -0.3852 -2.5807  
## Professnl -0.4024 0.3514 -0.5924  
## PDA -0.9967 0.3256 -0.5583  
## Bus\_Week -1.2479 0.1140 0.1745  
## PC\_Mag -0.0551 0.2075 0.3053  
## Fiel.Stream -0.1314 1.2222 -0.0063  
## M\_Gourmet 0.2046 0.4655 -1.0074  
##   
## Proportion of trace:  
## LD1 LD2 LD3   
## 0.67 0.21 0.12

Una vez estimada la función discriminante podemos ver las medias de las variables descriptivas en los segmentos:

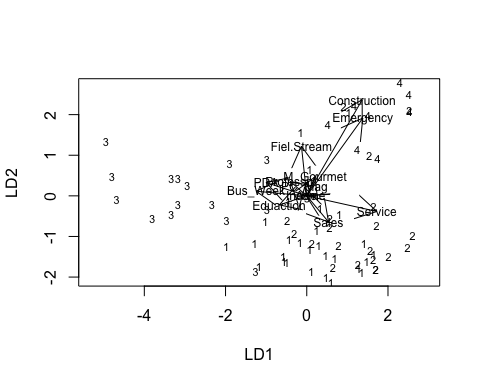
t(pda.des.lda$means)

## 1 2 3 4  
## Age 34.464 39.20 36.000 36.56  
## Eduaction 2.357 1.85 3.267 1.89  
## Income 39.000 42.05 68.933 35.56  
## Construction 0.071 0.05 0.067 0.56  
## Emergency 0.036 0.10 0.000 0.33  
## Sales 0.536 0.20 0.133 0.00  
## Service 0.107 0.50 0.000 0.00  
## Professnl 0.143 0.10 0.667 0.11  
## PDA 0.214 0.15 0.867 0.11  
## Bus\_Week 0.286 0.10 0.733 0.00  
## PC\_Mag 0.357 0.30 0.667 0.22  
## Fiel.Stream 0.071 0.15 0.133 0.67  
## M\_Gourmet 0.036 0.05 0.333 0.00

#Ahora vamos a calcular los valores medios de las variables en los grupos  
#Observemos primero los valores medios. Con esta información ya sería suficiente   
#para caracterizar a los grupos. Su representación en el espacio de las funciones   
#discriminante nos permiten visualizar la tabla de medias y ver su asociación con los grupos.

También podemos visualizar el resultado

plot(pda.des.lda, dimen=2)  
?lda  
lda.arrows <- function(x, myscale = 1, tex = 0.75, choices = c(1,2), ...){  
 ## adds `biplot` arrows to an lda using the discriminant function values  
 heads <- coef(x)  
 arrows(x0 = 0, y0 = 0,   
 x1 = myscale \* heads[,choices[1]],   
 y1 = myscale \* heads[,choices[2]], ...)  
 text(myscale \* heads[,choices], labels = row.names(heads),   
 cex = tex)  
}  
lda.arrows(pda.des.lda)

 También podemos construir un gráfico sintético:

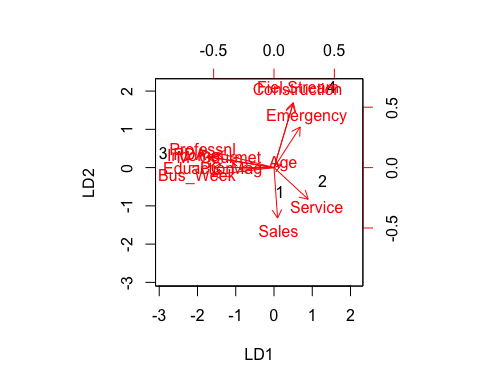
#para interpretar el significado de las funciones discriminantes  
#estimamos su correlación con los descriptores  
pda.des.puntos<-predict(pda.des.lda, pda.des)$x  
cor(pda.des, pda.des.puntos)

## LD1 LD2 LD3  
## Age 0.077 0.03304 -0.2851  
## Eduaction -0.623 0.00094 0.1102  
## Income -0.668 0.10709 -0.4053  
## Construction 0.196 0.65519 0.2488  
## Emergency 0.271 0.42002 0.0179  
## Sales 0.037 -0.52229 0.4898  
## Service 0.354 -0.32762 -0.6683  
## Professnl -0.590 0.15320 -0.1677  
## PDA -0.695 0.11732 -0.1936  
## Bus\_Week -0.638 -0.07559 0.0057  
## PC\_Mag -0.355 -0.01577 -0.0759  
## Fiel.Stream 0.200 0.67248 0.0493  
## M\_Gourmet -0.455 0.07720 -0.2318

#Para visualizar los segmentos  
#También podemos representar gráficamente los centros de los grupos en el espacio de las funciones discriminantes.   
#Para ello tenemos, primero, la función predict que toma como argumento un objeto de la clase lda y utilizar   
#las funciones discriminantes obtenidas para predecir los valores de un conjunto de valores,   
#concretamente nos interesa conocer la puntuación de las medias en el espacio de las funciones discriminantes.  
predict(pda.des.lda, pda.des.lda$means)$x

## LD1 LD2 LD3  
## 1 0.15 -0.64 0.67  
## 2 1.27 -0.35 -0.88  
## 3 -2.90 0.40 -0.32  
## 4 1.52 2.12 0.41

#Vamos ya podemos representar visualmente la caracterización de los grupos.   
#Para ello utilizamos la función biplot con dos grupos de datos,   
#la puntuación de los centros de los grupos en las funciones discriminantes   
#y la correlación de las variables discriminantes con las funciones discriminantes.  
pda.des.cor<-cor(pda.des, pda.des.puntos)  
biplot(predict(pda.des.lda, pda.des.lda$means)$x, pda.des.cor)



También podemos comprobar la calidad de la predicción realizada por la función discriminante:

#Comprobamos la calidad dela prediccion realizada por la función discriminante  
  
options(digits=4)  
pda.des.predict<-predict(pda.des.lda, pda.des)$class  
table(segmento=pda.bases.norm.kmeans4$cluster, lda=pda.des.predict)

## lda  
## segmento 1 2 3 4  
## 1 23 3 0 2  
## 2 5 12 0 3  
## 3 3 0 12 0  
## 4 1 0 0 8

## predicción de la pertenencia a un segmento con un conjunto nuevo de datos

# Anexo

# Referencias

Fraley, C., Raftery, A. E., Murphy, T. B., & Scrucca, L. (2012). Mclust Version 4 for R: Normal Mixture Modeling for Model-Based Clustering, Classification, and Density Estimation. Technical Report No. 597, Department of Statistics, University of Washington.

Fraley, C., & Raftery, A. E. (1998). How many clusters? Which clustering Method? Answers via model-based cluster analysis. Computer Journal, 41, 578–588.

Helsen, K., & Green, P. E. (1991). A Computational Study of Replicated Clustering with an Application to Market Segmentation. Decision Sciences, 22, 1124–1141.

Wishart, D. (1998). Efficient hierarchical cluster analysis for data mining and knowledge discovery. Computing Science and Statistics, 30.

Wishart, D. (1999). Clustering Methods for Large Data Problems. In International Statistical Institute (pp. 437–440). Helsinki.

Celeux, G., Biernacki, C., & Govaert, G. (2006). Choosing models in model-based clustering and discriminant analysis.

Wedel, M., & Kamakura, W. A. (1998). Market Segmentation: Conceptual and Methodological Foundations. Boston: Kluwer Academic.

Bartholomew, D. J., Knott, M., & Moustaki, I. (2011). Latent Variable Models and Factor Analysis: A Unified Approach (3 edition.). Chichester, West Sussex: Wiley.

Hoek, J., Gendall, P., & Esslemont, D. (1996). Market Segmentation. A Search for the Holy Grail? Journal of Marketing Practice, 2(1), 25–34.

Johnson, R. M. (1971). Market Segmentation: A Strategic Management Tool. Journal of Marketing Research, 8(February), 13–18.

Winter, F. W. (1979). A Cost-Benefit Approach to Market Segmentation. Journal of Marketing, 43(Fall), 103–111.

Mahajan, V., & Jain, A. K. (1978). An Approach to Normative Segmentation. Journal of Marketing Research, 15(August), 338–345.

Klastorin, T. D. (1983). Assesing Cluster Analysis Results. Journal of Marketing Research, 20(February), 92–98.

Funkhouser, G. R. (1983). A Note on the Reliability of Certain Clustering Algorithms. JMR, Journal of Marketing Research, 20(1), 99.

Kaufman, L., & Rousseeuw, P. J. (1990). Finding Groups in Data–An Introduction to Cluster Analysis. New York: John Wiley & Sons.