Anexo IX. Clustering: método de Ward

Hemos realizado un análisis clustering con varios métodos jerárquicos y no jerárquicos, de tal manera que valoraremos cual es el mejor de todos y, por tanto, aquel que nos sirve mejor para agrupar nuestro conjunto de datos. En el presente documento se tratará del método jerárquico Ward que forma los clusters de forma que se maximice la homogeneidad intra-clusters, es decir, que se produzca un menor incremento en la Suma de Cuadrados (varianza) intra-cluster

Vamos a cargar los datos que nos harán falta para realizar un análisis clustering.

```
load("C:/Users/losaa/OneDrive/Escritorio/Estudios/Proyecto II/ncdata.RData")
load("C:/Users/losaa/OneDrive/Escritorio/Estudios/Proyecto II/desc_data2.RData")
```

Prepararemos los datos de tal manera que vamos a quedarnos con solo las variables numéricas con valor de popularidad mayor que 70 y además eliminaremos las variables que nos muestran la duración de la canción, la popularidad y el año en el que salió, ya que estas no nos son de gran ayuda para encontrar una agrupación en los datos.

```
variablesnc = desc_data$variable[desc_data$type ==
'numerical'] variablesnc = variablesnc[-15] # Quitamos la
columna classical hola = ncdata[ncdata$popularity > 70,]
datos =
hola[,variablesnc]
datos =
datos[,-c(3,8,12)]
```

Imputamos los valores faltantes por la media y escalamos los datos ya que encontramos variables medidos en unidades diferentes y con magnitudes distintas.

```
missingVar = colnames(datos)[apply(datos, 2, function (x) sum(is.na(x))) > 0]
for (v in missingVar) {
  datos[is.na(datos[,v]),v] = mean(datos[,v], na.rm = TRUE)
}
datos = scale(datos, center = TRUE, scale = TRUE)
```

A continuación, definimos nuestra matriz de distancias con la distancia euclídea ya que pretendemos encontrar datos que sean similares y esta puede ser una buena medida de distancia.

```
midist <- get_dist(datos, stand = FALSE, method = "euclidean")
fviz_dist(midist, gradient = list(low = "#00AFBB", mid = "white", high = "#FC4E07")) + ggtitle("Martiz de distancias
Popularidad > 70")
```

Realizaremos varios cálculos del estadístico de Hopkins para posteriormente hacer un resumen de los resultados obtenidos y valorar entre que valores se encuentra y si existe o no agrupamiento en los datos.

```
set.seed(100)
myN = c(20, 35, 50, 65)
myhopkins = NULL
myseed = sample(1:1000, 10)
for (i in myN) {
  for (j in myseed) {
    tmp = get_clust_tendency(data = datos, n = i, graph = FALSE, seed = j)
    myhopkins = c(myhopkins, tmp$hopkins_stat)
  }
}
summary(myhopkins)
```

```
##      Min. 1st Qu. Median       Mean 3rd Qu.
Max. ## 0.8334 0.8506 0.8562 0.8562 0.8617
0.8789
```

Como podemos observar el coeficiente de Silhouette medio es alto y oscila entre 0.83 y 0.87 , lo cual nos muestra que existe agrupamiento en los datos. La pregunta que nos hacemos es si el método Ward que emplearemos en este documento nos servirá para encontrar dicho agrupamiento.

A continuación, hemos aplicado el método Ward y hemos creado diferentes grupos que corresponden a varias opciones respecto al número de clusters, que varía entre 2 y 7 según hemos creado en el código que aparece seguidamente. Además mostraremos la cantidad de canciones que contiene cada cluster según los diferentes grupos definidos.

```
clust1          <-          hclust(midist,
method="ward.D2")        grupos1        <-
cutree(clust1, k=2)
grupos2   <-   cutree(clust1,
k=3)        grupos3        <-
cutree(clust1,  k=4)  grupos4
<-     cutree(clust1,     k=5)
grupos5   <-   cutree(clust1,
k=6)        grupos6        <-
cutree(clust1,           k=7)
table(grupos1)
```

```
## grupos1
##      1       2
## 2563 544
```

```
table(grupos2)
```

```
## grupos2
##      1       2       3
## 2563 106 438
```

```
table(grupos3)
```

```
## grupos3
##    1    2    3 4
## 2079 106 484   438
```

```
table(grupos4)
```

```
## grupos4
##    1    2    3 4    5
## 407 106 1672   484  438
```

```
table(grupos5)
```

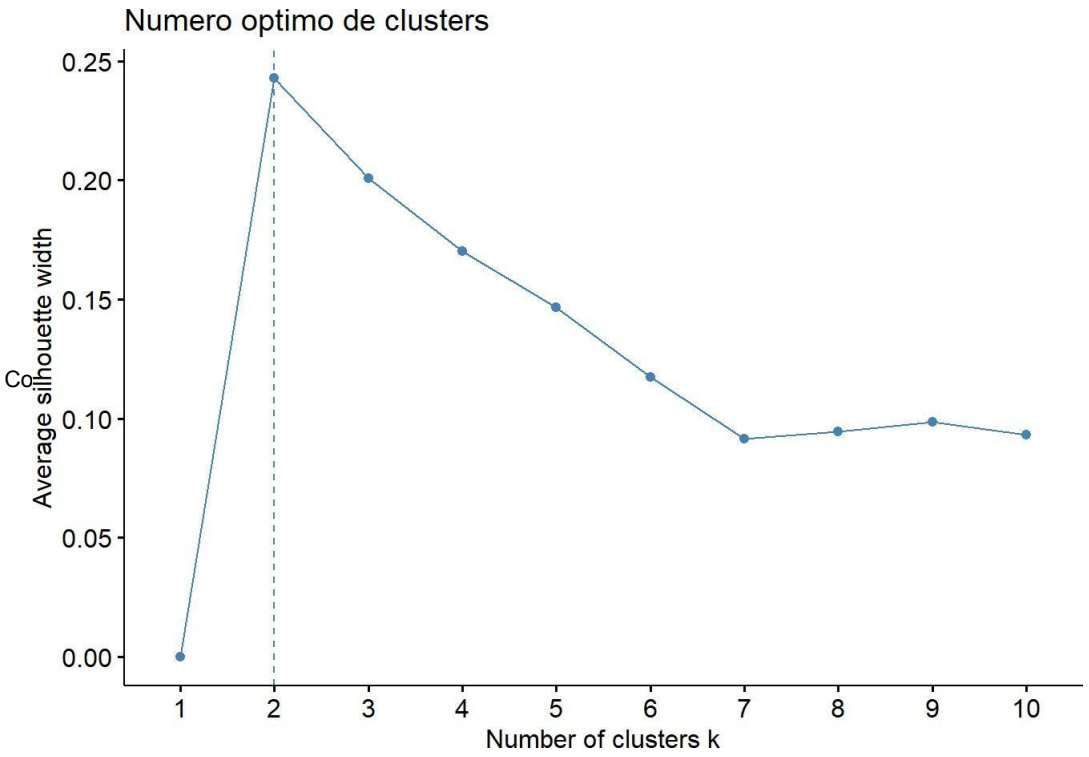
```
## grupos5
##    1    2    3 4    5        6
## 407 106 564    1108 484    438
```

```
table(grupos6)
```

```
                4
```

Para validar cuál sería el número óptimo de clusters hemos creado un gráfico con estos en el eje X y en el eje Y el coeficiente de Silhouette. Nos interesa el coeficiente de Silhouette más alto, el cual indicará que con ese número de clusters existe un mayor agrupamiento en nuestros datos.

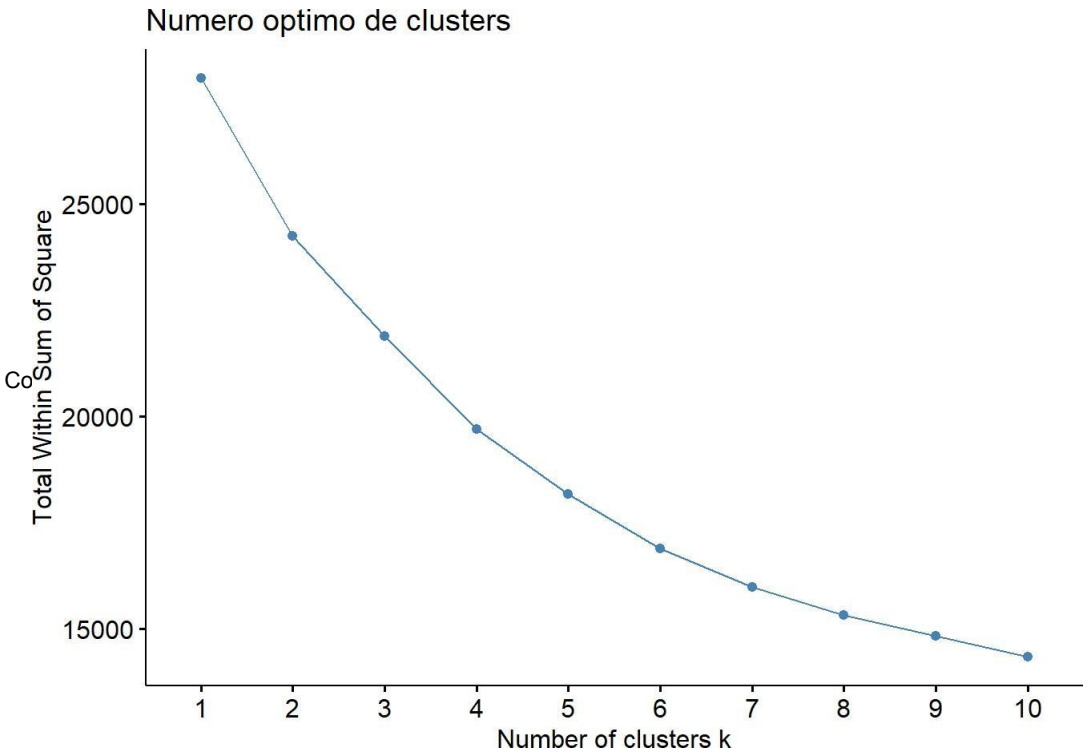
```
fviz_nbclust(x = datos, FUNcluster = hcut, method = "silhouette", hc_method = "ward.D2",
             k.max = 10, verbose = FALSE) +
  labs(title = "Numero optimo de clusters")
```



que 2 es el mejor número de clusters según el método de Ward. Aunque 3 no parece una mala opción.

Seguidamente, hacemos lo mismo pero con la suma de cuadrados intra-cluster: esta, al contrario que con Silhouette preferimos que el valor sea el menor posible.

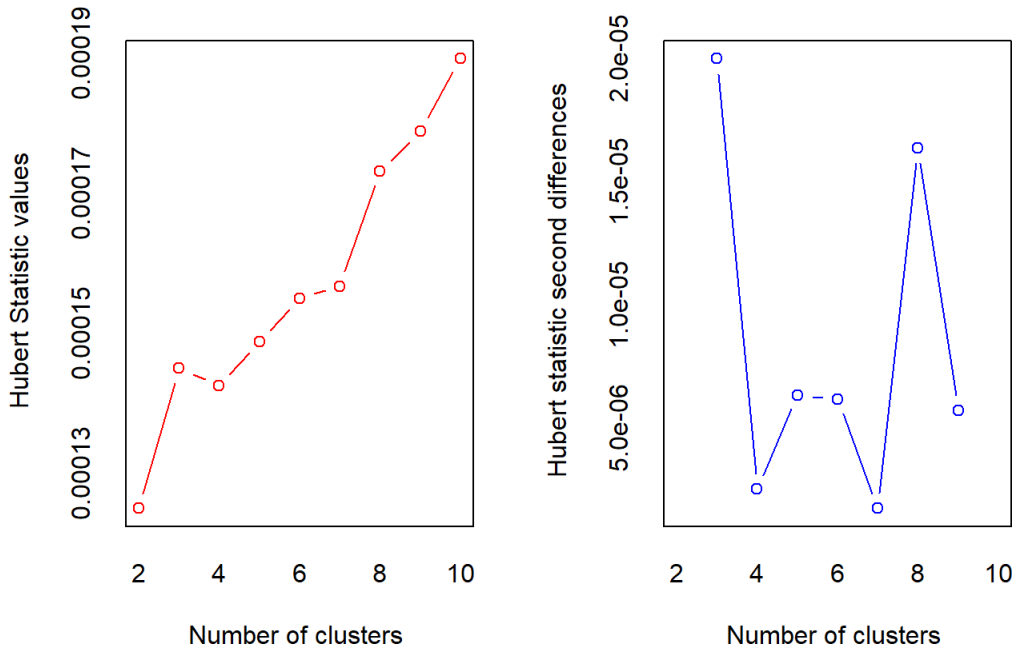
```
fviz_nbclust(x = datos, FUNcluster = hcut, method = "wss", hc_method = "ward.D2",
             k.max = 10, verbose = FALSE) +
  labs(title = "Numero optimo de clusters")
```



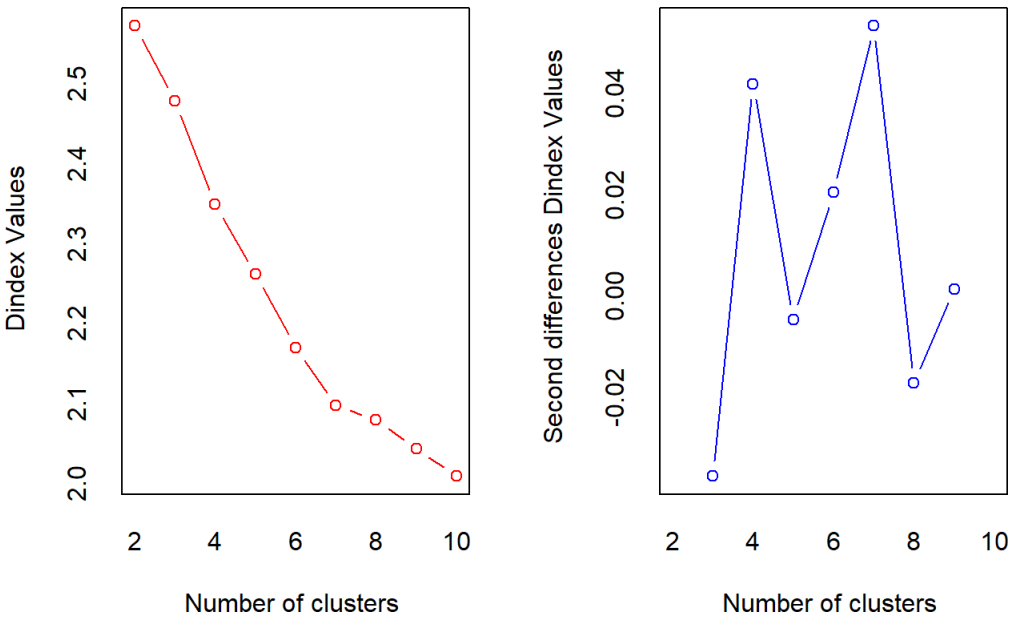
interesaría tener 5 o 6 clusters según la suma de cuadrados intra-cluster.A partir de 6 no varía mucho la diferencia por lo que suponemos que los datos en este punto empiezan a sobre ajustarse.

Por último, vamos a realizar la conclusión comprobando todos los índices posibles que nos proporciona la librería NbClust y confirmaremos el número de clusters que utilizaremos.

```
res.nbclust <- NbClust(data = datos, diss = midist, distance = NULL, min.nc = 2, max.nc = 10, method = "ward.D2", inde
x ="all")
```




```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##      In the plot of Hubert index, we seek a significant knee that corresponds to a
##      significant increase of the value of the measure i.e the significant peak in Hubert ## index
second differences plot.
##
```



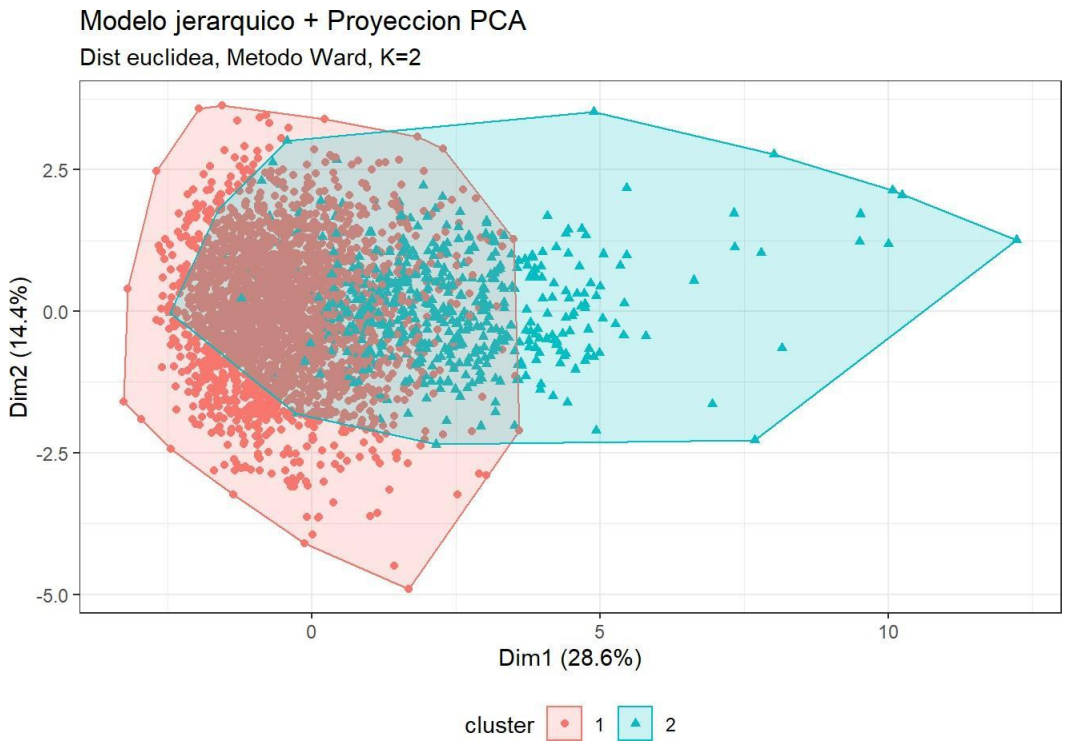
```
## *** : The D index is a graphical method of determining the number of clusters.
##     In the plot of D index, we seek a significant knee (the significant peak in Dindex ##  second
differences plot) that corresponds to a significant increase of the value of ##       the measure.
##
##
*******************************************************************
## * Among all indices:
## * 3 proposed 2 as the best number of
clusters ## * 6 proposed 3 as the best number
of clusters ## * 4 proposed 4 as the best
number of clusters ## * 2 proposed 6 as the
best number of clusters ## * 2 proposed 7 as
the best number of clusters ## * 1 proposed 8
as the best number of clusters ## * 3 proposed
10 as the best number of clusters ##
##     ***** Conclusion ***** ##
## * According to the majority rule, the best number of clusters is 3
##
##
##   *******************************************************************
```

Como conclusión obtenemos que para el método ward, observando todos los índices, de acuerdo a la gran mayoría la mejor opción es utilizar 3 clusters.

Seguidamente, vamos a graficar la proyección en PCA de los 6 diferentes grupos que hemos creado y vamos a ver si hay alguno que separa nuestros datos considerablemente.

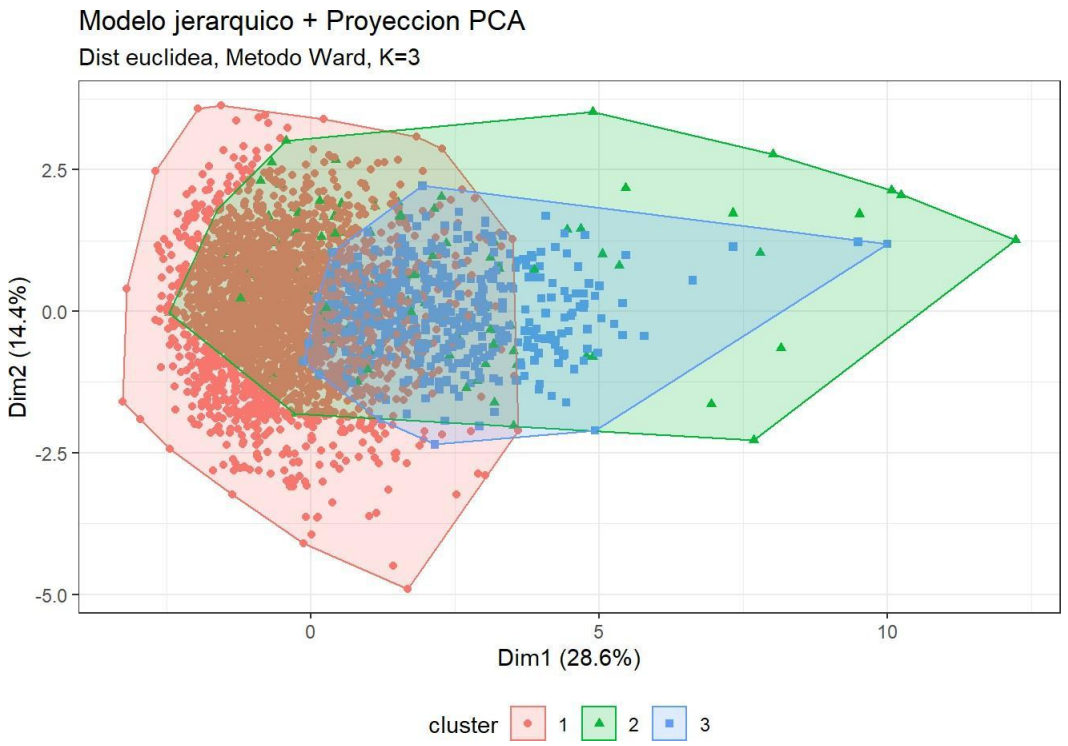
k = 2

```
fviz_cluster(object = list(data=datos, cluster=grupos1), stand = FALSE, ellipse.type = "convex", geom = "point", show.
clust.cent = FALSE,labelsize = 8) + labs(title = "Modelo jerarquico + Proyeccion PCA",subtitle = "Dist euclidea, Meto
do Ward, K=2") + theme_bw() + theme(legend.position = "bottom")
```



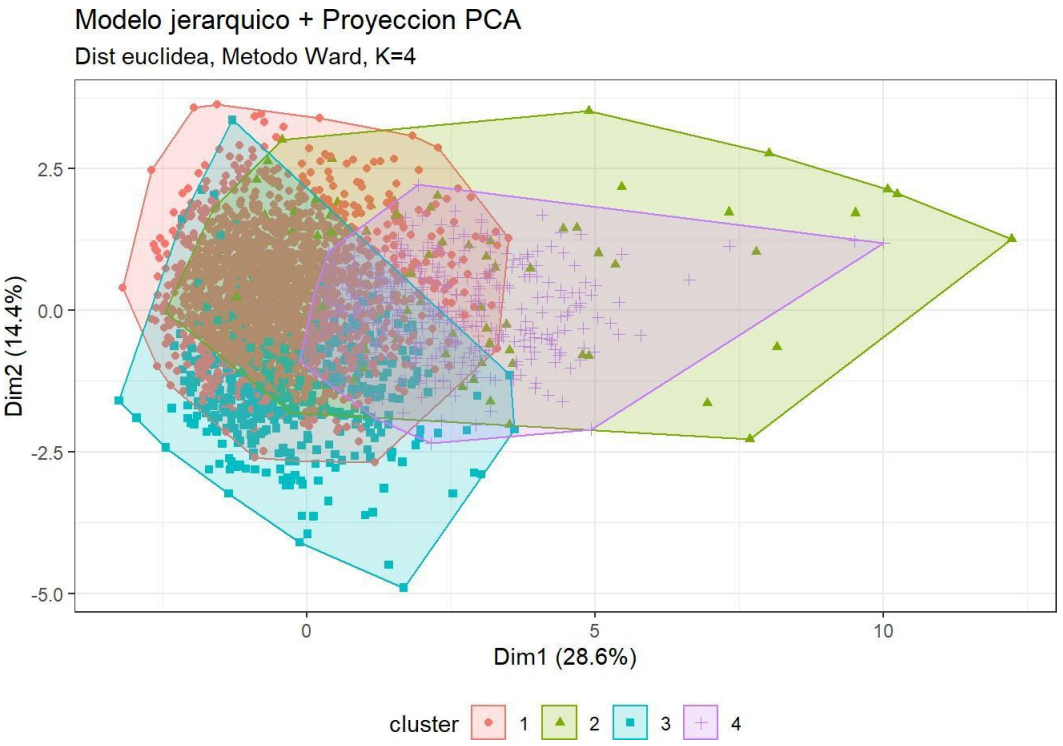
k = 3

```
fviz_cluster(object = list(data=datos, cluster=grupos2), stand = FALSE, ellipse.type = "convex", geom = "point", show.
clust.cent = FALSE,labelsize = 8) + labs(title = "Modelo jerarquico + Proyeccion PCA",subtitle = "Dist euclidea, Meto
do Ward, K=3") + theme_bw() + theme(legend.position = "bottom")
```



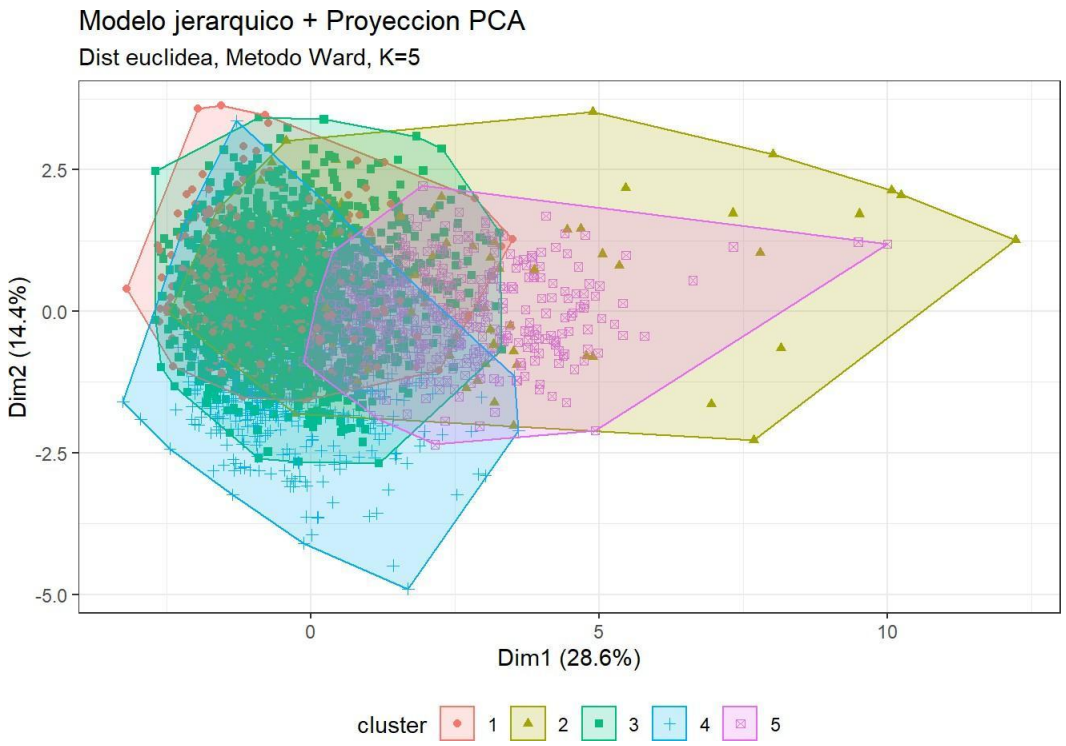
k = 4

```
fviz_cluster(object = list(data=datos, cluster=grupos3), stand = FALSE, ellipse.type = "convex", geom = "point", show.
clust.cent = FALSE,labelsize = 8) + labs(title = "Modelo jerarquico + Proyeccion PCA",subtitle = "Dist euclidea, Meto
do Ward, K=4") + theme_bw() + theme(legend.position = "bottom")
```



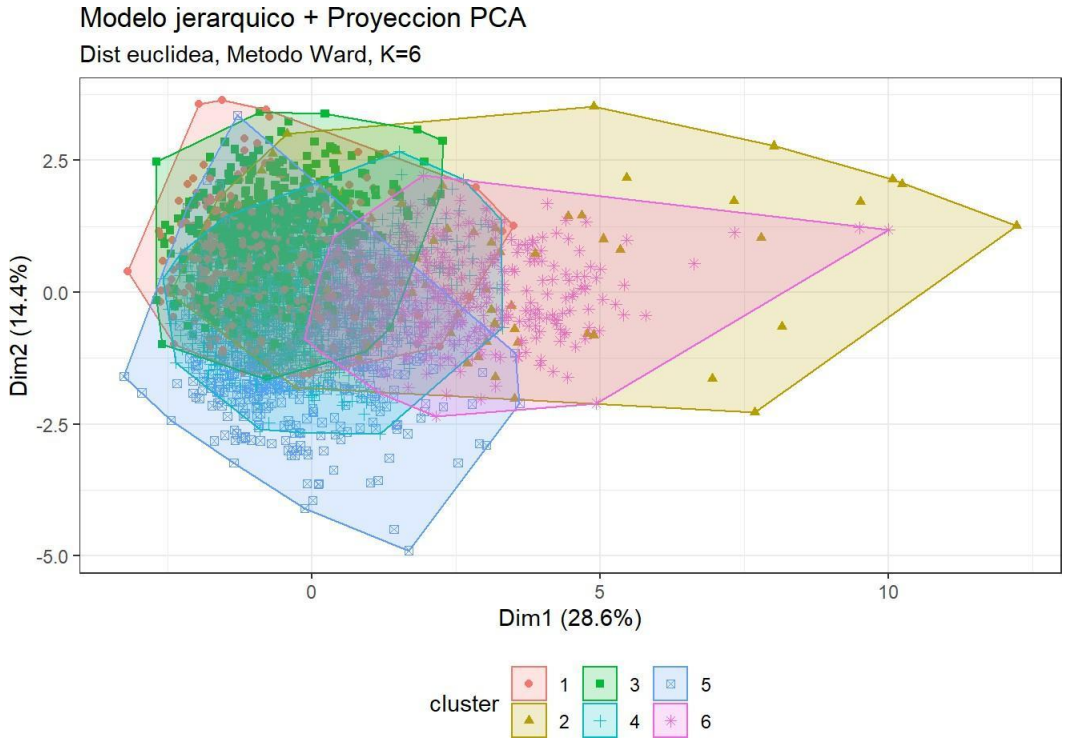
k = 5

```
fviz_cluster(object = list(data=datos, cluster=grupos4), stand = FALSE, ellipse.type = "convex", geom = "point", show.
clust.cent = FALSE,labelsize = 8) + labs(title = "Modelo jerarquico + Proyeccion PCA",subtitle = "Dist euclidea, Meto
do Ward, K=5") + theme_bw() + theme(legend.position = "bottom")
```



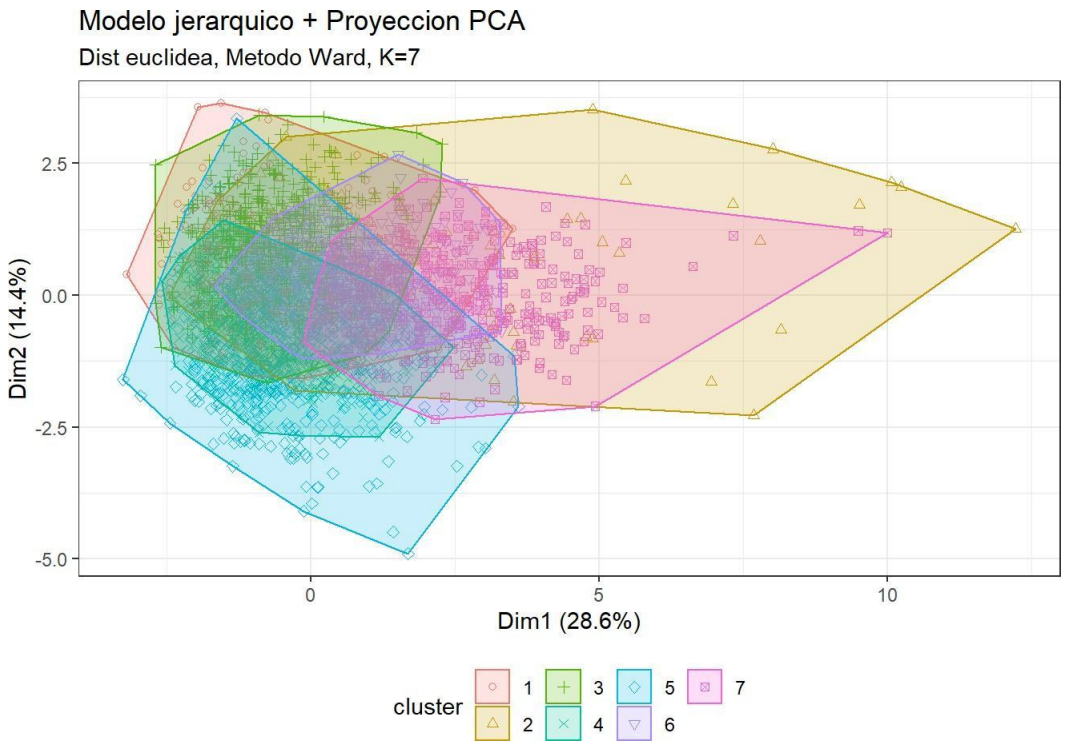
k = 6

```
fviz_cluster(object = list(data=datos, cluster=grupos5), stand = FALSE, ellipse.type = "convex", geom = "point", show.
clust.cent = FALSE,labelsize = 8) + labs(title = "Modelo jerarquico + Proyeccion PCA",subtitle = "Dist euclidea, Meto
do Ward, K=6") + theme_bw() + theme(legend.position = "bottom")
```



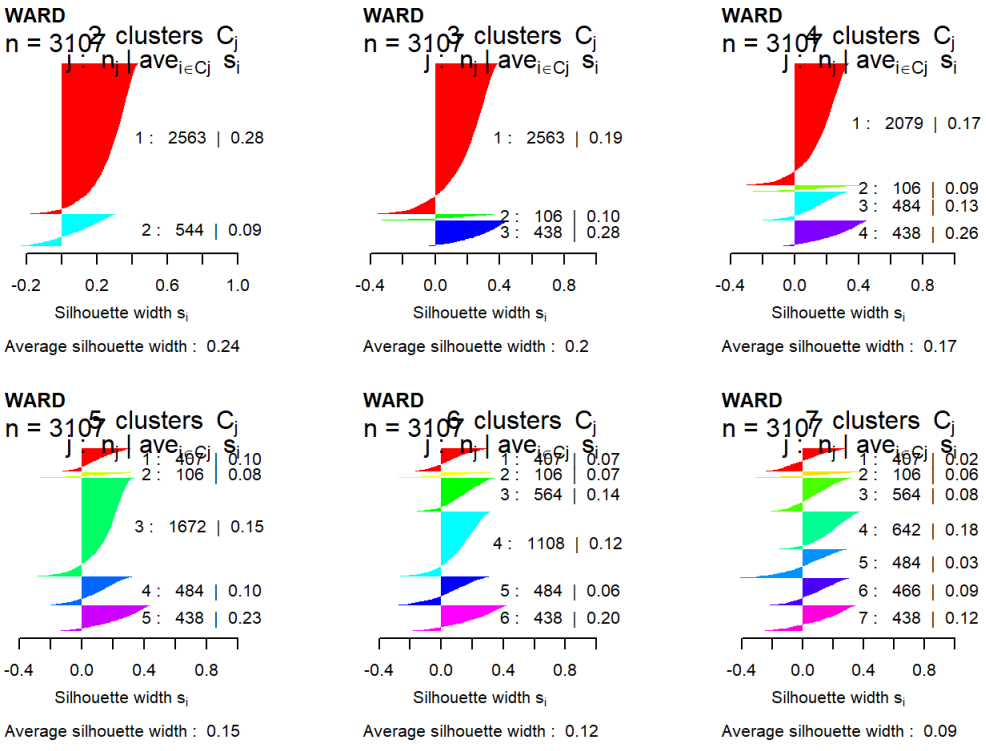
k = 7

```
fviz_cluster(object = list(data=datos, cluster=grupos6), stand = FALSE, ellipse.type = "convex", geom = "point", show.
clust.cent = FALSE,labelsize = 8) + labs(title = "Modelo jerarquico + Proyeccion PCA",subtitle = "Dist euclidea, Meto
do Ward, K=7") + theme_bw() + theme(legend.position = "bottom")
```



Por último, observaremos el coeficiente de Silhouette para cada individuo en cada uno de los grupos creados con diferente número de clusters.

```
par(mfrow = c(2,3))
plot(silhouette(grupos1, midist), col=rainbow(2), border=NA, main =
"WARD") plot(silhouette(grupos2, midist), col=rainbow(3), border=NA, main
= "WARD") plot(silhouette(grupos3, midist), col=rainbow(4), border=NA,
main = "WARD") plot(silhouette(grupos4, midist), col=rainbow(5),
border=NA, main = "WARD") plot(silhouette(grupos5, midist),
col=rainbow(6), border=NA, main = "WARD") plot(silhouette(grupos6,
midist), col=rainbow(7), border=NA, main = "WARD")
```



Como conclusión se puede ver que 3 es el número de cluster con menos individuos mal asignados. Por lo que si utilizásemos este método emplearíamos 3 clusters. No obstante, como hemos visto en la proyección PCA, Ward no nos ayuda a separar los grupos. De todas maneras, nos sirve para tener una idea de en qué número fijar los clusters si aplicamos un método no jerárquico.