Anexo X. Clustering: algoritmo de k-medias

Carga y preparación de datos

Cargamos los datos, escogemos las variables numéricas, las escalamos, y por problemas de espacio en memoria RAM seleccionamos aquellos con popularidad mayor a 70:

```
load("C:/DANIEL/UNIVERSIDAD/CD2/PROY II/Proyecto_R/Datos/ncdata.RData")
load("C:/DANIEL/UNIVERSIDAD/CD2/PROY II/Proyecto_R/Datos/desc_data.RData")

songs = ncdata[desc_data$type == "numerical" | desc_data$type == "binary"]
songs = songs[,-length(colnames(songs))]
songs = songs[,-14]
songs = songs[songs$popularity > 70,]
songs = scale(songs)
```

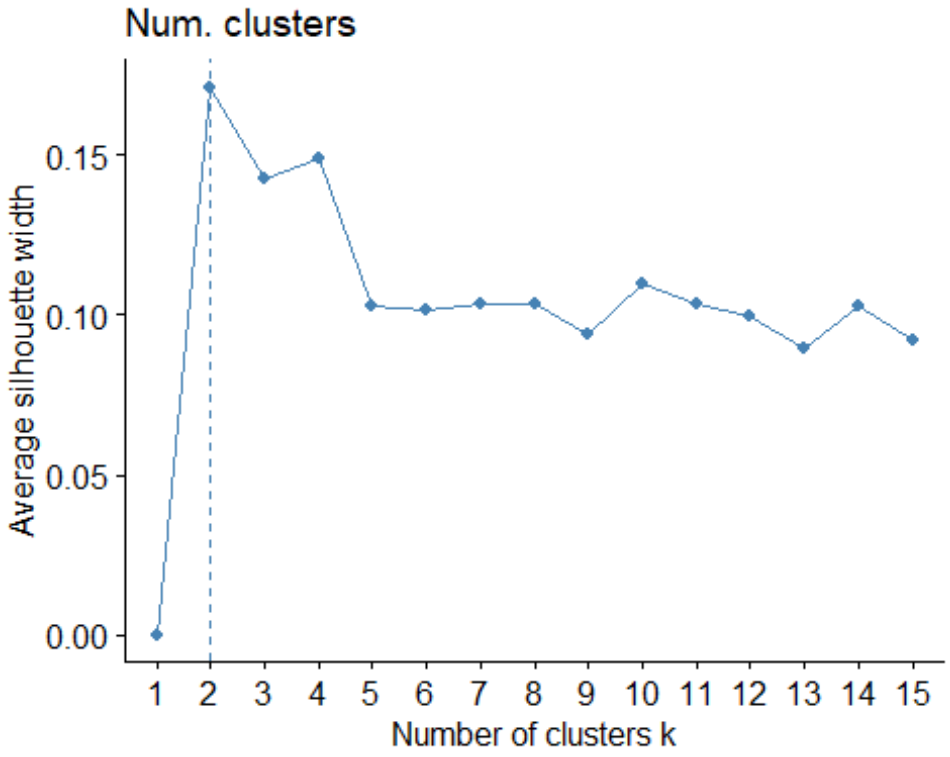
Vamos a obtener la matriz de distancias. En este caso, como queremos canciones similares, tiene sentido que intentemos agrupar las canciones que tengan valores cercanos en las diferentes variables (acústica, ruido, tempo, etc.). Por lo tanto, vamos a elegir como medida de distancia la distancia euclídea:

```
dist_matrix = get_dist(songs, method = "euclidean")
```

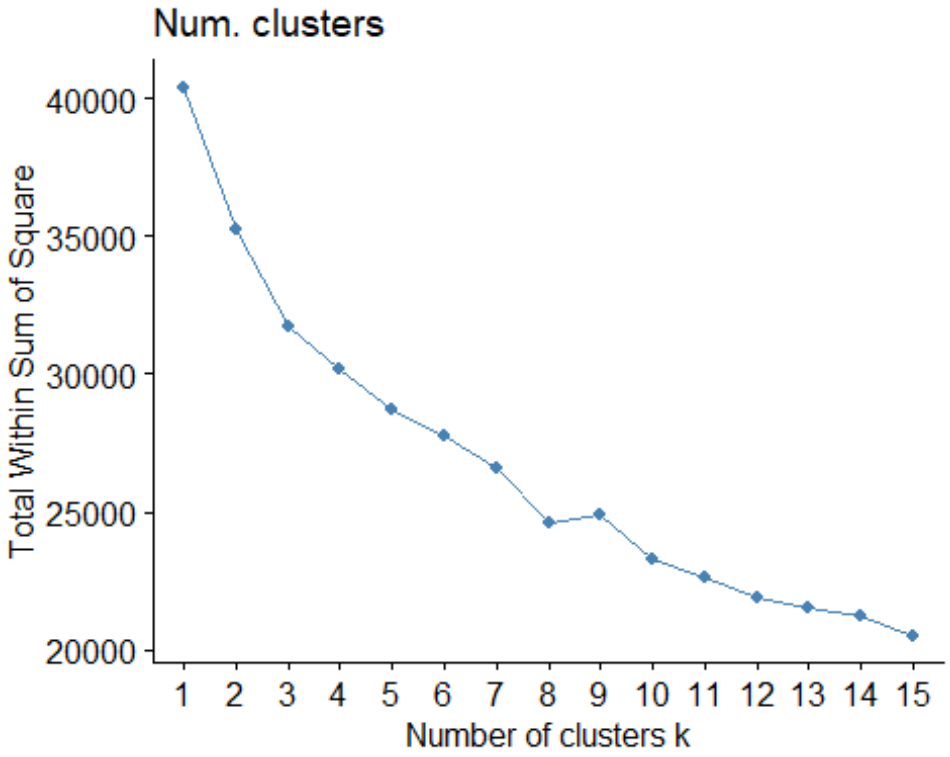
Nº óptimo de clusters para el algoritmo k-medias

Una vez tenemos la matriz de distancias, vamos a determinar mediante el coeficiente de Silhouette y la suma de cuadrados intracluster el nº óptimo de clusters para el algoritmo de k-medias:

```
fviz_nbclust(x = songs, FUNcluster = kmeans, method = "silhouette",
             k.max = 15, verbose = FALSE) +
  labs(title = "Num. clusters")
```

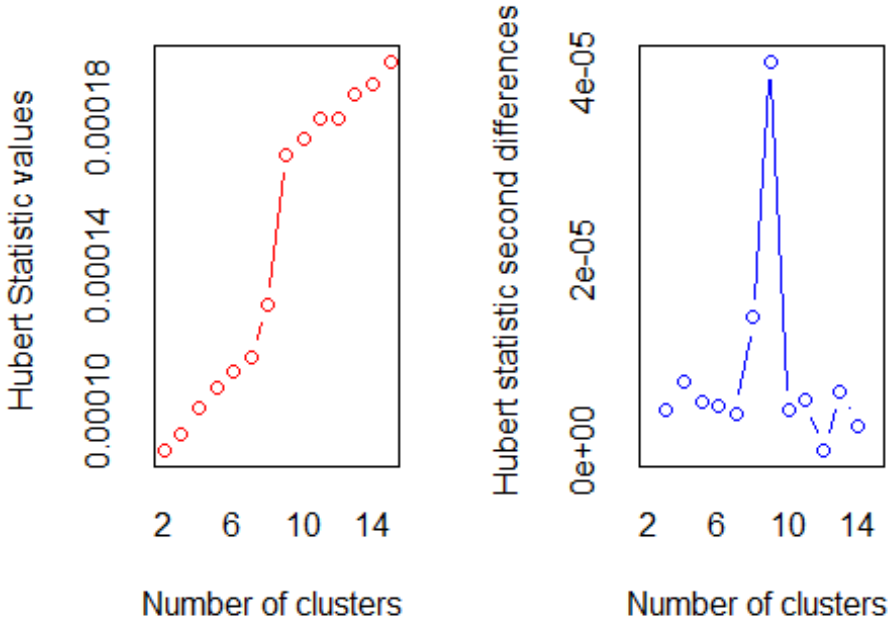



```
fviz_nbclust(x = songs, FUNcluster = kmeans, method = "wss",
             k.max = 15, verbose = FALSE) +
  labs(title = "Num. clusters")
```

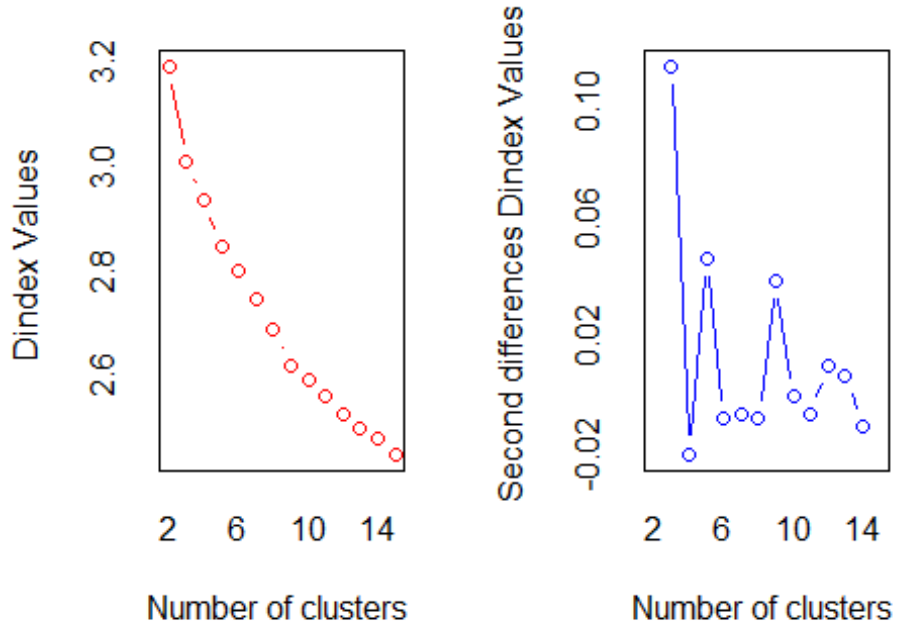


Parece que los dos criterios coinciden en 2 como el nº óptimo de cluster, no obstante vamos a probar todos los métodos posibles a ver qué resultados obtenemos. Esto lo hacemos debido a que, conociendo la naturaleza de nuestros datos y el objetivo que buscamos con esto, no buscamos agrupar en 2 grandes grupos nuestras canciones, si no crear grupos más pequeños compuestos por canciones del mismo género. Así pues, tras aplicar el algoritmo de clúster calculando cada vez un número de grupos diferentes obtenemos los siguientes resultados:

```
res.nbclust = NbClust(data = songs, diss = dist_matrix, distance = NULL,
                      min.nc = 2, method = "kmeans", index ="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:
## * 6 proposed 2 as the best number of clusters
## * 9 proposed 3 as the best number of clusters
## * 1 proposed 8 as the best number of clusters
## * 4 proposed 9 as the best number of clusters
## * 3 proposed 15 as the best number of clusters
##
##                    ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  3
##
##
## *******************************************************************
```

Vamos a probar con 3, 4, 6 y 9 clusters porque en los diagramas anteriores parece que aunque no es el número óptimo de agrupaciones, los estadísticos no son muy desfavorables.

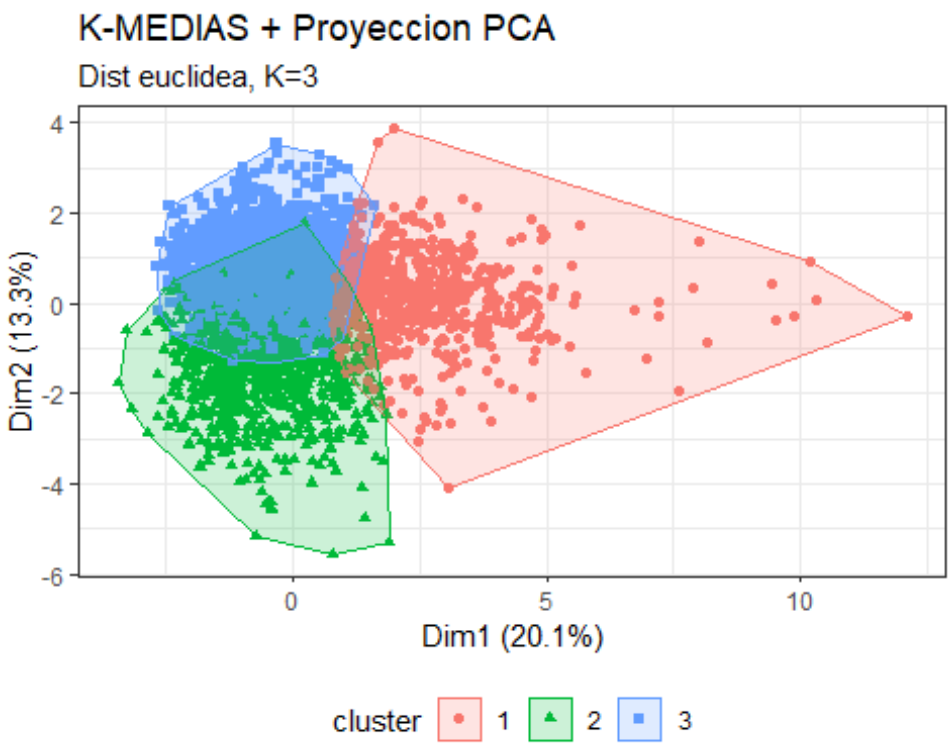
Realización del Clustering

```
set.seed(0)
clustering_3 <- kmeans(songs, centers = 3, nstart = 50, iter.max = 50)
clustering_4 <- kmeans(songs, centers = 4, nstart = 50, iter.max = 50)
clustering_6 <- kmeans(songs, centers = 6, nstart = 50, iter.max = 50)
clustering_9 <- kmeans(songs, centers = 9, nstart = 50, iter.max = 50)
table(clustering_3$cluster)

##
##    1    2    3
##  585  951 1571

table(clustering_4$cluster)

##
##    1    2    3    4
##  600   79 1500  928

table(clustering_6$cluster)

##
##   1   2   3   4   5   6
## 909 491 631 353  74 649

table(clustering_9$cluster)

##
##   1   2   3   4   5   6   7   8   9
## 637  73 209 532 371 292 341 480 172
```

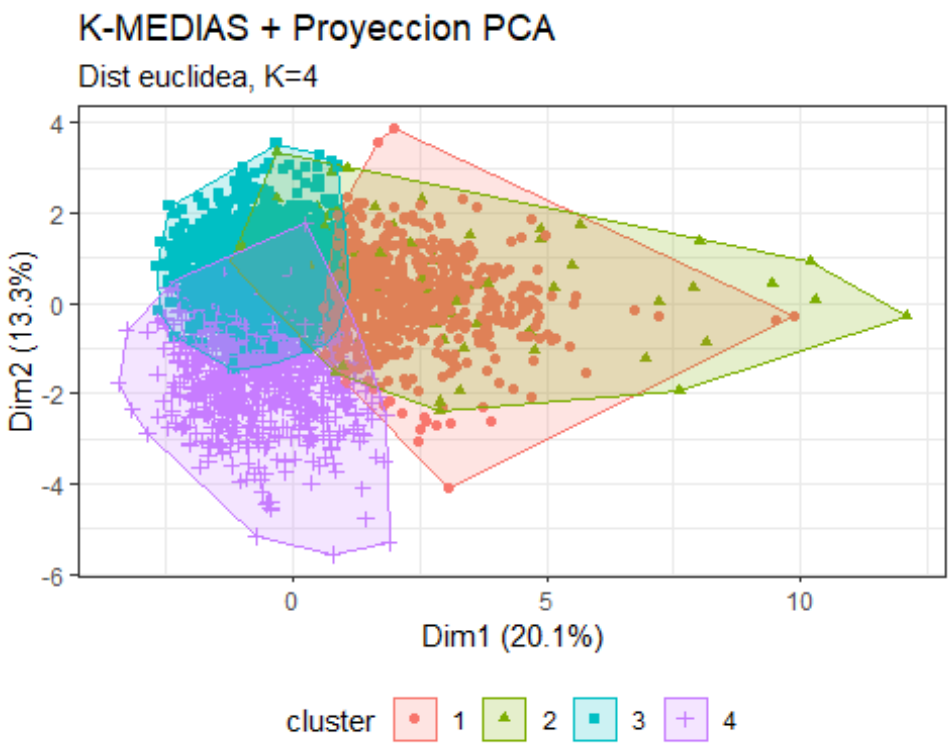
Resultados y validación

En primer lugar vamos a visualizar los resultados con los gráficos de scores en las 2 primeras componentes principales.

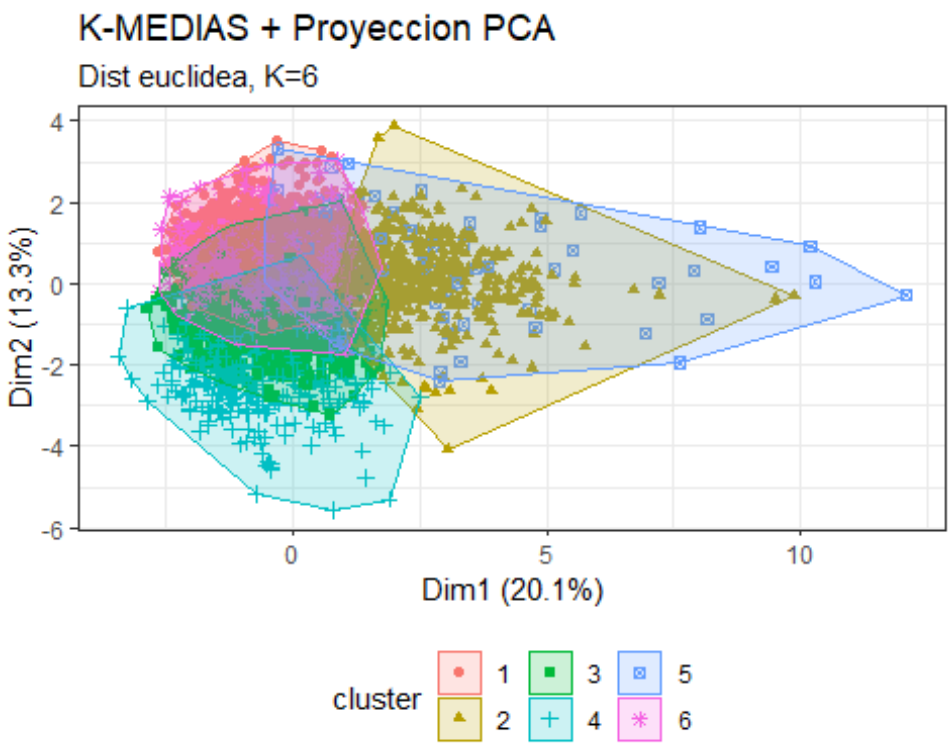
```
fviz_cluster(object = list(data=songs, cluster=clustering_3$cluster), stand =
FALSE,
             ellipse.type = "convex", geom = "point", show.clust.cent =
FALSE,
             labelsize = 8)  +
  labs(title = "K-MEDIAS + Proyeccion PCA",
       subtitle = "Dist euclidea, K=3") +
  theme_bw() +
  theme(legend.position = "bottom")
```



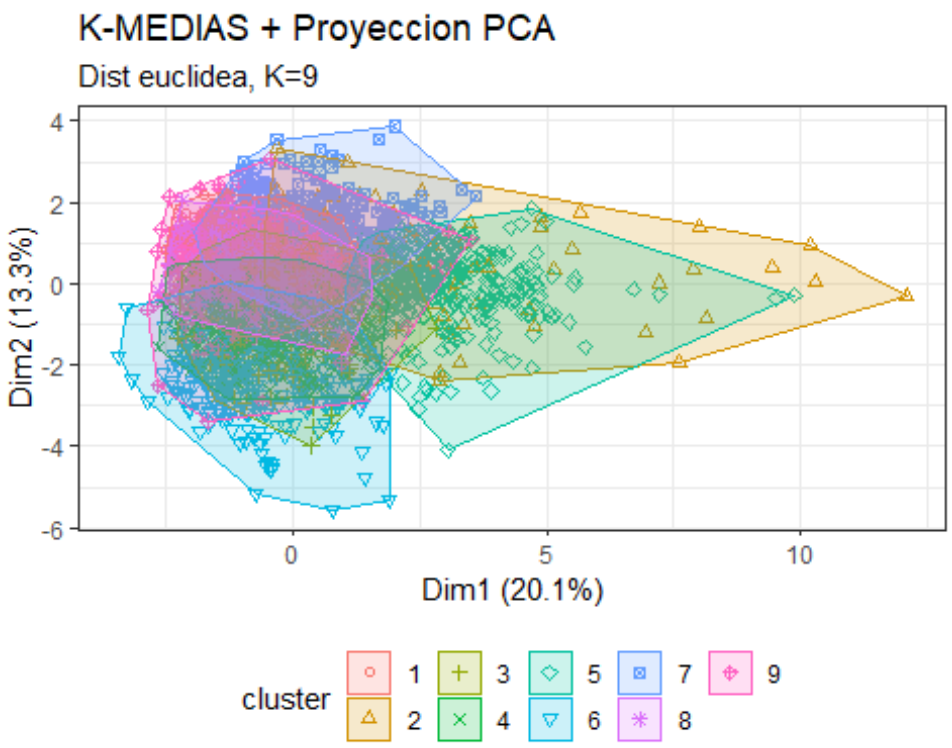

```
fviz_cluster(object = list(data=songs, cluster=clustering_4$cluster), stand =
FALSE,
             ellipse.type = "convex", geom = "point", show.clust.cent =
FALSE,
             labelsize = 8)  +
  labs(title = "K-MEDIAS + Proyeccion PCA",
       subtitle = "Dist euclidea, K=4") +
  theme_bw() +
  theme(legend.position = "bottom")
```




```
fviz_cluster(object = list(data=songs, cluster=clustering_6$cluster), stand =
FALSE,
             ellipse.type = "convex", geom = "point", show.clust.cent =
FALSE,
             labelsize = 8)  +
  labs(title = "K-MEDIAS + Proyeccion PCA",
       subtitle = "Dist euclidea, K=6") +
  theme_bw() +
  theme(legend.position = "bottom")
```

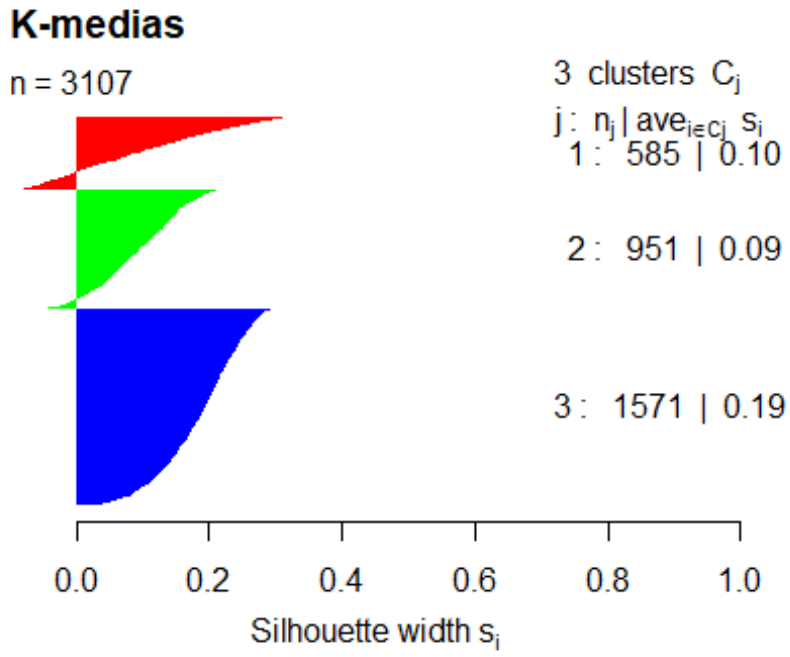



```
fviz_cluster(object = list(data=songs, cluster=clustering_9$cluster), stand =
FALSE,
             ellipse.type = "convex", geom = "point", show.clust.cent =
FALSE,
             labelsize = 8)  +
  labs(title = "K-MEDIAS + Proyeccion PCA",
       subtitle = "Dist euclidea, K=9") +
  theme_bw() +
  theme(legend.position = "bottom")
```

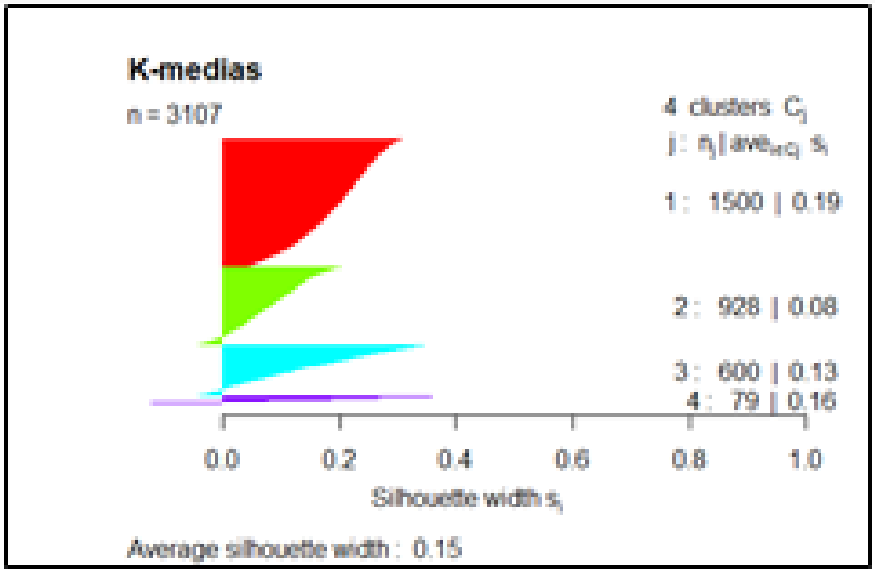


A la vista de los gráficos obtenidos, parece ser que la mejor agrupación se hace en 3 grandes grupos, ya que al aumentar el número de clusters hay mucho solape entre ellos lo que indica que hay muchas canciones mal clasificadas. Esto lo podemos comprobar con los coeficientes de Silhouette:

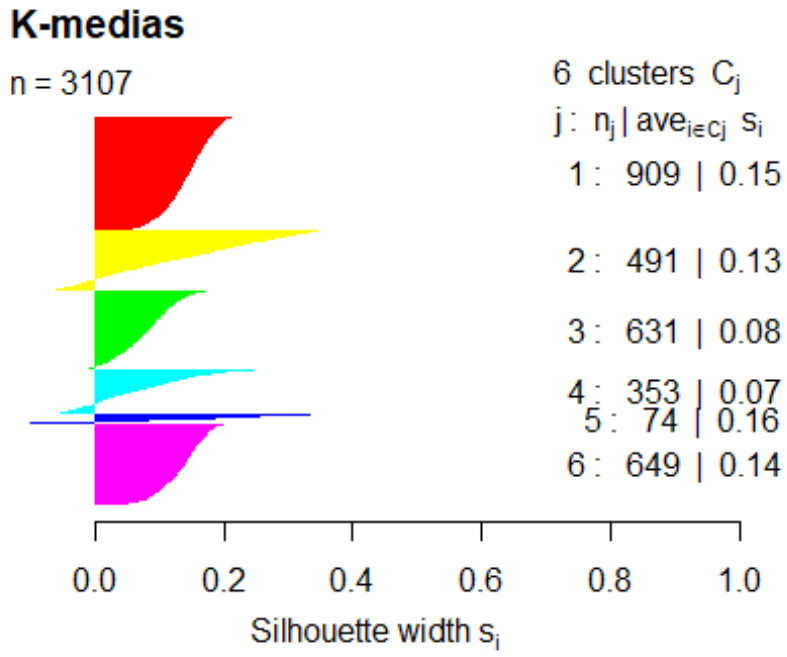
```
plot(silhouette(clustering_3$cluster, dist_matrix), col=rainbow(3),
border=NA, main = "K-medias")
```



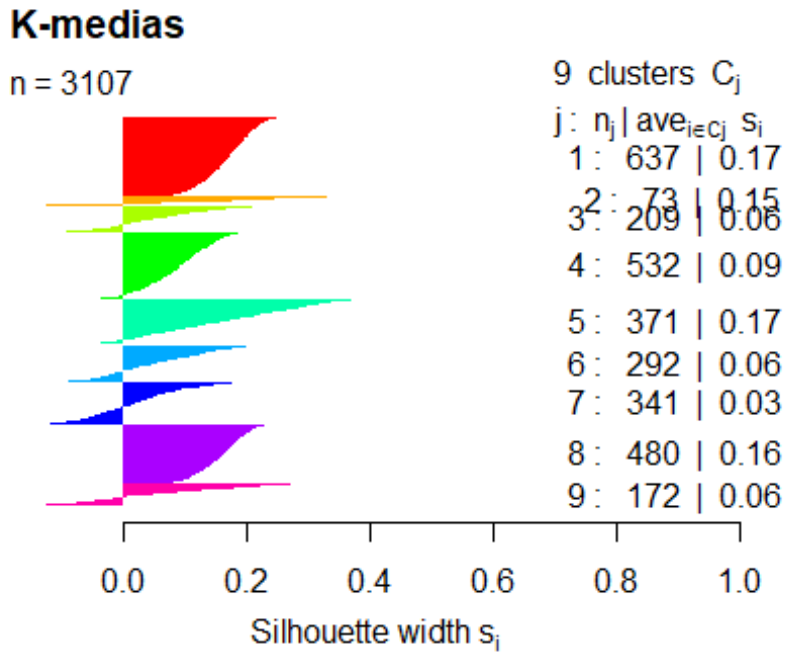

```
plot(silhouette(clustering_5$cluster, dist_matrix), col=rainbow(5),
border=NA, main = "K-medias")
```




```
plot(silhouette(clustering_6$cluster, dist_matrix), col=rainbow(6),
border=NA, main = "K-medias")
```




```
plot(silhouette(clustering_9$cluster, dist_matrix), col = rainbow(9),
border=NA, main = "K-medias")
```



Conclusiones

A la vista de los gráficos, conforme aumentamos el número de clusters aumentan dramáticamente las canciones mal clasificadas (coeficiente de Silhouette negativo). Es por ello que hacer el clustering con más de 5 centroides nos proporciona un resultado bastante pobre.

Sin embargo, con solo 3 o 4 grupos no podemos hacer recomendaciones ya que estos grupos tienen canciones muy variadas. En los siguientes anexos veremos qué soluciones podemos proponer a estos problemas ya que, como hemos comentado en anteriores ocasiones, estamos muy restringidos en términos de software.