Anexo XII. Clustering con variables incorrelacionadas

Carga y filtrado de los datos

Cargamos tanto ncdata como desc_data para poder trabajar fácilmente con las variables que queramos:

```
load('ncdata.RData')
load('desc_data.RData')
```

Ahora filtramos los datos, en este ejemplo vamos a seleccionar las canciones que:

- Duran entre 30 segundos y 5 minutos
- Tienen una popularidad mayor que 70

Además, solo vamos a seleccionar algunas variables, ya que como algunas están muy relacionadas pueden dificultar el análisis clustering, por tanto este análisis lo vamos a realizar teniendo en cuenta las variables:

- Acousticness -Instrumentalness -Speechiness -Loudness

Aplicando este filtrado, nuestro conjunto de datos sobre el que realizar el clustering sería el siguiente:

```
songs = ncdata[ncdata$duration_ms > 18000,]
songs = songs[songs$duration_ms < 300000,]
songs = songs[songs$popularity > 70,]
songs = songs[,c('acousticness','instrumentalness','speechiness','loudness')]
songs=scale(songs)

aux_data = ncdata[ncdata$duration_ms > 18000,]
aux_data = aux_data[aux_data$duration_ms < 300000,]
aux_data = aux_data[aux_data$popularity > 70,]
```

Como podemos ver, se nos queda un data set de alrededor de 3000 canciones y las 4 variables que hemos indicado en el párrafo anterior. Ahora ya podemos pasar a realizar el análisis clustering con estos datos.

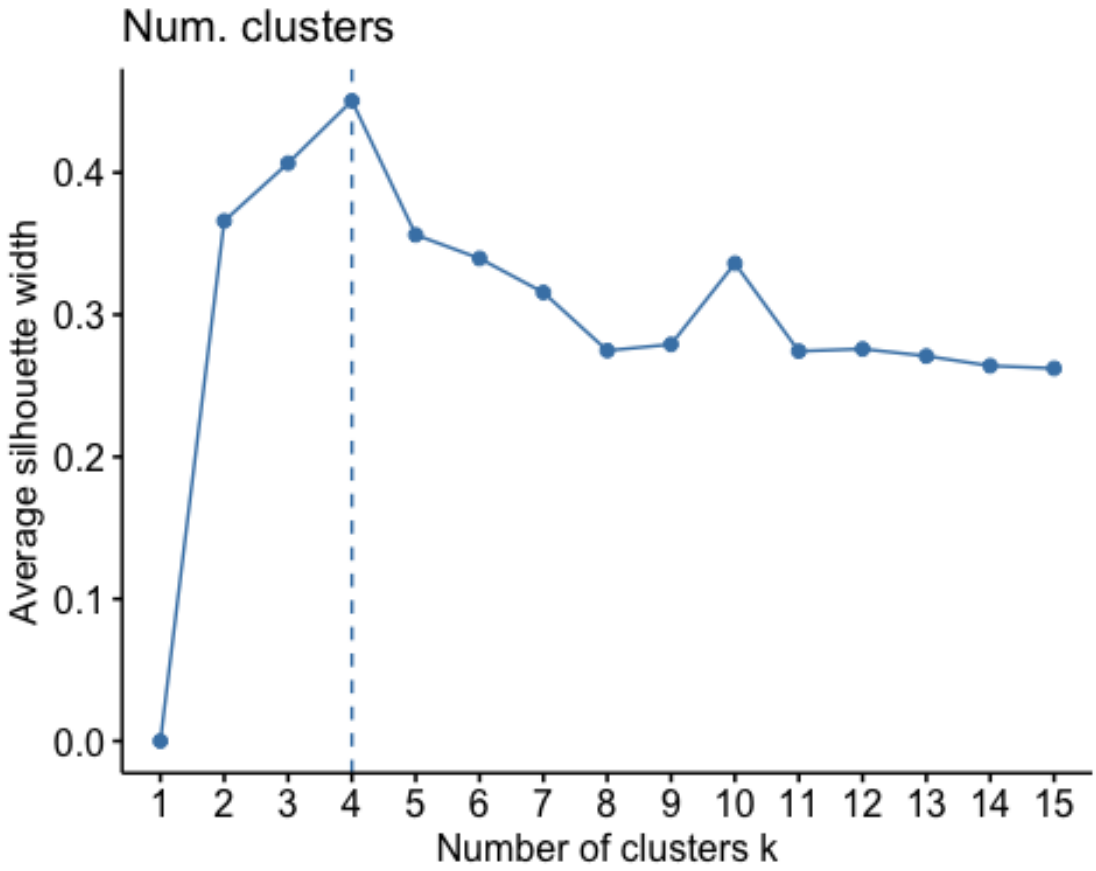
Análisis clustering - Elección de clusters

En esta sección vamos a ver en cuántos clusters dividir nuestras canciones y qué método utilizar para ello. En primer lugar vamos a calcular la matriz de distancias y graficar los coeficientes de Silhouette y la suma de cuadrados intracluster para ver cual es el número óptimo. El coeficiente de Silhouette nos permite cuantificar cómo de buena es la asignación que se ha hecho de los elementos en los clusters, comparando la similitud de cada elemento con el resto de su cluster frente a los de otros clusters. Este número oscila entre -1 y 1,

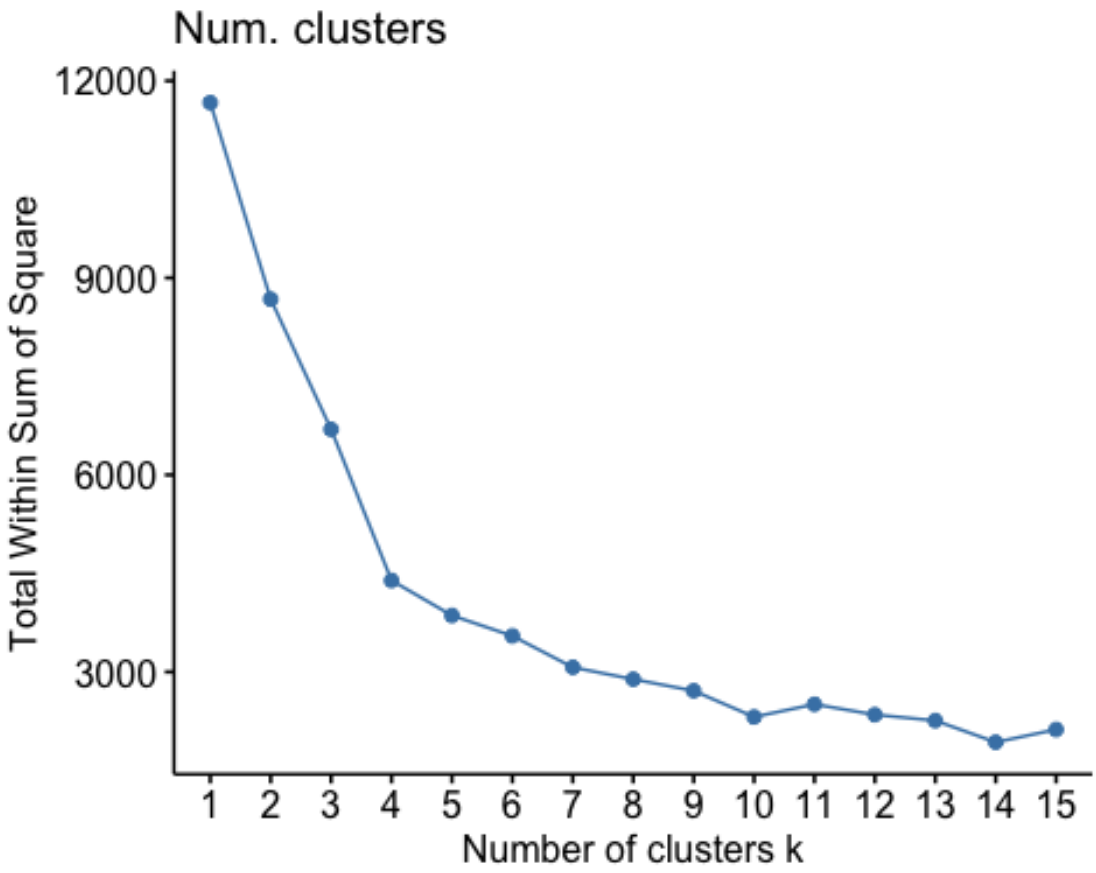
siendo 1 undicativo de una asignación correcta. Luego cuanto mayor sea el valor mejor asignación se habrá realizado en promedio.

Por otro lado, con el gráfico de la Suma de Cuadrados intra-cluster queremos un valor lo menor posible, siempre evitando aumentar innecesariamente el número de cluster cuando la variación es mínima. Luego, en este caso buscamos maximizar la homogeneidad intra-cluster, es decir, obtener la menor varianza posible dentro de cada cluster.

```
midist = get_dist(songs, method = "euclidean")
distancias=as.data.frame(as.matrix(midist),row.names=aux_data$id)
colnames(distancias)=aux_data$id
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")
```



Observando ambos gráficos, llegamos a la misma conclusión, el número óptimo de clusters para nuestro conjunto de datos es 4. Por tanto, nuestro análisis separará en 4 grandes grupos nuestras canciones en base a sus atributos musicales.

Vamos a utilizar el método k-medias, ya que es el método que más se suele utilizar en los casos en los que ya sabemos el número de clústers en los que separar los datos.

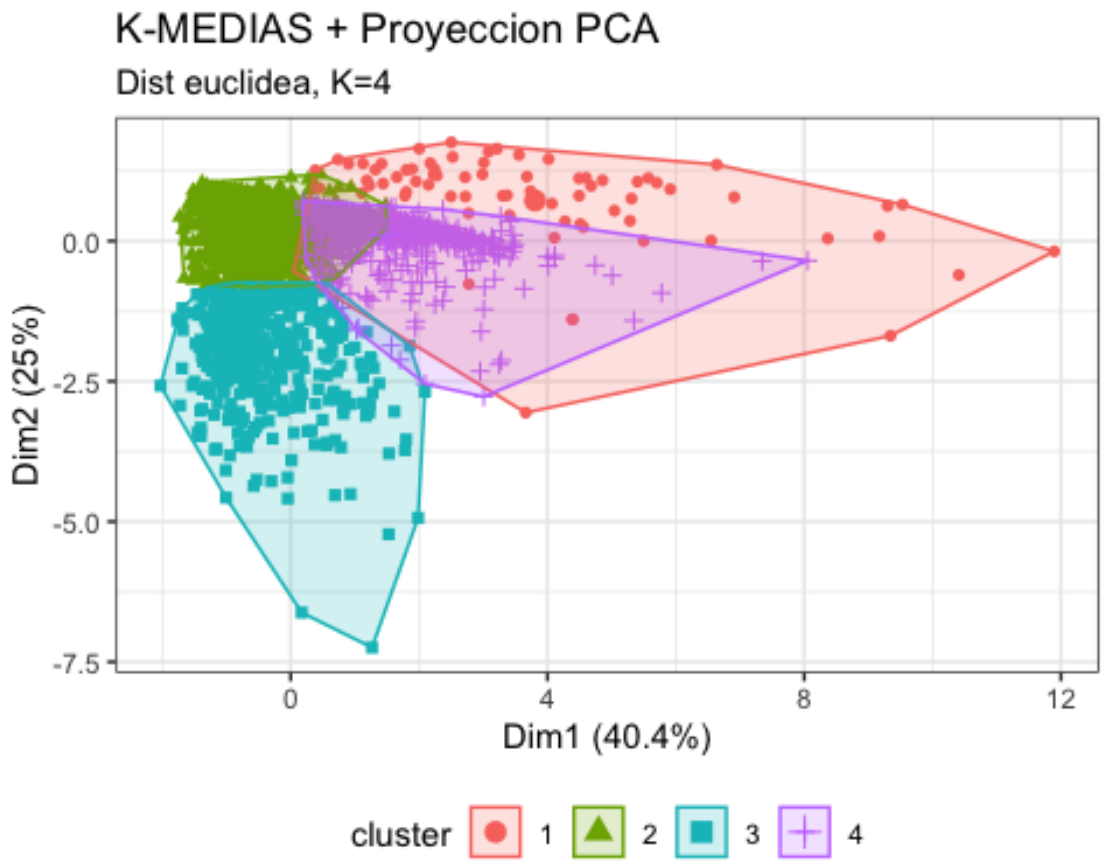
4 Clusters

```
set.seed(12710)
clustering_4 <- kmeans(songs, centers = 4, nstart = 50, iter.max = 50)
table(clustering_4$cluster)

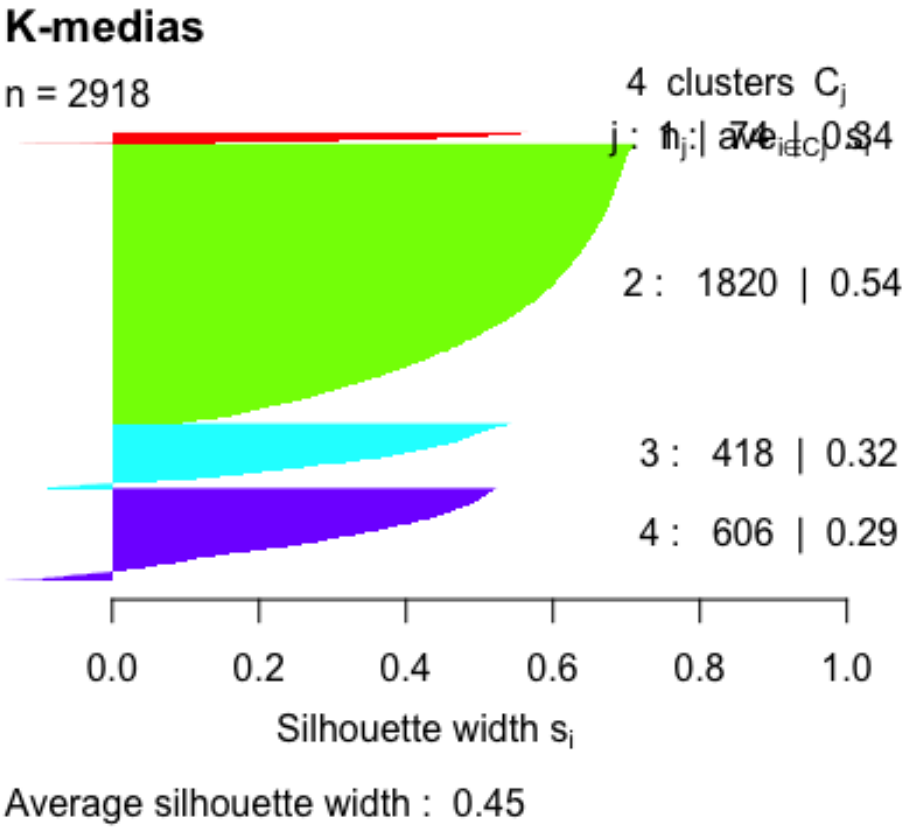
##
##    1    2    3    4
##   74 1820  418  606
```

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

```
    theme_bw() +
    theme(legend.position = "bottom")
```




```
plot(silhouette(clustering_4$cluster, midist), col=rainbow(4), border=NA,
main = "K-medias")
```



Ahora vamos a ver la formación de los clusters, en el ejemplo de k=4:

```
c1 = aux_data[which(clustering_4$cluster == 1),]
c2 = aux_data[which(clustering_4$cluster == 2),]
c3 = aux_data[which(clustering_4$cluster == 3),]
c4 = aux_data[which(clustering_4$cluster == 4),]
c1
```

```
## # A tibble: 74 x 26
##    acousticness artists danceability duration_ms energy explicit id
##           <dbl> <chr>          <dbl>       <dbl>  <dbl>    <dbl> <chr>
##  1    0.0000151 Gorill…        0.689      233867  0.739        0 0q6L…
##  2    0.0000416 Surf C…        0.346      147036  0.944        0 0HUT…
##  3    0.0000559 The Sm…        0.404      258467  0.72         0 6GtX…
##  4    0.000132  Route …        0.814      259934  0.622        0 61UQ…
##  5    0.000202  U2            0.54       295516  0.429        0 6ADS…
##  6    0.000269  Ramones       0.385      134467  0.783        0 4KcH…
##  7    0.000547  Lipps …        0.906      239253  0.63         1 7723…
##  8    0.000563  The St…        0.486      219754  0.666        0 57Xj…
##  9    0.000638  Soda S…        0.536      212747  0.749        0 2lpI…
## 10    0.000779  Marily…        0.621      218827  0.834        1 2aIB…
## # … with 64 more rows, and 19 more variables: instrumentalness <dbl>,
## #   key <dbl>, liveness <dbl>, loudness <dbl>, mode <dbl>, name <chr>,
```

```
## #   popularity <dbl>, release_date <chr>, speechiness <dbl>, tempo <dbl>,
## #   valence <dbl>, year <dbl>, album <chr>, album_id <chr>, artist_ids
<chr>,
## #   track_number <dbl>, disc_number <dbl>, time_signature <dbl>,
## #   classical <dbl>
```

```
c2
```

```
## # A tibble: 1,820 x 26
##    acousticness artists danceability duration_ms energy explicit id
##           <dbl> <chr>          <dbl>       <dbl>  <dbl>    <dbl> <chr>
##  1    0.0000105 KANABO…        0.436      240133  0.934        0 21z1…
##  2    0.0000183 Foo Fi…        0.465      235293  0.919        0 5OQs…
##  3    0.0000223 Korn           0.353      255733  0.898        1 6W21…
##  4    0.0000264 Green …        0.38       176346  0.988        1 6nTi…
## # … with 1,810 more rows, and 19 more variables: instrumentalness <dbl>,
## #   key <dbl>, liveness <dbl>, loudness <dbl>, mode <dbl>, name <chr>,
## #   popularity <dbl>, release_date <chr>, speechiness <dbl>, tempo <dbl>,
## #   valence <dbl>, year <dbl>, album <chr>, album_id <chr>, artist_ids
<chr>,
## #   track_number <dbl>, disc_number <dbl>, time_signature <dbl>,
## #   classical <dbl>
```

```
c3
```

```
## # A tibble: 418 x 26
##    acousticness artists danceability duration_ms energy explicit id
##           <dbl> <chr>          <dbl>       <dbl>  <dbl>    <dbl> <chr>
##  1     0.000107 Drake          0.766      181573  0.442        1 4Kz4…
##  2     0.000248 Kanye …        0.529      104591  0.9          0 2QpG…
##  3     0.000677 Saweet…        0.899      126446  0.811        1 5KBA…
##  4     0.000813 Jay Ro…        0.645      229670  0.705        1 51rX…
## # … with 596 more rows, and 19 more variables: instrumentalness <dbl>,
## #   key <dbl>, liveness <dbl>, loudness <dbl>, mode <dbl>, name <chr>,
## #   popularity <dbl>, release_date <chr>, speechiness <dbl>, tempo <dbl>,
## #   valence <dbl>, year <dbl>, album <chr>, album_id <chr>, artist_ids
<chr>,
## #   track_number <dbl>, disc_number <dbl>, time_signature <dbl>,
## #   classical <dbl>
```

Y también podemos ver los centroides de cada cluster:

```
clustering_4$centers
```

```
##   acousticness instrumentalness speechiness     loudness
## 1    0.6842331        5.7777741  -0.3187100 -1.346870413
## 2   -0.4733182       -0.1550215  -0.3237321  0.360017132
## 3   -0.1520269       -0.1737853   2.0942236  0.001440251
## 4    1.4428269       -0.1200889  -0.4333475 -0.917763687
```

Viendo la tabla de los centroides de los grupos, podemos hacer una descripción de las canciones que están dentro de cada grupo:

- Grupo 1: este grupo destaca por su alto valor de 'instrumentalness', es decir, son pistas de audio con poco contenido vocal, con bases musicales muy marcadas.
- Grupo 2: en este grupo las 4 características tienen valores cercanos a 0, no hay ninguna que sobre salga por encima del resto. En este grupo agrupamos las canciones que tienen tanto una base instrumental como vocal.
- Grupo 3: este grupo representa todo lo contrario al grupo 1. En este caso, agrupamos las pistas con fuerte contenido vocal. Según la descripción de la API de Spotify, 'cuanto más similar a un discurso sea la canción, más alto será su valor en esta variable'. Es por esto que encontramos aquí grandes éxitos de rap y trap estadounidense o latino.
- Grupo 4: en este grupo están representadas las pistas con alto contenido acústico. Además, estas pistas tienden a tener valores bajos en 'loudness', lo que implica que en general son canciones poco ruidosas o estridentes.

En conclusión, seleccionando para el clúster sólo 4 variables con proyecciones no correlacionadas en el PCA sí que hemos conseguido una separación que puede servirnos para recomendar canciones similares. Así, tenemos 4 grupos de canciones con rasgos principales claramente diferentes, lo que nos ayuda a discriminar entre gustos musicales. Con un software más potente quizás podríamos haber calculado grupos más específicos, pero dadas las restricciones que tenemos y el tiempo limitado del proyecto no hemos podido profundizar tanto en este análisis como nos hubiera gustado.