Análisis clúster k-medias.

Segmentando las empresas eólicas.

Vamos a considerar una serie de 4 variables que caracterizan a un grupo de 100 empresas de producción eléctrica mediante tecnología eólica. Nuestro objetivo es segmentar este conjunto de empresas, haciendo grupos homogéneos (conglomerados), y caracterizando a dichos grupos; de modo que se establezca una serie de tipologías de empresas con unas características definidas. Es decir, no se pretende tanto agrupar estos casos en particular como utilizar estas empresas concretas a modo de muestra para dividir toda la población de empresas eólicas españolas en distintas subpoblaciones con características diferentes.

Las variables clasificadoras son: el grado de liquidez (LIQUIDEZ), la rentabilidad económica (RENECO), la rentabilidad financiera (RENFIN), y la solvencia (SOLVENCIA).

Vamos a utilizar el método no-jerárquico de agrupación de k-medias, que es el más extendido en este tipo de análisis.

Preparando Datos.

Abriremos R-Studio y crearemos nuestro **proyecto** siguiendo la instrucción File → New Project. Luego nos preguntará si crea el proyecto en una nueva carpeta o en una ya existente. Vamos a crearlo, por ejemplo, en el disco extraíble D, carpeta R, subcarpeta "kmedias", que ya está creada. Nos saldrá una ventana para buscar la carpeta y, cuando la encontremos, pulsamos Open y Create Project.

Vamos a ir a la carpeta del proyecto y vamos a guardar en ella los dos archivos de esta práctica: un archivo de Microsoft® Excel® llamado "eolica_100_mv.xlsx" y un *script* denominado "kmedias_eolica.R". Si abrimos el archivo de Microsoft® Excel®, comprobaremos que se compone de tres hojas. La primera muestra el criterio de búsqueda de casos en la base de datos Sabi®; la segunda recoge la descripción de las variables consideradas, y la tercera (hoja "Datos") guarda los datos que debemos importar desde R-Studio. Estos datos se corresponden con diferentes variables económico-financieras de 100 empresas productoras de electricidad mediante generación eólica.

Luego vamos a cerrar el archivo de Microsoft® Excel® y volveremos a R-Studio. Vamos a abrir nuestro script "componentes_eolica.R" con File → Open File… Este script contiene el programa que vamos a ir ejecutando en la práctica.

La primera línea / instrucción en el script es:

```
rm(list = ls())
```

La instrucción tiene como objeto limpiar el *Environment* de objetos de anteriores sesiones de trabajo. Para importar los datos, ejecutaremos el código:

```
#Importando
library(readxl)
eolicos <- read_excel("eolica_100_mv.xlsx", sheet = "Datos")
eolicos <- data.frame(eolicos, row.names = 1)
summary (eolicos)
```

Podemos observar cómo, en el *Environment,* ya aparece un objeto. Este objeto es una estructura de datos tipo *data frame*, se llama "eolicos" y contiene 13 columnas, una por cada una de las variables almacenadas en el archivo de Microsoft® Excel®.

R ha considerado la primera columna (NOMBRE) como una variable de tipo cualitativo. En realidad, no es una variable, sino el nombre de los casos (empresas). Para evitar que R tome los nombres de los individuos como una variable, podemos redefinir nuestro *data frame* diciéndole que tome esa primera columna como los nombres de los individuos o casos (filas):

```
eolicos <- data.frame(eolicos, row.names = 1)
```

En la línea anterior hemos asignado al *data frame* "eolicos" los propios datos de "eolicos"; pero indicando que la primera columna de datos no es una variable; sino el nombre de los individuos, casos o filas.

Vemos que ya no aparece NOMBRE como variable, y en el *Environment* ya aparece el *data frame* "eolicos" con 100 observaciones (casos), pero con 12 variables (una menos).

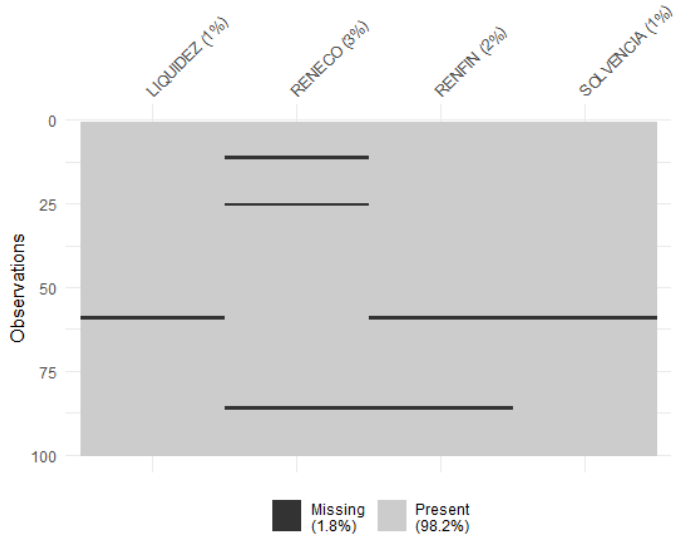
En nuestro análisis solo vamos a considerar como variables clasificadoras para construir los grupos o conglomerados las variables LIQUIDEZ, RENECO, RENFIN y SOLVENCIA. Por ello, crearemos con ellas un nuevo *data frame* llamado "originales":

```
#Seleccionando variables clasificadoras para el analisis

library(dplyr)
originales<-select(eolicos, LIQUIDEZ, RENECO, RENFIN, SOLVENCIA)
summary (originales)
```

El siguiente paso será localizar los posibles ***missing values***, ya que para realizar el análisis es necesario que todos los casos posean dato para todas las variables originales. Para tener una idea general, se puede utilizar la función vis_miss() del paquete visdat, que nos localizará gráficamente los *missing values* de las diferentes variables, y calculará el porcentaje de casos que supone, con respecto al total de observaciones:

```
# Identificando missing values.

library(visdat)
vis_miss(originales)
```

El resultado del código anterior es el siguiente gráfico:



Del gráfico anterior se desprende que existen 7 *missing values* que afectan a 4 casos. Para localizarlos, podemos filtrar nuestro *data frame* con las herramientas de dplyr:

```
originales %>% filter(is.na(LIQUIDEZ) | is.na(RENECO) | is.na(RENFIN) |
is.na(SOLVENCIA)) %>%
  select(LIQUIDEZ, RENECO, RENFIN, SOLVENCIA)
```

Los casos detectados con algún *missing value* son:

```
                              LIQUIDEZ RENECO RENFIN SOLVENCIA
Viesgo Renovables SL.            0.272     NA    3.2    65.883
Sargon Energias SLU              0.188     NA   26.9   -12.811
Parques Eolicos San Lorenzo SL      NA 13.806     NA        NA
Brulles Eolica SL               12.200     NA     NA    55.284
```

Ante la existencia de *missing values*, se puede actuar de varios modos. Por ejemplo, **se puede intentar obtener por otro canal de información el conjunto de valores** que no están disponibles, **o recurrir a alguna estimación**. En caso de que esto sea difícil, se puede optar, simplemente, por **eliminar** estos casos, en especial cuando representan un porcentaje muy reducido respecto al total de casos. En nuestro ejemplo, supondremos que hemos optado por esta última vía, y eliminaremos estos casos con el código:

```
originales <- originales %>%
  filter(! is.na(LIQUIDEZ) & ! is.na(RENECO) & ! is.na(RENFIN) & !
is.na(SOLVENCIA))
```

Podemos verificar en el *Environment* que el *data frame* "originales" ha pasado a tener 96 casos.

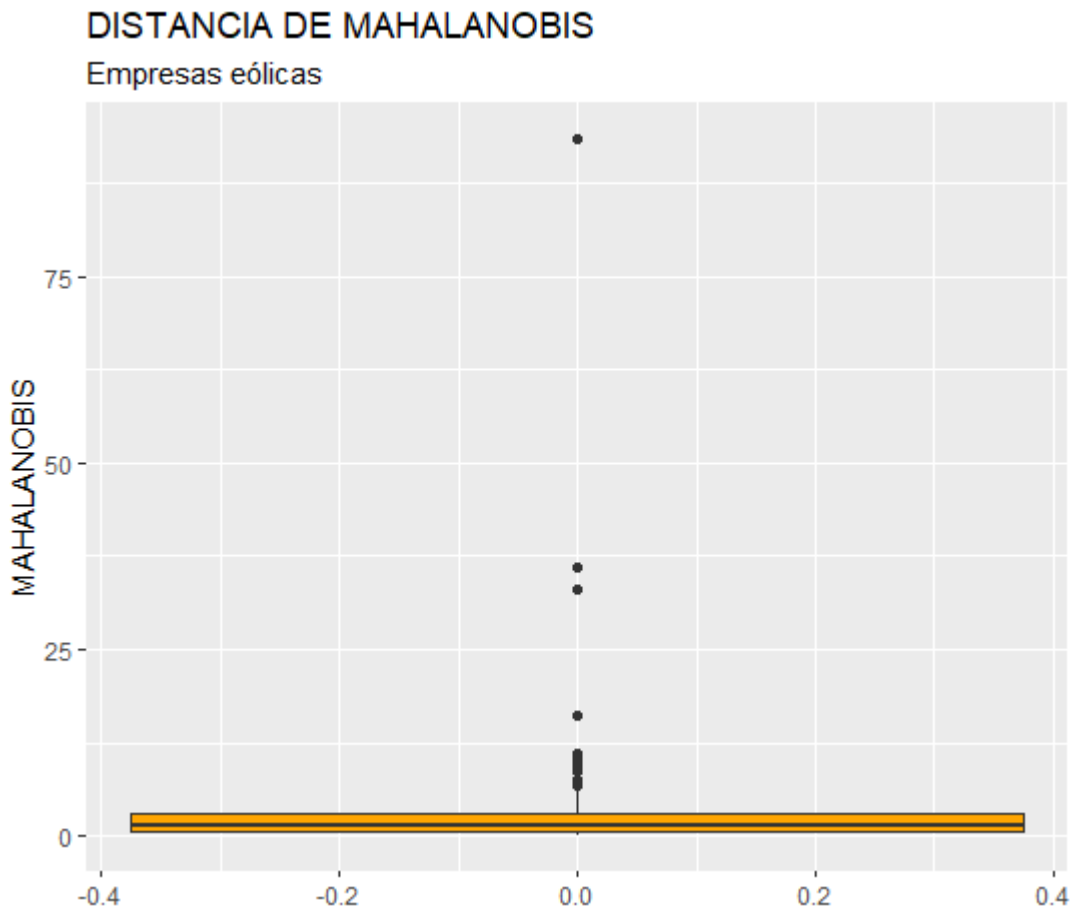
El siguiente paso será la identificación de *outliers*. Para realizar este proceso, y dado que en nuestro análisis contamos con 4 variables, primero "resumiremos" el valor que toman dichas variables para cada caso, mediante el cálculo de la *distancia de Mahalanobis*. De hecho, las distancias de los diferentes casos se almacenarán en un vector, al que llamaremos "MAHALANOBIS". Una vez calculado, lo añadiremos al *data frame* "originales_maha" mediante la función de pegado de columnas, cbind():

```
# Identificando outliers.

MAHALANOBIS <- mahalanobis(originales[],
                           center = colMeans(originales[]),
                           cov=cov(originales[]))
originales_maha <- cbind(originales, MAHALANOBIS)
```

A continuación, construiremos un diagrama de caja de la variable MAHALANOBIS a partir de la función ggplot() del paquete ggplot2:

```
library (ggplot2)
ggplot(data = originales_maha, map = (aes(y = MAHALANOBIS))) +
  geom_boxplot(fill = "orange") +
  ggtitle("DISTANCIA DE MAHALANOBIS", subtitle = "Empresas eólicas") +
  ylab("MAHALANOBIS")
```

El gráfico de caja resultante será:



En el gráfico se observa que existen, por encima de la caja, varios *outliers*. Para identificarlos de modo concreto, hemos de calcular los cuartiles primero y tercero de la variable MAHALANOBIS y pasar el correspondiente filtro:

```
Q1M <- quantile (originales_maha$MAHALANOBIS, c(0.25))
Q3M <- quantile (originales_maha$MAHALANOBIS, c(0.75))
originales_maha %>% filter(MAHALANOBIS > Q3M + 1.5*IQR(MAHALANOBIS) |
MAHALANOBIS < Q1M - 1.5*IQR(MAHALANOBIS)) %>% select(MAHALANOBIS)
```

Tras ejecutar el código anterior, se obtiene el siguiente listado de 13 casos que se comportan, según su *distancia de Mahalanobis* observada, como *outliers*:

	MAHALANOBIS
Global Power Generation SA.	6.715550
Saeta Yield SA.	6.686588
Parque Eolico Santa Catalina SL	16.291607
WPD Wind Investment SL.	11.126058
Renovalia Reserve SL.	7.687337
Molinos Del Ebro SA	32.989960
Luria De Energias SA	7.512327
Parque Eolico Sierra De Las Carbas SL	9.388162
Tarraco Eolica SA	8.699467

```
Elecdey Lezuza SA                    36.094927
Eolica Navarra SL                    93.413215
Sierra De Selva SL                    9.896801
El Paramo Parque Eolico SL           10.859703
```

Al estar interesados en definir tipologías, más que en agrupar todos los casos de la muestra, se podrá optar por **eliminar** estos *outliers* para evitar distorsiones en los resultados:

```
originales_so <- originales_maha %>% filter(MAHALANOBIS <= Q3M +
1.5*IQR(MAHALANOBIS) & MAHALANOBIS >= Q1M - 1.5*IQR(MAHALANOBIS))
originales_so <- originales_so %>% select(-MAHALANOBIS)
```

Con el código anterior, se ha creado un nuevo *data frame*, "originales_so", en el que ya no aparecen los *outliers*. Por tanto, puede observarse en el *Environment* cómo este *data frame* se compone de 83 casos. Por último, eliminaremos también del mismo la variable MAHALANOBIS, dejando solo las 4 variables originales clasificadoras.

Clúster no-jerárquico de k-medias.

Los métodos de agrupación usualmente se basan en la **distancia euclídea**. Como la distancia euclídea es sensible a las unidades de medida de las diferentes variables clasificadoras, es preciso trabajar con las **variables tipificadas**, que obtendremos creando, por ejemplo, un *data frame* "zoriginales" con la función scale(). Luego aplicaremos el método seleccionado a este *data frame,* en lugar de al *data frame* que contiene los datos originales sin tipificar:

```
zoriginales <- data.frame(scale(originales_so))
summary (zoriginales)
```

Este nuevo *data frame* contiene las mismas variables; pero tipificadas (obsérvese, en el summary(), las medias de las variables).

A la hora de realizar un análisis clúster por el método de k-medias, una cuestión importante consiste en determinar con **cuántos grupos** hemos de quedarnos. Existen algoritmos y paquetes de R que aconsejan un número (por ejemplo, NbClust() del paquete NbClust); si bien conviene supeditar el resultado a los propios intereses del investigador según los objetivos que persiga.

Para ejecutar NbClust(), el código será:

```
#Numero de grupos

library(NbClust)
NbClust(data = zoriginales, min.nc=2, max.nc= 10, method = "kmeans")
```

En el código anterior, los argumentos “min.nc” y “max.nc” delimitan el número mínimo y máximo de grupos a formar, respectivamente. De la información que aporta NbClust(), interesa especialmente el texto en el que se aconseja el número de grupos a formar. En este caso, este texto es:

```
*******************************************************************
* Among all indices:
* 5 proposed 2 as the best number of clusters
* 4 proposed 3 as the best number of clusters
* 1 proposed 4 as the best number of clusters
* 1 proposed 5 as the best number of clusters
* 1 proposed 6 as the best number of clusters
* 1 proposed 7 as the best number of clusters
* 2 proposed 8 as the best number of clusters
* 4 proposed 9 as the best number of clusters
* 4 proposed 10 as the best number of clusters

                   ***** Conclusion *****

* According to the majority rule, the best number of clusters is  2
  *******************************************************************
```

La función aconseja prioritariamente realizar 2 grupos. No obstante, y dado que parece un número de grupos demasiado pequeño, en la práctica optaremos por formar 3. El parámetro “k =” controlará el número de grupos, por lo que añadiremos el código:

```
k <- 3  #poner aquí número de grupos decidido.
```

Para realizar el procedimiento de k-medias, se ha escogido la función KMeans_rcpp() del paquete ClusterR. Esta función tiene la ventaja de que, para obtener las *semillas* o *centroides* iniciales, permite hacer uso del procedimiento ***k-medias++***, que obtiene, en general, mejores resultados que otros procedimientos, como los puramente aleatorios. El código a ejecutar será el siguiente:

```
# Aplicando k-means con inicializacion kmeans++

library (ClusterR)

set.seed(123)

cluster_k <-KMeans_rcpp (zoriginales,
```

```
                    clusters = k,
                    num_init = 10,
                    max_iters = 100,
                    initializer = "kmeans++")
```

La función set.seed() fija la secuencia de generación de números aleatorios, de modo que, si se ejecuta varias veces el mismo trozo de código, se obtendrá la misma solución.

La función para aplicar el algoritmo de **k-medias** con la inicialización *k-medias++* es KMeans_rcpp(). El primer argumento es el *data frame* con las variables clasificadoras (en sus versiones tipificadas). El segundo es el número de clústeres o grupos a obtener (que lo hemos asignado anteriormente al parámetro "**k**"). El tercero, "num_init", es el número de veces que se repite el procedimiento a fin de retener la mejor solución. "Max_iters =" fija el máximo de iteraciones del procedimiento de k-medias hasta obtener una solución estable. "Initializer =" define el método de obtención de las semillas (en nuestro caso, *k-means++*). La solución final se guarda en el objeto "cluster_k", como se puede apreciar en el *Environment*.

Dentro de la solución, el vector con el grupo de pertenencia de cada empresa se obtiene con el elemento **"$clusters"**. Conviene guardar ese vector como factor para luego utilizarlo en los gráficos. En concreto, se guardará en el factor **"whatcluster_k"**, que añadiremos al *data frame* **"originales_so"** mediante la función de pegado de columnas, cbind():

```
whatcluster_k <-as.factor(cluster_k$clusters)
levels(whatcluster_k)
originales_so <-cbind(originales_so, whatcluster_k)
summary(originales_so)
```

Vamos a continuación a **caracterizar los grupos** formados en función de las medias de las variables originales. Así, se podrán mostrar en pantalla las medias de cada grupo de las distintas variables originales, usando las funciones by_group() y summarise() de dplyr. Los decimales se ajustarán utilizando la función round(). Toda la información se asigna al *data frame* "tablamedias", para poder posteriormente representado en una tabla mediante las facilidades que ofrecen los paquetes knitr y kableExtra.

```
tablamedias <- originales_so %>%
  group_by(whatcluster_k) %>% summarise(obs = length(whatcluster_k),
                       Liquidez = round(mean(LIQUIDEZ),2),
                       Rentabilidad_Economica = round(mean(RENECO),2),
                       Rentabilidad_Financiera = round(mean(RENFIN),2),
                       Solvencia = round(mean(SOLVENCIA),2))
```

```
library (knitr)
library (kableExtra)
knitr.table.format = "html"

tablamedias %>%
  kable(caption = "Método de k-medias. Medias de variables",
        col.names = c("Clúster", "Observaciones", "Liquidez", "Rent.
Eco.", "Rent. Fin.", "Solvencia")) %>%
  kable_styling(full_width = F, bootstrap_options = "striped",
"bordered", "condensed", position = "center", font_size = 12) %>%
  row_spec(0, bold= T, align = "c") %>%
  row_spec(1:k, bold= F, align = "c")
```

El resultado obtenido es el siguiente:

Método de k-medias. Medias de variables

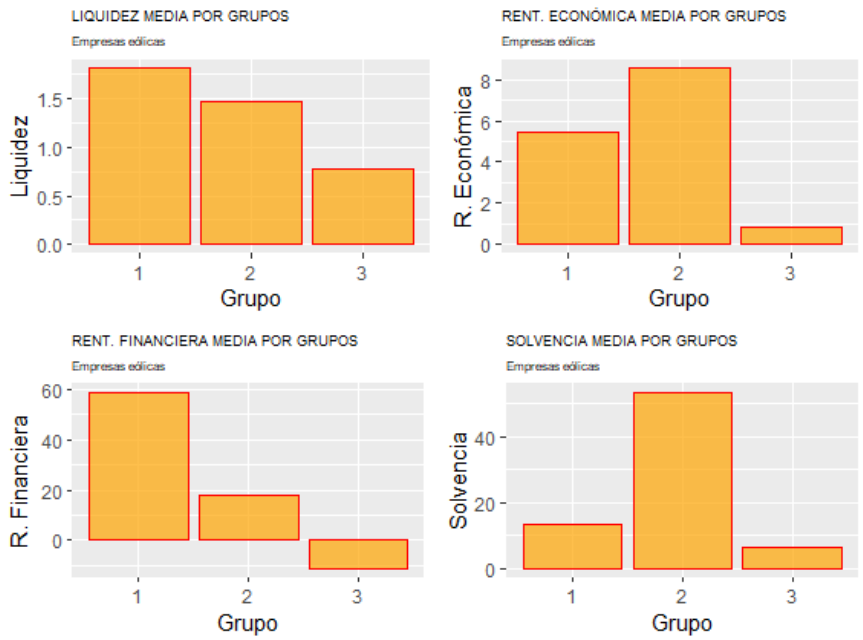
Clúster	Observaciones	Liquidez	Rent. Eco.	Rent. Fin.	Solvencia
1	25	1.81	5.47	58.76	13.50
2	28	1.47	8.53	18.14	53.41
3	30	0.78	0.84	-11.43	6.46

Obviamente, también se podrían comparar las medias de los grupos, para cada variable, con un simple gráfico de barras:

```
gliquid <- ggplot(data = tablamedias, map = (aes(y = Liquidez, x =
whatcluster_k))) +
  geom_bar(stat = "identity", colour = "red", fill = "orange", alpha =
0.7) +
  ggtitle("LIQUIDEZ MEDIA POR GRUPOS", subtitle = "Empresas eólicas") +
  xlab ("Grupo") +
  ylab("Liquidez") +
  theme(plot.title=     element_text(size=7),     plot.subtitle     =
element_text(size = 6))

greneco   <-   ggplot(data   =   tablamedias,   map   =   (aes(y   =
Rentabilidad_Economica, x = whatcluster_k))) +
  geom_bar(stat = "identity", colour = "red", fill = "orange", alpha =
0.7) +
  ggtitle("RENT. ECONÓMICA MEDIA POR GRUPOS", subtitle = "Empresas
eólicas") +
  xlab ("Grupo") +
  ylab("R. Económica") +
  theme(plot.title=     element_text(size=7),     plot.subtitle     =
element_text(size = 6))

grenfin   <-   ggplot(data   =   tablamedias,   map   =   (aes(y   =
Rentabilidad_Financiera, x = whatcluster_k))) +
  geom_bar(stat = "identity", colour = "red", fill = "orange", alpha =
0.7) +
  ggtitle("RENT. FINANCIERA MEDIA POR GRUPOS", subtitle = "Empresas
eólicas") +
```

```
  xlab ("Grupo") +
  ylab("R. Financiera") +
  theme(plot.title=      element_text(size=7),      plot.subtitle      =
element_text(size = 6))

gsolven <- ggplot(data = tablamedias, map = (aes(y = Solvencia, x =
whatcluster_k))) +
  geom_bar(stat = "identity", colour = "red", fill = "orange", alpha =
0.7) +
  ggtitle("SOLVENCIA MEDIA POR GRUPOS", subtitle = "Empresas eólicas")
+
  xlab ("Grupo") +
  ylab("Solvencia") +
  theme(plot.title=      element_text(size=7),      plot.subtitle      =
element_text(size = 6))

library (patchwork)

(gliquid + greneco) / (grenfin + gsolven)
```

Los 4 gráficos se han unido en uno solo con las facilidades del paquete patchwork, obteniéndose la siguiente composición:



De la tabla y gráficos anteriores se pueden extraer varias conclusiones que caracterizan a los diversos grupos. El grupo 1 se caracteriza sobre todo por su gran rentabilidad financiera media y, en menor medida, por su liquidez. El grupo 2 tiene, en media, y de modo muy destacado, una mayor solvencia y rentabilidad financiera. El grupo 3 destaca por poseer, en media, una muy baja rentabilidad económica, una baja solvencia y una rentabilidad financiera negativa.

Componentes principales.

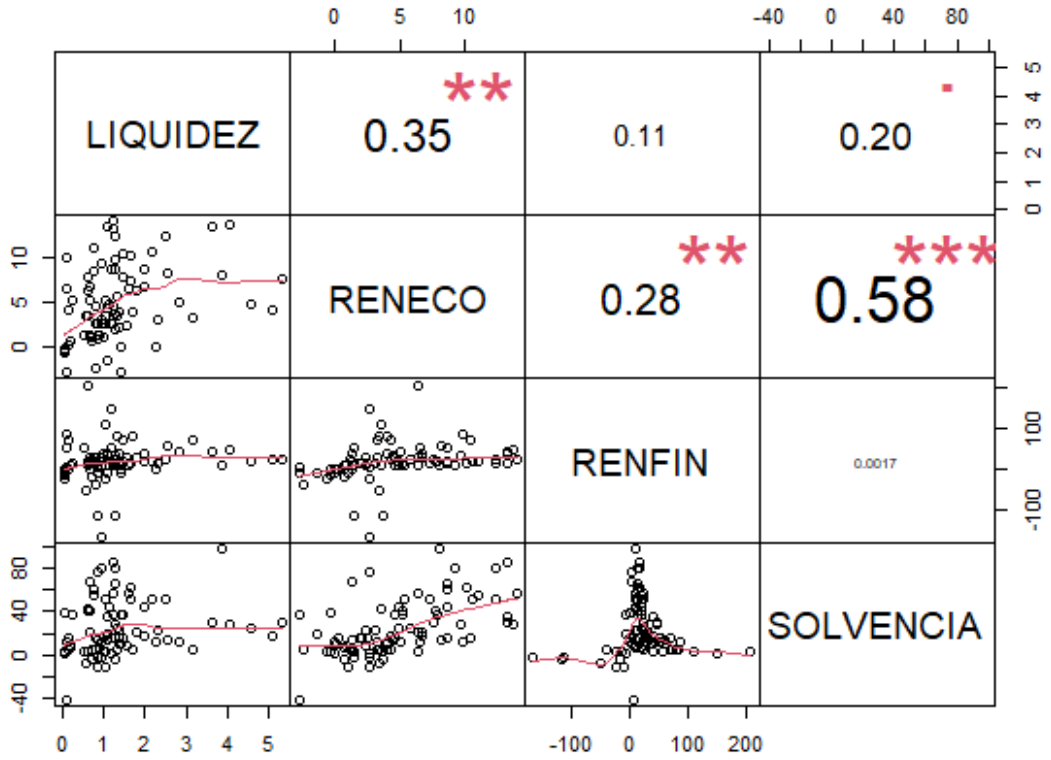
Un problema que acarrea el usar las variables originales como clasificadoras, cuando son más de 2, es que es difícil trazar un gráfico de dispersión que nos dé una idea precisa de los grupos formados en cuanto a su composición y a lo "separados" que se encuentran unos de otros.

Una solución alternativa a esta limitación puede consistir en utilizar para realizar el gráfico de dispersión a partir de las dos primeras **componentes principales** de las variables originales, siempre y cuando se den las condiciones para aplicar esta técnica y ambas componentes recojan una elevada proporción de la *comunalidad* o varianza total de las variables originales.

La condición básica para realizar componentes principales es la existencia de algunas variables con altas correlaciones. Para comprobar esto, recurriremos a la función chart.Correlation() del paquete PerformanceAnalytics. Antes, hemos de crear un *data frame* que contenga solo las variables originales, es decir, que no contenga a "whatcluster_k", pues es un factor (no es una variable en escala métrica). Este *data frame* será llamado, por ejemplo, "originales_cp":

```
originales_cp <- originales_so %>% select(-whatcluster_k)
library(PerformanceAnalytics)
chart.Correlation(originales_cp, histogram = F, pch = 18)
```

El resultado es el gráfico que se muestra a continuación. En él se comprueba la existencia de algunas correlaciones relativamente elevadas entre RENECO y SOLVENCIA, LIQUIDEZ y RENFIN:



La obtención de las componentes se va a realizar mediante la función prcomp(). Es conveniente que activemos el argumento "scale" con "T" (True), para que las variables originales sean consideradas en sus versiones tipificadas. Vamos a asignar los resultados a un objeto de nombre, por ejemplo, "componentes". La sintaxis es la siguiente:

```
componentes <- prcomp(originales_cp, scale=T)
summary (componentes)
```

Se obtienen los siguientes resultados:

```
Importance of components:
                          PC1    PC2    PC3     PC4
Standard deviation     1.3598 0.9997 0.9004 0.58387
Proportion of Variance 0.4622 0.2498 0.2027 0.08523
Cumulative Proportion  0.4622 0.7121 0.9148 1.00000
```

Como comprobamos, las dos primeras componentes acumulan más de un 71% de la varianza total o *comunalidad* (información) que las variables originales contienen sobre el comportamiento de las empresas de la muestra ("Cumulative Proportion"). La "Standard deviation" es la raíz cuadrada de los autovalores asociados a cada componente. Por tanto, podríamos utilizar esas dos componentes principales para clasificar a las

empresas, en lugar de las 4 variables originales, con la ventaja de que dos variables pueden ser fácilmente representadas en un gráfico de dispersión.

Las *cargas* o coeficientes de cada componente se obtienen pidiendo a nuestro objeto "componentes" el elemento "rotation":

```
round(componentes$rotation, 4)
```

Veremos cómo aparecen, por columnas, las componentes, por filas las variables originales, y en las intersecciones, los coeficientes o *cargas*. Así:

```
             PC1      PC2      PC3      PC4
LIQUIDEZ  0.4396 -0.0537  0.8823 -0.1594
RENECO    0.6523  0.0209 -0.1913  0.7332
RENFIN    0.2764 -0.8853 -0.2430 -0.2842
SOLVENCIA 0.5521  0.4614 -0.3548 -0.5969
```

No vamos a emplear un método para obtener el número de componentes retenidas sugerido, ya que en este caso lo que queremos es que sean 2 dichas componentes, a fin de obtener una representación gráfica bidimensional. Además, sabemos que las dos componentes recogen ya una proporción apreciable del comportamiento (varianza total o *comunalidad*) de los individuos.

Para utilizar las puntuaciones de las dos primeras componentes, hemos de obtener sus puntuaciones de cada caso (empresa). Estas puntuaciones están guardadas en la matriz "x" del objeto "prcomp" creado ("componentes"). Vamos a crear un *data frame* con las variables originales (con los *outliers* excluidos), el grupo de pertenencia, y las puntuaciones de ambas componentes (a la primera componente la hemos llamado "Componente.1" y a la segunda "Componente.2"), denominado "Componentes" (con la "C" en mayúsculas):

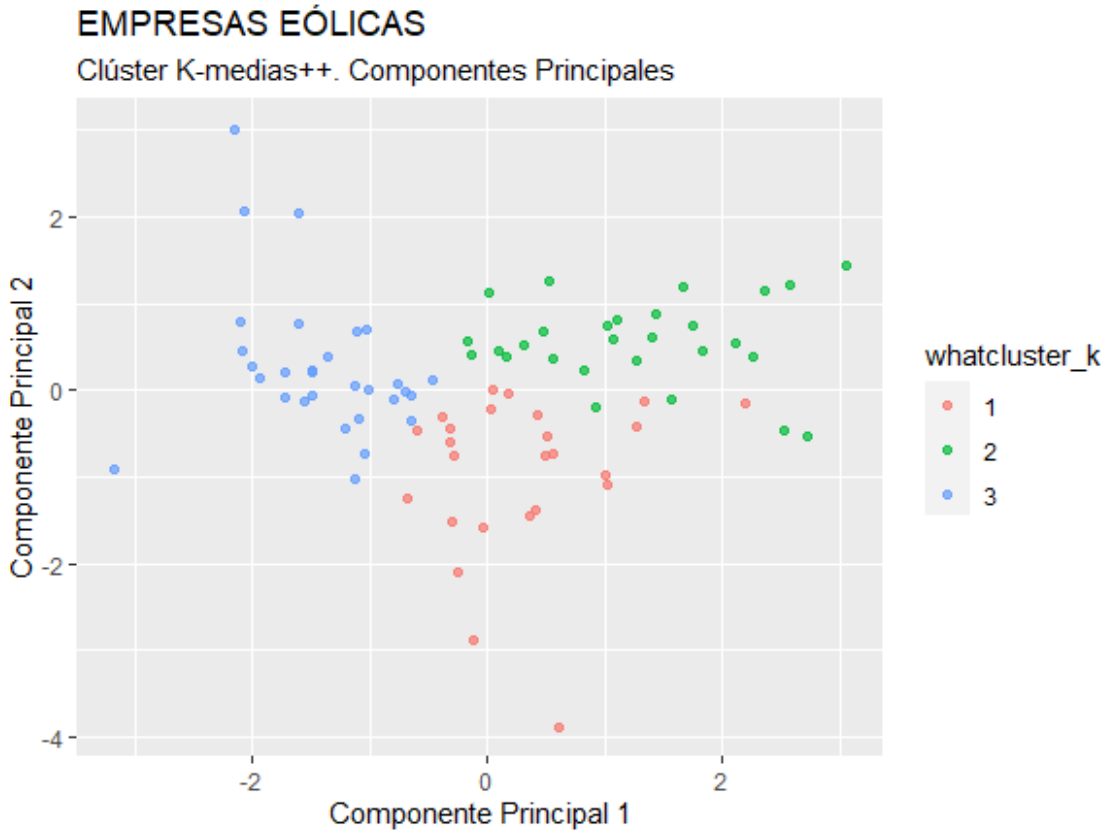
```
Componente.1 <-componentes$x[,1]
Componente.2 <-componentes$x[,2]
Componentes <- cbind(originales_so, Componente.1, Componente.2)
summary (Componentes)
```

Para realizar el gráfico de dispersión se utilizará la función ggplot() del paquete ggplot2:

```
ggplot(data = Componentes, map = (aes(x = Componente.1, y = Componente.2,
col = whatcluster_k))) +
  geom_point(alpha = 0.7) +
  ggtitle("EMPRESAS EÓLICAS", subtitle = "Clúster K-medias++.
Componentes Principales") +
```

```
xlab("Componente Principal 1") +
ylab("Componente Principal 2")  ylab("Componente Principal 2")
```

El resultado obtenido es el siguiente gráfico:



En el gráfico se aprecia cómo los casos del **grupo 1** están, en general, centrados en cuanto a la componente 1, con algunos casos más hacia la derecha (esta componente posee las mayores cargas, positivas, en las variables LIQUIDEZ, RENECO y SOLVENCIA); y en la parte inferior con respecto al eje de la componente 2 (cuyo mayor carga es para RENFIN, con signo negativo), lo que hace pensar en una buena rentabilidad financiera, en general. El grupo 2 tiene sus casos centrados verticalmente (componente 2, RENFIN); y tales casos se concentran en la zona derecha del gráfico(componente 1), lo que lleva a pensar en buenos comportamientos en términos de rentabilidad económica, solvencia y liquidez. El caso opuesto, respecto a esta componente 1, muestra el grupo 3, con una concentración en la parte izquierda del gráfico, lo que lleva a pensar en una peor situación en estas variables; mientras que, además, algunos casos tienden a concentrarse en la parte superior, lo que denotaría una mala situación en cuanto a la rentabilidad financiera.

Caracterización de los grupos.

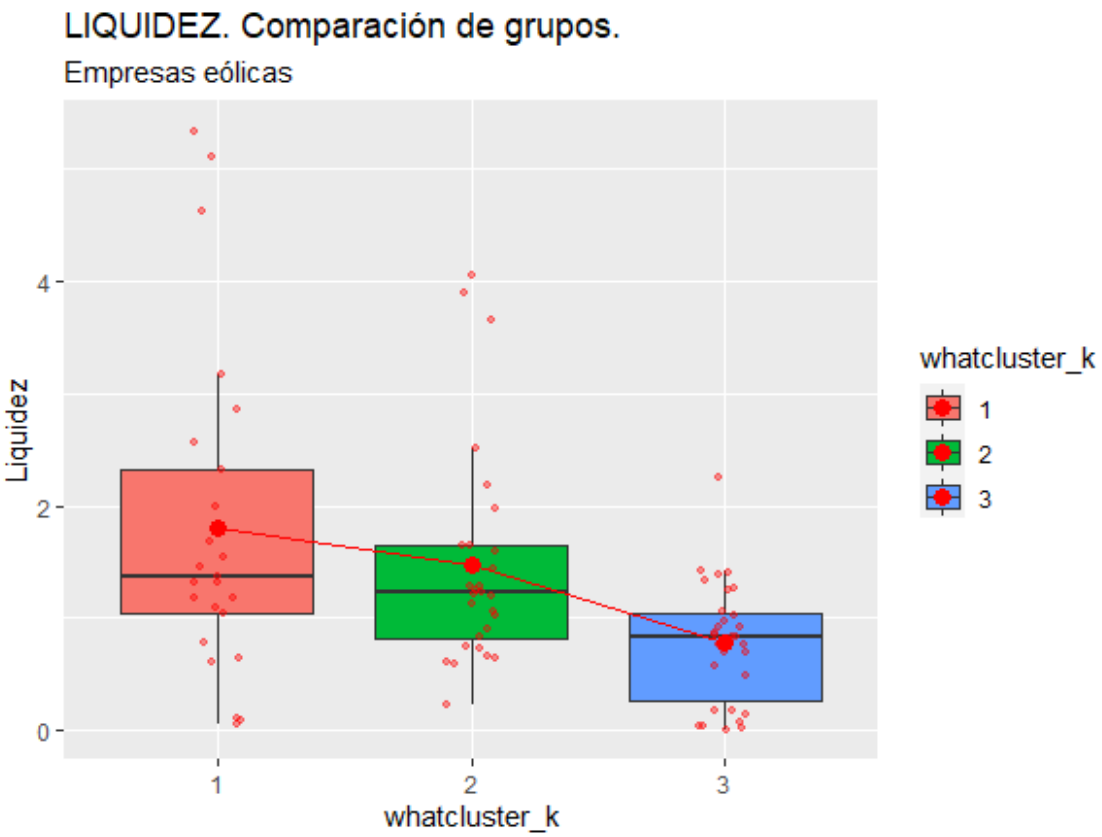
Cobra especial relevancia en el clúster de k-medias, cuando lo que se pretende es detectar diversas tipologías de casos, el caracterizar los grupos formados en cuanto a las variables originales utilizadas en el análisis. Uno de los modos, como ya se vio, es calcular el valor medio de las variables en los grupos. Como complemento a ello, y teniendo en cuenta que los grupos se pueden considerar submuestras que representan a subpoblaciones, se puede aplicar alguna prueba de comparaciones múltiples de las medias de los grupos formados, de modo que se pueda afirmar, para cierta significación estadística (usualmente 0,05), **si las diferencias en las medias de los grupos para cada variable son estadísticamente relevantes** (significativas) o no.

Una prueba clásica para llevar a cabo esta tarea es aplicar el *test de comparaciones múltiples de Tuckey*. No obstante, y dado que esta prueba requiere del cumplimiento de ciertos requisitos previos (normalidad de los grupos, varianzas homogéneas); hemos optado por la *prueba robusta de Kruskal-Wallis*.

Para ilustrar el caso, se pueden graficar previamente los box-plot, y luego realizar la prueba. En esta práctica realizaremos el análisis para la variable LIQUIDEZ. Para obtener el gráfico, ejecutaremos el código:

```
#Variable LIQUIDEZ

#Box-Plot

ggplot(data = originales_so, map = (aes(x = whatcluster_k, y = LIQUIDEZ,
fill = whatcluster_k))) +
  geom_boxplot(outlier.shape = NA) +
  stat_summary(fun = "mean",
               geom = "point",
               size = 3,
               col = "red") +
  stat_summary(fun = "mean",
               geom = "line",
               col = "red",
               map = (aes(group = TRUE))) +
  geom_jitter(width = 0.1,
              size = 1,
              col = "red",
              alpha = 0.40) +
  ggtitle("LIQUIDEZ. Comparación de grupos.", subtitle = "Empresas
eólicas") +
  ylab("Liquidez")
```

El resultado será el gráfico siguiente:



Se aprecia, como ya se vio en los gráficos de barras, cómo el grupo 1 posee una liquidez media (punto rojo grande) mayor que los otros dos grupos, teniendo el grupo 3 el menor indicador de liquidez medio. Para comprobar si estas diferencias son significativas (para una significación de 0,05), se procederá a realizar la prueba de comparaciones múltiples de Kruskal-Wallis. Esta prueba se realiza con una llamada a la función kruskalmc() del paquete pgirmess:

```
library(pgirmess)
Datos.KMC<-kruskalmc(originales_so$LIQUIDEZ ~
originales_so$whatcluster_k)
Datos.KMC
```

La solución se guarda en un objeto, por ejemplo “Datos.KMC”. En el argumento de la función se pone la variable analizada (LIQUIDEZ), la “tilde” de la “ñ”, y el factor que identifica la pertenencia a los diferentes grupos (whatcluster_k). Luego se llama al objeto creado con la solución. EL resultado obtenido es el siguiente:

```
Multiple comparison test after Kruskal-Wallis
```

```
p.value: 0.05
Comparisons
    obs.dif critical.dif difference
1-2    4.76     15.87806      FALSE
1-3   22.56     15.62641       TRUE
2-3   17.80     15.16292       TRUE
```

Del resultado se desprende que las diferencias entre la liquidez media son significativas en el caso del grupo 3, con respecto a los otros dos grupos ("difference" TRUE). La diferencia entre la liquidez de los grupos 1 y 2 NO es significativa ("difference" FALSE).

Se propone realizar el mismo procedimiento para el resto de variables originales.

Updated: 15/11/2022