Análisis clúster jerárquico.

Segmentando las empresas eólicas.

Vamos a considerar una serie de 6 variables que caracterizan a un grupo de 20 empresas de producción eléctrica mediante tecnología eólica. Nuestro objetivo es segmentar este conjunto de empresas, haciendo grupos homogéneos (conglomerados), y caracterizando a dichos grupos. Las variables clasificadoras son: la solvencia (SOLVENCIA), los fondos propios (FPIOS), el margen de beneficio (MARGEN), el resultado del ejercicio (RES), la rentabilidad financiera (RENFIN) y el grado de apalancamiento (APALANCA).

Dado que son pocos casos, vamos a utilizar métodos jerárquicos de agrupación de casos. En concreto, utilizaremos el **método de Ward**, usualmente empleado 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 "componentes", 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_20.xlsx" y un *script* denominado "cluster_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 20 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:

```
# DATOS

library(readxl)
eolicos <- read_excel("eolica_60.xlsx", sheet = "Datos")
```

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 11 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 20 observaciones (casos), pero con 10 variables (una menos).

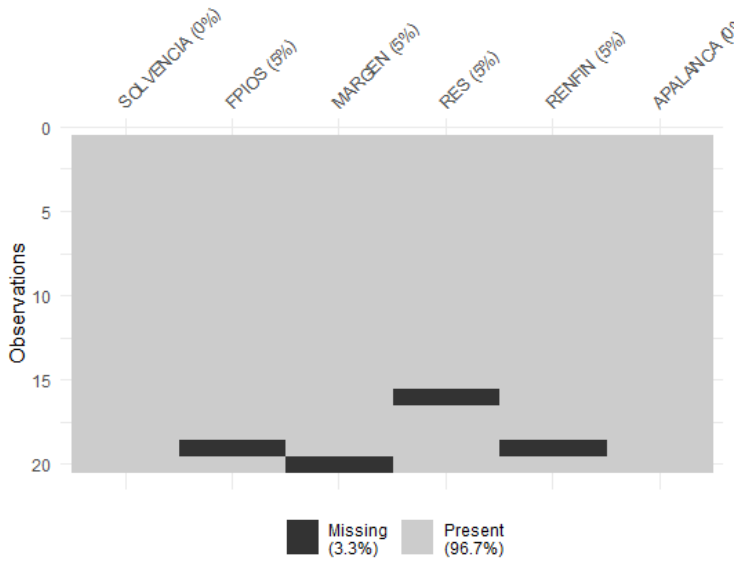
En nuestro análisis solo vamos a considerar como variables clasificadoras para construir los grupos o conglomerados las variables SOLVENCIA, FPIOS, MARGEN, RES, RENFIN y APALANCA. Por ello, crearemos con ellas un nuevo *data frame* llamado "originales":

```
# Seleccionando variables para el analisis

library(dplyr)
originales <- eolicos %>% select(SOLVENCIA, FPIOS, MARGEN, RES, RENFIN,
APALANCA)
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 4 *missing values* que afectan a 4 casos. Para localizarlos, podemos filtrar nuestro *data frame* con las herramientas de dplyr:

```
originales %>% filter(is.na(RES) | is.na(FPIOS) | is.na(RENFIN) |
is.na(MARGEN) | is.na(SOLVENCIA) | is.na(APALANCA)) %>%
  select(RES, FPIOS, RENFIN, MARGEN, SOLVENCIA, APALANCA)
```

Los casos detectados con algún *missing value* son:

```
                                    RES    FPIOS RENFIN MARGEN SOLVENCIA  APALANCA
Biovent Energia SA                   NA  70033.0 11.952 22.792    38.082   141.163
Parque Eolico Santa Catalina SL 3645.278       NA     NA 31.780    -1.126 -6265.496
WPD Wind Investment SL.         -850.068 108023.8 -1.049     NA    99.082     0.000
```

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(RES) & ! is.na(FPIOS) & ! is.na(RENFIN) & !
is.na(MARGEN) & ! is.na(SOLVENCIA) & ! is.na(APALANCA))
```

Podemos verificar en el *Environment* que el *data frame* "originales" ha pasado a tener 17 casos.

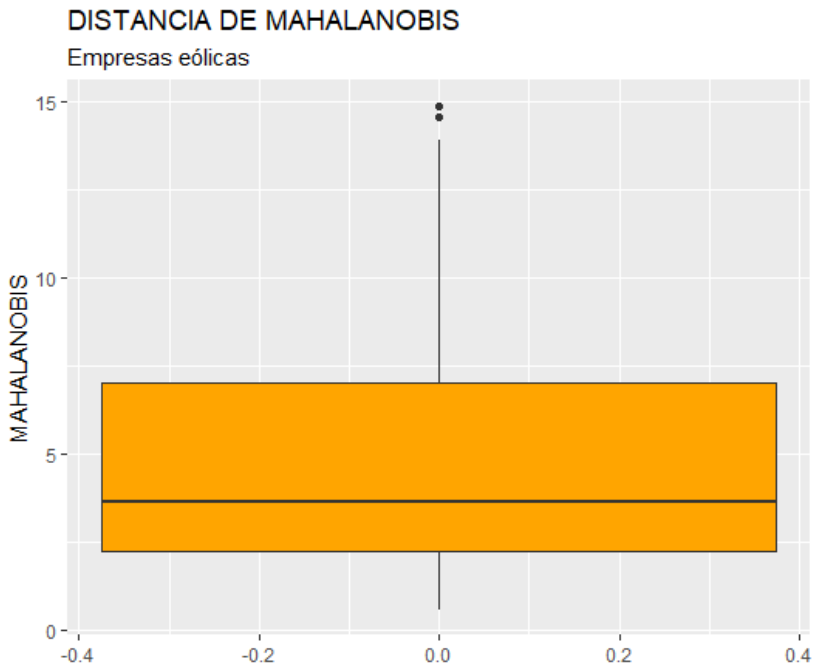
El siguiente paso será la identificación de *outliers*. Para realizar este proceso, y dado que en nuestro análisis contamos con 6 variables, primero "resumiremos" el valor que toman dichas variables para cada caso, mediante el cálculo de la *distancia de Mahalanobis*. De hecho, las distancias de los 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, 2 *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 casos que se comportan, según su *distancia de Mahalanobis* observada, como *outliers*:

```
                                MAHALANOBIS
Holding De Negocios De GAS SL.     14.89967
Elawan Energy SL.                  14.56272
```

Clúster jerárquico con variables originales.

En el desarrollo de otras técnicas, en este punto localizaríamos y eliminaríamos los *outliers*. En este caso **no** lo vamos a hacer, ya que queremos agrupar **todos los casos** que tenemos en el análisis. Precisamente, si hay algún caso que permanece aislado, sin agruparse con otros, en el proceso de agrupación, quizá se trate de un candidato a *outlier*.

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**, lo que haríamos 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:

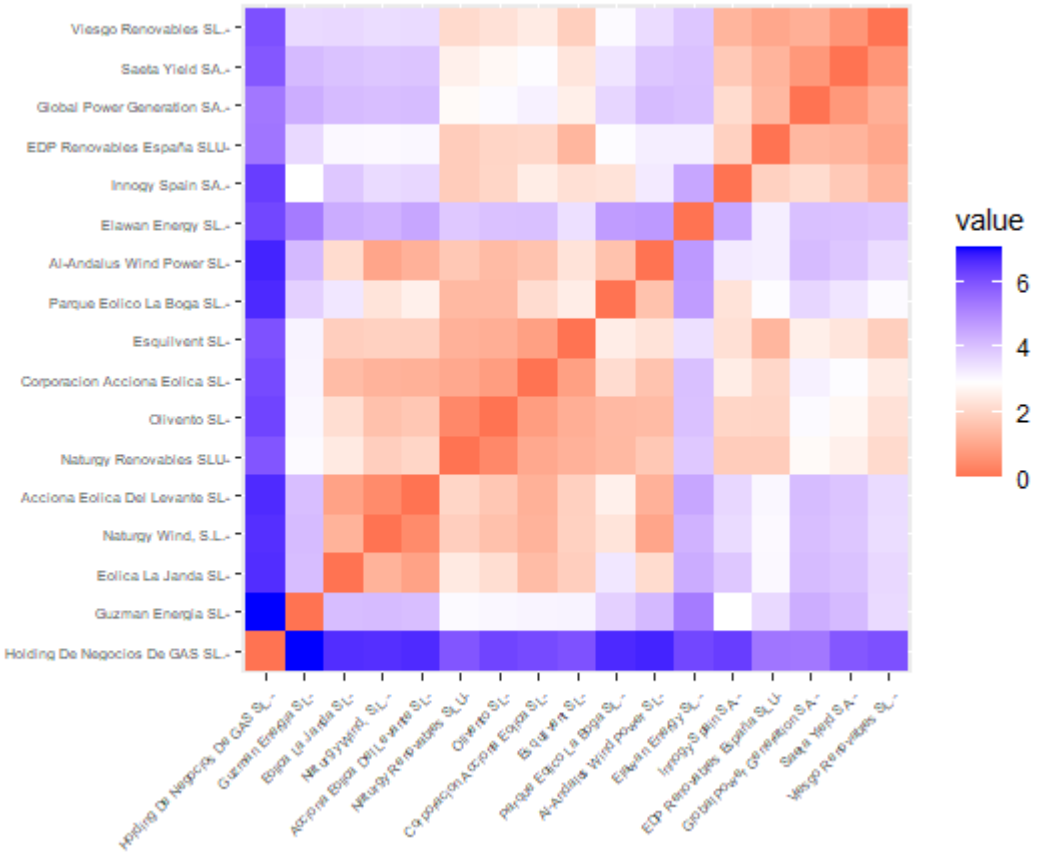
```
zoriginales <- data.frame(scale(originales))
summary (zoriginales)
```

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

Previamente a aplicar un método de agrupación, conviene calcular la **matriz de distancias** entre los casos, a la que llamaremos "d". Para visualizarla, una opción es representarla con el *gráfico de temperatura* que ofrece la función fviz_dist() de la librería factoextra:

```
d <- dist(zoriginales)
library (factoextra)
fviz_dist(d, lab_size = 5)
```

El resultado es este:

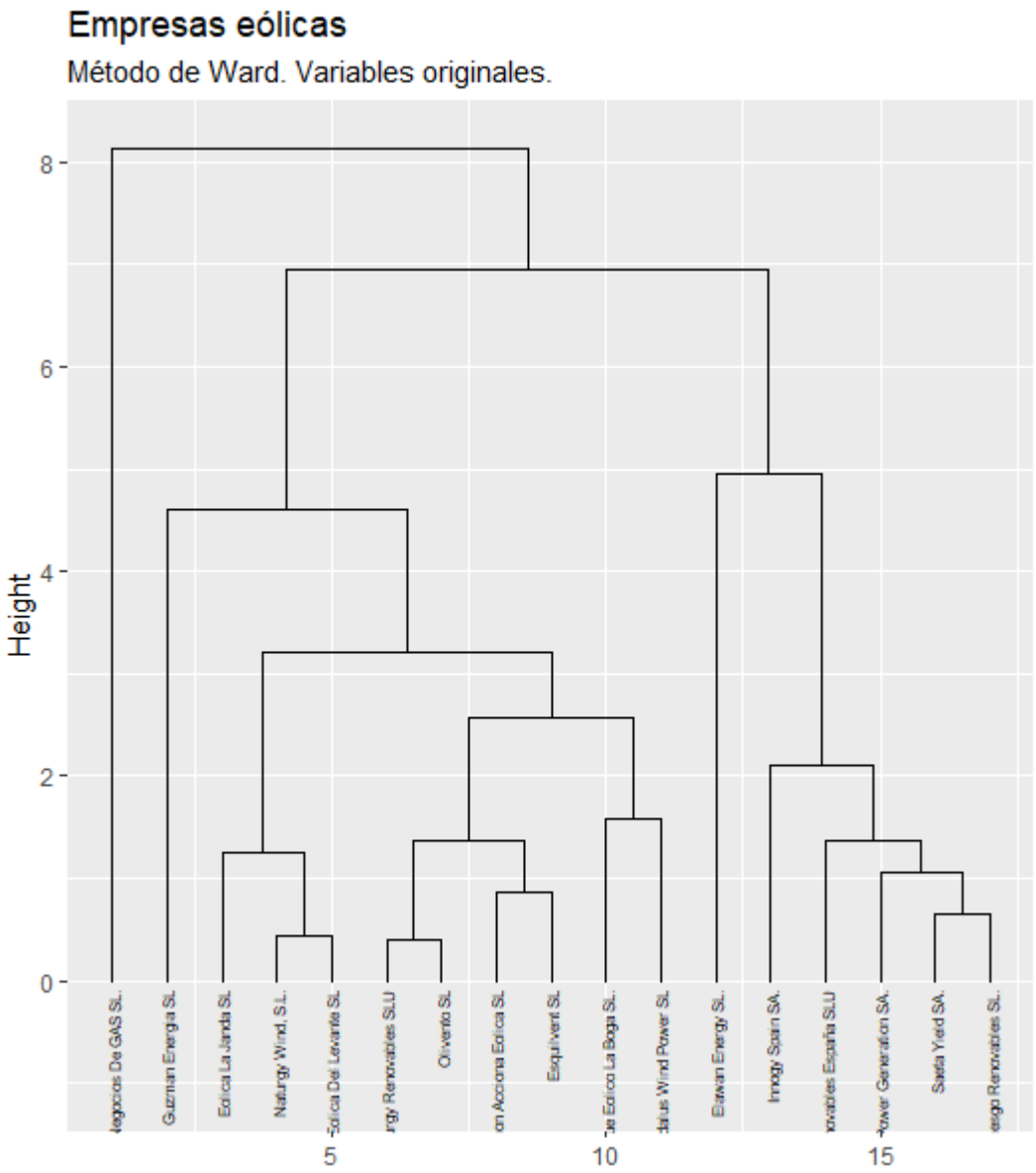


Los casos con intersecciones en tonos más rojizos tenderán a agruparse con mayor facilidad (o a agruparse antes); mientras que los casos cuya intersección está en un tono azulado tenderán a pertenecer a grupos diferentes (o a agruparse más tarde). Puede observarse cómo las distancias de las dos empresas que fueron identificadas como *outliers* ("Holding de Negocios de Gas" y "Elawan Energy") matienen grandes distancias (casillas azuladas) con el resto de empresas.

Vamos a realizar el análisis clúster jerárquico mediante uno de los métodos más habituales, el de ***Ward***, como es común en las aplicaciones prácticas este método proporciona grupos muy homogéneos). La función a utilizar es hclust(). La solución se guardará en el **objeto "cluster_j"**. Luego se visualizará el ***dendograma*** construido con la función fviz_dend() del paquete factoextra, que permite personalizar el gráfico en un lenguaje semejante al utilizado con ggplot2:

```
cluster_j<-hclust(d, method="ward.D2")
fviz_dend(cluster_j, cex=0.4, rect = FALSE) +
  labs(title = "Empresas eólicas",
       subtitle = "Método de Ward. Variables originales + outliers") +
  theme_grey()
```

La solución obtenida será:



El eje vertical del dendograma recoge las distancias entre los casos y/o grupos previos que se van agrupando. Por otro lado, en este ejemplo, es interesante observar que las dos empresas *outliers* ("Holding de Negocios de Gas" y "Elawan Energy") se agrupan con el resto en una fase muy tardía del proceso de agrupamiento (muy cerca del único grupo).

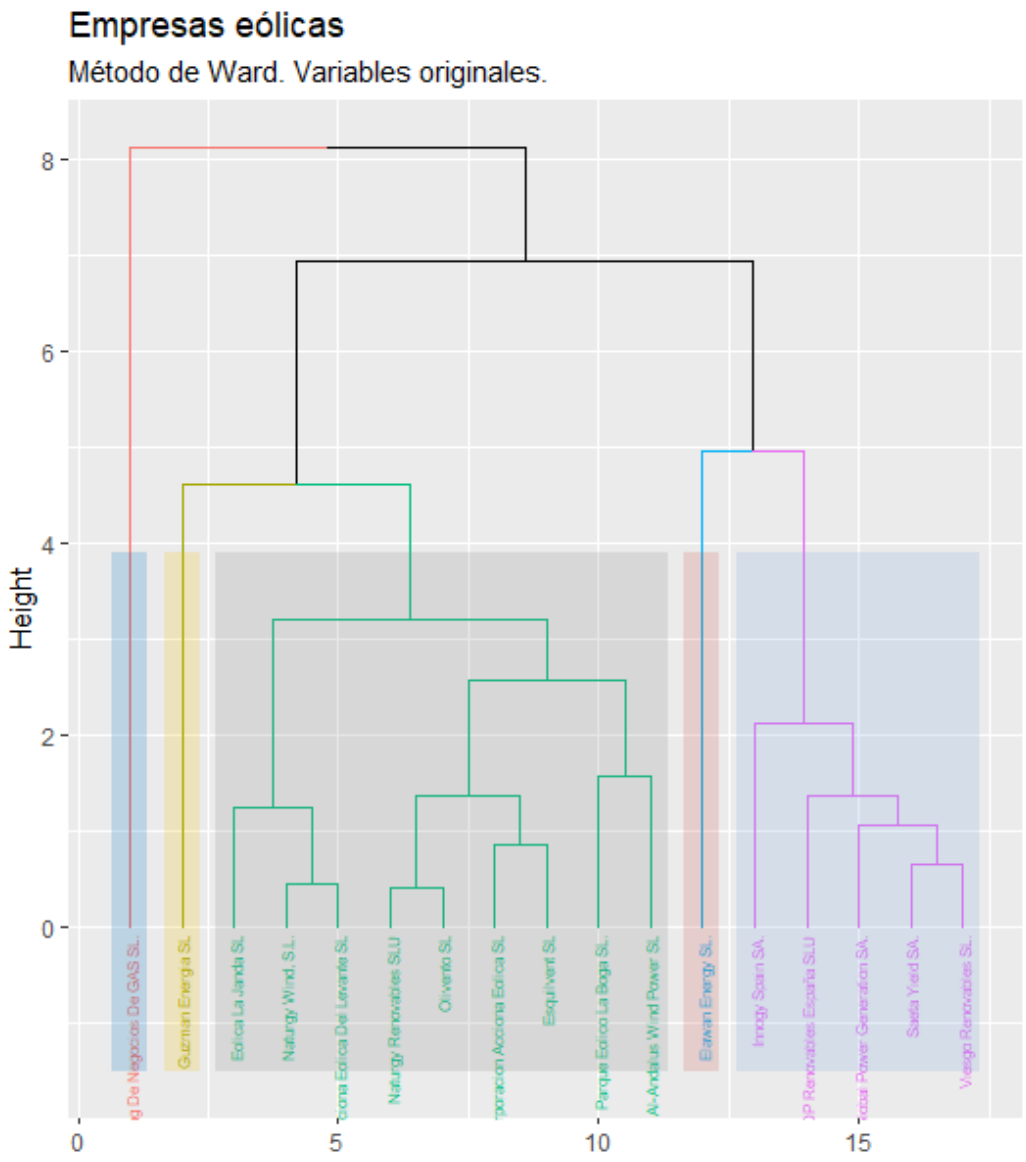
Una cuestión importante consiste en determinar con **cuántos grupos** hemos de quedarnos. Aunque existen algoritmos y paquetes de R que aconsejan un número (por ejemplo, NbClust() del paquete NbClust); puede ser preferible que el propio investigador decida el número de grupos a crear. Esto es así dado que el dendograma informa de la sucesiva

agrupación de casos y grupos precedentes. Además, ya que son muy pocos los casos a agrupar, las opciones (número de grupos) que se tienen son reducidas.

En este ejemplo, un número de grupos razonable podría ser 5, que contaría con el aval de mantener individualizados a las empresas *outliers*. Si se acepta esta opción, se podrá visualizar de nuevo el dendograma coloreando los grupos formados, con el código siguiente:

```
fviz_dend(cluster_j, cex=0.4, k=3, kcolors = "jco", rect = TRUE,
rect_border = "jco", rect_fill = TRUE)+
  labs(title = 'Dendograma 50 empresas eólicas', subtitle = 'Método de
Ward') +
  theme_grey()
```

El parámetro “k =” controla el número de grupos, y el argumento “rect = TRUE” activa los rectángulos para diferenciar los diferentes grupos, rectángulos que toman colores diferentes dependiendo de la paleta de colores dispuesta en el argumento “kcolors =”.



El resultado se muestra en el dendograma anterior. En él se aprecia cómo hay un grupo de 9 empresas (gris con texto en verde), luego hay un grupo de 5 empresas (azul pálido con texto en morado), y luego hay 3 empresas que no se han agrupado con ninguna otra: el *outlier* que mostraba una mayor distancia de Mahalanobis (y distancias euclídeas con el resto de empresas), “Holding de Negocios de Gas” (grupo azul con letra naranja), el otro outlier, “Elawan Energy” (grupo rosado con letra en azul) y, finalmente, la empresa “Guzmán Energía” (grupo amarillo).

Con lo visto hasta ahora, puede afirmarse que el análisis clúster, a partir de la observación de su dendograma, puede ser considerado un método de detección de *outliers*, por sí mismo.

Vamos a continuación a **identificar con mayor detalle los casos** que integran cada uno de los grupos, así como a **caracterizar tales grupos** en función de las medias de las variables originales. Para ello, crearemos el vector de valores enteros que indica el grupo al que pertenece cada caso (empresa). A este vector se le llamará, por ejemplo, "whatcluster_j", y se construye mediante la función cutree(), donde el primer argumento es el nombre del objeto que guarda la solución del análisis clúster ("cluster_j"), y el segundo argumento es el número de grupos que hemos decidido crear (k = 5). Conviene convertir esta variable en un factor, para que deje de ser variable métrica, a efectos de incorporar posteriormente una leyenda en gráficos posteriores, con la función as.factor(). Finalmente, ese factor se incorporará al *data_frame* "originales" (importante: no a "zoriginales"; sino al *data frame* que contiene a las variables no tipificadas) con la función cbind().

```
whatcluster_j <- cutree(cluster_j, k=5)
whatcluster_j <- as.factor(whatcluster_j)
levels(whatcluster_j)
originales <- cbind(originales, whatcluster_j)
```

Una vez incorporado al *data frame* el grupo de pertenencia de cada empresa, se podrán sacar 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 %>%
  group_by(whatcluster_j) %>% summarise(obs = length(whatcluster_j),
                        Solvencia = round(mean(SOLVENCIA),2),
                        Fondos_Propios = round(mean(FPIOS),0),
                        Margen = round(mean(MARGEN),2),
                        Resultado = round(mean(RES),0),
                        Rentabilidad_Financiera = round(mean(RENFIN),2),
                        Apalancamiento = round(mean(APALANCA),2))

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

tablamedias %>%
  kable(caption = "Método de Ward. 5 grupos. Medias de variables") %>%
  kable_styling(full_width = F, bootstrap_options = "striped",
"bordered", "condensed", position = "center", font_size = 11) %>%
```

```
  row_spec(0, bold= T, align = "c") %>%
  row_spec(1:5, bold= F, align = "c")
```

El resultado obtenido es el siguiente:

Método de Ward. 5 grupos. Medias de variables

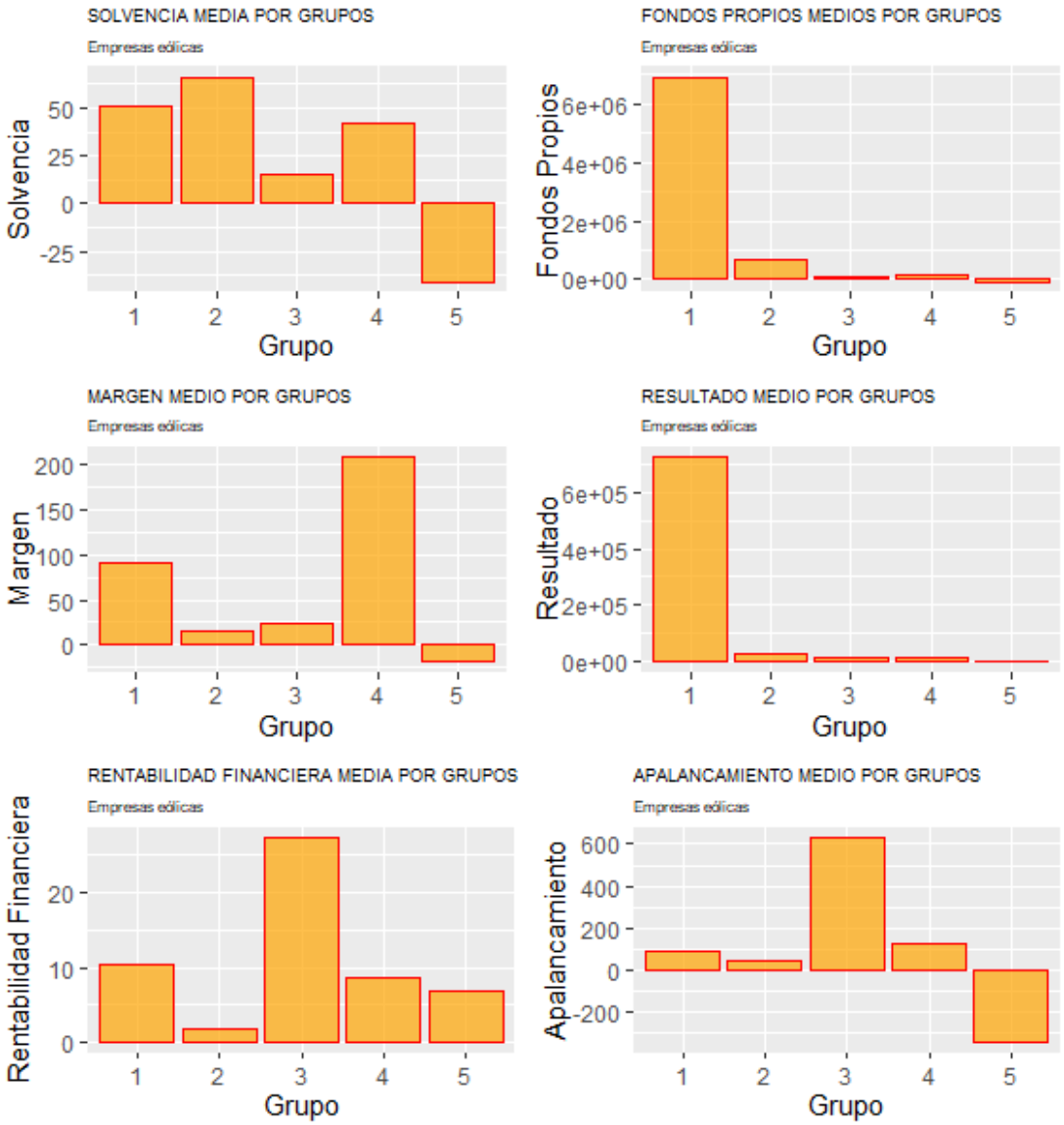
whatcluster_j	obs	Solvencia	Fondos_Propios	Margen	Resultado	Rentabilidad_Financiera	Apalancamiento
1	1	51.17	6904824	91.15	727548	10.29	91.96
2	5	66.07	679149	15.93	21691	1.85	49.83
3	9	14.98	76425	23.95	13153	27.20	628.85
4	1	42.01	186302	208.36	12819	8.61	123.77
5	1	-40.74	-77533	-19.19	-5661	6.90	-343.54

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

```
gsolve <- ggplot(data = tablamedias, map = (aes(y = Solvencia, x =
whatcluster_j))) +
           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))


gfpios <- ggplot(data = tablamedias, map = (aes(y = Fondos_Propios, x =
whatcluster_j))) +
           geom_bar(stat = "identity", colour = "red", fill = "orange",
alpha = 0.7) +
          ggtitle("FONDOS  PROPIOS  MEDIOS  POR  GRUPOS",  subtitle  =
"Empresas eólicas") +
          xlab ("Grupo") +
          ylab("Fondos Propios") +
          theme(plot.title=  element_text(size=7),  plot.subtitle  =
element_text(size = 6))

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

gresul <- ggplot(data = tablamedias, map = (aes(y = Resultado, x =
whatcluster_j))) +
```

```
  geom_bar(stat = "identity", colour = "red", fill = "orange", alpha =
0.7) +
  ggtitle("RESULTADO MEDIO POR GRUPOS", subtitle = "Empresas eólicas")
+
  xlab ("Grupo") +
  ylab("Resultado") +
  theme(plot.title=     element_text(size=7),     plot.subtitle     =
element_text(size = 6))

grentf   <-   ggplot(data   =   tablamedias,   map   =   (aes(y   =
Rentabilidad_Financiera, x = whatcluster_j))) +
  geom_bar(stat = "identity", colour = "red", fill = "orange", alpha =
0.7) +
  ggtitle("RENTABILIDAD  FINANCIERA  MEDIA  POR  GRUPOS",  subtitle  =
"Empresas eólicas") +
  xlab ("Grupo") +
  ylab("Rentabilidad Financiera") +
  theme(plot.title=     element_text(size=7),     plot.subtitle     =
element_text(size = 6))

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

library (patchwork)

(gsolve + gfpios) / (gmargen + gresul) / (grentf + gapala)
```

Los 6 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 (que en realidad es una única empresa *outlier*, "Holding de negocios de gas"). Se caracteriza sobre todo por poseer unos fondos propios, y resultado muy superior al resto de los grupos (a la media, si son grupos de más de una empresa). El grupo 2 tiene, en media, una mayor solvencia y menor apalancamiento; mientras que la rentabilidad financiera en la menor. El grupo 3 destaca por poseer, en media, la mayor rentabilidad financiera y grado de apalancamiento (y, por tanto, la menor solvencia). El grupo 4 (que en realidad solo se compone de la empresa "Elawan Energy", el otro caso *outlier*), destaca por tener un elevado margen. Finalmente, el grupo 5, formado únicamente por la empresa "Guzmán Energía", destaca por alcanzar valores negativos en solvencia, apalancamiento, y margen.

Del mismo modo, se pueden presentar en diferentes tablas la información de cada grupo:

```
originales %>% filter(whatcluster_j == "1")%>%
  select ( SOLVENCIA, FPIOS, MARGEN, RES, RENFIN, APALANCA) %>%
  kable(caption = "Método de Ward. Grupo 1.") %>%
  kable_styling(full_width = F, bootstrap_options = "striped",
"bordered", "condensed", position = "center", font_size = 12) %>%
  row_spec(0, bold= T, align = "c")
```

Con lo que se obtendrá:

Método de Ward. Grupo 1.

	SOLVENCIA	FPIOS	MARGEN	RES	RENFIN	APALANCA
Holding De Negocios De GAS SL.	51.174	6904824	91.152	727548	10.287	91.964

Cambiando el grupo (“whatcluster_j”) en el filtro, se obtendrán el resto:

Método de Ward. Grupo 2.

	SOLVENCIA	FPIOS	MARGEN	RES	RENFIN	APALANCA
Global Power Generation SA.	86.917	1740487.00	22.403	39995.000	1.603	1.044
EDP Renovables España SLU	56.960	726783.00	47.193	67033.000	11.338	67.028
Saeta Yield SA.	83.489	665319.56	16.258	2084.476	0.432	17.067
Viesgo Renovables SL.	65.883	177707.00	11.818	4609.000	3.200	13.330
Innogy Spain SA.	37.096	85447.21	-18.025	-5268.573	-7.302	150.688

Método de Ward. Grupo 3.

	SOLVENCIA	FPIOS	MARGEN	RES	RENFIN	APALANCA
Naturgy Renovables SLU	16.274	318475.00	20.442	42737.000	12.043	494.729
Corporacion Acciona Eolica SL	15.737	136064.00	20.091	29592.000	28.990	422.263
Olivento SL	15.304	58341.00	16.629	7388.175	16.684	534.761
Parque Eolico La Boga SL.	9.646	29316.80	1.001	11.940	1.684	921.591
Naturgy Wind, S.L.	10.388	28418.00	39.575	8500.000	38.018	824.537
Al-Andalus Wind Power SL	8.591	21466.12	12.582	4403.214	27.350	1019.616
Acciona Eolica Del Levante SL	11.557	21769.00	27.520	6853.000	43.139	743.754
Esquilvent SL	30.938	48769.13	39.476	9010.214	24.633	218.275
Eolica La Janda SL	16.428	25206.75	38.256	9880.091	52.261	480.122

Método de Ward. Grupo 4.

	SOLVENCIA	FPIOS	MARGEN	RES	RENFIN	APALANCA
Elawan Energy SL.	42.01	186302	208.357	12818.98	8.605	123.771

Método de Ward. Grupo 5.

	SOLVENCIA	FPIOS	MARGEN	RES	RENFIN	APALANCA
Guzman Energia SL	-40.745	-77532.7	-19.193	-5661.463	6.904	-343.542

Componentes principales.

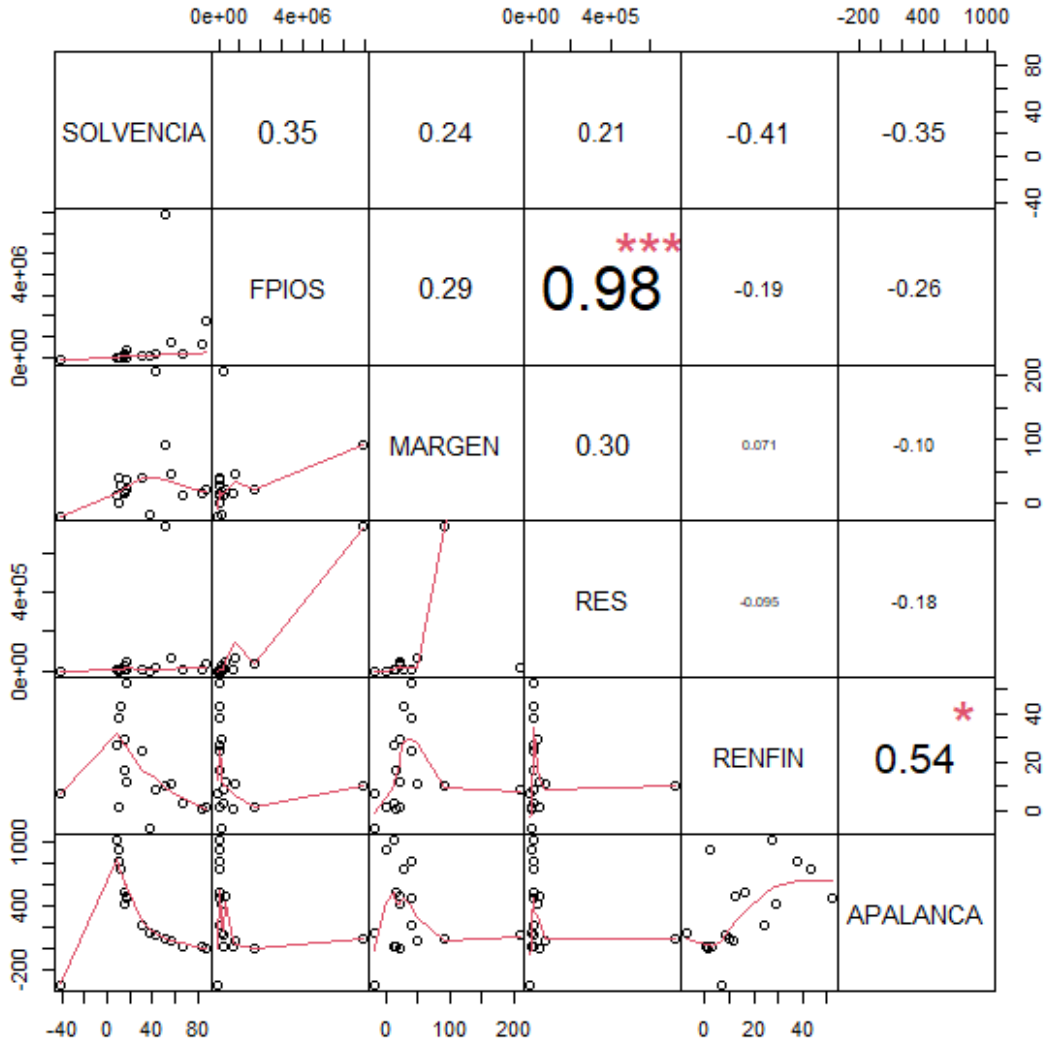
Un problema que acarrea el usar las variables originales como clasificadoras, cuando son más de 2, es que es difícil trazar un gráfico de dispersión que nos dé una idea precisa de los grupos formados en cuanto a su composición y a lo "separados" que se encuentran unos de otros.

Una idea alternativa puede ser 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_j", pues es un factor (no es una variable en escala métrica). Este data frame será llamado, por ejemplo, "originales_cp":

```
originales_cp <- originales %>% select(-whatcluster_j)
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 elevadas correlaciones. Destacan el caso de FPIOS con RES, y de RENFIN con APALANCA:



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     PC5     PC6
Standard deviation     1.6035 1.2188 0.9530 0.7869 0.63683 0.10069
Proportion of Variance 0.4286 0.2476 0.1514 0.1032 0.06759 0.00169
Cumulative Proportion  0.4286 0.6761 0.8275 0.9307 0.99831 1.00000
```

Como comprobamos, las dos primeras componentes acumulan ya más de un 67% de la varianza total o comunalidad (información) que las variables originales guardan 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     PC5     PC6
SOLVENCIA -0.3914  0.2801  0.3789 -0.7186  0.3110 -0.1078
FPIOS     -0.5460 -0.3179 -0.2863 -0.0746  0.0238  0.7161
MARGEN    -0.2666 -0.2925  0.8058  0.3217 -0.2990  0.0326
RES       -0.5022 -0.4142 -0.3158  0.0321 -0.0387 -0.6885
RENFIN     0.3097 -0.5886  0.1592 -0.0006  0.7295  0.0142
APALANCA   0.3599 -0.4658 -0.0031 -0.6111 -0.5288  0.0204
```

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.

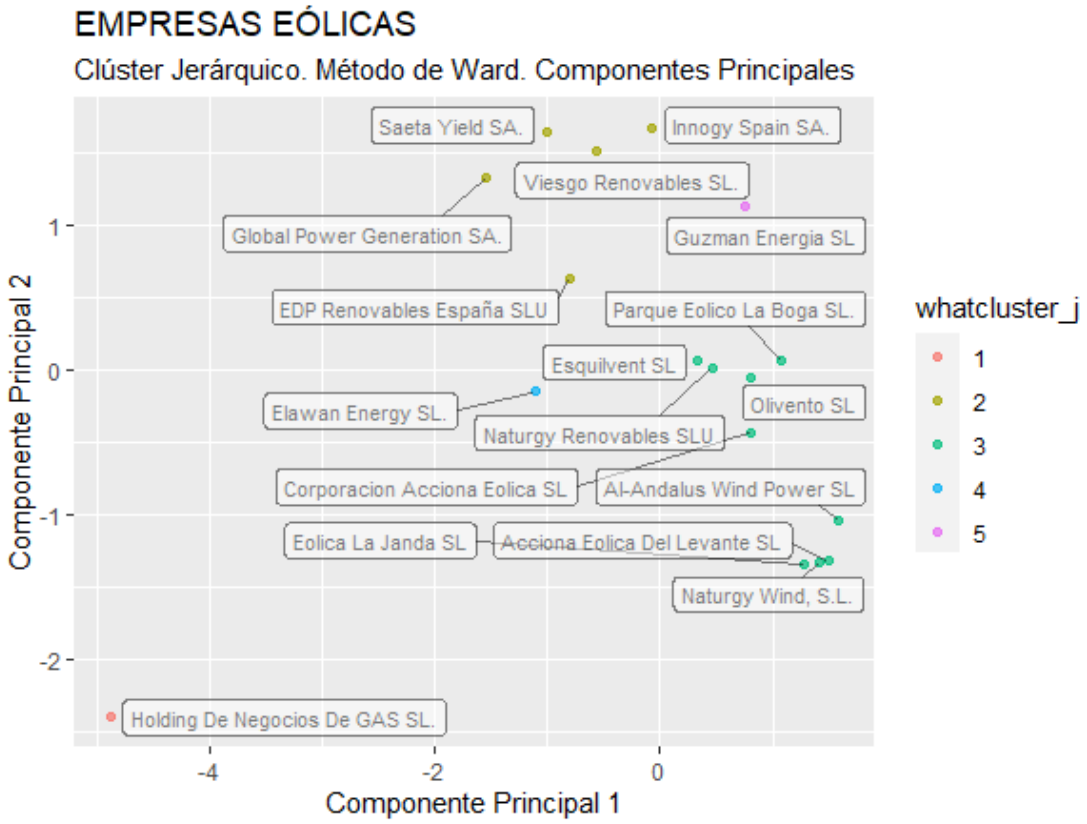
Para utilizar las puntuaciones de las dos primeras componentes, hemos de obtener sus puntuaciones de cada caso (empresa). Estas puntuaciones están guardadas en la matriz "x" del objeto "prcomp" creado ("componentes"). Vamos a crear un *data frame* con las variables originales, el grupo de pertenencia, 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, Componente.1, Componente.2)
summary (Componentes)
```

Para realizar el gráfico de dispersión, además de la función ggplot() del paquete ggplot2, se utilizará geom_label_repel() del paquete ggrepel; que permitirá etiquetar los casos de un modo cómodo, evitando etiquetas superpuestas:

```
ggplot(data = Componentes, map = (aes(x = Componente.1, y = Componente.2,
col = whatcluster_j))) +
  geom_point(alpha = 0.7) +
  geom_label_repel(aes(label = row.names(Componentes)), size = 3, color
= "black", alpha = 0.5) +
  ggtitle("EMPRESAS EÓLICAS", subtitle = "Clúster Jerárquico. Método de
Ward. Componentes Principales") +
  xlab("Componente Principal 1") +
  ylab("Componente Principal 2")
```

El resultado obtenido es el siguiente gráfico:



En el gráfico se puede observar el claro papel de *outlier* de la empresa "Holding de Negocios", sobre todo en lo que respecta a la primera componente principal. Esto se debe a su especial comportamiento en las variables FPIOS y RES, que son, precisamente, las variables con mayor peso (con signo negativo) en tal componente. En cambio, en el gráfico no existe, por ejemplo, una posición muy singular de la empresa "Guzmán Energía", a pesar de constituir un grupo particular. Esto se puede deber a que su

comportamiento especial se da en las variables SOLVENCIA y APALANCA; pero la primera de ellas no tiene un peso notable en ninguna de las dos primeras componentes.

Updated: 07/11/2022