Minería de datos I

José Domingo Mateo Vázquez - Máster MECOFIN

Sobre la Base de datos

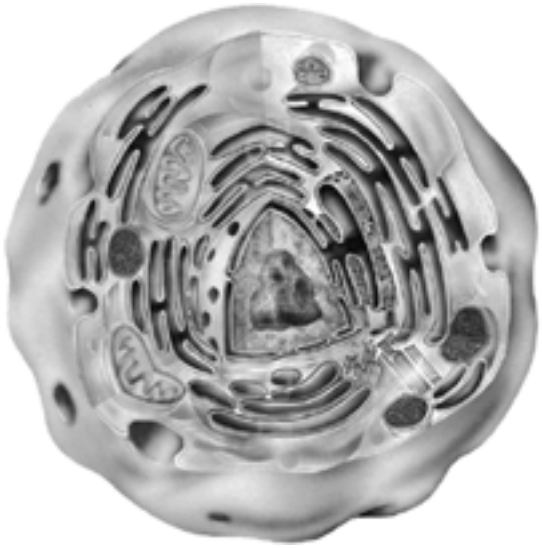
La autoría de esta Base de datos es de William Wolberg, Nick Street y Olvi Mangasarian. Las variables han sido calculadas gracias a la imagen digitalizada de una biopsia del tejido mamario y describen características del núcleo de las células. Las técnicas usadas están descritas en el artículo de K.P. Bennett y O.L. Mangasarian: "Robust Linear Programming Discrimination of Two Linearly Inseparable Sets", Optimization Methods and Software 1, 1992, 23-34

La base de datos ha sido descargada de la web Kaggle.com, en la dirección: http://bit.ly/2qse1el. También está disponible a través del servidor ftp de la UW CS. También puede encontrarse en el repositorio de Machine Learning de la UCI en la dirección: http://bit.ly/1L1zT4y

Información de las variables:
1) ID
2) Diagnóstico (M = Maligno, B = Benigno)
3-32) Diez características calculadas para cada núcleo de la célula:

a) El radio (radius) - La media de la distancia desde el centro hasta los puntos del perímetro
b) La textura (texture) - La desviación estándar de los valores en escala de grises
c) El perímetro (perimeter)
d) El área (area)
e) La suavidad (smoothness) - Variación local en las longitudes del radio
f) Compactación (compactness) - Perímetro^2/(área-1)
g) Concavidad (concavity) - Gravedad de las partes cóncavas del contorno
h) Puntos cóncavos (concave points) -Número de porciones cóncavas del contorno
i) Simetría (symmetry)
j) Dimensión fractal (fractal dimensión) - "Aproximación de los bordes costeros"-1

La media, el error estándar y la peor o la más grande de cada una de estas características han sido calculadas por ordenador para cada imagen, resultando en 30 variables diferentes. Todos los valores aparecen con cuatro cifras significativas. No hay valores perdidos. En total hay 357 casos benignos y 212 malignos.



K-MEANS

Aunque la finalidad del algoritmo K-Means no es la de la clasificación, la base de datos incorpora una variable que diferencia entre tumores benignos y malignos. Por ello voy a usar el algoritmo con ésta finalidad. Mi idea es acabar encontrando la configuración que mejor clasifique los datos. Que maximice los aciertos.

Lo primero que hago es borrar la memoria de R, cargar la base de datos y eliminar las variables cualitativas id y diagnosis:

```
rm(list= ls())
setwd("D:/Universidad/Master/Mineria de datos I/05 - Trabajo final")
datos <- read.csv("cancer_winsconsin.csv", header = TRUE)
bdkmeans <- datos[ -c(1:2) ]
```

Para ver como se relacionan las variables entre sí, hago un diagrama de dispersión con el siguiente comando:

```
#plot(bdkmeans[,]) Silencio el gráfico porque es muy grande y da error. Lo adjunto aparte.
```

El resultado del diagrama de dispersión muestra una gran varianza entre algunas de las variables, así como la existencia de valores atípicos que pueden influir negativamente en nuestro modelo de clasificación.
Para eliminar estos efectos voy a probar diferentes transformaciones de los datos: obteniendo su logaritmo, estandarizándolos y eliminando los outliers igualando su valor a los datos de determinados percentiles.

```
#Calculo del logaritmo
bdkmeans <- log10(bdkmeans)
#Estandarizador
bdkmeans <- scale(bdkmeans)
bdkmeans <- data.frame(bdkmeans[,colSums(is.na(bdkmeans))<nrow(bdkmeans)])
#Eliminador de outliers
percentilup <- 85
percentildown <- 15
for (columna in colnames(bdkmeans)){
  up <-c("bdkmeans$",columna,"[bdkmeans$",columna,">quantile(bdkmeans$",columna,",",
         percentilup*0.01,")] <- quantile(bdkmeans$",columna,",", percentilup*0.01,")")
  down <-c("bdkmeans$",columna,"[bdkmeans$",columna,"<quantile(bdkmeans$",columna,", ",
           percentildown*0.01,")] <- quantile(bdkmeans$",columna,", ",
           percentildown*0.01,")")
  up <-paste(up, collapse="")
  down <-paste(down, collapse="")
  eval(parse(text=up))
  eval(parse(text=down))
}
```

En este punto si volvemos a dibujar el diagrama de dispersión y podemos ver los cambios:

```
#plot(bdkmeans[,]) Silencio el gráfico porque es muy grande y da error. Lo adjunto aparte.
```

Podemos comprobar en el diagrama de dispersión, como con los outliers al 5% los datos son heterocedásticos, y cómo se transforman en datos homocedásticos cuando eliminamos los outliers al 15%. En este punto ya los datos parecen correctos y procedo a hacer el K-Means. En cuanto al número de grupos le indico que cree dos, para ver si es capaz de clasificar por un lado los tumores malignos y por el otro los benignos. El K-Means es un algoritmo iterativo que resuelve un problema de optimización.

```
kmeans_cancer <-  kmeans(bdkmeans,2, iter.max =1000, nstart = 1000)
```

El resultado que devuelve es un mínimo local que no nos garantiza que la solución sea la mejor globalmente, por ello pueden obtenerse soluciones diferentes en función del punto en el que comencemos a iterar. Para intentar alcanzar un mínimo global, le indico que comience a iterar en mil puntos diferentes y que haga un

máximo de mil iteraciones en cada uno de los inicios.

Creando una tabla de contingencia podemos comprobar cómo ha clasificado el algoritmo los grupos y su correspondencia con la diagnosis.

```
tabla <- table(datos$diagnosis, kmeans_cancer$cluster)
tabla
```

```
##
##       1   2
##   B  10 347
##   M 190  22
```

Después de esto podemos calcular el porcentaje de acierto que tuvo el algoritmo a la hora de agrupar los casos en los grupos benigno y maligno.

```
aciertos <- max(tabla[1,]) + max(tabla[2,])
fallos <- min(tabla[1,])+ min(tabla[2,])
por_ac = aciertos*100/569
por_fa = 100-por_ac
por_ac
```

```
## [1] 94
```

Este caso en concreto es el que mejor clasifica los datos. Hice además de éste otras pruebas con diferentes ajustes que paso a resumir en la siguiente tabla:

Normalizados	Logaritmo	Sin outliers al 5%	Sin outliers al 10%	Sin outliers al 15%	% de acierto
					85,41%
1					91,03%
2	1				93,05%
		1			86,82%
			1		87,34%
				1	89,10%
1		2			91,74%
1			2		93,67%
1				2	93,84%
2		1			91,22%
2			1		92,44%
2				1	93,32%
3	2	1			92,97%
3	2		1		93,14%
3	2			1	92,97%
2	1	3			93,50%
2	1		3		94,03%
2	1			3	94,37%**

Los números identifican el orden en el que se hicieron las transformaciones. **El modelo con mayor porcentaje de acierto
El mejor resultado se obtiene con las siguientes transformaciones de los datos:
1) La aplicación del logaritmo
2) La normalización de los datos
3) La eliminación de los outliers al 15%

Por último, he hecho una función que sirve para clasificar los nuevos casos. La función es muy simple, simplemente coge la posición de los centroides que devuelve el K-Means y comprueba cuál es el centroide más próximo: el de los tumores malignos o el de los tumores benignos. El código de la función es el siguiente:

```
clasificador <- function(nuevo_punto){
  if (dist(rbind(kmeans_cancer$centers[1],nuevo_punto))
      <dist(rbind(kmeans_cancer$centers[2],nuevo_punto))){
    return ("M")

  }else if (dist(rbind(kmeans_cancer$centers[2],nuevo_punto))
            <dist(rbind(kmeans_cancer$centers[1],nuevo_punto))){
    return ("B")
  }
}
```

Y podemos comprobar cómo clasifica los puntos con cualquier entrada de la base de datos:

```
clasificador(bdkmeans[100,])
```

```
## [1] "B"
```

```
clasificador(bdkmeans[507,])
```

```
## [1] "M"
```

Como puede comprobarse, el clasificador final que hemos obtenido es bastante bueno, ya que es capaz de acertar un 94,53% de los casos. En un trabajo posterior habría que mejorar el modelo e intentar reducir los casos en los que el modelo clasifica como benignos tumores malignos. Si bien estos casos son muy pocos -1,76%, 10 casos sobre un total de 569-, estos errores pueden ser fatales para la persona que los sufre. Estaríamos reduciendo sus posibilidades de sobrevivir. A su vez, sería interesante comprobar si esta clasificación errónea se debe a que estos tumores malignos se encuentran en un estado inicial de desarrollo que motiva sus valores atípicos. Otra posible vía de estudio sería centrarse en cada una de las variables de forma individual, y ver si alguna de ellas arroja valores que puedan alertarnos de que el tumor es maligno.

CLUSTER JERÁRQUICO

El clúster jerárquico o dendrograma es otro método de aprendizaje no supervisado. En este caso también he decidido separar las observaciones en dos grupos: los benignos y los malignos, y ver cómo el algoritmo los separa. En primer lugar, vuelvo a cargar los datos de la base de datos, sin las transformaciones del ejercicio anterior:

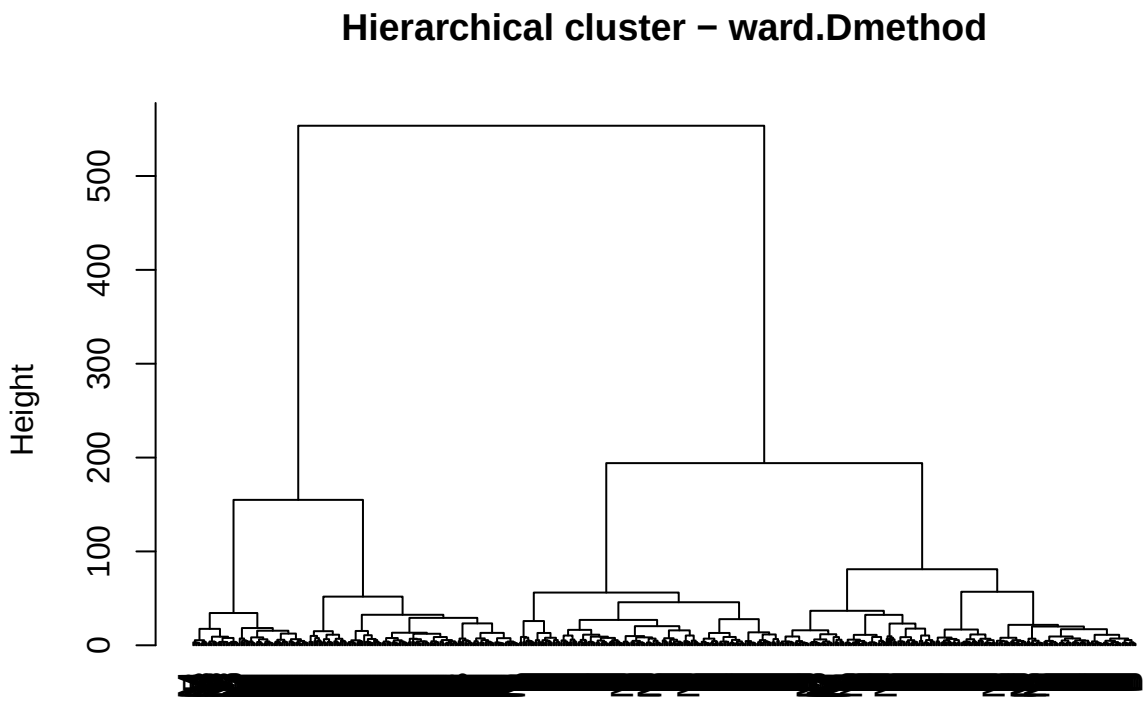
```
bdkmeans <- datos[ -c(1:2) ]
```

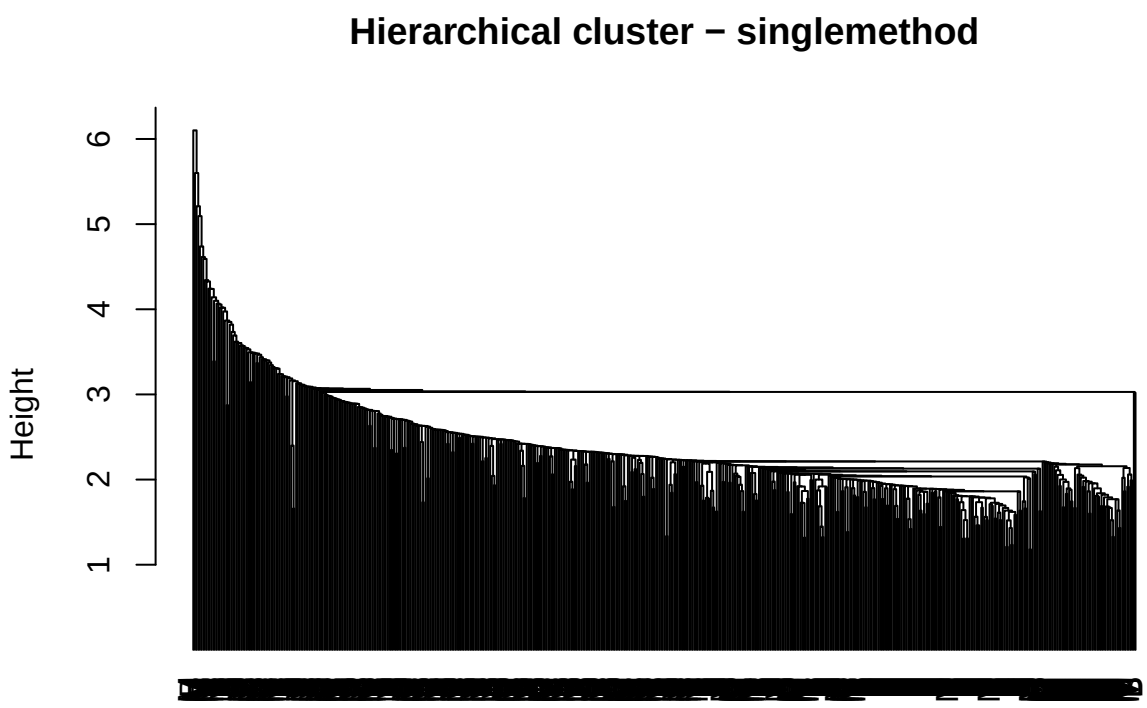
Como la covarianza de mis variables es muy dispar, he optado por transformarlas calculándoles el logaritmo y normalizándolos.

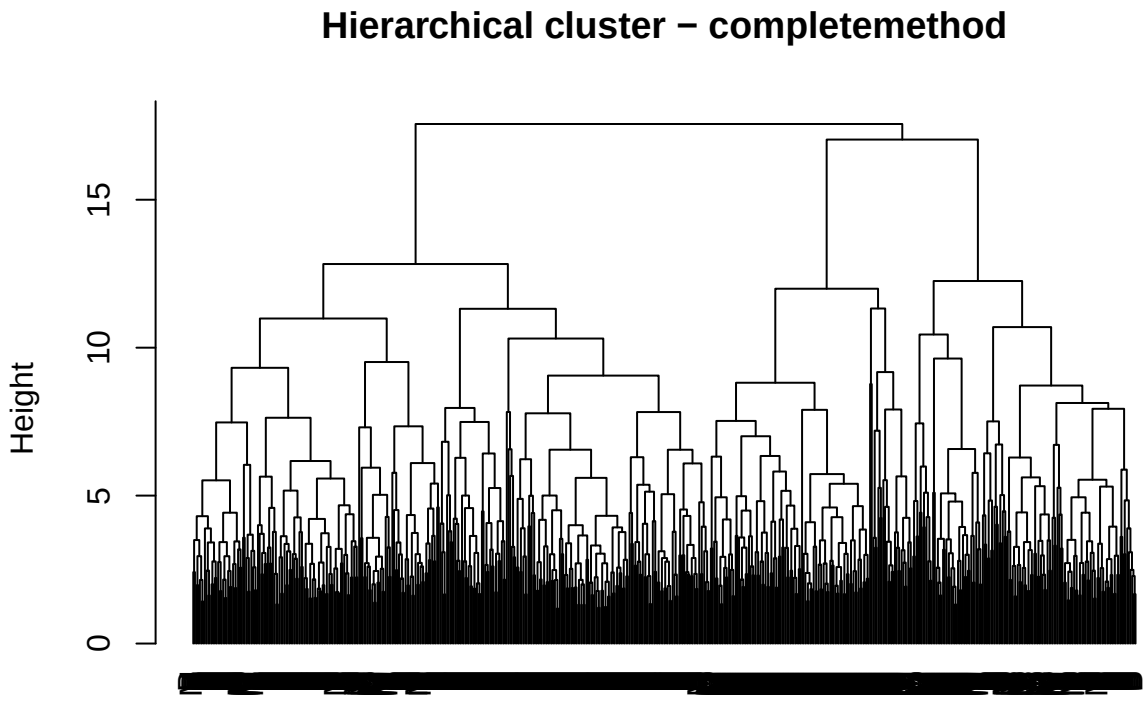
```
bdkmeans <- log10(bdkmeans)
bdkmeans <- scale(bdkmeans)
bdkmeans <- data.frame(bdkmeans[,colSums(is.na(bdkmeans))<nrow(bdkmeans)])
```

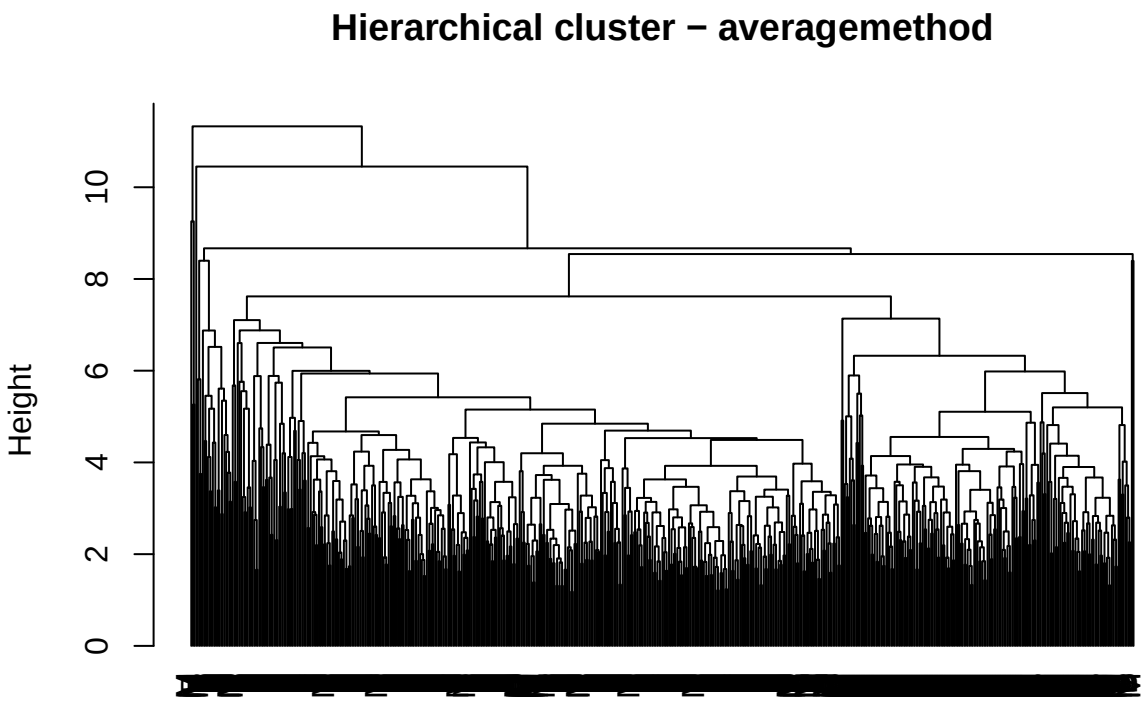
Tras esto, hay que decidir con qué tipo de medida de disimilitud queremos hacer el clúster jerárquico. La función que R tiene implementada permite diferentes opciones. He optado por crear un bucle y realizar el dendrograma con todas las opciones.

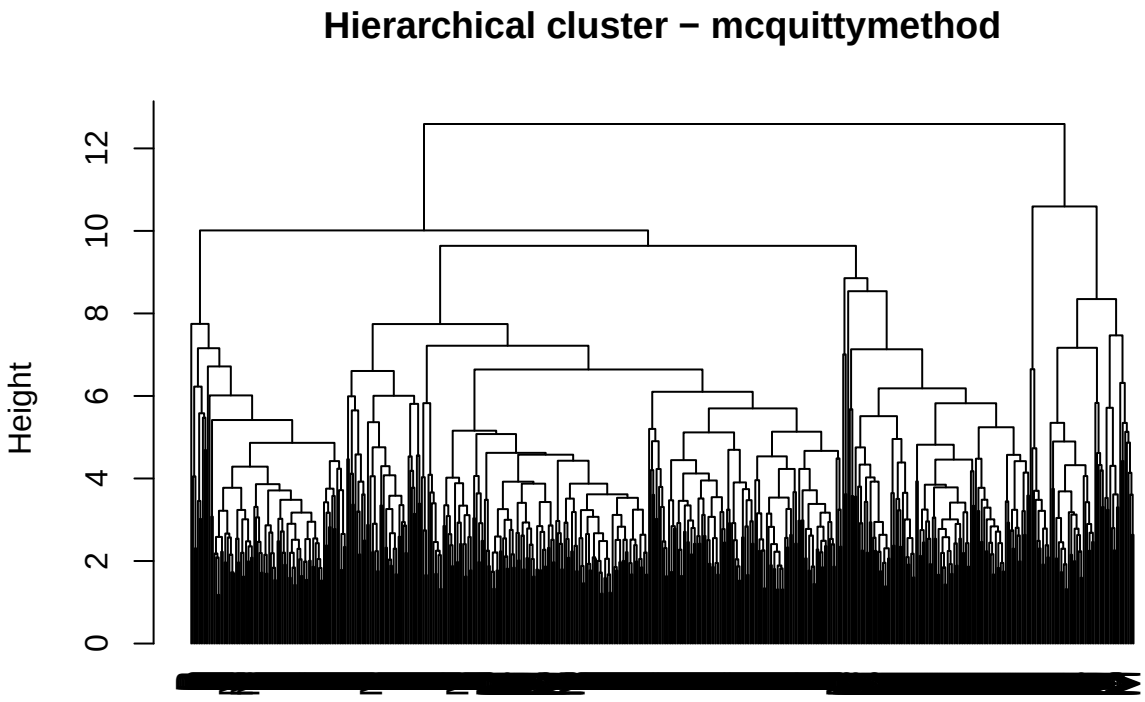
```
for (i in c("ward.D","single","complete","average","mcquitty","median","centroid")){
  hc <- hclust(dist(bdkmeans), method=i)
  par(mar=c(0, 4, 4, 2))
  plot(hc, xlab="", sub="", hang = -1,labels=datos$diagnosis,
       main=paste(c("Hierarchical cluster - ", i, "method"),collapse=""))
}
```

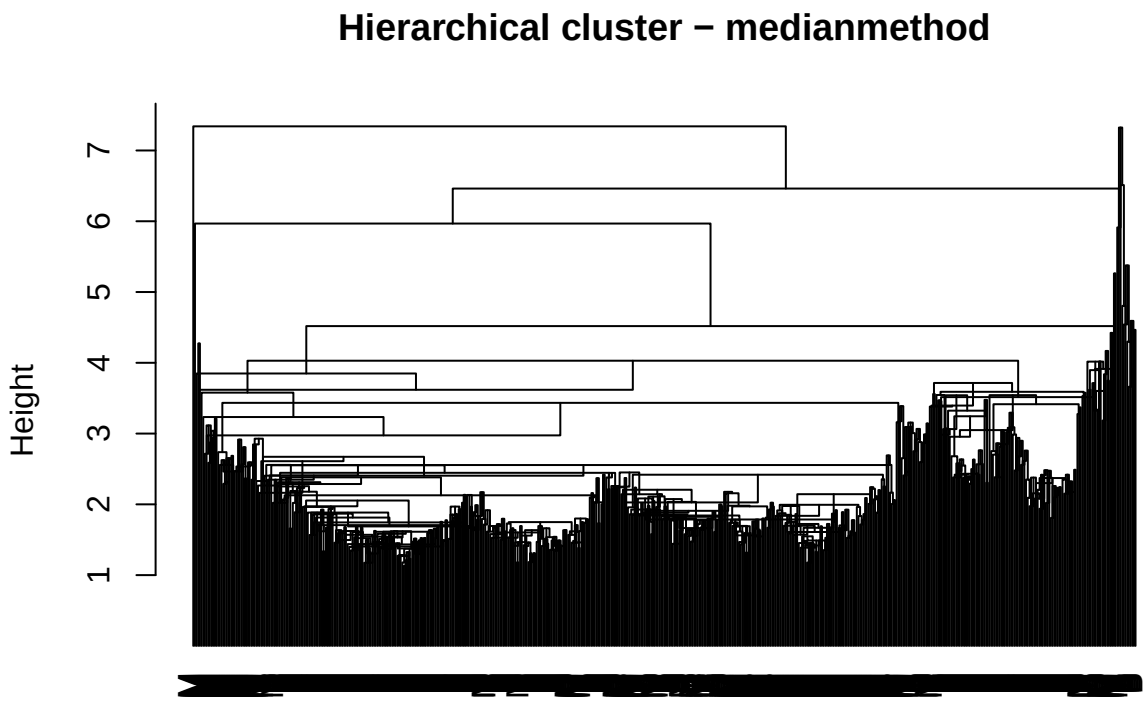


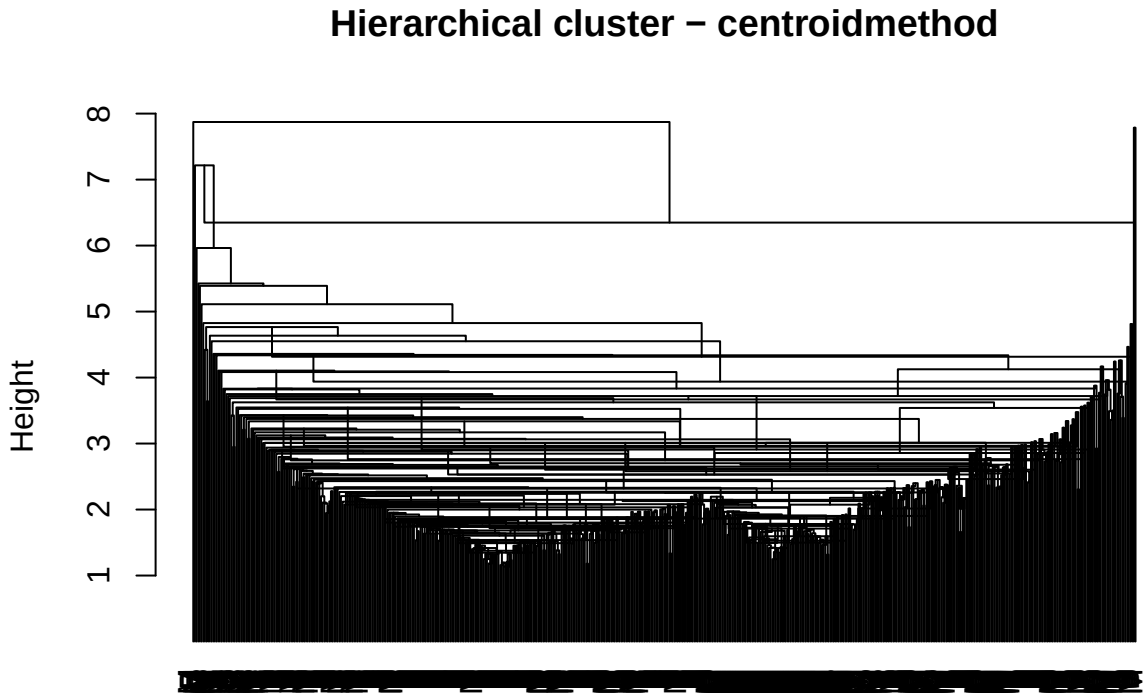

Hierarchical cluster − singlemethod
Height
1
2
3
4
5
6


Hierarchical cluster − completemethod
Height
0
5
10
15


Hierarchical cluster − averagemethod
Height
0
2
4
6
8
10

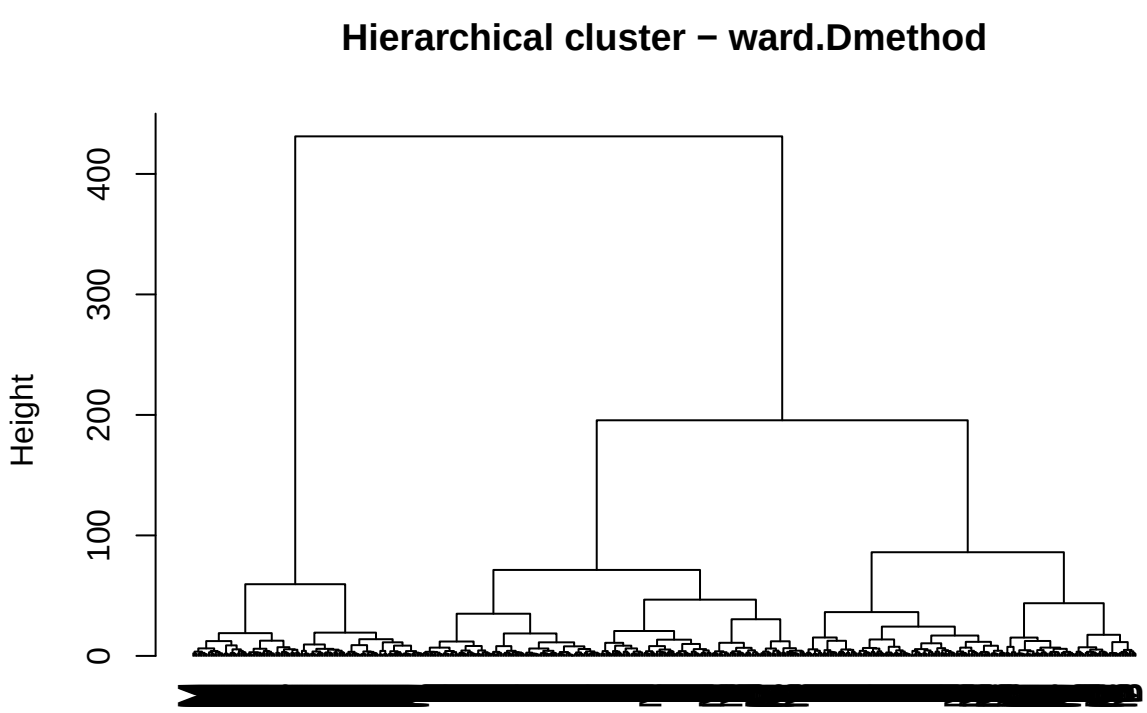

Hierarchical cluster − mcquittymethod
Height
0
2
4
6
8
10
12

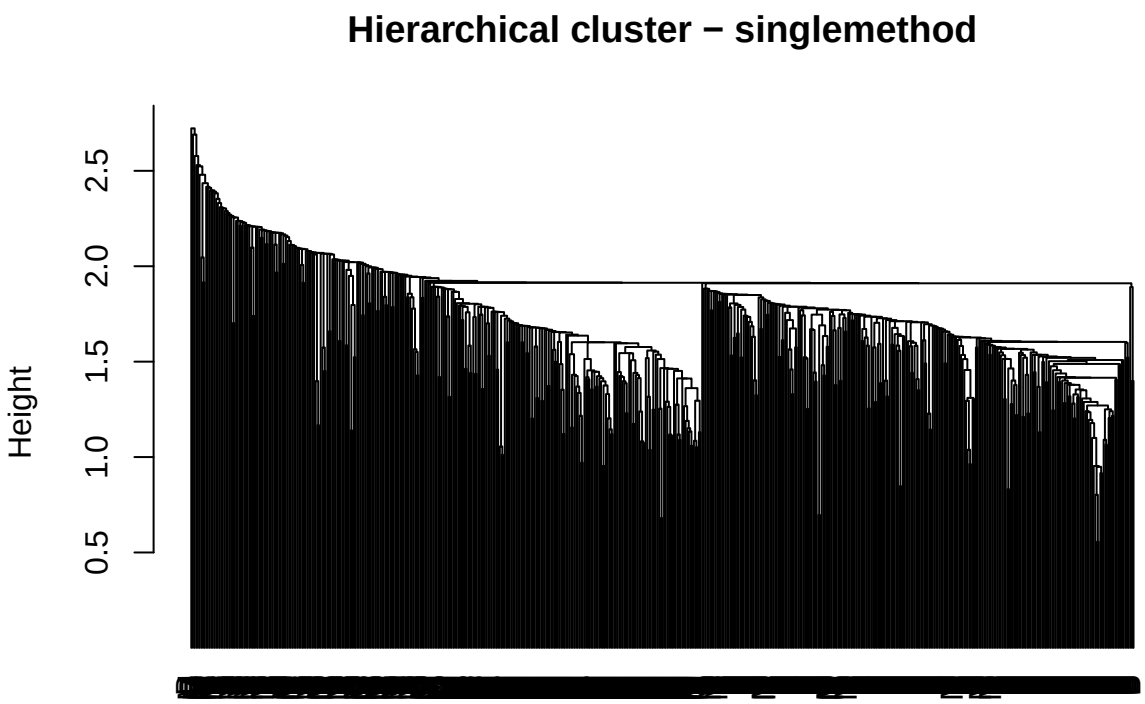


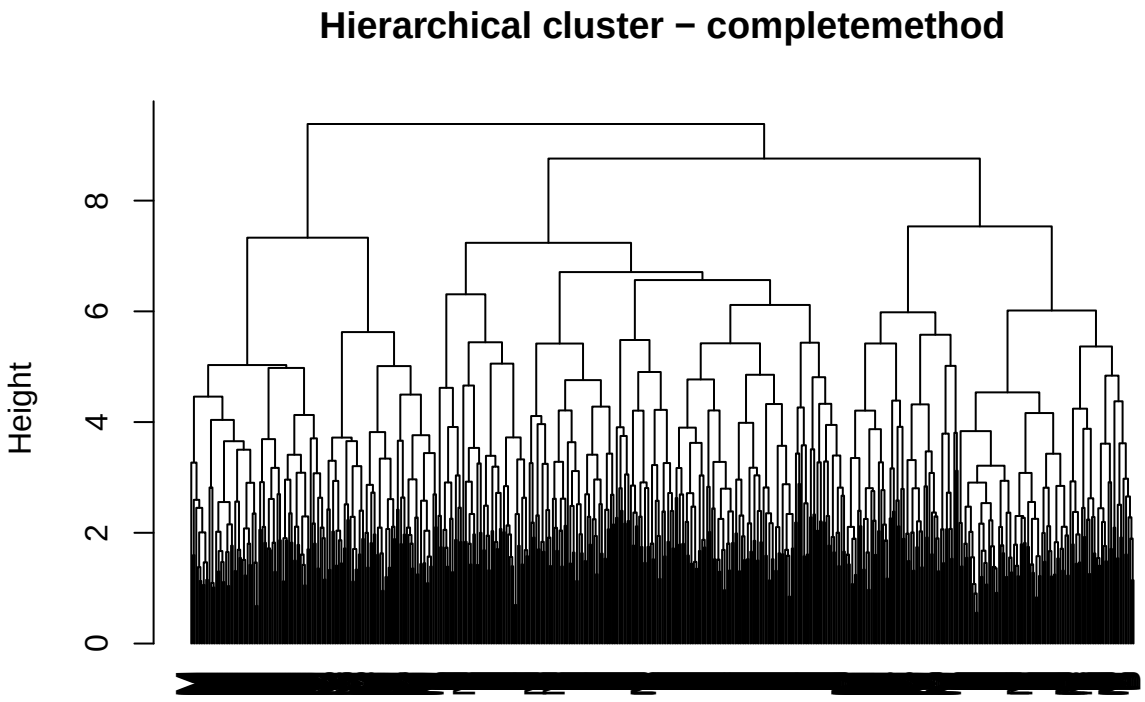


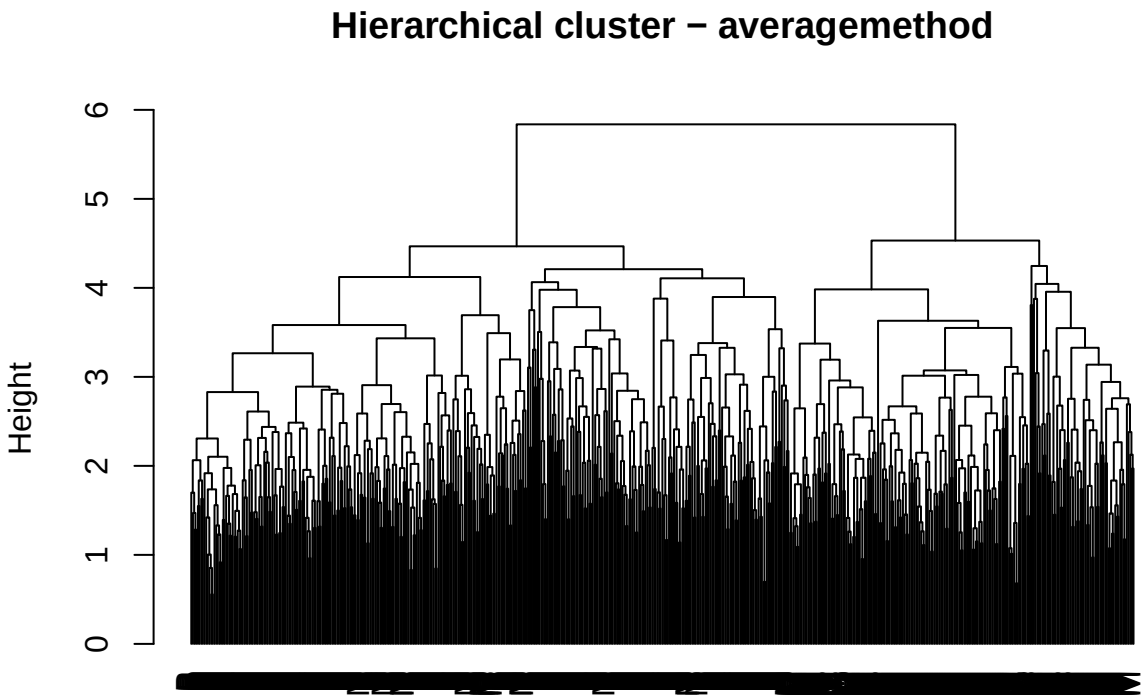
En general, ninguno de los resultados parece bueno. Todos tienen muchísimas ramificaciones y mezclan bastante los casos benignos y malignos. No sabría por qué rama del clúster cortar. Por ello pruebo a transformar los outliers de la muestra al percentil 15 por abajo y al 85 por arriba y volver a realizar los dendrogramas.

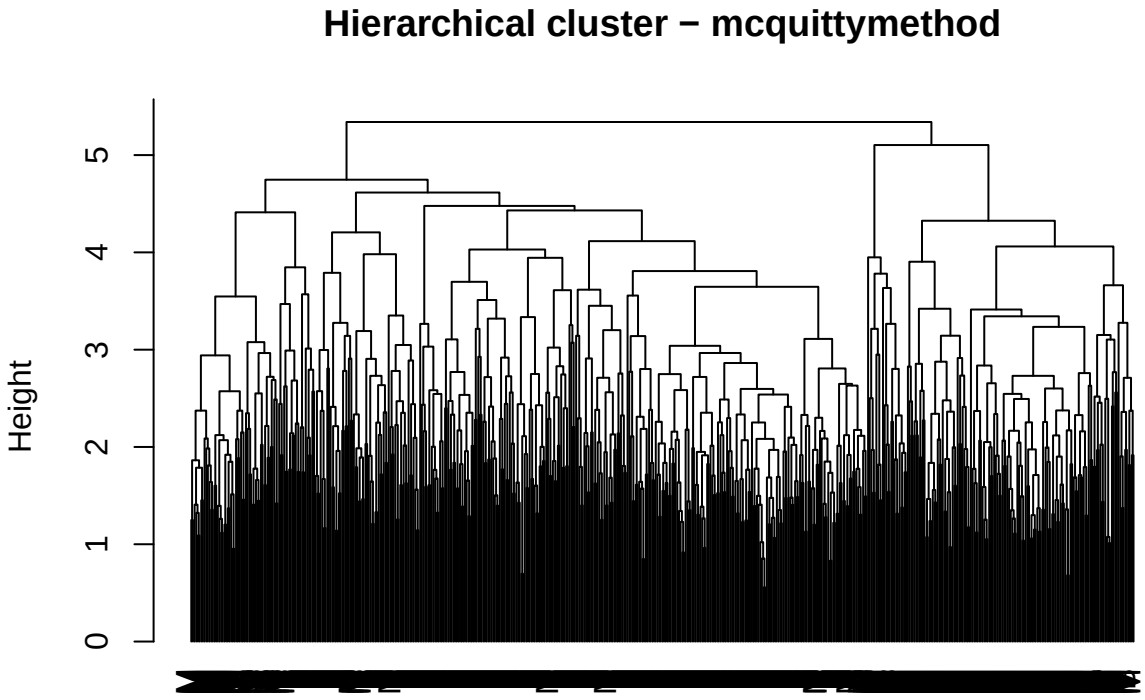
```
percentilup <- 85
percentildown <- 15
for (columna in colnames(bdkmeans)){
  up <-c("bdkmeans$",columna,"[bdkmeans$",columna,">quantile(bdkmeans$",columna,",",
         percentilup*0.01,")] <- quantile(bdkmeans$",columna,",", percentilup*0.01,")")
  down <-c("bdkmeans$",columna,"[bdkmeans$",columna,"<quantile(bdkmeans$",columna,", ",
         percentildown*0.01,")] <- quantile(bdkmeans$",columna,", ",
         percentildown*0.01,")")
  up <-paste(up, collapse="")
  down <-paste(down, collapse="")
  eval(parse(text=up))
  eval(parse(text=down))
}

for (i in c("ward.D","single","complete","average","mcquitty","median","centroid")){
  hc <- hclust(dist(bdkmeans), method=i)
  par(mar=c(0, 4, 4, 2))
  plot(hc, xlab="", sub="", hang = -1,labels=datos$diagnosis,
       main=paste(c("Hierarchical cluster - ", i, "method"),collapse=""))
}
```

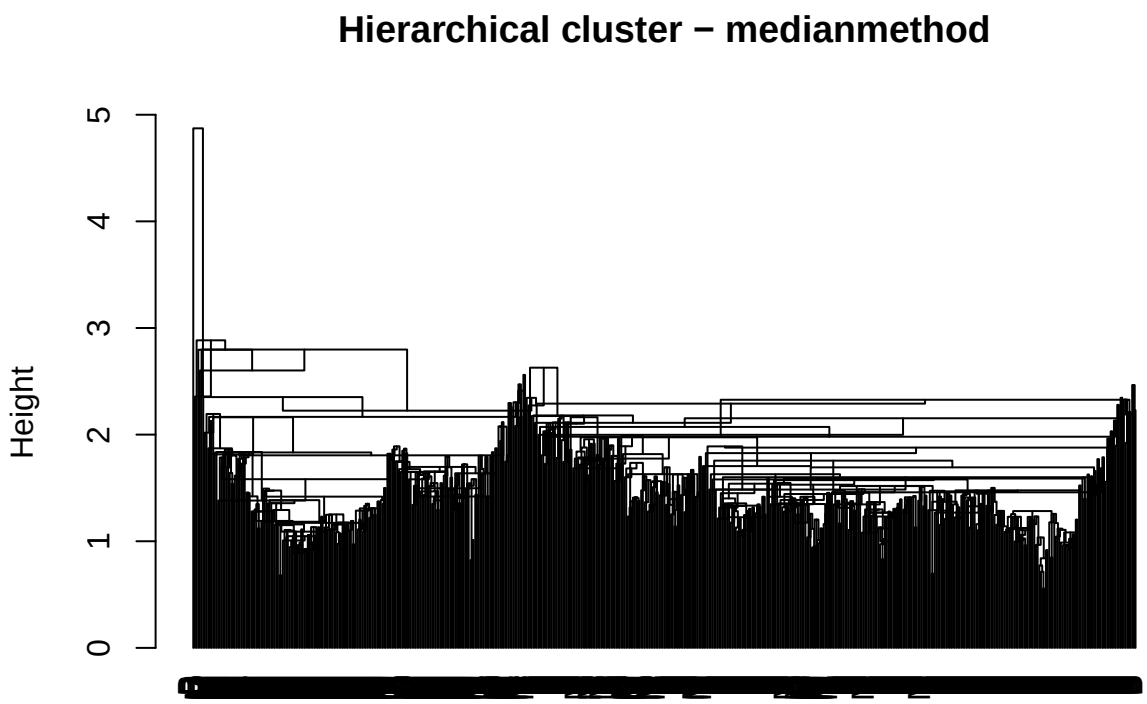


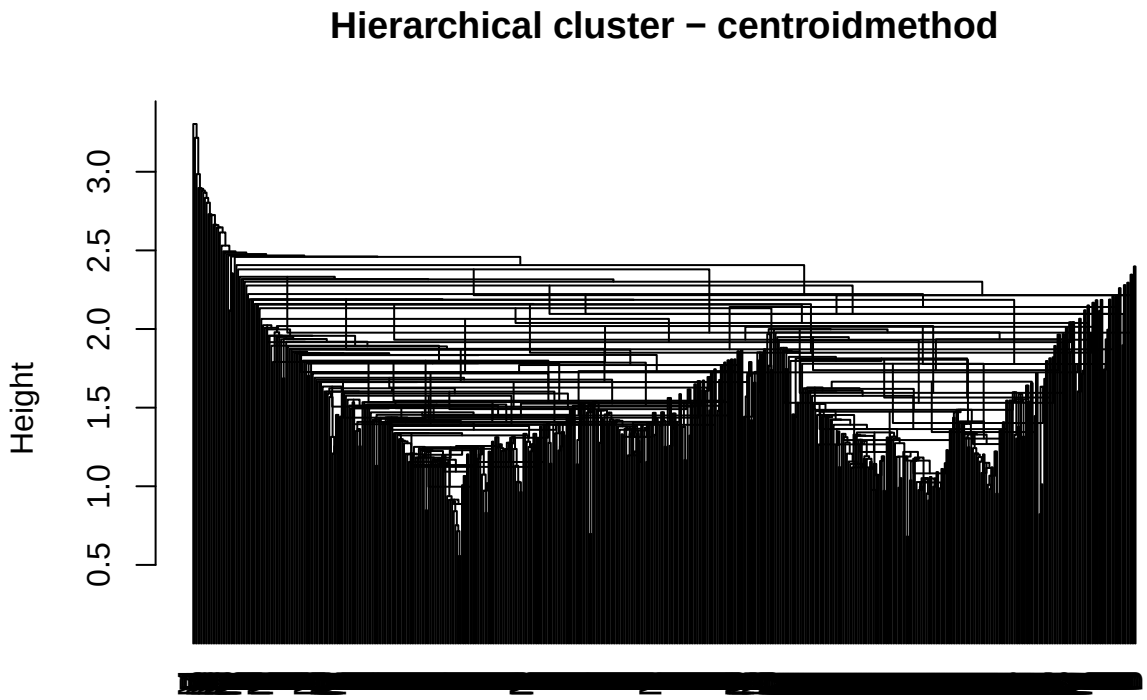

Hierarchical cluster − singlemethod
Height
0.5
1.0
1.5
2.0
2.5


Hierarchical cluster − completemethod
Height
0
2
4
6
8


Hierarchical cluster − averagemethod
Height
0
1
2
3
4
5
6


Hierarchical cluster − mcquittymethod
Height
0
1
2
3
4
5





De todos los métodos el que mejor clasifica es el de la media. Divide los dos grupos de una forma bastante correcta y es fácil identificar el lugar dónde podría cortar las ramas para separar ambos grupos. Para afinar un poco más, en un trabajo más en profundidad habría que analizar los motivos por los que dentro de las dos ramificaciones principales hay algunas ramas con grupos de individuos que pertenecen al otro grupo. Creo que esto es señal de que comparten alguna característica en común que los hace identificables.

ANALÍSIS DE COMPONENTES PRINCIPALES

El análisis de componentes principales nos permite reducir la dimensionalidad de los datos. Reseteo la base de datos y compruebo el resultado de la matriz de covarianza y correlación.

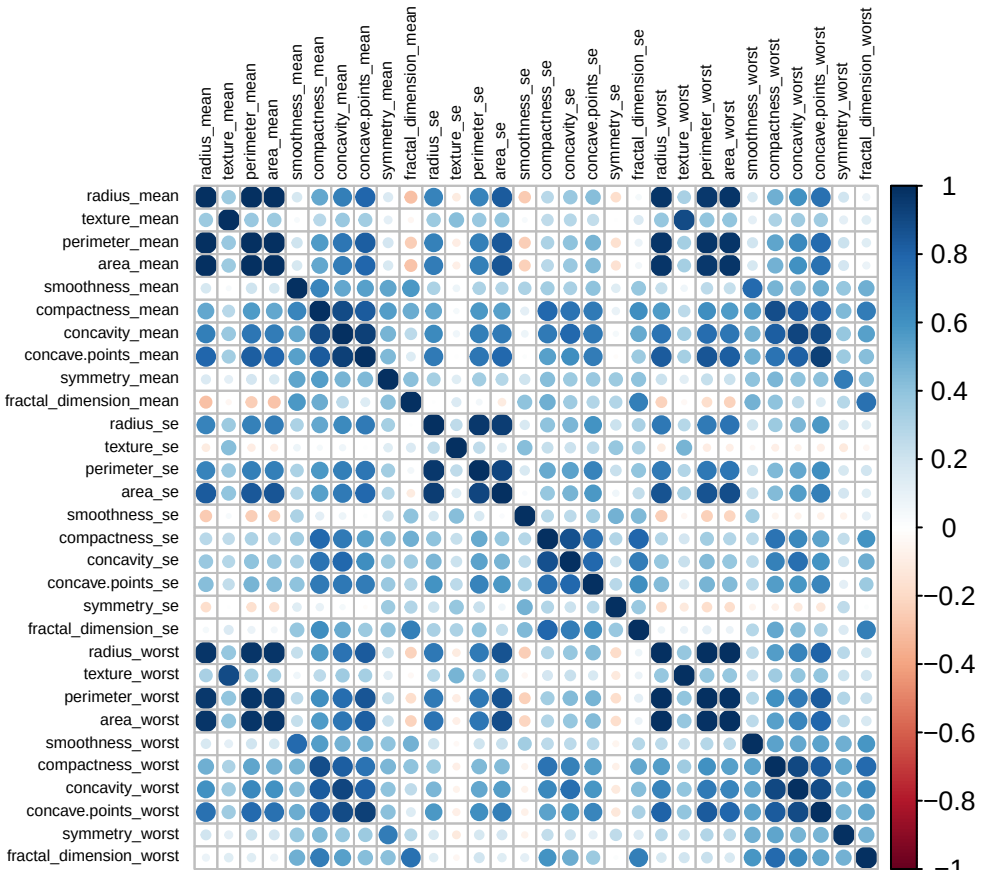
```
bdkmeans <- datos[ -c(1,2) ]
mat_cov <- cov(bdkmeans)
```

La matriz de covarianza muestra resultados muy dispares entre las variables y esto puede influir en el análisis. Por ello procedo a normalizar los datos y a quitar los outliers.

```
bdkmeans <- data.frame(scale(bdkmeans))
percentilup <- 85
percentildown <- 15
for (columna in colnames(bdkmeans)){
  up <-c("bdkmeans$",columna,"[bdkmeans$",columna,">quantile(bdkmeans$",columna,",",
         percentilup*0.01,")] <- quantile(bdkmeans$",columna,",", percentilup*0.01,")")
  down <-c("bdkmeans$",columna,"[bdkmeans$",columna,"<quantile(bdkmeans$",columna,", ",
           percentildown*0.01,")] <- quantile(bdkmeans$",columna,", ", percentildown*0.01,
           ")")
  up <-paste(up, collapse="")
  down <-paste(down, collapse="")
  eval(parse(text=up))
  eval(parse(text=down))
}
```

Vuelvo a calcular la matriz de covarianza y la matriz de correlación. Dibujo esta última para ver la relación entre las variables.

```
mat_cov <- cov(bdkmeans)
mat_cor <- cov2cor(mat_cov)
corrplot::corrplot(mat_cor, tl.cex=0.5, tl.col="black")
```



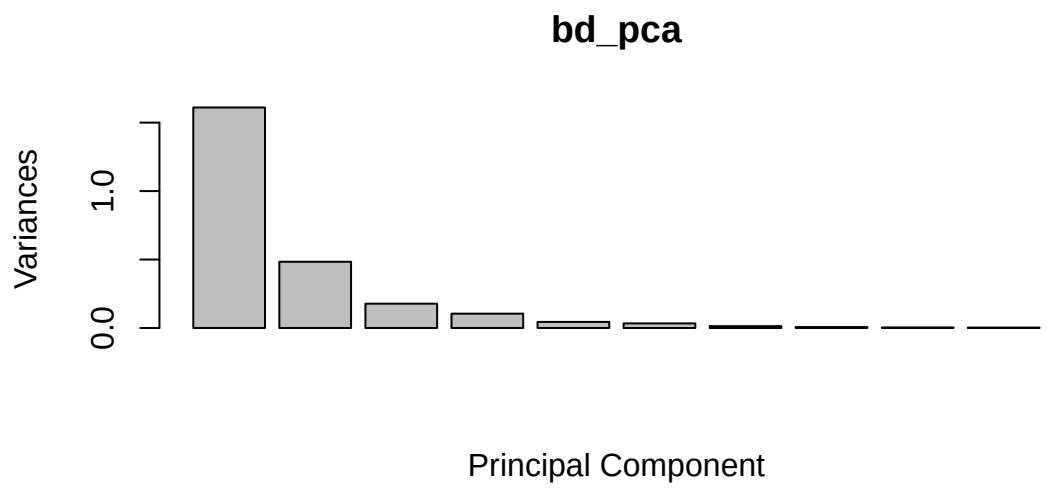
Tras esto realizo el análisis de componentes principales sobre la matriz de correlación.

```
bd_pca <- prcomp(mat_cor)
summary(bd_pca)
```

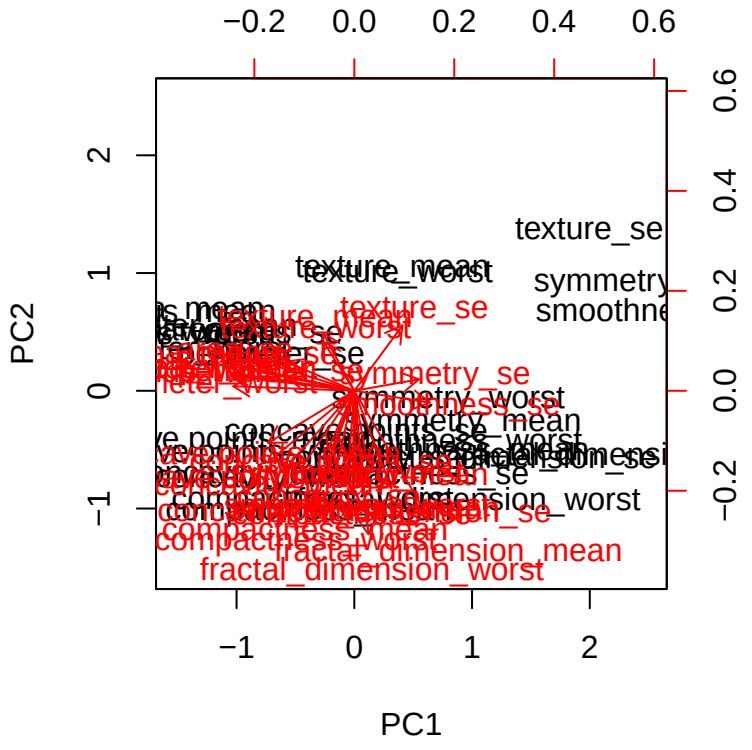
```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     1.269 0.696 0.4214 0.3238 0.2108 0.1838 0.1159
## Proportion of Variance 0.647 0.194 0.0713 0.0421 0.0179 0.0136 0.0054
## Cumulative Proportion  0.647 0.841 0.9128 0.9549 0.9728 0.9864 0.9918
##                           PC8     PC9     PC10     PC11     PC12     PC13
## Standard deviation     0.08228 0.06290 0.05649 0.04412 0.03793 0.03679
## Proportion of Variance 0.00272 0.00159 0.00128 0.00078 0.00058 0.00054
## Cumulative Proportion  0.99449 0.99608 0.99736 0.99815 0.99872 0.99927
##                           PC14     PC15     PC16     PC17     PC18     PC19
## Standard deviation     0.02988 0.01693 0.01439 0.01124 0.00967 0.00874
## Proportion of Variance 0.00036 0.00012 0.00008 0.00005 0.00004 0.00003
## Cumulative Proportion  0.99963 0.99974 0.99982 0.99987 0.99991 0.99994
##                           PC20     PC21     PC22     PC23     PC24     PC25
## Standard deviation     0.00752 0.00583 0.00497 0.00353 0.00275 0.00207
## Proportion of Variance 0.00002 0.00001 0.00001 0.00001 0.00000 0.00000
## Cumulative Proportion  0.99997 0.99998 0.99999 0.99999 1.00000 1.00000
##                           PC26      PC27     PC28     PC29      PC30
## Standard deviation     0.00114 0.000745 0.00025 8.4e-05 1.99e-16
## Proportion of Variance 0.00000 0.000000 0.00000 0.0e+00 0.00e+00
## Cumulative Proportion  1.00000 1.000000 1.00000 1.0e+00 1.00e+00
```

Del análisis podemos inferir como entre los tres primeros componentes principales acumulan un 91,28% de la varianza explicada. Si añadimos el cuarto, acumulan un 95,5% de la varianza explicada. Voy a quedarme con los 4 primeros, aunque como los 3 primeros acumulan mucha varianza voy a utilizarlos para dibujar gráficos con ellos. Podemos ver la varianza y el biplot del análisis a continuación.

```
plot(bd_pca, xlab = "Principal Component", type = "b")
```




```
biplot(bd_pca, scale = 0)
```

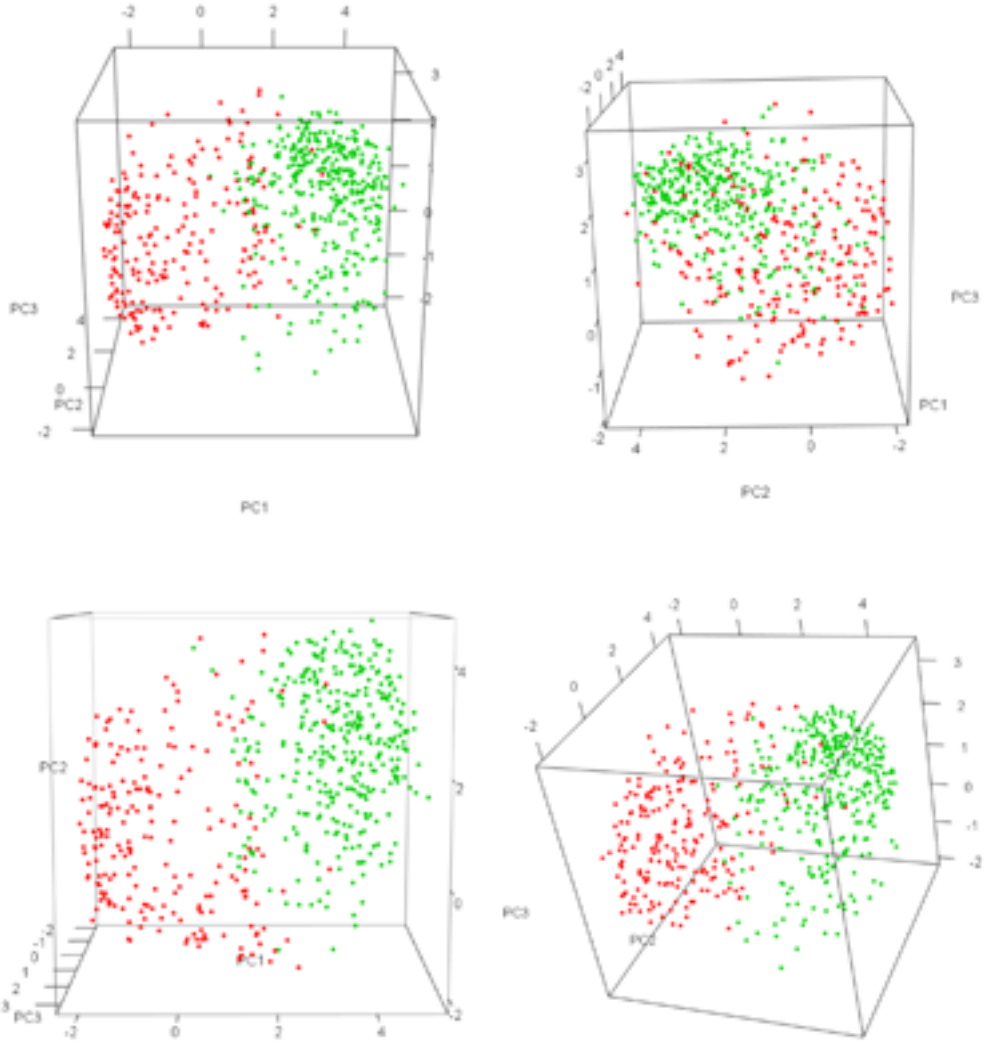


El siguiente paso es comprobar lo que pesan los componentes principales en los datos.

```
bdkmeans_pca <- scale(as.matrix(bdkmeans),center = bd_pca$center,
                      scale = bd_pca$scale) %*% bd_pca$rotation[ , 1:4]
```

Y con esto ya podemos dibujar en el espacio los datos con las 3 primeras componentes principales.

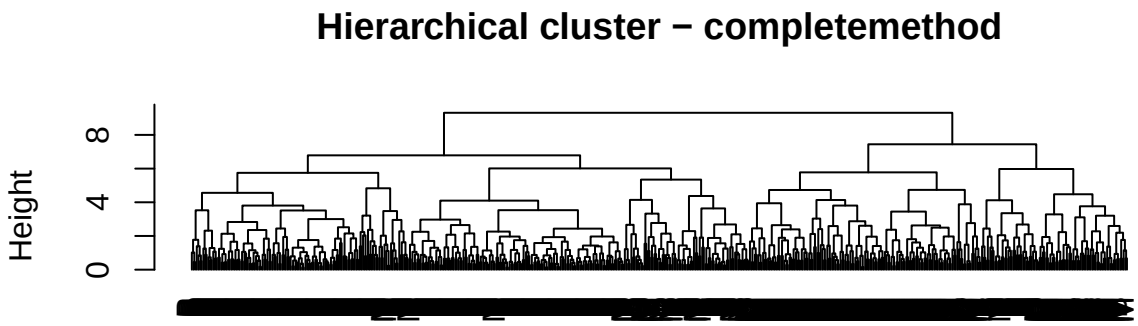
```
diagnosis <- datos$diagnosis
diagnosis[diagnosis=="B"] <- 3
diagnosis[diagnosis=="M"] <- 2
rgl::plot3d(bdkmeans_pca[,1],bdkmeans_pca[,2],bdkmeans_pca[,3],
       col=diagnosis, size=5, xlab= "PC1",  ylab= "PC2",  zlab= "PC3")
#Esta libreria crea un objeto interactivo que no puede reproducirse aqui directamente.
#Hago capturas de pantalla y las adjunto a continuación.
```

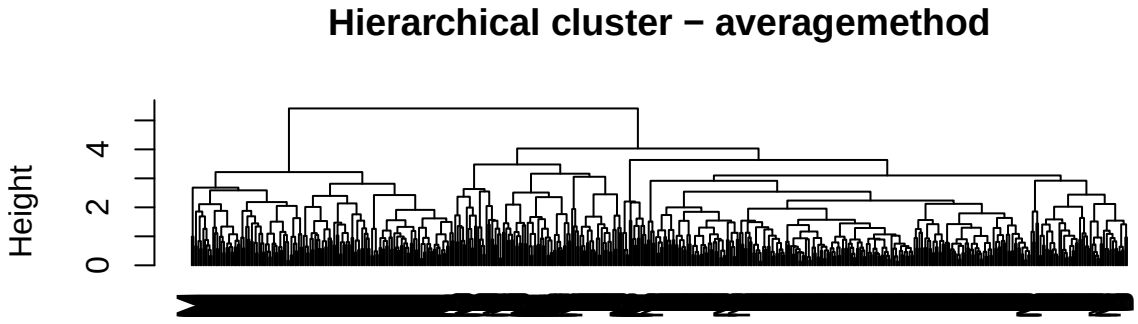


En los gráficos aparecen pintados de verde los casos benignos y en rojo los casos malignos. Puede inferirse por la distribución de ambos grupos que la extracción de los componentes principales ha sido un éxito y se siguen apreciando las diferencias entre ambos grupos. Del resultado podemos inferir que hay una región intermedia entre tumores benignos y malignos dónde se entremezclan ambos casos. Podemos ver también como hay algunos casos de tumores malignos que son muy difíciles de detectar, ya que presentan las características de los tumores benignos.

Por último, pruebo a hacer un clúster jerárquico por el método completo y el de la media sobre el resultado de los componentes principales.

```
for (i in c("complete","average")){
  hc <- hclust(dist(bdkmeans_pca), method=i)
  plot(hc, hang = -1,labels=datos$diagnosis, xlab="", sub="",
       main=paste(c("Hierarchical cluster - ", i, "method"),collapse=""))
}
```





ANÁLISIS FACTORIAL

El análisis factorial es una técnica estadística que permite extraer variables latentes que explican las relaciones entre un conjunto de variables. Para llevarlo a cabo en primer lugar reseteo la base de datos, cargo la librería necesaria para poder realizarlo y calculo las matrices de covarianza y correlación.

```
library(psych)
```

```
## Warning: package 'psych' was built under R version 3.3.3
```

```
bdkmeans <- datos[ -c(1,2) ]
bdkmeans <- data.frame(scale(bdkmeans))
mat_cov <- cov(bdkmeans)
mat_cor <- cov2cor(mat_cov)
```

A continuación, realizo el test de Kaiser-Meyer-Olkin. Este test nos permite saber si nuestros datos cumplen las características necesarias para poderles hacer un análisis factorial.

```
KMO(mat_cor)
```

```
## Kaiser-Meyer-Olkin factor adequacy
## Call: KMO(r = mat_cor)
## Overall MSA =  0.83
## MSA for each item =
##             radius_mean            texture_mean          perimeter_mean
##                    0.83                    0.64                    0.85
##               area_mean         smoothness_mean        compactness_mean
##                    0.86                    0.81                    0.88
##          concavity_mean     concave.points_mean           symmetry_mean
##                    0.89                    0.90                    0.83
##  fractal_dimension_mean               radius_se              texture_se
##                    0.83                    0.83                    0.48
##            perimeter_se                 area_se           smoothness_se
##                    0.84                    0.85                    0.64
##          compactness_se            concavity_se       concave.points_se
##                    0.87                    0.83                    0.84
##             symmetry_se    fractal_dimension_se            radius_worst
##                    0.58                    0.81                    0.82
##           texture_worst         perimeter_worst              area_worst
##                    0.60                    0.88                    0.82
##        smoothness_worst       compactness_worst         concavity_worst
##                    0.75                    0.85                    0.90
##    concave.points_worst          symmetry_worst fractal_dimension_worst
##                    0.89                    0.69                    0.81
```

Del resultado del test inferimos:
1. Valores +90% - concave.points_mean, concavity_worst
2. Valores 80-90% - radius_mean, perimeter_mean, area_mean, smoothness_mean, compactness_mean, concavity_mean, symmetry_mean, fractal_dimension_mean, radius_se, perimeter_se, area_se, compactness_se, concavity_se, concave.points_se, fractal_dimension_se, radius_worst, perimeter_worst, area_worst, compactness_worst, concave.points_worst, fractal_dimension_worst
3. Valores 70-80% - smoothness_worst, symmetry_worst
4. Valores 60-70% - texture_mean, somoothness_se, texture_worst
5. Valores -60% - texture_se, symmetry_se

El resultado del test en su conjunto es positivo pero las variables que están por debajo del 60% son mediocres para realizar el análisis factorial, por ello procedo a eliminarlas y a recalcular las matrices de covarianza y

correlación.

```
bdkmeans <- bdkmeans[ -c(12,19) ]
mat_cov <- cov(bdkmeans)
mat_cor <- cov2cor(mat_cov)
```

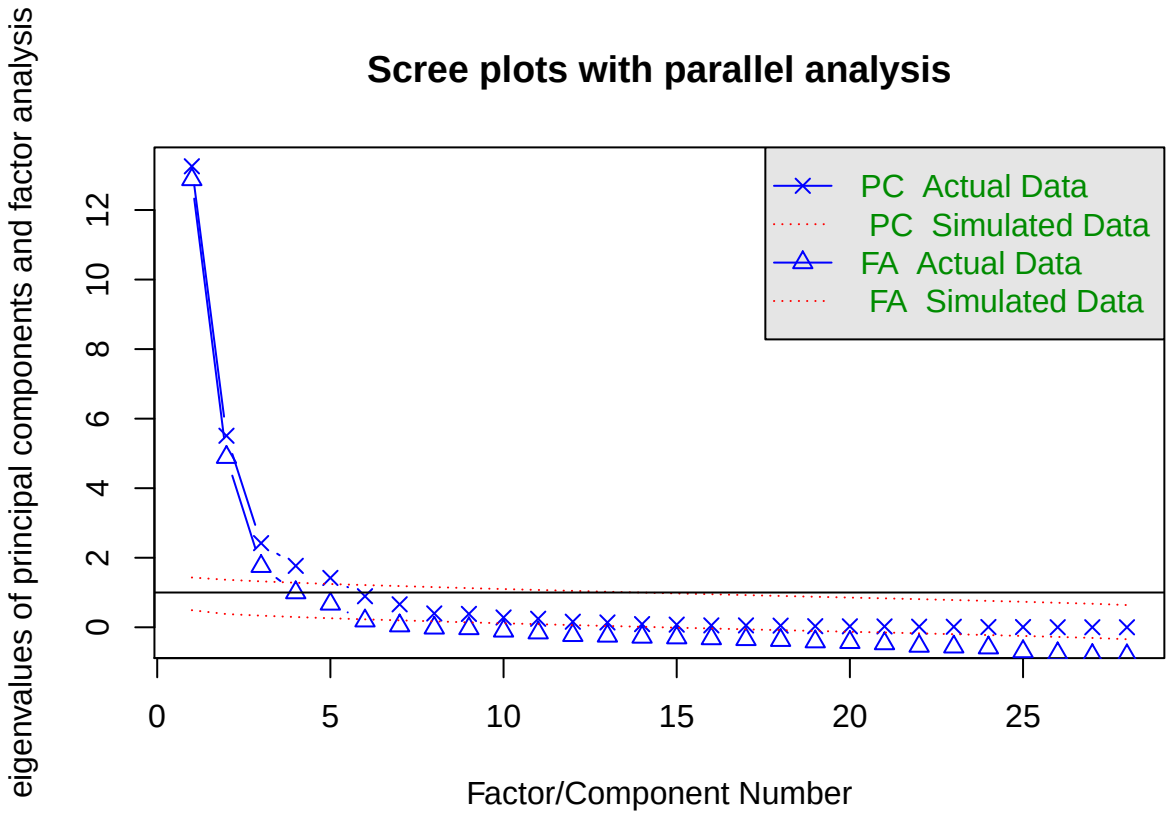
Realizo ahora el test de Barlett.

```
cortest.bartlett(mat_cor, n=569)
```

```
## $chisq
## [1] 37763
##
## $p.value
## [1] 0
##
## $df
## [1] 378
```

El test de Bartlett supone que las varianzas son iguales en todos los grupos o muestras. Si fuese así no podríamos realizar el análisis factorial, pero con los resultados que obtenemos rechazamos esa hipótesis. Podemos realizar el análisis factorial. Ahora hay que decidir el número de factores que vamos a escoger.

```
fa.parallel(mat_cor, n.obs=569, fa="both", n.iter=100,
            main="Scree plots with parallel analysis")
```




```
## Parallel analysis suggests that the number of factors =  5  and the number of components =  5
```

Del análisis extraemos que debemos usar 5 factores. Hacemos el análisis factorial y comprobamos la

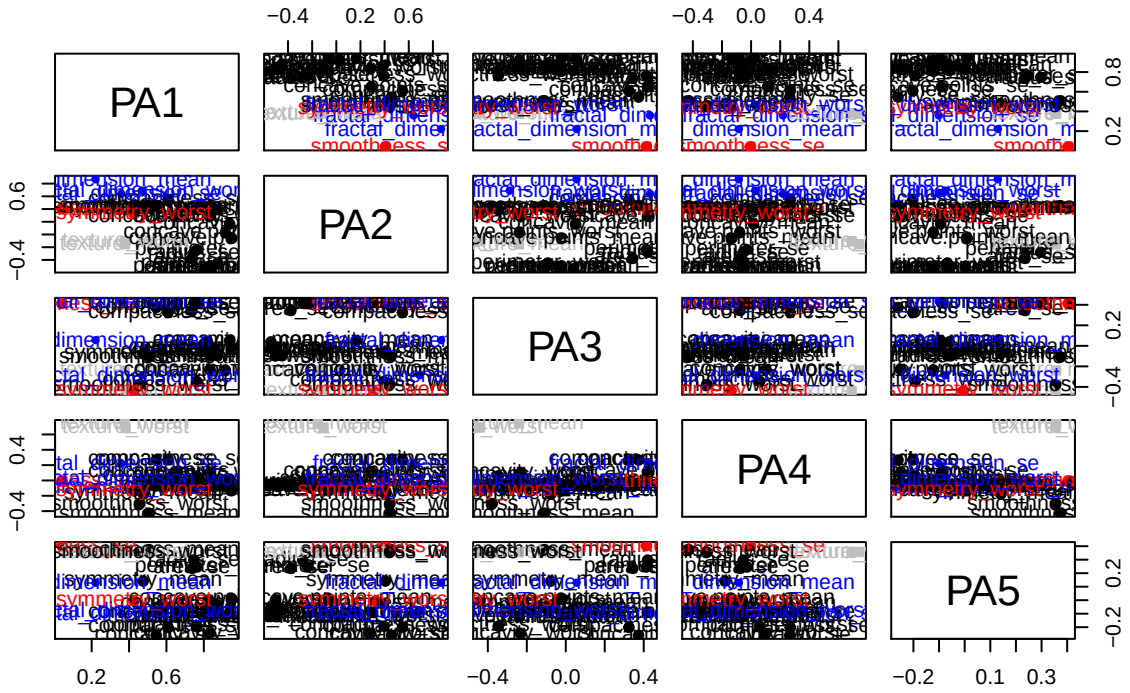
distribución de las variables en función de los factores.

```
fa <- fa(mat_cor, nfactors=5, rotate="none", fm="pa")
fa
```

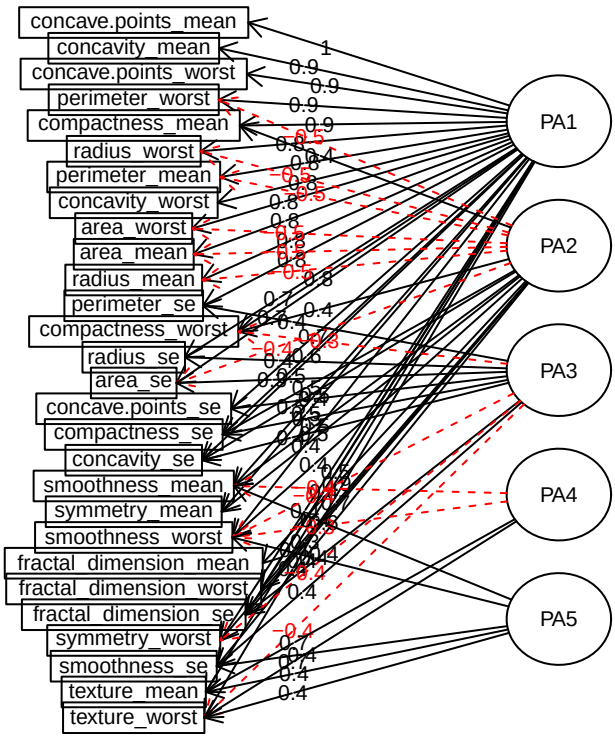
```
## Factor Analysis using method =  pa
## Call: fa(r = mat_cor, nfactors = 5, rotate = "none", fm = "pa")
## Standardized loadings (pattern matrix) based upon correlation matrix
##                           PA1   PA2   PA3   PA4   PA5   h2    u2 com
## radius_mean              0.81 -0.54 -0.02 -0.05 -0.12 0.96 0.038 1.8
## texture_mean             0.38 -0.15 -0.24  0.74  0.36 0.90 0.099 2.4
## perimeter_mean           0.84 -0.50 -0.02 -0.05 -0.12 0.97 0.035 1.7
## area_mean                0.81 -0.54  0.04 -0.06 -0.06 0.96 0.037 1.8
## smoothness_mean          0.51  0.45 -0.11 -0.43  0.37 0.80 0.204 3.9
## compactness_mean         0.87  0.39 -0.06 -0.07 -0.03 0.91 0.089 1.4
## concavity_mean           0.94  0.17  0.05  0.02 -0.09 0.92 0.081 1.1
## concave.points_mean      0.95 -0.06 -0.02 -0.13  0.01 0.92 0.075 1.0
## symmetry_mean            0.48  0.41 -0.08 -0.18  0.14 0.45 0.550 2.5
## fractal_dimension_mean   0.22  0.87  0.06 -0.08  0.13 0.83 0.173 1.2
## radius_se                0.75 -0.28  0.42 -0.05  0.30 0.90 0.096 2.3
## perimeter_se             0.76 -0.24  0.41 -0.03  0.25 0.87 0.126 2.0
## area_se                  0.74 -0.38  0.34 -0.08  0.24 0.86 0.136 2.2
## smoothness_se            0.04  0.41  0.41 -0.01  0.40 0.50 0.499 3.0
## compactness_se           0.61  0.55  0.32  0.27 -0.20 0.89 0.110 3.2
## concavity_se             0.55  0.46  0.37  0.27 -0.26 0.78 0.216 3.8
## concave.points_se        0.65  0.29  0.42  0.11 -0.09 0.70 0.297 2.3
## fractal_dimension_se     0.36  0.64  0.43  0.22 -0.10 0.78 0.217 2.8
## radius_worst             0.84 -0.50 -0.10 -0.06 -0.05 0.98 0.024 1.7
## texture_worst            0.38 -0.11 -0.44  0.69  0.35 0.95 0.047 3.0
## perimeter_worst          0.87 -0.45 -0.09 -0.05 -0.07 0.98 0.016 1.6
## area_worst               0.83 -0.50 -0.04 -0.06  0.00 0.94 0.057 1.7
## smoothness_worst         0.46  0.44 -0.38 -0.30  0.36 0.77 0.232 4.6
## compactness_worst        0.76  0.39 -0.35  0.06 -0.19 0.90 0.104 2.1
## concavity_worst          0.83  0.28 -0.23  0.10 -0.25 0.90 0.105 1.6
## concave.points_worst     0.92  0.03 -0.24 -0.08 -0.12 0.92 0.084 1.2
## symmetry_worst           0.43  0.32 -0.44 -0.14 -0.01 0.50 0.502 3.1
## fractal_dimension_worst 0.47  0.69 -0.29  0.03 -0.08 0.79 0.213 2.2
##
##                         PA1  PA2  PA3  PA4  PA5
## SS loadings           13.15 5.34 2.19 1.63 1.22
## Proportion Var         0.47 0.19 0.08 0.06 0.04
## Cumulative Var         0.47 0.66 0.74 0.80 0.84
## Proportion Explained   0.56 0.23 0.09 0.07 0.05
## Cumulative Proportion  0.56 0.79 0.88 0.95 1.00
##
## Mean item complexity =  2.3
## Test of the hypothesis that 5 factors are sufficient.
##
## The degrees of freedom for the null model are  378  and the objective function was  68
## The degrees of freedom for the model are 248  and the objective function was  22
##
## The root mean square of the residuals (RMSR) is  0.03
## The df corrected root mean square of the residuals is  0.04
##
## Fit based upon off diagonal values = 1
```

```
## Measures of factor score adequacy
##                                                 PA1  PA2  PA3  PA4  PA5
## Correlation of scores with factors              1.00 0.99 0.99 0.97 0.96
## Multiple R square of scores with factors        1.00 0.98 0.98 0.94 0.92
## Minimum correlation of possible factor scores   0.99 0.97 0.97 0.88 0.85
```

```
factor.plot(fa, labels=rownames(fa$loadings))
```



```
fa.diagram(fa,simple=FALSE,e.size=.08, rsize=.8)
```



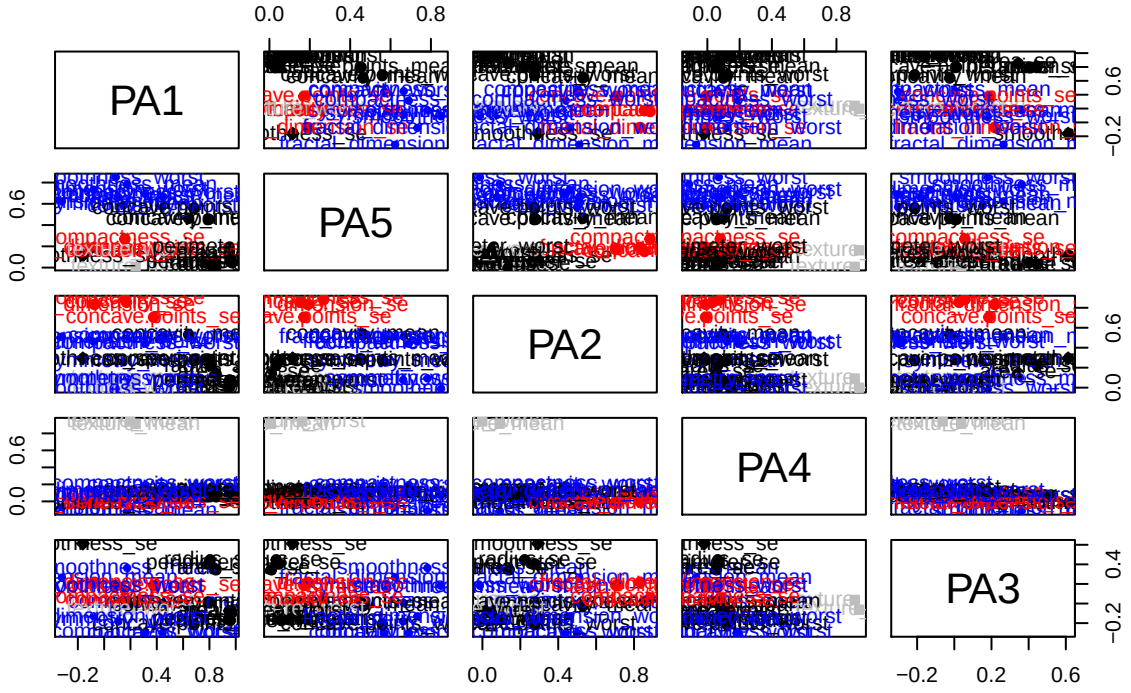
Se puede ver como las variables están muy pegadas y no tienen mucha varianza. Para evitarlo y aumentar la varianza entre los factores utilizamos la rotación ortogonal varimax.

```
fa.varimax <- fa(mat_cor, nfactors=5, rotate="varimax", fm="pa")
fa.varimax
```

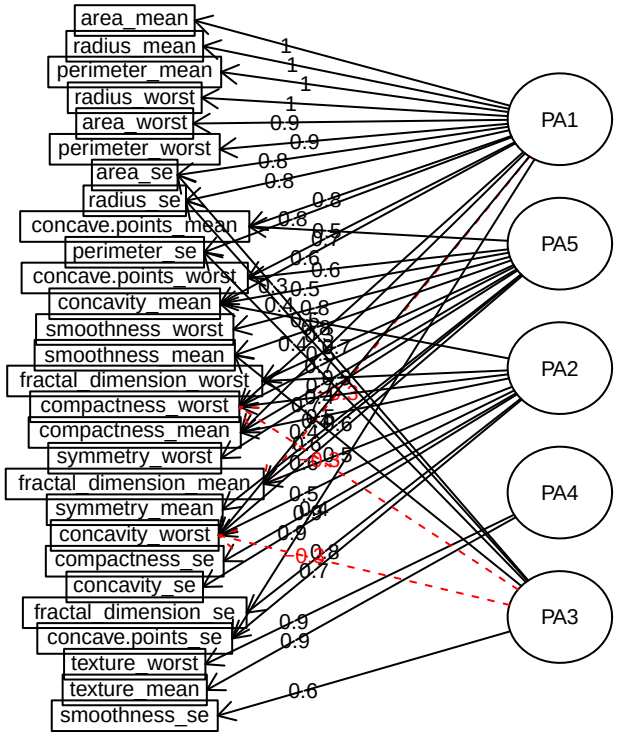
```
## Factor Analysis using method =  pa
## Call: fa(r = mat_cor, nfactors = 5, rotate = "varimax", fm = "pa")
## Standardized loadings (pattern matrix) based upon correlation matrix
##                            PA1  PA5   PA2   PA4   PA3   h2    u2 com
## radius_mean               0.96 0.06  0.04  0.09 -0.17 0.96 0.038 1.1
## texture_mean              0.24 0.01  0.09  0.91  0.04 0.90 0.099 1.2
## perimeter_mean            0.96 0.10  0.07  0.09 -0.16 0.97 0.035 1.1
## area_mean                 0.97 0.05  0.05  0.09 -0.09 0.96 0.037 1.0
## smoothness_mean           0.17 0.78  0.09 -0.11  0.36 0.80 0.204 1.6
## compactness_mean          0.45 0.67  0.51  0.06  0.01 0.91 0.089 2.7
## concavity_mean            0.64 0.46  0.53  0.10 -0.02 0.92 0.081 2.9
## concave.points_mean       0.80 0.45  0.28  0.06  0.01 0.92 0.075 1.9
## symmetry_mean             0.15 0.58  0.25 -0.02  0.15 0.45 0.550 1.7
## fractal_dimension_mean   -0.32 0.62  0.51 -0.07  0.27 0.83 0.173 2.9
## radius_se                 0.81 0.03  0.22  0.07  0.45 0.90 0.096 1.8
## perimeter_se              0.79 0.05  0.28  0.07  0.40 0.87 0.126 1.8
## area_se                   0.85 0.01  0.14  0.06  0.34 0.86 0.136 1.4
## smoothness_se            -0.16 0.11  0.29 -0.02  0.61 0.50 0.499 1.7
## compactness_se            0.16 0.27  0.88  0.08  0.06 0.89 0.110 1.3
## concavity_se              0.17 0.15  0.85  0.03  0.03 0.78 0.216 1.1
## concave.points_se         0.38 0.17  0.71 -0.01  0.19 0.70 0.297 1.8
## fractal_dimension_se     -0.07 0.19  0.83 -0.01  0.22 0.78 0.217 1.3
## radius_worst              0.95 0.15  0.01  0.14 -0.16 0.98 0.024 1.2
## texture_worst             0.20 0.16  0.00  0.94 -0.06 0.95 0.047 1.2
## perimeter_worst           0.95 0.19  0.06  0.15 -0.16 0.98 0.016 1.2
## area_worst                0.95 0.13  0.01  0.14 -0.07 0.94 0.057 1.1
## smoothness_worst          0.09 0.85 -0.01  0.07  0.18 0.77 0.232 1.1
## compactness_worst         0.31 0.69  0.44  0.19 -0.30 0.90 0.104 2.8
## concavity_worst           0.44 0.57  0.50  0.18 -0.30 0.90 0.105 3.7
## concave.points_worst      0.68 0.56  0.28  0.12 -0.21 0.92 0.084 2.6
## symmetry_worst            0.11 0.66  0.08  0.09 -0.18 0.50 0.502 1.3
## fractal_dimension_worst -0.08 0.73  0.46  0.12 -0.13 0.79 0.213 1.9
##
##                         PA1  PA5  PA2  PA4  PA3
## SS loadings           10.00 5.31 4.65 1.98 1.60
## Proportion Var         0.36 0.19 0.17 0.07 0.06
## Cumulative Var         0.36 0.55 0.71 0.78 0.84
## Proportion Explained   0.42 0.23 0.20 0.08 0.07
## Cumulative Proportion  0.42 0.65 0.85 0.93 1.00
##
## Mean item complexity =  1.7
## Test of the hypothesis that 5 factors are sufficient.
##
## The degrees of freedom for the null model are  378  and the objective function was  68
## The degrees of freedom for the model are 248  and the objective function was  22
##
## The root mean square of the residuals (RMSR) is  0.03
## The df corrected root mean square of the residuals is  0.04
##
```

```
## Fit based upon off diagonal values = 1
## Measures of factor score adequacy
##                                                  PA1  PA5  PA2  PA4  PA3
## Correlation of scores with factors                 1 0.97 0.98 0.98 0.98
## Multiple R square of scores with factors           1 0.95 0.95 0.96 0.97
## Minimum correlation of possible factor scores      1 0.90 0.91 0.92 0.93
```

```
factor.plot(fa.varimax, labels=rownames(fa$loadings))
```



```
fa.diagram(fa.varimax,simple=FALSE, e.size=.08, rsize=.8)
```



Por último realizamos el análisis factorial con la rotación oblicua promax.

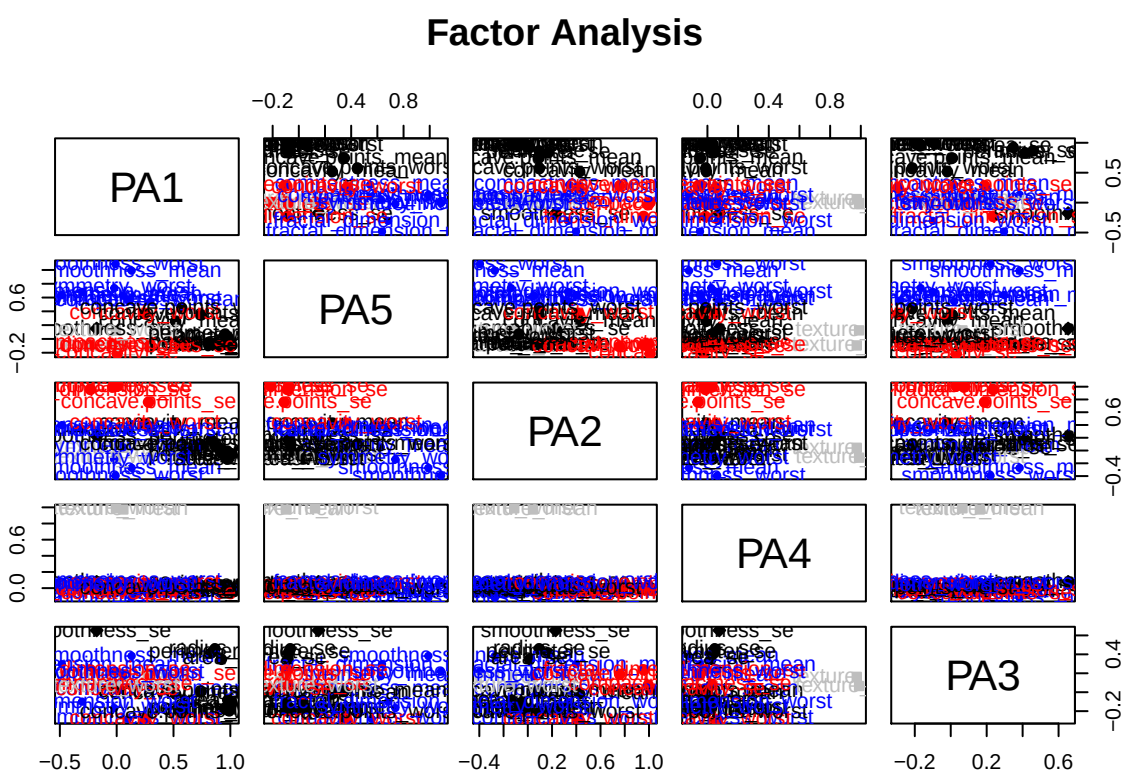
```
fa.promax <- fa(mat_cor, nfactors=5, rotate="promax", fm="pa")
```

```
fa.promax
```

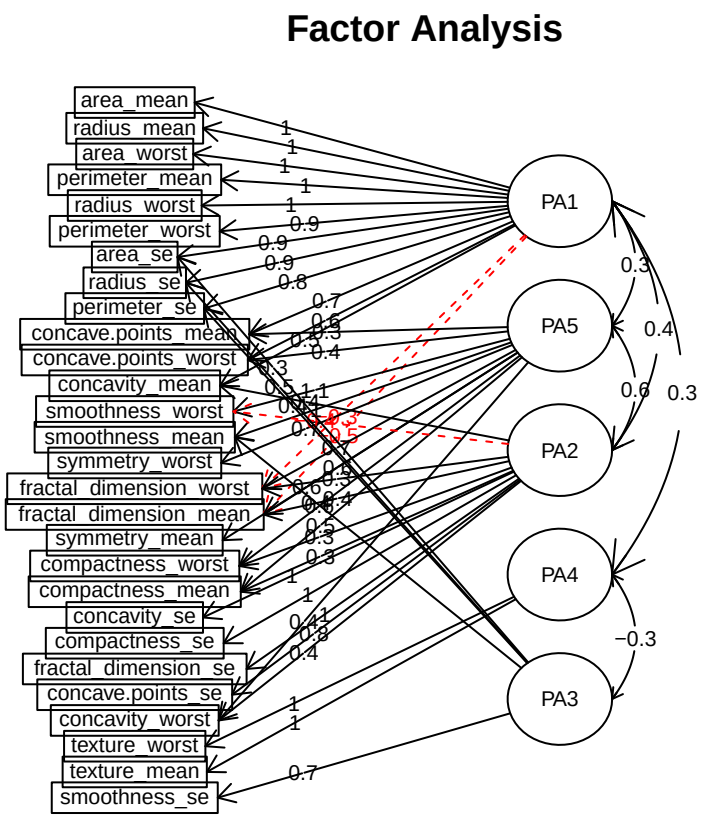
```
## Factor Analysis using method =  pa
## Call: fa(r = mat_cor, nfactors = 5, rotate = "promax", fm = "pa")
##
##  Warning: A Heywood case was detected.
## Standardized loadings (pattern matrix) based upon correlation matrix
##                            PA1   PA5   PA2   PA4   PA3   h2    u2 com
## radius_mean               0.99 -0.09 -0.05 -0.03 -0.20 0.96 0.038 1.1
## texture_mean              0.06 -0.09  0.04  0.97  0.16 0.90 0.099 1.1
## perimeter_mean            0.98 -0.05 -0.02 -0.03 -0.18 0.97 0.035 1.1
## area_mean                 1.01 -0.10 -0.05 -0.02 -0.11 0.96 0.037 1.0
## smoothness_mean           0.13  0.99 -0.27 -0.12  0.38 0.80 0.204 1.5
## compactness_mean          0.30  0.54  0.34 -0.03  0.02 0.91 0.089 2.3
## concavity_mean            0.52  0.26  0.43 -0.01 -0.02 0.92 0.081 2.4
## concave.points_mean       0.74  0.35  0.09 -0.05  0.01 0.92 0.075 1.5
## symmetry_mean             0.06  0.63  0.05 -0.04  0.17 0.45 0.550 1.2
## fractal_dimension_mean   -0.48  0.64  0.40 -0.05  0.30 0.83 0.173 3.1
## radius_se                 0.86 -0.06  0.09  0.05  0.46 0.90 0.096 1.6
## perimeter_se              0.83 -0.08  0.17  0.04  0.41 0.87 0.126 1.6
## area_se                   0.92 -0.08  0.02  0.02  0.35 0.86 0.136 1.3
## smoothness_se            -0.18  0.15  0.23  0.08  0.65 0.50 0.499 1.6
## compactness_se           -0.03 -0.07  0.99  0.03  0.07 0.89 0.110 1.0
## concavity_se              0.01 -0.21  1.01 -0.02  0.03 0.78 0.216 1.1
## concave.points_se         0.28 -0.11  0.76 -0.06  0.19 0.70 0.297 1.5
## fractal_dimension_se     -0.23 -0.09  0.96 -0.01  0.24 0.78 0.217 1.3
## radius_worst              0.97  0.04 -0.13  0.03 -0.17 0.98 0.024 1.1
## texture_worst            -0.01  0.12 -0.11  0.99  0.07 0.95 0.047 1.1
## perimeter_worst           0.95  0.06 -0.08  0.02 -0.17 0.98 0.016 1.1
## area_worst                0.98  0.03 -0.13  0.04 -0.08 0.94 0.057 1.1
## smoothness_worst         -0.01  1.09 -0.39  0.06  0.22 0.77 0.232 1.4
## compactness_worst         0.10  0.56  0.31  0.08 -0.28 0.90 0.104 2.3
## concavity_worst           0.25  0.37  0.42  0.05 -0.29 0.90 0.105 3.5
## concave.points_worst      0.57  0.45  0.10 -0.01 -0.21 0.92 0.084 2.3
## symmetry_worst           -0.02  0.74 -0.14  0.02 -0.17 0.50 0.502 1.2
## fractal_dimension_worst -0.30  0.68  0.33  0.07 -0.10 0.79 0.213 2.0
##
##                         PA1  PA5  PA2  PA4  PA3
## SS loadings            9.99 5.25 4.68 1.94 1.67
## Proportion Var         0.36 0.19 0.17 0.07 0.06
## Cumulative Var         0.36 0.54 0.71 0.78 0.84
## Proportion Explained   0.42 0.22 0.20 0.08 0.07
## Cumulative Proportion  0.42 0.65 0.85 0.93 1.00
##
##  With factor correlations of
##       PA1   PA5   PA2   PA4   PA3
## PA1  1.00  0.35  0.38  0.33 -0.14
## PA5  0.35  1.00  0.64  0.22 -0.16
## PA2  0.38  0.64  1.00  0.21 -0.02
## PA4  0.33  0.22  0.21  1.00 -0.31
## PA3 -0.14 -0.16 -0.02 -0.31  1.00
```

```
##
## Mean item complexity =  1.6
## Test of the hypothesis that 5 factors are sufficient.
##
## The degrees of freedom for the null model are  378  and the objective function was  68
## The degrees of freedom for the model are 248  and the objective function was  22
##
## The root mean square of the residuals (RMSR) is  0.03
## The df corrected root mean square of the residuals is  0.04
##
## Fit based upon off diagonal values = 1
## Measures of factor score adequacy
##                                                  PA1  PA5  PA2  PA4  PA3
## Correlation of scores with factors                 1 0.99 0.99 0.99 0.99
## Multiple R square of scores with factors           1 0.97 0.97 0.97 0.99
## Minimum correlation of possible factor scores      1 0.94 0.95 0.94 0.97
```

```
factor.plot(fa.promax, labels=rownames(fa$loadings))
```



```
fa.diagram(fa.promax,simple=FALSE, e.size=.08, rsize=.8)
```



Como conclusión decir que del análisis factorial derivan las siguientes relaciones de variables y factores:

FACTOR 1	FACTOR 2	FACTOR 3
area_mean	concavity_mean	area_se
area_se	concavity_se	radius_se
area_worst	concavity_worst	perimeter_se
radius_mean	fractal_dimension_worst	smoothness_se
radius_se	fractal_dimension_mean	smoothness_mean
radius_worst	fractal_dimension_se	
perimeter_mean	compactness_worst	
perimeter_worst	compactness_mean	
perimeter_se	compactness_se	
concave.points_mean	concave.points_se	
concave.points_worst		
concavity_mean		
TAMAÑO	**FORMA**	**GRADO DE DESVIACIÓN**

FACTOR 4	FACTOR 5
texture_worst	concave.points_mean
texture_mean	concave.points_worst
	smoothness_mean
	smoothness_worst
	symmetry_mean
	symmetry_worst
	fractal_dimension_mean
	fractal_dimension_worst
	compactness_mean
	compactness_worst
	concavity_worst
TEXTURA	**GRADO DE ASIMETRÍA**

Aparece abajo de cada factor, en negrita, el nombre que le he dado a cada una de las variables latentes en función de las variables con las que se relaciona.