Técnicas de Agrupación y de Reducción de la Dimensión

Práctica 2

ÍNDICE

RESUMEN EJECUTIVO

Debido al elevado número de características que presentan hoy en día los vehículos, el potencial cliente puede encontrarse abrumado ante tal cantidad de información. Por medio de este estudio, se pretende dotar al consumidor y al vendedor de una rápida herramienta de búsqueda, dependiendo de sus prioridades.

Mediante el estudio estadístico de los datos, se presenta una solución ya estudiada y contrastada de las diferentes prestaciones de los distintos automóviles. En esta, se podrán de manifiesto las características de los diferentes modelos desde una perspectiva de Data Science, otorgando mayor valor añadido a las posibles transacciones que resultasen del empleo de esta herramienta.

Por tanto, este estudio está dirigido al consumidor que se encuentra en búsqueda de vehículo, a los agentes vendedores, que encontraran en este un instrumento de apoyo a sus ventas, y a todos aquellos amantes del motor que buscan conocer más a fondo las posibilidades que pueden ofrecer los modelos de las marcas que comercializan todo-terrenos en España.

OBJETIVO DEL TRABAJO

El objetivo del trabajo es llevar a cabo un análisis de los todo-terreno que estaban a la venta en España hace unos años. Se procederá del siguiente modo:

- Realizar una reducción de la dimensión, si fuese posible, determinando las variables más asociadas entre sí y sus factores subyacentes;
- Agrupación de los diferentes todo-terrenos en el menor número de grupos según las puntuaciones factoriales que se desprenderán del análisis anterior.

DESCRIPCION DE LA BASE DE DATOS

Los datos que manejamos están contenidos en las siguientes variables:

- marca: Nombre de la marca del todo-terreno
- modelo: Nombre del modelo del todo-terreno
- pvp_euro: Precio de Venta al Publico, expresado en euros
- cilindro: Numero de cilindros
- cc: Cilindrada (en centimetros cubicos)
- potencia: Potencia (CV)
- rpm: Revoluciones Por Minuto
- peso: Peso en kg
- plazas: Numero de plazas
- cons90: Consumo a 90 km/h
- cons120: Consumo a 120 km/h
- consurb: Consumo urbano
- velocida: Velocidad maxima
- acelerac: Aceleracion de 0 a 100 (en segundos)
- acel2: Tiempo de acelaración, expresado como “mayor a 10 segundos” o “menor a 10 segundos”

CUESTIONES PREVIAS

Nos encontramos ante una base de datos con 125 modelos de todo-terreno con 15 variables. Algunas de estas variables presentan un importante número de valores perdidos.

Debido a la cantidad de estos valores contenidos en estas variables, no se considera oportuno eliminar las observaciones que contengan uno o más de estos. Para poder trabajar con todas las observaciones se han imputados dichos valores, siguiendo el procedimiento que se encuentra contenido y explicado en ANEXO II: Tratamiento de los valores perdidos.

Para realizar el análisis, se ha prescindido de la variable "acel2".

Dicha variable contenía el tiempo de aceleración de los vehículos expresado en "mayor a 10 segundos" y "menor a 10 segundos". La primera opción agrupaba al 97.6% de los todo-terrenos, por lo tanto, se decide prescindir de ella dado la poca capacidad explicativa de la variable.

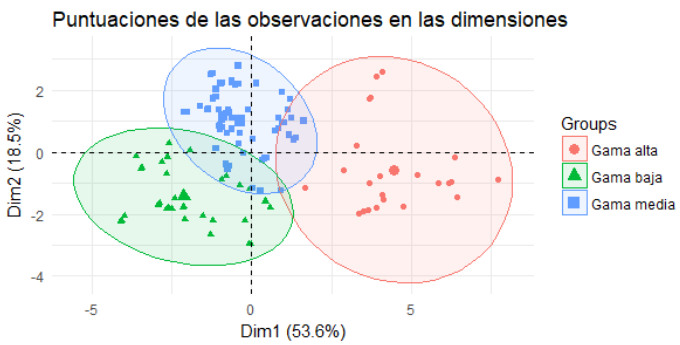
CONCLUSIONES

Existen variables correlacionadas entre sí en los diferentes modelos de todo-terreno. Podemos ver brevemente un ejemplo en la sección "Dos grupos:" del Anexo III. Aquí, pueden observarse dos grupos donde la correlación es alta. En uno se incluyen la información del número de cilindros, la potencia, la velocidad máxima y la cilindrada. En otro, el precio, el peso, los consumos a 90 y a 120 km/h y el consumo urbano. Son correlaciones lógicas ya que, por ejemplo, a mayor potencia, mayor será la velocidad, la cilindrada y el número de cilindros del vehículo.

Tras el estudio de los datos (ANEXO III: Análisis exploratorio de los datos y ANEXO IV: Calculo del índice KMO y del test de esfericidad de Barlett) y la realización de los análisis que pueden comprobarse en los anexos, se concluye que se puede realizar una reducción a dos dimensiones. Se ha realizado por medio de un análisis de componentes principales, quedando explicada el 72.2% de la varianza (ANEXO V: Análisis de componentes principales).

Tomando los datos de éste análisis, se ha realizado una agrupación en tres clusters claramente diferenciados (ANEXO VI: Agrupación). Se dividen en tres grupos:

- Todo-terrenos de gama baja
- Todo-terrenos de gama media
- Todo-terrenos de gama alta



Observando la media y la mediana de cada grupo en las diferentes variables, se observan diferencias en la potencia, numero de cilindros, cilindrada, velocidad máxima y consumos (ANEXO VII: Diferencias en los clusters).

Por supuesto, la diferencia más notable aparece en el precio, siendo determinante en la agrupación. Sus medianas son de 15910 euros en la gama baja, 25303 en la gama media y 39657 en la gama alta (ANEXO VII: Diferencias en los clusters).

Tampoco puede obviarse la marca del coche más común en los diferentes grupos, siendo Suzuki en la gama baja, Nissan en la gama media y Mercedes en el caso de la gama alta (ANEXO VII: Diferencias en los clusters).

ANEXO I: Carga de los datos y librerías

```
# LIBRERIAS UTILIZADAS:

library(memisc)
library(mice)
library(VIM)
library(missForest)
library(Hmisc)
library(corrplot)
library(PerformanceAnalytics)
library(ppcor)
library(psych)
library(FactoMineR)
library(factoextra)
library(cluster)
library(fpc)

setwd("C:/Users/mbarr/Desktop/CUNEF/Tecnicas de Agrupacion y Reduccion de
la Dimension/Practica 2/")

data = as.data.set(spss.system.file('tterreno_euro.sav'))

colnames(data)

##  [1] "marca"    "modelo"   "pvp_euro" "cilindro" "cc"       "potencia"
##  [7] "rpm"      "peso"     "plazas"   "cons90"   "cons120"  "consurb"
## [13] "velocida" "acelerac" "acel2"

data = as.data.frame(data)

summary(data)

##         marca                      modelo       pvp_euro     cilindro
##  NISSAN    :19   Montero La. TDI 2.8 :  3   Min.   : 9113   4:91
##  SUZUKI    :19   Maverick 2.7 TD GLS :  2   1st Qu.:18140   6:31
##  LAND ROVER:15   Monterey 3.2i V6 24V:  2   Median :24867   8: 3
##  MITSUBISHI:15   Montero Co. TDI 2.5 :  2   Mean   :26696
##  JEEP      :10   Montero Co. TDI 2.8 :  2   3rd Qu.:31169
##  OPEL      : 9   Montero Corto 3.0 GL:  2   Max.   :69461
##  (Other)   :38   (Other)             :112
##        cc           potencia          rpm            peso      plazas
##  Min.   :1298   Min.   : 64.0   Min.   :3600   Min.   : 930   2: 6
##  1st Qu.:2184   1st Qu.: 95.0   1st Qu.:4000   1st Qu.:1462   4:27
##  Median :2497   Median :112.0   Median :4500   Median :1750   5:61
##  Mean   :2570   Mean   :117.1   Mean   :4671   Mean   :1675   6: 2
##  3rd Qu.:2835   3rd Qu.:125.0   3rd Qu.:5200   3rd Qu.:1909   7:23
##  Max.   :5216   Max.   :225.0   Max.   :6500   Max.   :2320   8: 4
##                                                NA's   :2      9: 2
```

```
##      cons90          cons120         consurb         velocida
##  Min.   : 6.600   Min.   : 8.40   Min.   : 8.10   Min.   :120.0
##  1st Qu.: 7.800   1st Qu.:10.53   1st Qu.:10.43   1st Qu.:140.0
##  Median : 8.600   Median :12.20   Median :12.00   Median :146.5
##  Mean   : 8.897   Mean   :12.25   Mean   :12.59   Mean   :150.6
##  3rd Qu.: 9.700   3rd Qu.:13.90   3rd Qu.:13.57   3rd Qu.:160.8
##  Max.   :13.700   Max.   :18.50   Max.   :22.10   Max.   :196.0
##  NA's   :10       NA's   :15      NA's   :7       NA's   :3
##     acelerac                    acel2
##  Min.   : 9.40   Menor a 10 segundos:  3
##  1st Qu.:13.20   Mayor a 10 segundos:122
##  Median :15.60
##  Mean   :15.43
##  3rd Qu.:18.50
##  Max.   :22.00
##  NA's   :46
```

```
str(data)
```

```
## 'data.frame':    125 obs. of  15 variables:
##  $ marca   : Factor w/ 17 levels "ASIA MOTORS",..: 1 1 1 2 3 4 4 4 4 4
...
##  $ modelo  : Factor w/ 111 levels "4 Runner 3.0 TD",..: 78 79 79 4 21
44 45 47 46 47 ...
##  $ pvp_euro: num  15164 14413 15164 31633 17956 ...
##  $ cilindro: Factor w/ 3 levels "4","6","8": 1 1 1 2 1 1 1 1 1 1 ...
##  $ cc      : num  1789 2184 2184 4300 1589 ...
##  $ potencia: num  85 72 72 193 95 124 100 100 100 100 ...
##  $ rpm     : num  5500 4250 4250 4400 6000 5200 4000 4000 4000 4000 ..
.
##  $ peso    : num  1220 1270 1270 1915 1250 ...
##  $ plazas  : Factor w/ 7 levels "2","4","5","6",..: 2 2 2 3 2 5 3 3 5
5 ...
##  $ cons90  : num  9 8 8 9.6 7.6 8.7 7.5 7.5 8.6 8.6 ...
##  $ cons120 : num  NA NA NA 12.6 11.9 12.3 11.8 11.8 13.1 13.1 ...
##  $ consurb : num  12 12 12 15.6 10.5 13.3 10.3 10.3 11.8 11.8 ...
##  $ velocida: num  160 130 130 180 150 160 140 140 145 145 ...
##  $ acelerac: num  NA NA NA 10.1 15.6 14 19 19 19.9 19.9 ...
##  $ acel2   : Factor w/ 2 levels "Menor a 10 segundos",..: 2 2 2 2 2 2
2 2 2 2 ...
```

Procedemos a transformación de las variables que aparecen como factores a character (en el caso de “marca” y “modelo”) y a numeric (cilindro y plazas). La variable “acel2” no ha sido modificada ya que la eliminaremos en el siguiente proceso.

```
data2 = data

data$marca = as.character(data$marca)
data$modelo = as.character(data$modelo)
```

```
data$cilindro = as.numeric(data$cilindro)
data$plazas = as.numeric(data$plazas)
```

Eliminacion de la variable “acel2”

Eliminamos esta variable y reordenamos las variables para mayor comodidad a la hora de la imputación de los valores perdidos:

```
data2 = data[c(1, 2, 3, 4, 5, 6, 7, 9, 8, 10, 11, 12, 13, 14)]
```

ANEXO II: Tratamiento de los valores perdidos

```
apply(is.na(data2), 2, sum)
```

```
##    marca   modelo pvp_euro cilindro       cc potencia      rpm   plaza
s
##        0        0        0        0        0        0        0
0
##     peso   cons90  cons120  consurb velocida acelerac
##        2       10       15        7        3       46
```

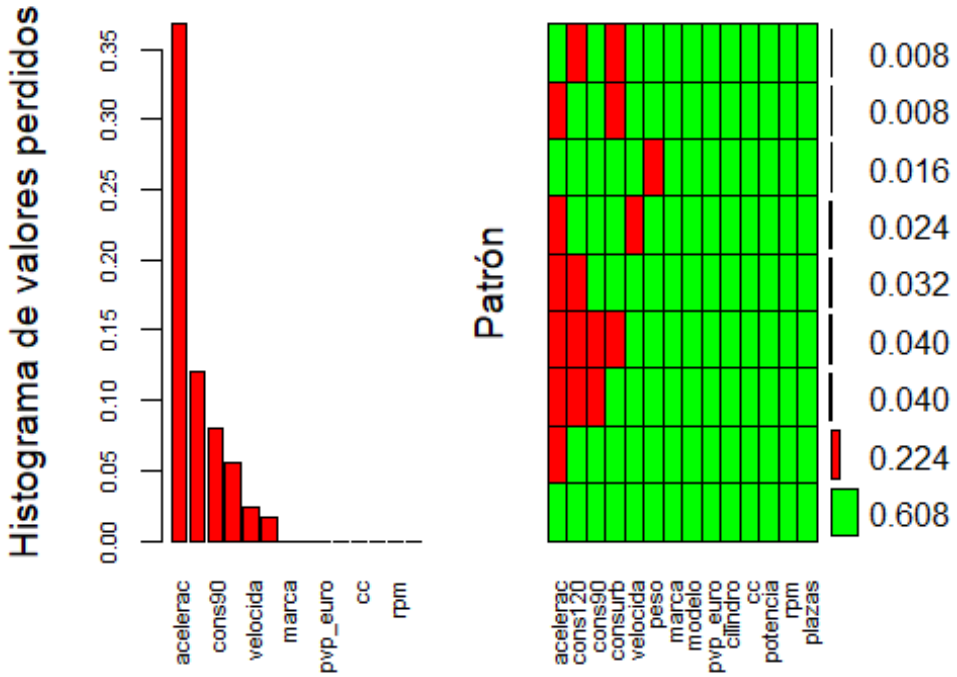
Observamos un importante número de valores perdidos. Son especialmente altos en las variables “acelerac”, “cons120” y “cons90”, constituyendo el 37%, 12% y el 8% de sus observaciones respectivamente. Analizamos por tanto éstos en busca de patrones que puedan esclarecer la situación:

```
md.pattern(data2)
```

```
##    pvp_euro cilindro cc potencia rpm plazas peso velocida consurb cons
90
## 76        1        1  1        1   1      1    1        1       1
1
##  2        1        1  1        1   1      1    0        1       1
1
## 28        1        1  1        1   1      1    1        1       1
1
##  1        1        1  1        1   1      1    1        1       0
1
##  4        1        1  1        1   1      1    1        1       1
1
##  1        1        1  1        1   1      1    1        1       0
1
##  3        1        1  1        1   1      1    1        0       1
1
##  5        1        1  1        1   1      1    1        1       1
0
##  5        1        1  1        1   1      1    1        1       0
0
##           0        0  0        0   0      0    2        3       7
10
##    cons120 acelerac marca modelo
## 76       1        1     0      0   2
##  2       1        1     0      0   3
## 28       1        0     0      0   3
##  1       0        1     0      0   4
##  4       0        0     0      0   4
##  1       1        0     0      0   4
```

```
##  3       1         0     0      0   4
##  5       0         0     0      0   5
##  5       0         0     0      0   6
##         15        46   125    125 333
```

```
aggr_plot = aggr(data2,
                 col = c('green','red'),
                 numbers = TRUE,
                 sortVars = TRUE,
                 labels = names(data2),
                 cex.axis = 0.7,
                 gap = 3,
                 ylab = c("Histograma de valores perdidos", "Patrón"))
```




```
##
##  Variables sorted by number of missings:
##  Variable Count
##  acelerac 0.368
##   cons120 0.120
##    cons90 0.080
##   consurb 0.056
##  velocida 0.024
##      peso 0.016
##     marca 0.000
##    modelo 0.000
##  pvp_euro 0.000
##  cilindro 0.000
```

```
##        cc 0.000
##  potencia 0.000
##       rpm 0.000
##    plazas 0.000
```

```
aggr_plot
```

```
##
##  Missings in variables:
##  Variable Count
##      peso     2
##    cons90    10
##   cons120    15
##   consurb     7
##  velocida     3
##  acelerac    46
```

Debido a que en el 60.8% de los casos las observaciones tienen todas las variables completas y que en el 22.4% la única variable que contiene valores perdidos es "acelerac", es recomendable proceder a la imputación de estos valores en vez de elimininarlos. Para ello, acudimos a la función "missForest", de la librería con el mismo nombre.

```
dataMF = missForest(data2[3:14])
```

```
##   missForest iteration 1 in progress...done!
##   missForest iteration 2 in progress...done!
##   missForest iteration 3 in progress...done!
```

```
head(dataMF$ximp)
```

```
##   pvp_euro cilindro   cc potencia  rpm plazas peso cons90 cons120 cons
urb
## 1 15163.93        1 1789       85 5500      2 1220    9.0 11.3840    1
2.0
## 2 14412.75        1 2184       72 4250      2 1270    8.0  9.4138    1
2.0
## 3 15163.93        1 2184       72 4250      2 1270    8.0  9.4318    1
2.0
## 4 31633.33        2 4300      193 4400      3 1915    9.6 12.6000    1
5.6
## 5 17956.40        1 1589       95 6000      2 1250    7.6 11.9000    1
0.5
## 6 29740.00        1 2389      124 5200      5 1750    8.7 12.3000    1
3.3
##   velocida acelerac
## 1      160   15.244
## 2      130   17.527
## 3      130   17.527
```

```
## 4      180   10.100
## 5      150   15.600
## 6      160   14.000
```

c

Una vez se ha completado la imputación, procedemos a la fusión de los datos completos con el resto del dataset:

```
tterreno = cbind(data2[,1:2], dataMF$ximp)
```

ANEXO III: Análisis exploratorio de los datos

Elegimos las variables que vamos a utilizar. Una vez hemos eliminado "acel2" por los motivos ya comentados, procedemos a eliminar también "marca" y "modelo" ya que no será posible realizar los cálculos pertinentes si se encuentran en el dataset. Calculamos la matriz de correlaciones y la matriz de correlaciones y p-valor y visualizamos ésta última:

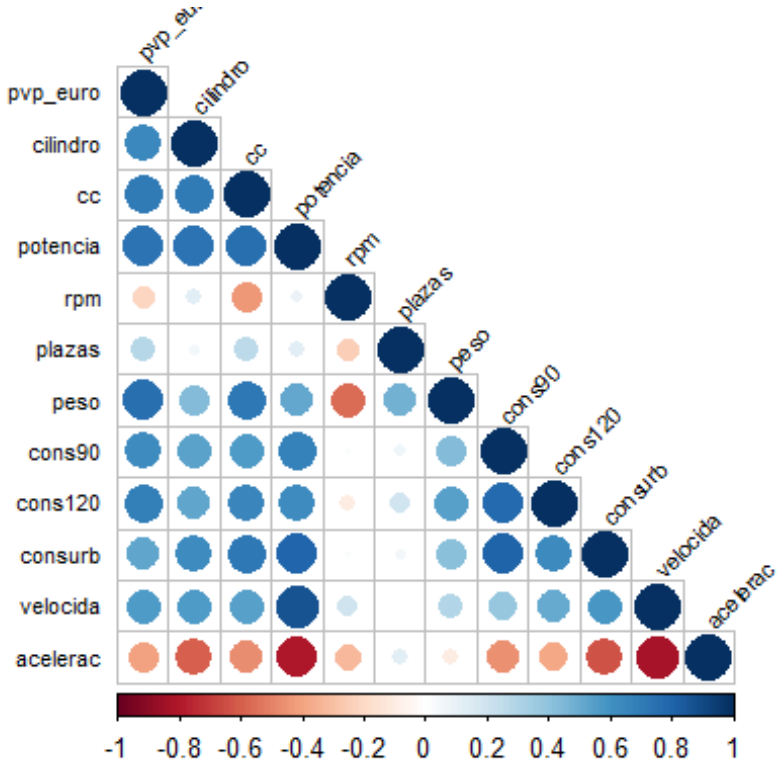
```
tterreno_variables = tterreno[,-1:-2]

cor.mat = round(cor(tterreno_variables), 2)
cor.mat.nds= rcorr(as.matrix(tterreno_variables))
cor.mat.nds
```

```
##          pvp_euro cilindro    cc potencia   rpm plazas  peso cons90
## pvp_euro     1.00     0.64  0.70     0.73 -0.22   0.28  0.75   0.62
## cilindro     0.64     1.00  0.70     0.73  0.13   0.05  0.43   0.53
## cc           0.70     0.70  1.00     0.75 -0.44   0.26  0.71   0.57
## potencia     0.73     0.73  0.75     1.00  0.08   0.13  0.51   0.67
## rpm         -0.22     0.13 -0.44     0.08  1.00  -0.25 -0.57   0.02
## plazas       0.28     0.05  0.26     0.13 -0.25   1.00  0.47   0.06
## peso         0.75     0.43  0.71     0.51 -0.57   0.47  1.00   0.43
## cons90       0.62     0.53  0.57     0.67  0.02   0.06  0.43   1.00
## cons120      0.68     0.52  0.65     0.62 -0.11   0.19  0.54   0.77
## consurb      0.52     0.62  0.71     0.79  0.02   0.05  0.41   0.80
## velocida     0.57     0.57  0.54     0.86  0.20  -0.01  0.29   0.38
## acelerac    -0.41    -0.60 -0.47    -0.80 -0.33   0.12 -0.11  -0.46
##          cons120 consurb velocida acelerac
## pvp_euro    0.68    0.52     0.57    -0.41
## cilindro    0.52    0.62     0.57    -0.60
## cc          0.65    0.71     0.54    -0.47
## potencia    0.62    0.79     0.86    -0.80
## rpm        -0.11    0.02     0.20    -0.33
## plazas      0.19    0.05    -0.01     0.12
## peso        0.54    0.41     0.29    -0.11
## cons90      0.77    0.80     0.38    -0.46
## cons120     1.00    0.63     0.50    -0.39
## consurb     0.63    1.00     0.58    -0.64
## velocida    0.50    0.58     1.00    -0.83
## acelerac   -0.39   -0.64    -0.83     1.00
##
## n= 125
##
##
## P
##          pvp_euro cilindro cc     potencia rpm    plazas peso   cons90
## pvp_euro          0.0000   0.0000 0.0000   0.0119 0.0019 0.0000 0.0000
## cilindro 0.0000            0.0000 0.0000   0.1471 0.5474 0.0000 0.0000
## cc       0.0000   0.0000          0.0000   0.0000 0.0037 0.0000 0.0000
## potencia 0.0000   0.0000   0.0000          0.3500 0.1628 0.0000 0.0000
```

```
## rpm      0.0119   0.1471   0.0000 0.3500          0.0042 0.0000 0.7955
## plazas   0.0019   0.5474   0.0037 0.1628   0.0042        0.0000 0.5393
## peso     0.0000   0.0000   0.0000 0.0000   0.0000 0.0000        0.0000
## cons90   0.0000   0.0000   0.0000 0.0000   0.7955 0.5393 0.0000
## cons120  0.0000   0.0000   0.0000 0.0000   0.2244 0.0373 0.0000 0.0000
## consurb  0.0000   0.0000   0.0000 0.0000   0.7918 0.5447 0.0000 0.0000
## velocida 0.0000   0.0000   0.0000 0.0000   0.0248 0.9037 0.0009 0.0000
## acelerac 0.0000   0.0000   0.0000 0.0000   0.0001 0.1916 0.2207 0.0000
##          cons120 consurb velocida acelerac
## pvp_euro 0.0000  0.0000  0.0000   0.0000
## cilindro 0.0000  0.0000  0.0000   0.0000
## cc       0.0000  0.0000  0.0000   0.0000
## potencia 0.0000  0.0000  0.0000   0.0000
## rpm      0.2244  0.7918  0.0248   0.0001
## plazas   0.0373  0.5447  0.9037   0.1916
## peso     0.0000  0.0000  0.0009   0.2207
## cons90   0.0000  0.0000  0.0000   0.0000
## cons120          0.0000  0.0000   0.0000
## consurb  0.0000          0.0000   0.0000
## velocida 0.0000  0.0000           0.0000
## acelerac 0.0000  0.0000  0.0000
```

Visualizamos mediante un correlograma para mayor facilidad:

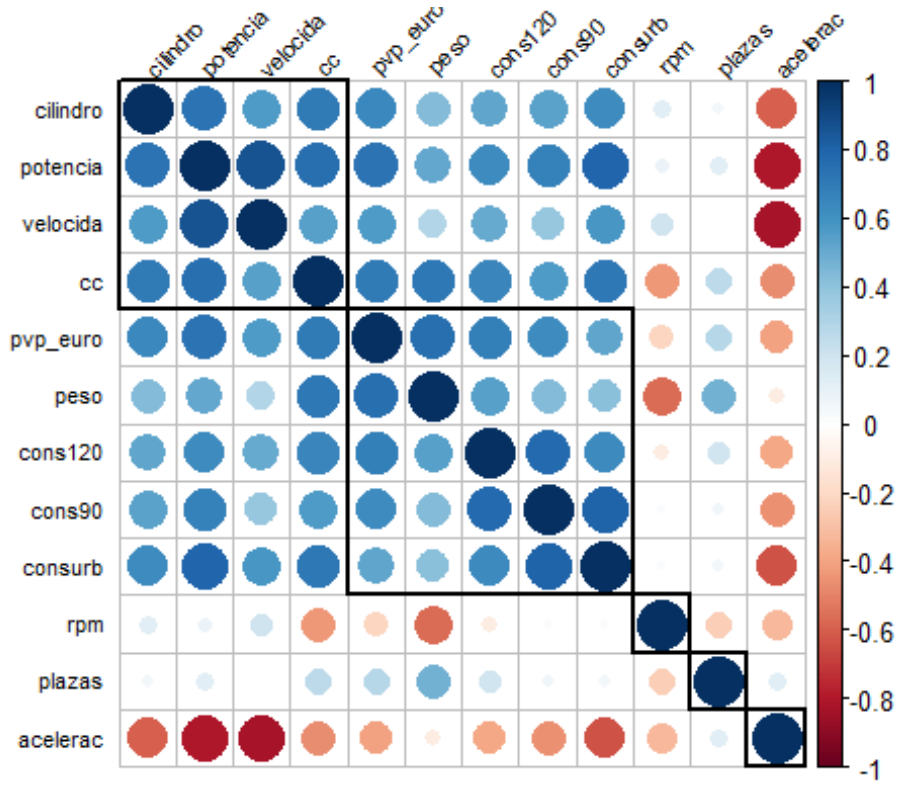
```
corrplot(cor.mat,
         type="lower",
         order="original",
         tl.col="black",
         tl.cex=0.7, tl.srt=45)
```



Dos grupos:

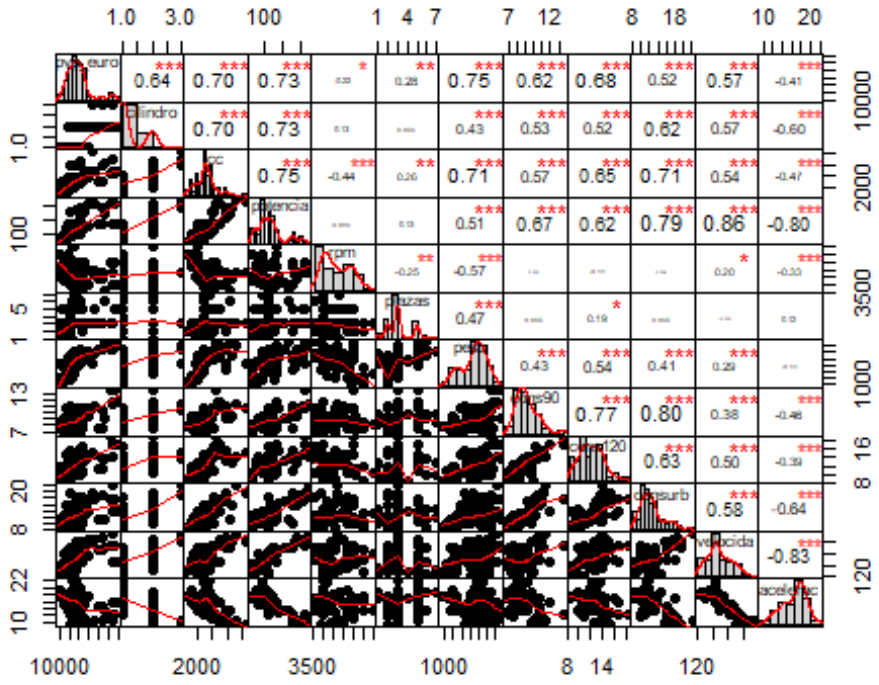
Como se ha comentado anteriormente, existen dos grupos con correlaciones importantes entre sus miembros:

```
corrplot(cor.mat, type="full", order="hclust", addrect = 5,
         tl.col="black", tl.cex=0.7, tl.srt=45)
```



Realizamos el grafico de correlaciones para buscar correlaciones significativas:

```
chart.Correlation(tterreno_variables, histogram=TRUE, pch=19)
```



ANEXO IV: Cálculo del índice KMO y del test de esfericidad de Barlett

Es importante el cálculo de este estadístico la permite medir la calidad de las correlaciones entre las variables para evaluar la idoneidad del análisis de componentes principales y factorial

```
# Creamos la matriz de correlaciones parciales

p.cor.mat = pcor(tterreno_variables)
p.cor.mat2=as.matrix(p.cor.mat$estimate)

# KMO global:

kmo.num = sum(cor.mat^2) - sum(diag(cor.mat^2))

kmo.denom = kmo.num + (sum(p.cor.mat2^2) - sum(diag(p.cor.mat2^2)))
kmo = kmo.num/kmo.denom
kmo

## [1] 0.7231829
```

El índice KMO se encuentra por encima de 0.7, lo que nos indica un valor aceptable para llevar a cabo PCA o factorial.

Calculamos ahora el MSA o KMO parcial para cada una de las variables:

```
p.cor.mat2=data.frame(p.cor.mat2)

rownames(p.cor.mat2) = c(rownames(cor.mat))
colnames(p.cor.mat2) = c(colnames(cor.mat))

for (j in 1:ncol(tterreno_variables)){
  kmo_j.num = sum(cor.mat[,j]^2) - cor.mat[j,j]^2
  kmo_j.denom = kmo_j.num + (sum(p.cor.mat2[,j]^2) - p.cor.mat2[j,j]^2)
  kmo_j = round(kmo_j.num/kmo_j.denom,4)
  print(paste(colnames(tterreno_variables)[j], "=", kmo_j))
}

## [1] "pvp_euro = 0.8244"
## [1] "cilindro = 0.7179"
## [1] "cc = 0.6939"
## [1] "potencia = 0.7819"
## [1] "rpm = 0.3238"
## [1] "plazas = 0.5509"
## [1] "peso = 0.7921"
## [1] "cons90 = 0.6377"
## [1] "cons120 = 0.7103"
```

```
## [1] "consurb = 0.8193"
## [1] "velocida = 0.6753"
## [1] "acelerac = 0.8979"
```

Pasamos a realizar el test de Barlett. Ya que no será válido si el número de observaciones supera las 100, muestreamos sobre 80 elegidas al azar:

```
set.seed(1234)

tterreno_variables.sam = tterreno_variables[sample(nrow(tterreno_variable
s), 80),]

print(cortest.bartlett(cor.mat, n=nrow(tterreno_variables.sam)))
```

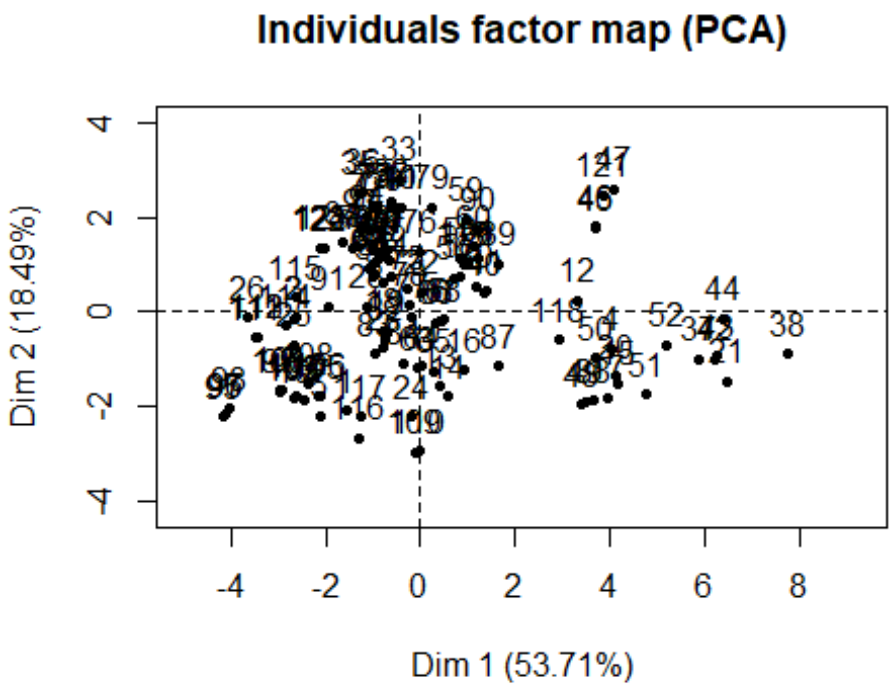
```
## $chisq
## [1] 1004.299
##
## $p.value
## [1] 8.998559e-168
##
## $df
## [1] 66
```

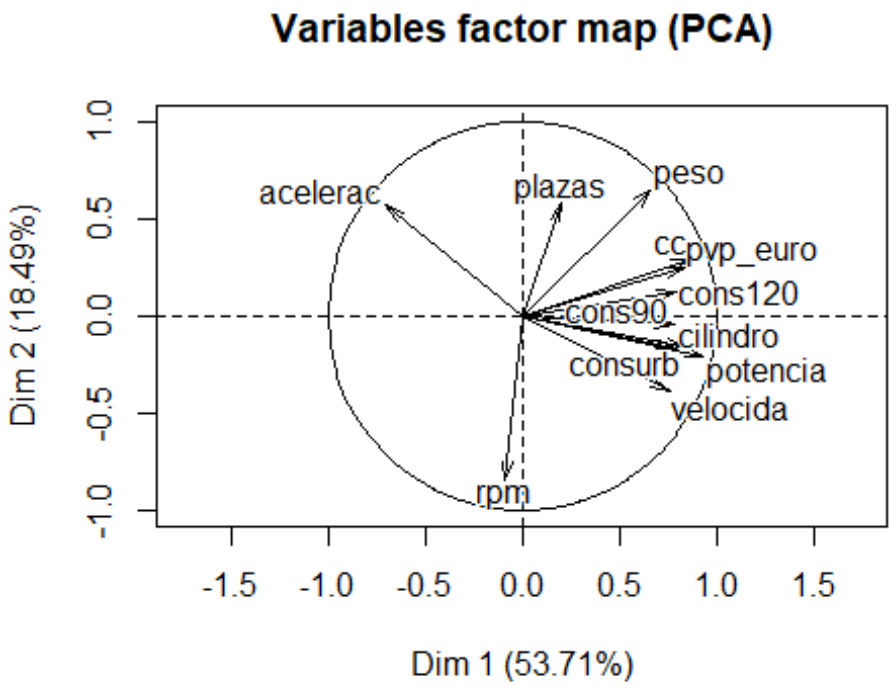
El p-valor es practicamente cero, lo que nos permite realizar el análisis de componentes principales.

ANEXO V: Análisis de componentes principales

Identificacion de los componentes principales

```
tterreno_variables.acp = PCA(tterreno_variables,
                             scale.unit = TRUE,
                             ncp = ncol(tterreno_variables),
                             graph = TRUE)
```



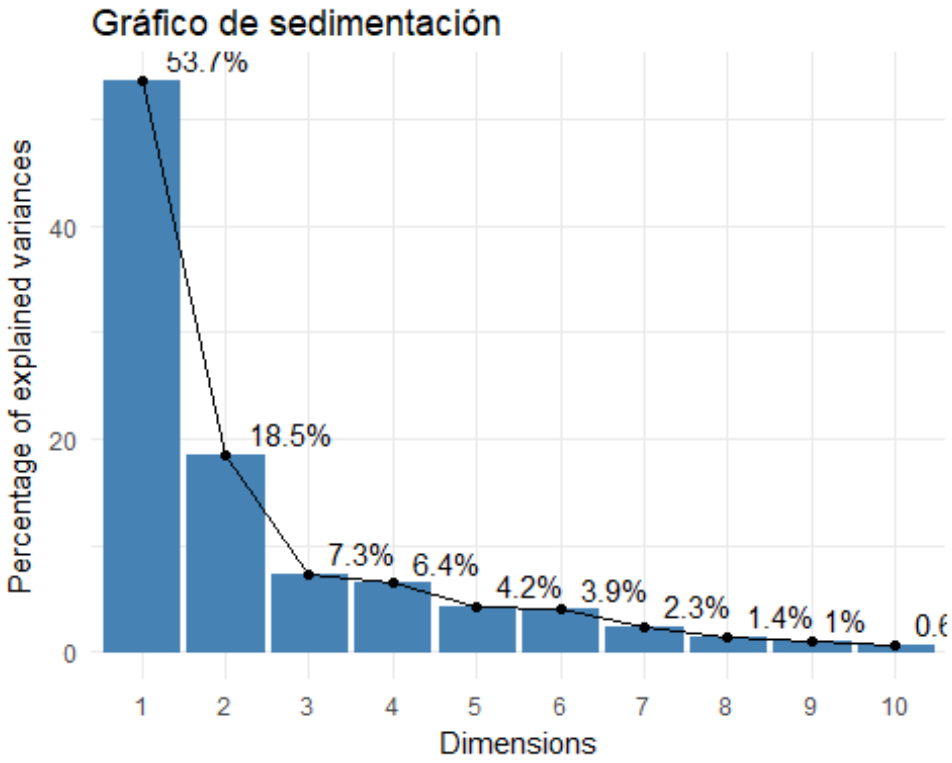



```
print(tterreno_variables.acp)

## **Results for the Principal Component Analysis (PCA)**
## The analysis was performed on 125 individuals, described by 12 variabl
```

```
es
## *The results are available in the following objects:
##
##    name               description
## 1  "$eig"             "eigenvalues"
## 2  "$var"             "results for the variables"
## 3  "$var$coord"       "coord. for the variables"
## 4  "$var$cor"         "correlations variables - dimensions"
## 5  "$var$cos2"        "cos2 for the variables"
## 6  "$var$contrib"     "contributions of the variables"
## 7  "$ind"             "results for the individuals"
## 8  "$ind$coord"       "coord. for the individuals"
## 9  "$ind$cos2"        "cos2 for the individuals"
## 10 "$ind$contrib"     "contributions of the individuals"
## 11 "$call"            "summary statistics"
## 12 "$call$centre"     "mean of the variables"
## 13 "$call$ecart.type" "standard error of the variables"
## 14 "$call$row.w"      "weights for the individuals"
## 15 "$call$col.w"      "weights for the variables"

tterreno_variables.acp$eig

##         eigenvalue percentage of variance
## comp 1  6.44501258             53.7084381
## comp 2  2.21913510             18.4927925
## comp 3  0.87697716              7.3081430
## comp 4  0.77151509              6.4292924
## comp 5  0.50400534              4.2000445
## comp 6  0.47347853              3.9456544
## comp 7  0.28115001              2.3429168
## comp 8  0.16333716              1.3611430
## comp 9  0.12050966              1.0042471
## comp 10 0.07442298              0.6201915
## comp 11 0.04552738              0.3793948
## comp 12 0.02492901              0.2077417
##         cumulative percentage of variance
## comp 1                           53.70844
## comp 2                           72.20123
## comp 3                           79.50937
## comp 4                           85.93867
## comp 5                           90.13871
## comp 6                           94.08437
## comp 7                           96.42728
## comp 8                           97.78842
## comp 9                           98.79267
## comp 10                          99.41286
## comp 11                          99.79226
## comp 12                         100.00000
```

Comprobamos que dos componentes explican un porcentaje muy importante de la varianza (72.2%). Comprobamos al mismo tiempo con la regla del codo:

```
fviz_eig(tterreno_variables.acp, addlabels=T, hjust=-0.3)+
labs(title="Gráfico de sedimentación")+
theme_minimal()
```



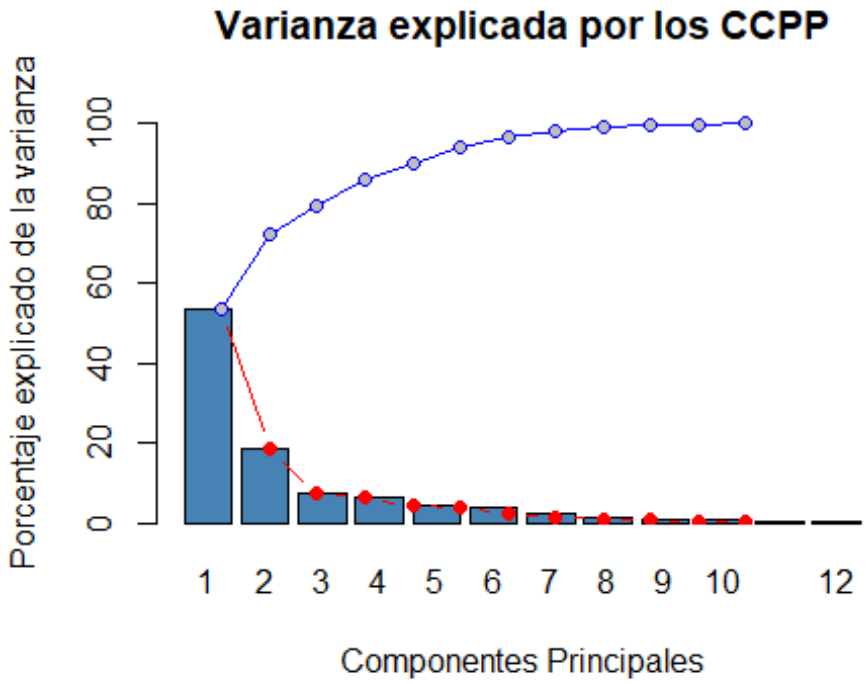
Con esta regla confirmamos lo expuesto anteriormente. Recurrir a tres dimensiones en este caso no es adecuado ya que en la tercera nos encontramos con un 7.3%, cuando para estimarse válida tendría que ser al menos del 8.33% (12 variables, 8.33% cada una).

Es una buena reducción de la dimensión dada la facilidad de su representación gráfica.

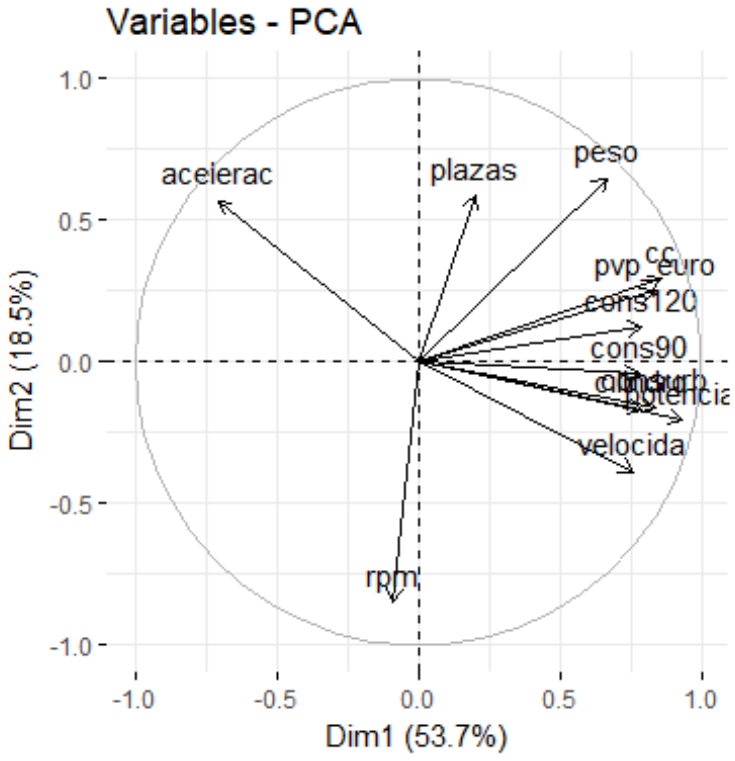
Representamos gráficamente los autovalores y el porcentaje de varianza explicada:

```
autoval= round(tterreno_variables.acp$eig, 2)


barplot(autoval[, 2], names.arg=1:nrow(autoval),
        main = "Varianza explicada por los CCPP",
        xlab = "Componentes Principales",
        ylab = "Porcentaje explicado de la varianza",
        col ="steelblue",
        ylim=c(0,105))
lines(x = 1:nrow(autoval), autoval[, 2],
      type="b", pch=19, col = "red")
```

```
lines(x = 1:nrow(autoval), autoval[, 3],
      type="o", pch=21, col = "blue", bg="grey")
```

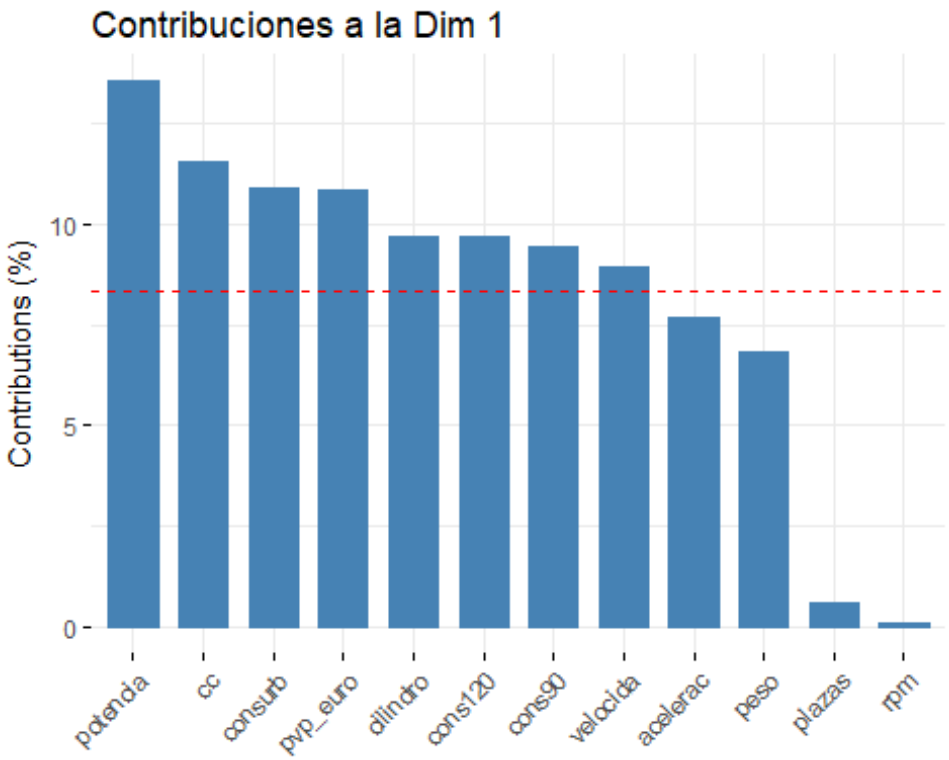



```
fviz_pca_var(tterreno_variables.acp)
```

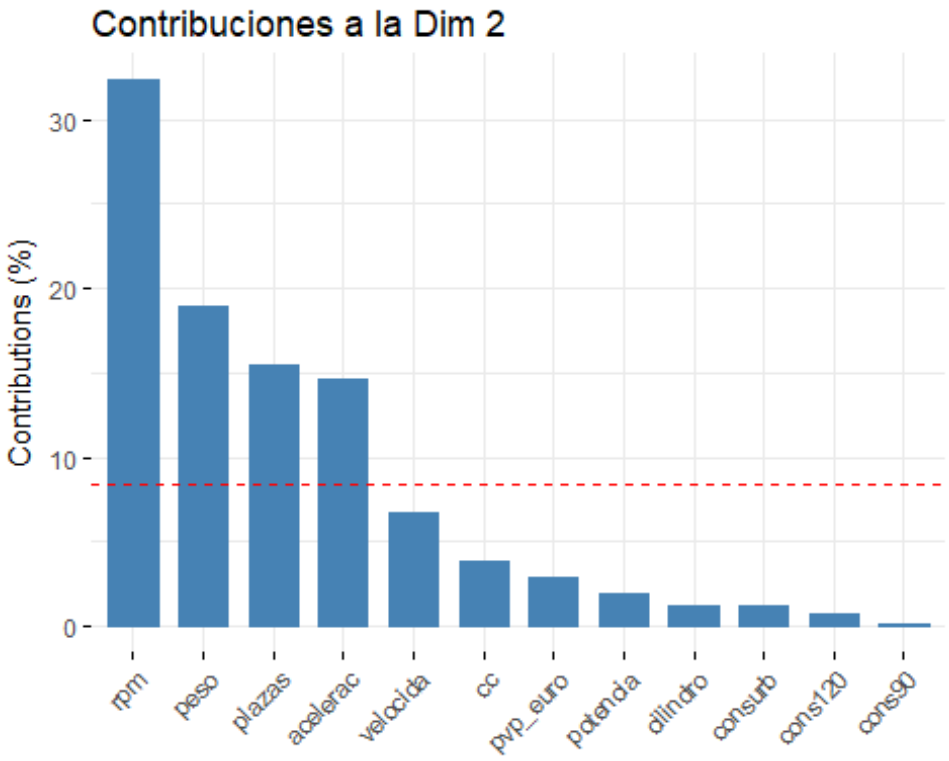


Pasamos a comprobar como contribuyen las variables a las dos dimensiones:

```
fviz_contrib(tterreno_variables.acp, choice="var", axes = 1 )+
  labs(title = "Contribuciones a la Dim 1")
```

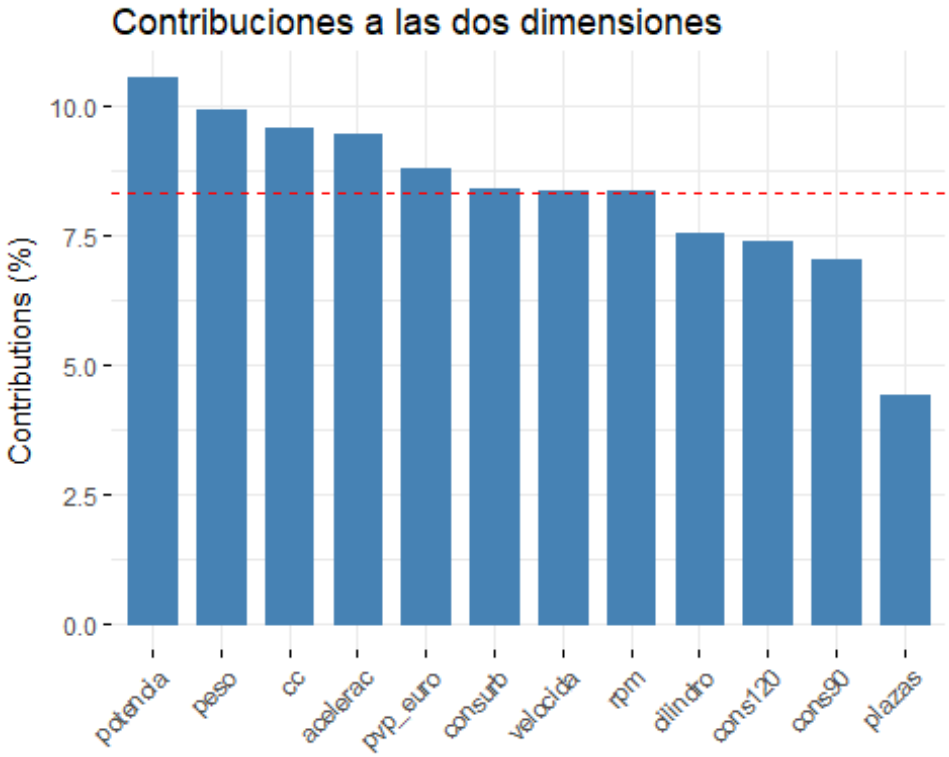



```
fviz_contrib(tterreno_variables.acp, choice="var", axes = 2 )+
  labs(title = "Contribuciones a la Dim 2")
```



Y a ambos ejes:

```
fviz_contrib(tterreno_variables.acp, choice="var", axes = 1:2) +
  labs(title = "Contribuciones a las dos dimensiones")
```



Las dimensiones escogidas serían:

```
tterreno_variables.acp$var$cor[,1:2]
```

```
##                Dim.1       Dim.2
## pvp_euro   0.83671854  0.25065323
## cilindro   0.79096100 -0.16688474
## cc         0.86357506  0.29303227
## potencia   0.93459832 -0.20422035
## rpm       -0.08823259 -0.84729436
## plazas     0.20100679  0.58626132
## peso       0.66443626  0.64912816
## cons90     0.77991337 -0.04304536
## cons120    0.79093932  0.12516733
## consurb    0.83871621 -0.15935896
## velocida   0.75943991 -0.38669900
## acelerac  -0.70452215  0.57048671
```

Y sus coordenadas en el espacio 2D que crean:

```
ACP = tterreno_variables.acp$ind$coord[,1:2]
ACP
```

```
##           Dim.1       Dim.2
## 1   -1.55482145 -2.07985517
## 2   -2.65960631 -0.14221476
## 3   -2.63784941 -0.13171317
## 4    4.03794845 -0.77294646
## 5   -2.10238904 -2.18670141
## 6    0.45772332 -0.17881465
## 7   -1.41914238  1.34209577
## 8   -1.34772296  1.38806410
## 9   -0.52891337  2.15097906
## 10  -0.45514189  2.18864098
## 11  -0.42201303  2.20555397
## 12   3.29540445  0.23024394
## 13   0.42439141 -1.57188113
## 14   0.58785985 -1.78847148
## 15   4.15157235 -1.52309378
## 16   0.93498150 -1.22592335
## 17   3.93393322 -1.79758772
## 18  -0.80371138 -0.39173963
## 19  -0.71738128 -0.45524279
## 20   4.11010911 -1.35732198
## 21   6.44941755 -1.46267541
## 22   0.03358555  0.41075585
## 23  -0.78847462 -0.76293457
## 24  -0.20780249 -2.21672981
## 25  -2.67449572 -0.68788484
```

```
## 26  -3.65818614 -0.12642170
## 27  -0.93562922  1.02450994
## 28  -1.12133069  0.10410374
## 29  -0.86218810  1.10873734
## 30  -0.99754078  0.72217045
## 31  -0.93572495  0.78763391
## 32  -0.61740076  2.32095526
## 33  -0.42835948  2.78857731
## 34  -1.63314470  1.45783368
## 35  -1.28576607  2.50568096
## 36  -1.23185054  2.54695133
## 37   5.86224985 -1.00775101
## 38   7.73004056 -0.89263856
## 39  -0.74060252  1.35029594
## 40   1.32779484  0.39503360
## 41   1.40856777  0.43626991
## 42   6.20338810 -0.99864266
## 43   6.24529761 -0.92684740
## 44   6.37607971 -0.13273585
## 45   3.69136213  1.74937381
## 46   3.71298985  1.78790589
## 47   4.09937133  2.59418296
## 48   3.37800330 -1.95488665
## 49   3.49482146 -1.89524848
## 50   3.69717233 -0.98103380
## 51   4.76609832 -1.74244038
## 52   5.18404833 -0.70784146
## 53  -0.76648132  0.62385399
## 54  -0.61191247  0.74858248
## 55  -0.61940264  2.25323512
## 56   0.72821914  0.70292406
## 57   0.84573745  0.76291967
## 58   0.83968416  1.11630598
## 59   0.95373331  1.95246839
## 60   1.13356274  1.36997148
## 61  -1.03276657  0.93815668
## 62  -1.07024739  0.89965859
## 63  -0.05317854 -1.19413280
## 64   0.03220511 -1.15054262
## 65   0.28071248 -1.24779858
## 66   0.33425856 -0.24184605
## 67   0.41366536 -0.20130718
## 68   0.47924001 -0.16782992
## 69  -0.82629749  1.24695045
## 70  -0.78923533  1.24754437
## 71  -0.71392694  1.28599091
## 72  -1.12752382  2.02861692
## 73  -0.97933176  2.21423481
## 74  -0.91375711  2.24771207
## 75  -0.26442600  0.50208427
```

```
## 76  -0.01865876  1.25762283
## 77  -0.20344178 -0.10118958
## 78  -0.24406618  0.15459406
## 79   0.21889814  2.19127158
## 80   1.17390654  0.54262701
## 81   1.22951996  1.11874515
## 82  -0.75419197 -0.56542388
## 83  -0.94041239 -0.88591688
## 84  -0.37365888 -1.08199905
## 85  -0.63891509  1.10382484
## 86  -1.01542394  1.36673321
## 87   1.66691872 -1.15573705
## 88   3.67183086 -1.87066633
## 89   1.63169385  1.01435112
## 90   1.20624210  1.72086187
## 91  -1.93729379  0.08632898
## 92  -0.98935769  1.28363231
## 93  -1.23310274  1.71691802
## 94  -1.13064235  1.76922623
## 95  -4.15282774 -2.20695968
## 96  -4.12192452 -2.17285581
## 97  -4.11616966 -2.16991783
## 98  -4.02785721 -2.04236043
## 99  -2.97680676 -1.68170324
## 100 -2.95071635 -1.65921997
## 101 -2.89881040 -1.64580246
## 102 -2.67545150 -1.82760751
## 103 -2.66027312 -1.81069506
## 104 -2.63370738 -1.77880558
## 105 -2.14164089 -1.79003235
## 106 -2.12432900 -1.77028751
## 107 -2.46524690 -1.86520350
## 108 -2.37708205 -1.53495197
## 109 -0.08900112 -2.96666859
## 110 -0.03556943 -2.94678085
## 111 -3.46864870 -0.54560988
## 112 -3.44627818 -0.53418926
## 113 -3.43551984 -0.52869689
## 114 -2.82708433 -0.30300351
## 115 -2.60681200  0.30709805
## 116 -1.30529534 -2.67102299
## 117 -1.23639162 -2.21295294
## 118  2.90685123 -0.59862820
## 119  0.92097861  1.00254063
## 120  0.99019262  1.02147199
## 121  3.88309240  2.46157240
## 122 -2.11934956  1.33504196
## 123 -2.09636278  1.34843030
## 124 -2.09045270  1.35144752
## 125 -2.04090430  1.33646600
```

```r
summary(ACP)
```

```
##      Dim.1              Dim.2         
##  Min.   :-4.1528   Min.   :-2.9667  
##  1st Qu.:-1.4191   1st Qu.:-1.2478  
##  Median :-0.6194   Median :-0.1264  
##  Mean   : 0.0000   Mean   : 0.0000  
##  3rd Qu.: 0.9537   3rd Qu.: 1.2836  
##  Max.   : 7.7300   Max.   : 2.7886
```

ANEXO VI: Agrupación

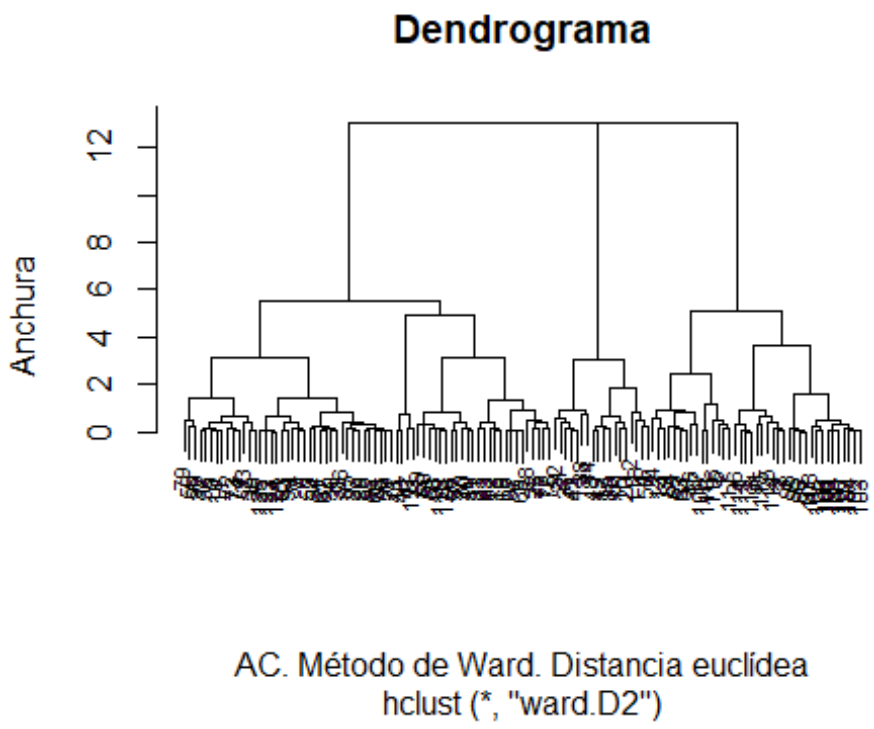
Una vez sabemos que podemos reducir la dimensión, pasamos a analizar si podemos agrupar las observaciones teniendo en cuenta el análisis realizado anteriormente.

Calculamos las distancias. No las representamos ya que debido al número de observaciones no se podrán obtener resultados claros de su visualización.

```
q.dist = get_dist(ACP, stand = TRUE, method = "euclidean")
```

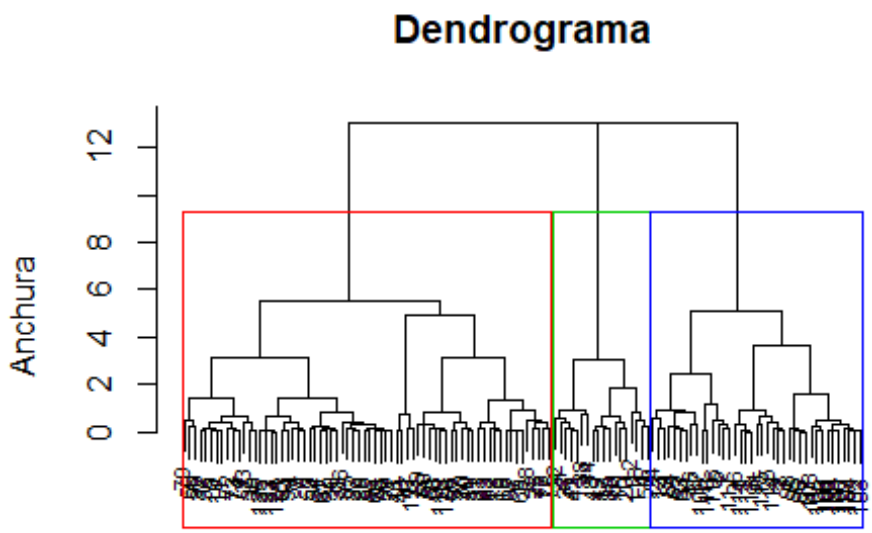
Creamos un dendograma que nos permita apreciar si existen agrupaciones:

```
q.hc = hclust(q.dist, method = "ward.D2")
plot(hclust(q.dist, method = "ward.D2"), cex=0.7, main="Dendrograma", yla
b="Anchura",
     xlab="AC. Método de Ward. Distancia euclídea")
```



Se pueden apreciar tres grupos distintos. Para comprobar, realizamos el mismo dendograma aplicando tres grupos:

```
plot(hclust(q.dist, method = "ward.D2"), cex=0.7, main="Dendrograma", yla
b="Anchura",
     xlab="AC. Método de Ward. Distancia euclídea")
rect.hclust(q.hc, k=3, border = 2:4)
```



Teniendo en cuenta estos resultados, dividimos los datos en tres grupos:

```
grp = cutree(q.hc, k = 3)
pam.q = pam(ACP, 3)
pam.q$medoids

##          Dim.1      Dim.2
## 108 -2.3770821 -1.5349520
## 4    4.0379484 -0.7729465
## 85  -0.6389151  1.1038248

clusters = fviz_cluster(pam.q, data=ACP, labelsize=8, stand=F, repel=TRUE
)
```

c

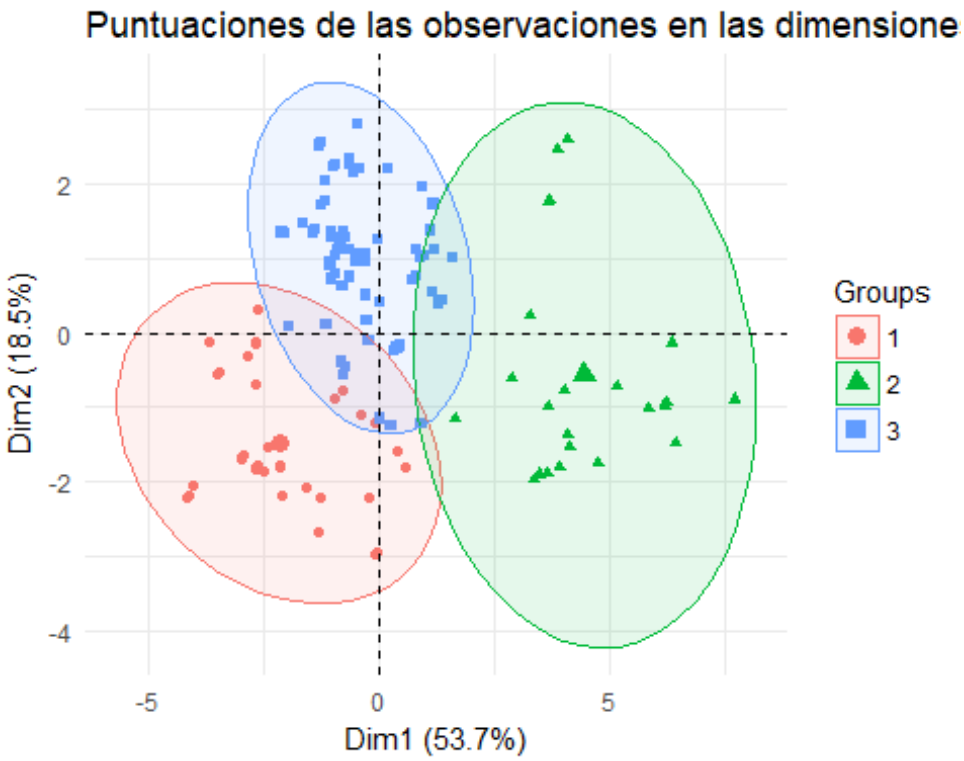
E incluimos el cluster en el que se agrupan los diferentes grupos en el data frame donde tenemos el resto de la informacion.

```
tterreno = cbind(tterreno, clusters$data$cluster)
```

Comprobamos si los clusters pueden diferenciarse claramente:

```
acp_g= fviz_pca_ind(tterreno_variables.acp, geom = "point",
                    habillage=tterreno$`clusters$data$cluster`, addEllips
```

```
es=T,
                     ellipse.level= 0.95)+
  labs(title = "Puntuaciones de las observaciones en las dimensiones")+
  theme_minimal()
print(acp_g)
```



ANEXO VII: Diferencias en los clusters

Una vez hemos comprobados como efectivamente los cluster agrupan diferentes observaciones de manera clara, podemos pasar a examinar las diferencias entre los grupos

```
grupo1 = subset(tterreno, tterreno$`clusters$data$cluster`==1)
summary(grupo1)
```

```
##     marca              modelo             pvp_euro        cilindro
##  Length:36          Length:36          Min.   : 9113   Min.   :1.000
##  Class :character   Class :character   1st Qu.:13508   1st Qu.:1.000
##  Mode  :character   Mode  :character   Median :15910   Median :1.000
##                                        Mean   :16134   Mean   :1.056
##                                        3rd Qu.:18190   3rd Qu.:1.000
##                                        Max.   :25553   Max.   :2.000
##        cc          potencia           rpm           plazas
##  Min.   :1298   Min.   : 64.00   Min.   :4250   Min.   :1.000
##  1st Qu.:1590   1st Qu.: 69.00   1st Qu.:5150   1st Qu.:2.000
##  Median :1905   Median : 95.00   Median :5450   Median :2.000
##  Mean   :1821   Mean   : 93.69   Mean   :5372   Mean   :2.306
##  3rd Qu.:1998   3rd Qu.:116.50   3rd Qu.:5600   3rd Qu.:3.000
##  Max.   :2464   Max.   :136.00   Max.   :6500   Max.   :3.000
##       peso          cons90          cons120         consurb
##  Min.   : 930   Min.   : 6.700   Min.   : 8.40   Min.   : 8.10
##  1st Qu.:1130   1st Qu.: 7.675   1st Qu.: 9.00   1st Qu.: 9.80
##  Median :1220   Median : 7.900   Median :10.60   Median :10.30
##  Mean   :1257   Mean   : 8.094   Mean   :10.44   Mean   :10.86
##  3rd Qu.:1325   3rd Qu.: 8.350   3rd Qu.:11.39   3rd Qu.:12.00
##  Max.   :1696   Max.   :10.600   Max.   :14.50   Max.   :15.90
##     velocida        acelerac     clusters$data$cluster
##  Min.   :127.0   Min.   :13.20   1:36
##  1st Qu.:136.5   1st Qu.:15.13   2: 0
##  Median :144.0   Median :17.05   3: 0
##  Mean   :145.7   Mean   :16.20
##  3rd Qu.:152.0   3rd Qu.:17.53
##  Max.   :170.0   Max.   :19.00
```

```
grupo2 = subset(tterreno, tterreno$`clusters$data$cluster`==2)
summary(grupo2)
```

```
##     marca              modelo             pvp_euro        cilindro
##  Length:23          Length:23          Min.   :21672   Min.   :2.00
##  Class :character   Class :character   1st Qu.:31676   1st Qu.:2.00
##  Mode  :character   Mode  :character   Median :39657   Median :2.00
##                                        Mean   :44436   Mean   :2.13
##                                        3rd Qu.:62362   3rd Qu.:2.00
##                                        Max.   :69461   Max.   :3.00
##        cc          potencia          rpm           plazas
##  Min.   :2959   Min.   :136.0   Min.   :3600   Min.   :2.000
```

```
##  1st Qu.:3182   1st Qu.:173.5   1st Qu.:4300   1st Qu.:3.000
##  Median :3497   Median :181.0   Median :4750   Median :3.000
##  Mean   :3618   Mean   :182.2   Mean   :4777   Mean   :3.391
##  3rd Qu.:3960   3rd Qu.:208.0   3rd Qu.:5350   3rd Qu.:3.000
##  Max.   :5216   Max.   :225.0   Max.   :5500   Max.   :6.000
##       peso          cons90         cons120         consurb
##  Min.   :1455   Min.   : 7.40   Min.   :11.00   Min.   : 9.80
##  1st Qu.:1778   1st Qu.:10.30   1st Qu.:13.90   1st Qu.:15.75
##  Median :1925   Median :10.80   Median :14.60   Median :17.30
##  Mean   :1944   Mean   :10.92   Mean   :14.94   Mean   :16.57
##  3rd Qu.:2130   3rd Qu.:11.60   3rd Qu.:16.20   3rd Qu.:18.30
##  Max.   :2320   Max.   :13.70   Max.   :18.50   Max.   :22.10
##     velocida      acelerac     clusters$data$cluster
##  Min.   :145   Min.   : 9.40   1: 0
##  1st Qu.:170   1st Qu.:10.55   2:23
##  Median :175   Median :11.50   3: 0
##  Mean   :173   Mean   :11.83
##  3rd Qu.:180   3rd Qu.:12.43
##  Max.   :196   Max.   :16.00
```

```
grupo3 = subset(tterreno, tterreno$`clusters$data$cluster`==3)
summary(grupo3)
```

```
##     marca              modelo             pvp_euro        cilindro
##  Length:66          Length:66          Min.   :11555   Min.   :1.000
##  Class :character   Class :character   1st Qu.:22228   1st Qu.:1.000
##  Mode  :character   Mode  :character   Median :25303   Median :1.000
##                                        Mean   :26276   Mean   :1.136
##                                        3rd Qu.:30252   3rd Qu.:1.000
##                                        Max.   :52880   Max.   :2.000
##        cc          potencia          rpm           plazas
##  Min.   :1998   Min.   : 68.0   Min.   :3600   Min.   :1.000
##  1st Qu.:2495   1st Qu.: 99.0   1st Qu.:4000   1st Qu.:3.000
##  Median :2499   Median :112.0   Median :4000   Median :3.000
##  Mean   :2613   Mean   :107.2   Mean   :4252   Mean   :3.682
##  3rd Qu.:2826   3rd Qu.:115.0   3rd Qu.:4500   3rd Qu.:5.000
##  Max.   :3059   Max.   :134.0   Max.   :5300   Max.   :7.000
##       peso          cons90          cons120         consurb
##  Min.   :1450   Min.   : 6.600   Min.   : 8.60   Min.   : 8.60
##  1st Qu.:1730   1st Qu.: 8.000   1st Qu.:10.88   1st Qu.:10.92
##  Median :1832   Median : 8.600   Median :12.30   Median :11.80
##  Mean   :1810   Mean   : 8.551   Mean   :12.05   Mean   :12.05
##  3rd Qu.:1911   3rd Qu.: 9.175   3rd Qu.:13.20   3rd Qu.:13.10
##  Max.   :2115   Max.   :10.600   Max.   :16.20   Max.   :18.10
##     velocida      acelerac     clusters$data$cluster
##  Min.   :120   Min.   :12.30   1: 0
##  1st Qu.:135   1st Qu.:15.26   2: 0
##  Median :145   Median :17.46   3:66
##  Mean   :145   Mean   :17.13
```

```
##  3rd Qu.:155   3rd Qu.:18.80
##  Max.   :169   Max.   :22.00
```

Se desprende claramente de las tres tablas anteriores que la agrupación está hecha siguiendo criterios de la gama del vehículo, tanto por el precio como por las características de los todo-terreno.

Marca más común en gama baja:

```
gama_baja = grupo1

sort(table(gama_baja$marca),decreasing=TRUE)[1]

## SUZUKI
##     19
```

Marca más común en gama media:

```
gama_media = grupo3

sort(table(gama_media$marca),decreasing=TRUE)[1]

## NISSAN
##     18
```

Marca más común en gama alta:

```
gama_alta = grupo2

sort(table(gama_alta$marca),decreasing=TRUE)[1]

## MERCEDES
##        6
```

Añadimos al dataset original una columna que recoja el "nombre" de los clusters:

```
tterreno$gama = 0
tterreno$gama = replace(tterreno$gama,
                        tterreno$`clusters$data$cluster` == 1,
                        "Gama baja")
tterreno$gama = replace(tterreno$gama,
                        tterreno$`clusters$data$cluster` == 2,
                        "Gama alta")
tterreno$gama = replace(tterreno$gama,
```

```
                        tterreno$`clusters$data$cluster` == 3,
                        "Gama media")
```