Machine Learning I. Trabajo de evaluación. Temas: 1-Conglomerados, 2-Reducción de Dimensionalidad, 4-Árboles

Jerónimo Carranza Carranza

8 de mayo de 2017

Contents

1 Conglomerados

Leer el fichero "Crimen.dat", que contiene el total de delitos por cada 100.000 habitantes para cada uno de los estados de EEUU más el distrito de Columbia (Año 1986). Aplicar y comparar tres técnicas de análisis de conglomerados (una de tipo jerárquico, otra de tipo partición y el método basado en mixturas de normales multivariantes.

1.1 Lectura de datos

```
df <- read.table("Crimen.dat", header=TRUE, sep=" ")
head(df)
```

```
##    Asesinato Abusos Atraco Agresión Robo_domicilio Hurto Robo_vehículo
## ME       2.0   14.8     28      102            803  2347           164
## NH       2.2   21.5     24       92            755  2208           228
## VT       2.0   21.8     22      103            949  2697           181
## MA       3.6   29.7    193      331           1071  2189           906
## RI       3.5   21.4    119      192           1294  2568           705
## CT       4.6   23.8    192      205           1198  2758           447
```

```
str(df)
```

```
## 'data.frame':    51 obs. of  7 variables:
##  $ Asesinato     : num  2 2.2 2 3.6 3.5 4.6 10.7 5.2 5.5 5.5 ...
##  $ Abusos        : num  14.8 21.5 21.8 29.7 21.4 23.8 30.5 33.2 25.1 38.6 ...
##  $ Atraco        : int  28 24 22 193 119 192 514 269 152 142 ...
##  $ Agresión      : int  102 92 103 331 192 205 431 265 176 235 ...
##  $ Robo_domicilio: int  803 755 949 1071 1294 1198 1221 1071 735 988 ...
##  $ Hurto         : int  2347 2208 2697 2189 2568 2758 2924 2822 1654 2574 ...
##  $ Robo_vehículo : int  164 228 181 906 705 447 637 776 354 376 ...
```

```
summary(df)
```

```
##    Asesinato          Abusos          Atraco          Agresión
##  Min.   : 1.000   Min.   :11.60   Min.   :  7.0   Min.   : 32.0
##  1st Qu.: 3.800   1st Qu.:23.45   1st Qu.: 69.0   1st Qu.:177.0
##  Median : 6.600   Median :30.50   Median :112.0   Median :252.0
##  Mean   : 7.251   Mean   :34.22   Mean   :154.1   Mean   :283.4
##  3rd Qu.: 9.700   3rd Qu.:43.75   3rd Qu.:207.0   3rd Qu.:385.5
##  Max.   :31.000   Max.   :72.70   Max.   :754.0   Max.   :668.0
##  Robo_domicilio     Hurto      Robo_vehículo
##  Min.   : 385   Min.   :1358   Min.   : 99.0
##  1st Qu.: 901   1st Qu.:2385   1st Qu.:211.5
##  Median :1159   Median :2822   Median :328.0
##  Mean   :1207   Mean   :2942   Mean   :393.8
##  3rd Qu.:1457   3rd Qu.:3400   3rd Qu.:544.5
##  Max.   :2221   Max.   :4373   Max.   :975.0
```

1.2 Normalización y Exploración de outliers

```
library(ggplot2)
library(reshape)
```

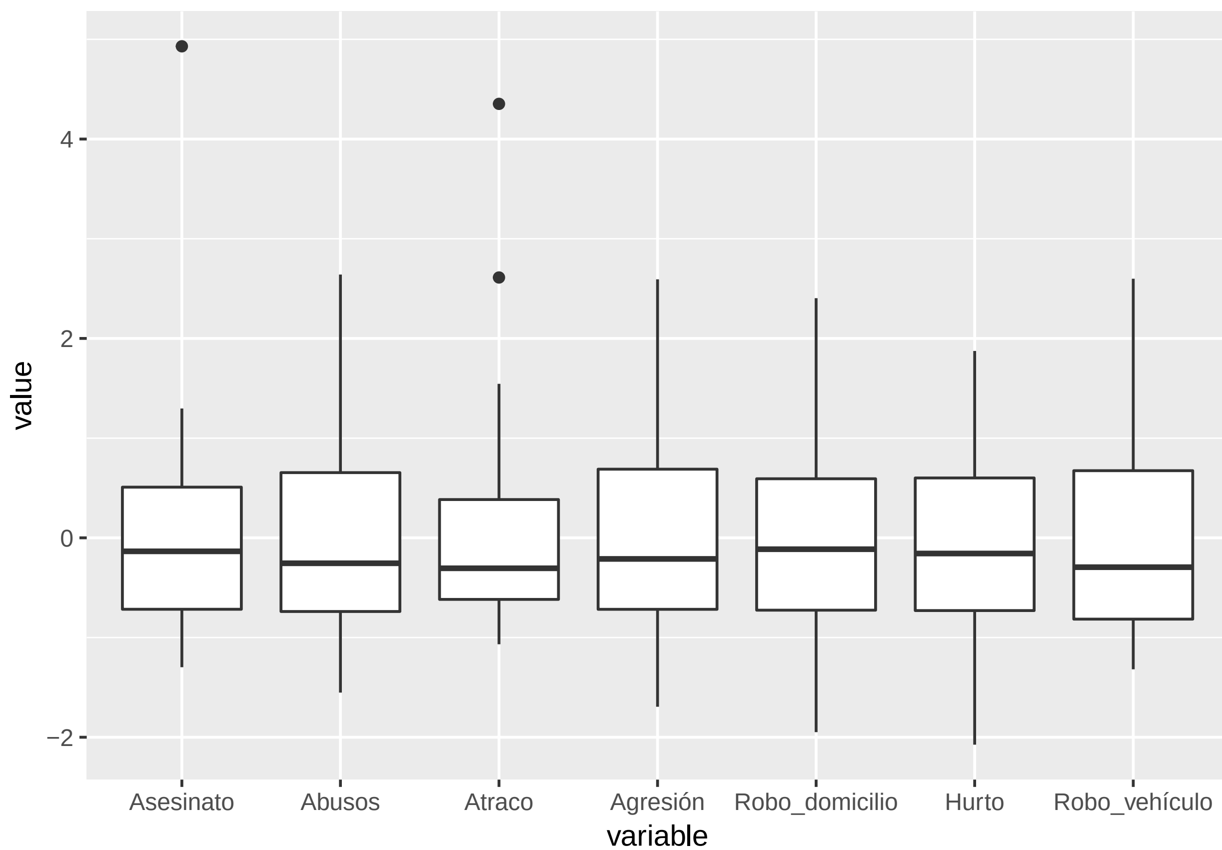
```r
zdf = as.data.frame(scale(df))
head(zdf)
```

```
##     Asesinato     Abusos     Atraco   Agresión Robo_domicilio      Hurto
## ME -1.0901250 -1.3326283 -0.9149710 -1.2225614    -0.95799233 -0.7793343
## NH -1.0486042 -0.8728090 -0.9439951 -1.2899748    -1.07179111 -0.9614093
## VT -1.0901250 -0.8522201 -0.9585071 -1.2158201    -0.61185438 -0.3208721
## MA -0.7579584 -0.3100450  0.2822737  0.3212049    -0.32261582 -0.9862972
## RI -0.7787189 -0.8796720 -0.2546724 -0.6158410     0.20607434 -0.4898482
## CT -0.5503544 -0.7149606  0.2750177 -0.5282036    -0.02152322 -0.2409687
##    Robo_vehículo
## ME    -1.0278140
## NH    -0.7416184
## VT    -0.9517933
## MA     2.2902662
## RI     1.3914332
## CT     0.2377072
```

```r
zplot = ggplot(melt(zdf), aes(x=variable, y=value)) + geom_boxplot()
```

```
## Using  as id variables
```

```r
print(zplot)
```



Se puede observar en el gráfico que existen dos variables con observaciones outliers, en concreto, Asesinato y Atraco. Se pueden extraer como los casos con valores normalizados mayor de 2 (observable gráficamente):

```r
zdf[(zdf$Asesinato>2 | zdf$Atraco>2) ,]
```

```
##    Asesinato     Abusos   Atraco  Agresión Robo_domicilio       Hurto
## NY 0.7160306 -0.2551412 2.611459 0.9953387     0.03300536 -0.02352669
## DC 4.9303935  1.2478504 4.352906 2.5930358     1.23500496  1.55751280
##    Robo_vehículo
## NY      1.087350
## DC      2.598821
```

```r
# zdf[(zdf$Asesinato>2 | zdf$Atraco>2) ,c("Atraco","Asesinato")]
df[(zdf$Asesinato>2 | zdf$Atraco>2) ,]
```

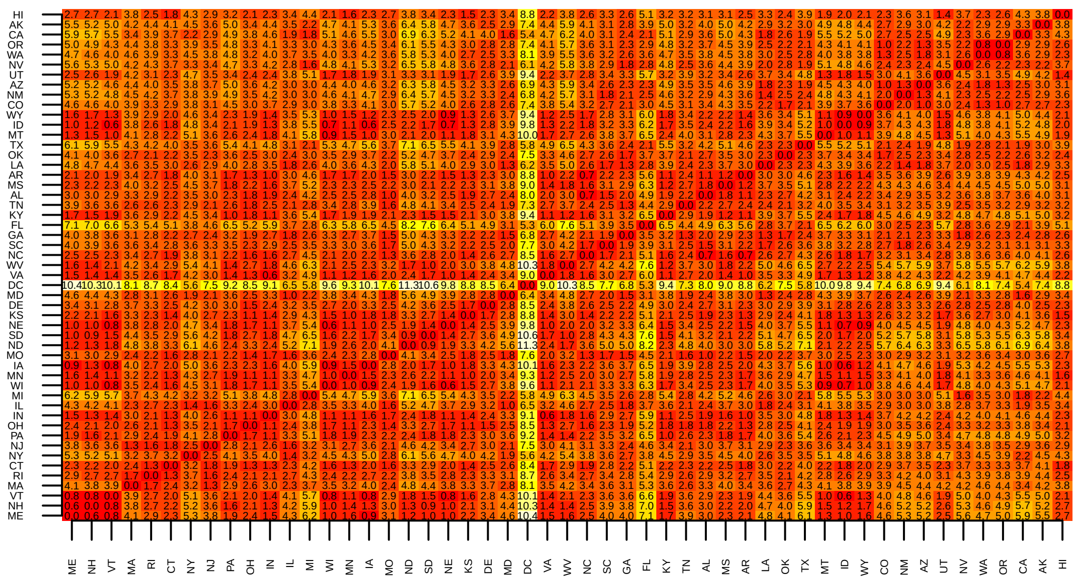
```
##    Asesinato Abusos Atraco Agresión Robo_domicilio Hurto Robo_vehículo
## NY      10.7   30.5    514      431           1221  2924           637
## DC      31.0   52.4    754      668           1728  4131           975
```

Los casos de outliers en dichas variables corresponden a: NY y DC, esto es, New Tork y District Columbia (Washington).

En el primer caso, New York, es outlier respecto a la variable Atraco, mientras que District Columbia, lo es tanto para Atraco como, especialmente, para Asesinato.

1.3 Cálculo y representación de la matriz de distancias entre estados.

```r
D = dist(zdf) # Distancia euclídea
dm <- data.matrix(D)
dim <- ncol(dm)
image(1:dim, 1:dim, dm, axes = FALSE, xlab="", ylab="")
axis(1, 1:dim, row.names(df), cex.axis = 0.3, las=2)
axis(2, 1:dim, row.names(df), cex.axis = 0.3, las=1)
text(expand.grid(1:dim, 1:dim), sprintf("%0.1f", dm), cex=0.3)
```

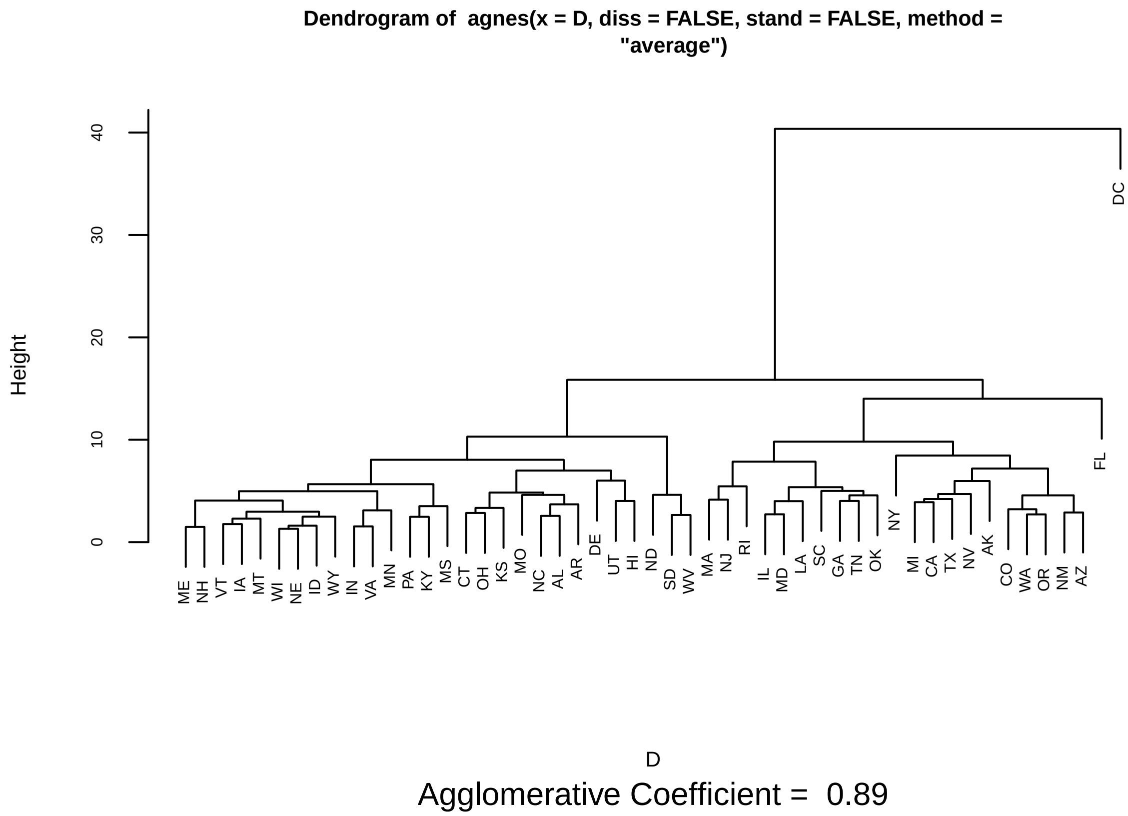


Es destacadamente distinguible la mayor distancia de DC al resto de estados en la representación de la matriz de distancias.

1.4 Técnica de agrupamiento jerárquico aglomerativo "agnes"

Se realiza un agrupamiento jerarquico con la matriz de distancias normalizada y con aglomeración por la media de grupo (average) con la función "agnes" del paquete "cluster".

```
library(cluster)
Agnes1 = agnes(D,diss=FALSE,stand=FALSE,method="average")

plot(Agnes1,FALSE,2,
     cex=0.5,cex.main=0.7,cex.axis=0.5,
     cex.lab=0.7)
```




```r
# library(ggdendro)
# ggdendrogram(Agnes1, rotate = FALSE, theme_dendro = TRUE,
#              cex = 0.4)
```

El dendrograma muestra muy claramente la agregación final, a mucha distancia, del distrito federal al resto de estados, entre los cuales, se distingue también muy claramente, Florida (FL).

El coeficiente de aglomeración es de 0,89 aproximadamente, que es relativamente alto.

```r
summary(Agnes1)
```

```
## Object of class 'agnes' from call:
##  agnes(x = D, diss = FALSE, stand = FALSE, method = "average")
## Agglomerative coefficient:  0.8944398
## Order of objects:
##  [1] ME NH VT IA MT WI NE ID WY IN VA MN PA KY MS CT OH KS MO NC AL AR DE
## [24] UT HI ND SD WV MA NJ RI IL MD LA SC GA TN OK NY MI CA TX NV AK CO WA
## [47] OR NM AZ FL DC
## Merge:
##       [,1] [,2]
##  [1,]  -14  -20
##  [2,]   -1   -2
##  [3,]  -11  -25
##  [4,]    1  -40
##  [5,]   -3  -16
##  [6,]    5  -39
##  [7,]   -9  -31
##  [8,]    4  -41
```

```
##  [9,]  -27  -33
## [10,]  -19  -26
## [11,]  -47  -48
## [12,]  -12  -23
## [13,]   -6  -10
## [14,]  -43  -44
## [15,]    6    8
## [16,]    3  -15
## [17,]  -42   11
## [18,]   13  -21
## [19,]    7  -34
## [20,]    9  -35
## [21,]  -13  -49
## [22,]   12  -36
## [23,]  -45  -51
## [24,]  -29  -32
## [25,]    2   15
## [26,]   -4   -8
## [27,]   21  -38
## [28,]   17   14
## [29,]   24  -37
## [30,]  -17   20
## [31,]  -18   10
## [32,]   27  -46
## [33,]   18   30
## [34,]   25   16
## [35,]  -28   29
## [36,]   22   35
## [37,]   26   -5
## [38,]   34   19
## [39,]   32  -50
## [40,]  -22   23
## [41,]   33   40
## [42,]   39   28
## [43,]   37   36
## [44,]   38   41
## [45,]   -7   42
## [46,]   43   45
## [47,]   44   31
## [48,]   46  -30
## [49,]   47   48
## [50,]   49  -24
## Height:
##  [1]  1.485257  4.057354  1.764794  2.293472  2.970018  1.299376  1.603844
##  [8]  2.485421  4.973368  1.537254  3.104387  5.660352  2.473323  3.515611
## [15]  8.043155  2.850941  3.342439  4.840688  4.613637  2.564110  3.694409
## [22]  6.987229  6.006600  4.019045 10.301613  4.617395  2.651501 15.855157
## [29]  4.145846  5.451281  7.857314  2.719254  4.002586  5.368022  5.000123
## [36]  4.022855  4.570229  9.810814  8.454734  3.905699  4.211590  4.700157
## [43]  5.969770  7.185865  3.213917  2.709407  4.568831  2.892128 14.003761
## [50] 40.363675
##
## 1275 dissimilarities, summarized :
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
```

```
##  1.2994  7.3441 11.2300 12.9480 16.9320 46.1750
## Metric :  euclidean
## Number of objects : 51
##
## Available components:
## [1] "order"     "height"    "ac"        "merge"     "diss"      "call"
## [7] "method"    "order.lab" "data"
```

El resumen global del análisis "agnes" muestra el coeficiente de agregación, el orden de agregación, los elementos agregados en cada etapa del proceso, las distancias a las que se produce la aglomeración en cada etapa y un resumen estadístico de las mismas.

Se puede utilizar la función rect de hclust para identificar distintos grupos dentro del dendrograma de agnes, bien por altura (distancia de separación) o número de grupos.

```r
plot(Agnes1,FALSE,2,main="",cex=0.5,cex.axis=0.5)
rect.hclust(Agnes1, k = 8, border="red")
```

La clasificación en 8 grupos muestra la segmentación en grupos individuales de DC, FL y NY, y los siguientes cinco grupos multiestado: ME-MS (15), CT-HI (10), ND-WV (3), MA-OK (10), MI-AZ (10).

```r
dfclus = data.frame(df,"grp" = cutree(Agnes1,k=8))

dfclus$grp = factor(dfclus$grp,labels = c('ME-MS','MA-OK','CT-HI','NY','MI-AZ','ND-WV','DC','FL'))

library(doBy)
bygrp = summaryBy(.~grp, data=dfclus, FUN=mean)
rownames(bygrp)=bygrp$grp
```

```r
library(knitr)
kable(bygrp[,2:5],digits = 2)
```

	Asesinato.mean	Abusos.mean	Atraco.mean	Agresión.mean
ME-MS	4.31	22.18	60.07	162.13
MA-OK	8.13	36.90	206.20	358.30
CT-HI	6.29	32.14	115.30	262.60
NY	10.70	30.50	514.00	431.00
MI-AZ	9.67	53.82	204.40	388.80
ND-WV	3.63	16.07	21.33	72.67
DC	31.00	52.40	754.00	668.00
FL	11.70	52.70	367.00	605.00

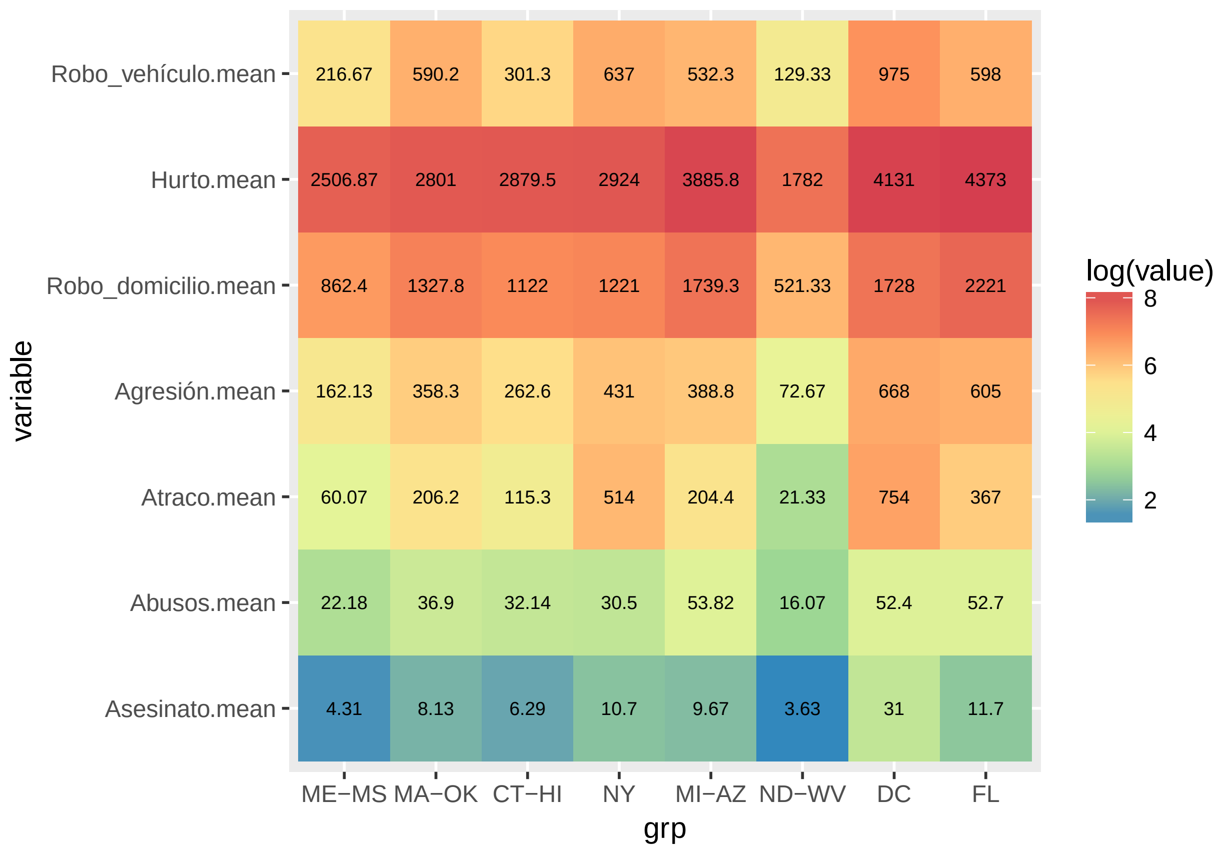
```r
kable(bygrp[,c(6,7,8)],digits = 2)
```

	Robo_domicilio.mean	Hurto.mean	Robo_vehículo.mean
ME-MS	862.40	2506.87	216.67
MA-OK	1327.80	2801.00	590.20
CT-HI	1122.00	2879.50	301.30
NY	1221.00	2924.00	637.00
MI-AZ	1739.30	3885.80	532.30
ND-WV	521.33	1782.00	129.33
DC	1728.00	4131.00	975.00
FL	2221.00	4373.00	598.00

```r
grpplot = ggplot(melt(bygrp), aes(x=grp , y=variable, fill=log(value))) + geom_tile() +
  scale_fill_distiller(palette = "Spectral") +
  geom_text(aes(label = round(value,2)), size=2.5)
```

```
## Using grp as id variables
```

```r
print(grpplot)
```



1.5 Técnica de partición k-medias

Utilizamos los datos normalizados previamente, zdf, y prefijamos 8 grupos.

```r
df.k = kmeans(zdf, centers = 8)
df.k
```

```
## K-means clustering with 8 clusters of sizes 6, 16, 1, 6, 6, 6, 3, 7
##
## Cluster means:
##    Asesinato       Abusos      Atraco    Agresión Robo_domicilio      Hurto
## 1  0.8821138  1.703094438  0.84824397  1.0773583      1.1848229  1.0630286
## 2 -0.7034624 -0.961598730 -0.75624535 -0.9284706     -1.0214114 -0.7165414
## 3  4.9303935  1.247850405  4.35290576  2.5930358      1.2350050  1.5575128
## 4  0.4080845 -0.289456079 -0.37560618  0.4571552     -0.1092431 -0.7485518
## 5  0.6745097  0.368245728  1.04536608  0.8043341      0.2740375 -0.0621685
## 6  0.1381992  0.775448933 -0.03941019  0.4706379      1.5479516  1.3536063
## 7 -0.6541564 -0.419852611  0.28711109 -0.1394532     -0.1463858 -0.5444270
## 8 -0.6185671  0.007612712 -0.28369649 -0.5965801     -0.2626682  0.2721346
##   Robo_vehículo
## 1     1.1872207
## 2    -0.8668289
## 3     2.5988208
## 4    -0.6164078
## 5     0.6431509
## 6     0.1870267
```

```
## 7     1.7968769
## 8    -0.3608760
##
## Clustering vector:
## ME NH VT MA RI CT NY NJ PA OH IN IL MI WI MN IA MO ND SD NE KS DE MD DC VA
##  2  2  2  7  7  8  5  7  2  8  2  5  1  2  8  2  4  2  2  2  8  8  5  3  2
## WV NC SC GA FL KY TN AL MS AR LA OK TX MT ID WY CO NM AZ UT NV WA OR CA AK
##  2  4  4  5  1  2  5  4  4  4  5  6  1  2  2  2  6  6  6  8  1  6  6  1  1
## HI
##  8
##
## Within cluster sum of squares by cluster:
## [1] 15.215314 18.327683  0.000000  6.969120  9.448288  8.644797  2.406303
## [8]  9.848342
##  (between_SS / total_SS =  79.8 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

```
df.k$cluster
```

```
## ME NH VT MA RI CT NY NJ PA OH IN IL MI WI MN IA MO ND SD NE KS DE MD DC VA
##  2  2  2  7  7  8  5  7  2  8  2  5  1  2  8  2  4  2  2  2  8  8  5  3  2
## WV NC SC GA FL KY TN AL MS AR LA OK TX MT ID WY CO NM AZ UT NV WA OR CA AK
##  2  4  4  5  1  2  5  4  4  4  5  6  1  2  2  2  6  6  6  8  1  6  6  1  1
## HI
##  8
```

Añadimos columna con la clasificación generada y comparamos con la anterior obtenida por agnes.

```
dfclus = dfclus[,1:8]
dfclus = data.frame(dfclus,"grpk" = df.k$cluster)
table(dfclus$grp,dfclus$grpk)
```

```
##
##           1  2  3  4  5  6  7  8
##   ME-MS   0 13  0  1  0  0  0  1
##   MA-OK   0  0  0  1  5  1  3  0
##   CT-HI   0  0  0  4  0  0  0  6
##   NY      0  0  0  0  1  0  0  0
##   MI-AZ   5  0  0  0  0  5  0  0
##   ND-WV   0  3  0  0  0  0  0  0
##   DC      0  0  1  0  0  0  0  0
##   FL      1  0  0  0  0  0  0  0
```

```
orderBy(~ grp + grpk, dfclus[,c(8,9)])
```

```
##      grp grpk
## ME ME-MS    2
## NH ME-MS    2
## VT ME-MS    2
## PA ME-MS    2
## IN ME-MS    2
## WI ME-MS    2
```

```
## IA ME-MS    2
## NE ME-MS    2
## VA ME-MS    2
## KY ME-MS    2
## MT ME-MS    2
## ID ME-MS    2
## WY ME-MS    2
## MS ME-MS    4
## MN ME-MS    8
## SC MA-OK    4
## IL MA-OK    5
## MD MA-OK    5
## GA MA-OK    5
## TN MA-OK    5
## LA MA-OK    5
## OK MA-OK    6
## MA MA-OK    7
## RI MA-OK    7
## NJ MA-OK    7
## MO CT-HI    4
## NC CT-HI    4
## AL CT-HI    4
## AR CT-HI    4
## CT CT-HI    8
## OH CT-HI    8
## KS CT-HI    8
## DE CT-HI    8
## UT CT-HI    8
## HI CT-HI    8
## NY    NY    5
## MI MI-AZ    1
## TX MI-AZ    1
## NV MI-AZ    1
## CA MI-AZ    1
## AK MI-AZ    1
## CO MI-AZ    6
## NM MI-AZ    6
## AZ MI-AZ    6
## WA MI-AZ    6
## OR MI-AZ    6
## ND ND-WV    2
## SD ND-WV    2
## WV ND-WV    2
## DC    DC    3
## FL    FL    1
```

Representamos hotmap de grupos de kmeans y medias de las variables.

```
# bygrpk = summaryBy(.~grpk, data=dfclus, FUN=mean)
# rownames(bygrpk)=bygrpk$grpk
#
# (bygrpk.melted = melt(bygrpk))
#
# grpkplot = ggplot(melt(bygrpk), aes(x=grpk , y=variable, fill=log(value))) + geom_tile() +
#   scale_fill_distiller(palette = "Spectral") +
```

```
#   geom_text(aes(label = round(value,2)), size=2.5)
#
# print(grpkplot)
```

1.6 Técnica de mixturas de normales multivariantes

Utilizamos los datos normalizados previamente, zdf, con la función de mclust para la obtención automática del mejor modelo de mixtura.

```
library(mclust)
```

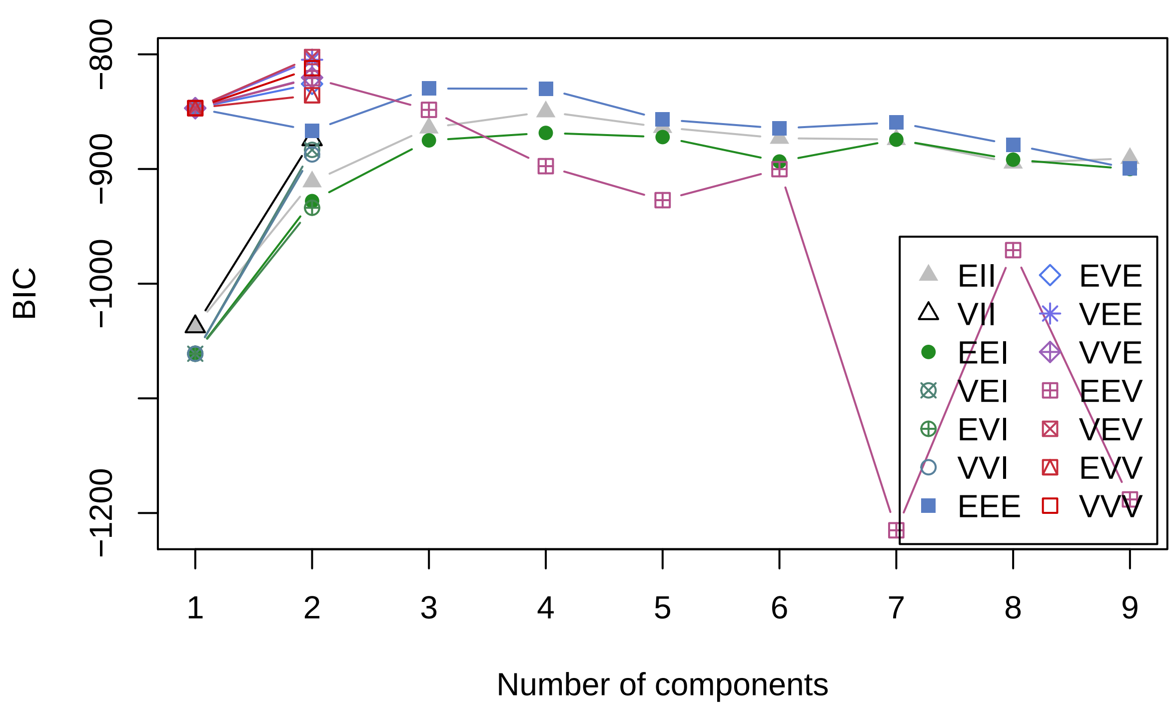
```
## Package 'mclust' version 5.2.3
```

```
## Type 'citation("mclust")' for citing this R package in publications.
```

```
df.m = Mclust(zdf)
df.m
```

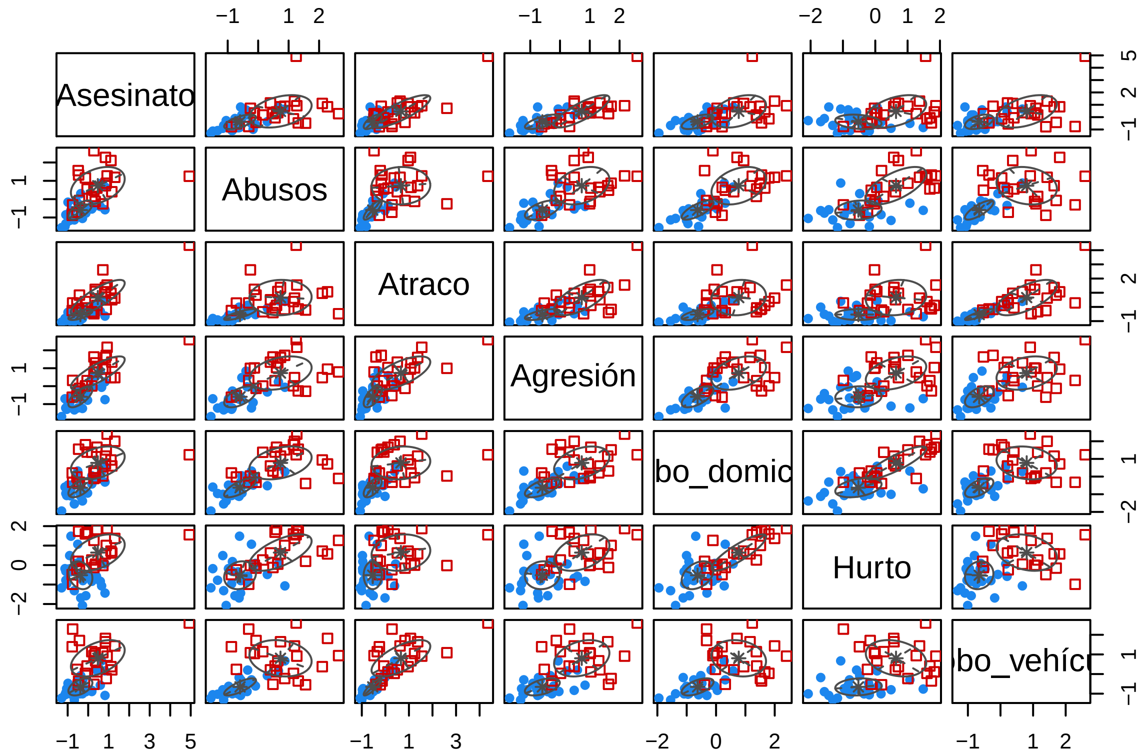
```
## 'Mclust' model object:
##  best model: ellipsoidal, equal shape (VEV) with 2 components
```

```
plot(df.m,what = 'BIC')
```



El resultado muestra como mejor modelo una mixtura de dos componentes. Hay un curioso repunte del indicador BIC para el modelo EEV para 8 componentes.

```
plot(df.m, what = 'classification')
```




```r
table(df.m$classification)
```

```
##
##  1  2
## 28 23
```

```r
df.m$classification
```

```
## ME NH VT MA RI CT NY NJ PA OH IN IL MI WI MN IA MO ND SD NE KS DE MD DC VA
##  1  1  1  2  2  2  2  2  1  1  1  2  2  1  1  1  1  1  1  1  1  2  2  2  1
## WV NC SC GA FL KY TN AL MS AR LA OK TX MT ID WY CO NM AZ UT NV WA OR CA AK
##  1  1  2  1  2  1  1  1  1  1  2  2  2  1  1  1  2  2  2  1  2  2  2  2  2
## HI
##  1
```

La clasificación divide los datos en dos grupos aproximadamente por mitad.

Incluimos el dato de clasificación (grpm) en el data.frame original.

```r
dfclus = dfclus[,1:9]
dfclus = data.frame(dfclus,"grpm" = df.m$classification)
table(dfclus$grp,dfclus$grpk)
```

```
##
##          1  2  3  4  5  6  7  8
##   ME-MS  0 13  0  1  0  0  0  1
##   MA-OK  0  0  0  1  5  1  3  0
##   CT-HI  0  0  0  4  0  0  0  6
##   NY     0  0  0  0  1  0  0  0
```

```
##   MI-AZ  5  0  0  0  0  5  0  0
##   ND-WV  0  3  0  0  0  0  0  0
##   DC     0  0  1  0  0  0  0  0
##   FL     1  0  0  0  0  0  0  0
```

```r
table(dfclus$grp,dfclus$grpm)
```

```
##
##          1  2
##   ME-MS 15  0
##   MA-OK  2  8
##   CT-HI  8  2
##   NY     0  1
##   MI-AZ  0 10
##   ND-WV  3  0
##   DC     0  1
##   FL     0  1
```

```r
table(dfclus$grpm,dfclus$grpk)
```

```
##
##      1  2  3  4  5  6  7  8
##   1  0 16  0  5  2  0  0  5
##   2  6  0  1  1  4  6  3  2
```

```r
orderBy(~ grp + grpk + grpm, dfclus[,c(8,9,10)])
```

```
##      grp grpk grpm
## ME ME-MS    2    1
## NH ME-MS    2    1
## VT ME-MS    2    1
## PA ME-MS    2    1
## IN ME-MS    2    1
## WI ME-MS    2    1
## IA ME-MS    2    1
## NE ME-MS    2    1
## VA ME-MS    2    1
## KY ME-MS    2    1
## MT ME-MS    2    1
## ID ME-MS    2    1
## WY ME-MS    2    1
## MS ME-MS    4    1
## MN ME-MS    8    1
## SC MA-OK    4    2
## GA MA-OK    5    1
## TN MA-OK    5    1
## IL MA-OK    5    2
## MD MA-OK    5    2
## LA MA-OK    5    2
## OK MA-OK    6    2
## MA MA-OK    7    2
## RI MA-OK    7    2
## NJ MA-OK    7    2
## MO CT-HI    4    1
## NC CT-HI    4    1
## AL CT-HI    4    1
```

```
## AR CT-HI    4    1
## OH CT-HI    8    1
## KS CT-HI    8    1
## UT CT-HI    8    1
## HI CT-HI    8    1
## CT CT-HI    8    2
## DE CT-HI    8    2
## NY    NY    5    2
## MI MI-AZ    1    2
## TX MI-AZ    1    2
## NV MI-AZ    1    2
## CA MI-AZ    1    2
## AK MI-AZ    1    2
## CO MI-AZ    6    2
## NM MI-AZ    6    2
## AZ MI-AZ    6    2
## WA MI-AZ    6    2
## OR MI-AZ    6    2
## ND ND-WV    2    1
## SD ND-WV    2    1
## WV ND-WV    2    1
## DC    DC    3    2
## FL    FL    1    2
```

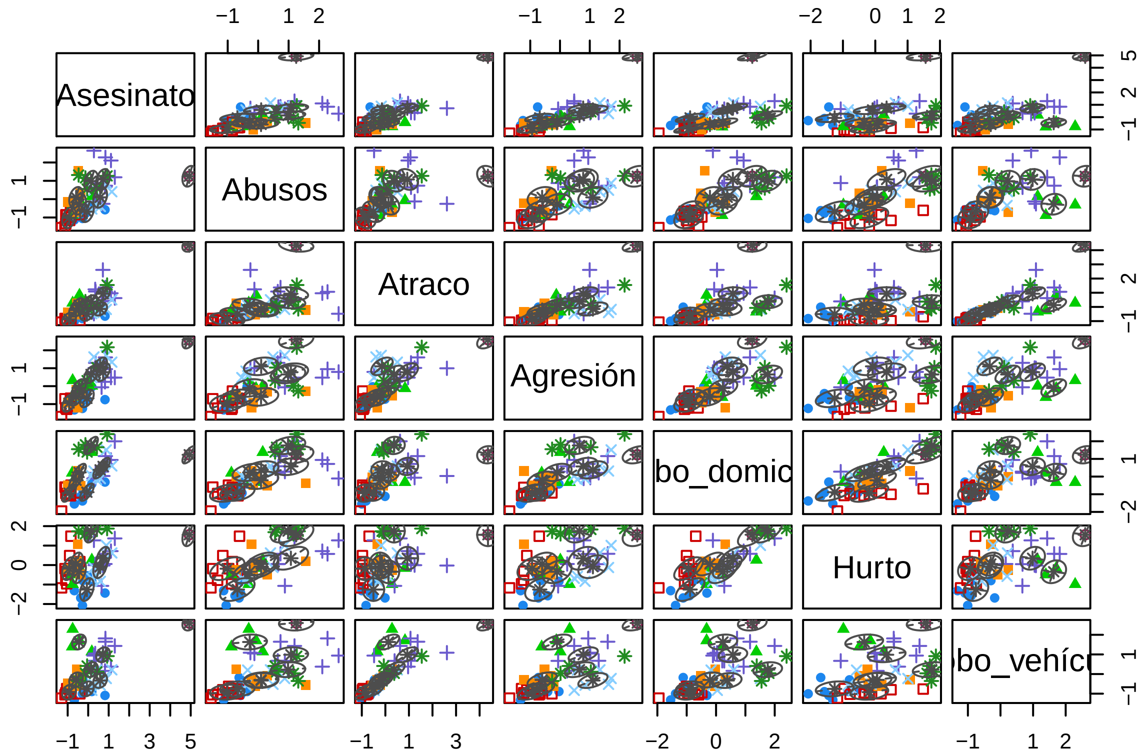
Forzamos a un modelo con ocho componentes y comparamos con los resultados anteriores.

```r
df.m8 = Mclust(zdf, G = 8)
df.m8
```

```
## 'Mclust' model object:
##  best model: ellipsoidal, equal volume, shape and orientation (EEE) with 8 components
```

El resultado es un modelo EEE, esto, elipsoidal con igual varianza, forma y orientación.

```r
plot(df.m8,what = 'classification')
```



Igualmente incorporamos el dato de clasificación (grpm8) al data.frame original y comporamos con los resultados anteriores.

```
dfclus = dfclus[,1:10]
dfclus = data.frame(dfclus,"grpm8" = df.m8$classification)
table(dfclus$grp,dfclus$grpk)
```

```
##
##          1  2  3  4  5  6  7  8
##   ME-MS  0 13  0  1  0  0  0  1
##   MA-OK  0  0  0  1  5  1  3  0
##   CT-HI  0  0  0  4  0  0  0  6
##   NY     0  0  0  0  1  0  0  0
##   MI-AZ  5  0  0  0  0  5  0  0
##   ND-WV  0  3  0  0  0  0  0  0
##   DC     0  0  1  0  0  0  0  0
##   FL     1  0  0  0  0  0  0  0
```

```
table(dfclus$grp,dfclus$grpm8)
```

```
##
##         1 2 3 4 5 6 7 8
##   ME-MS 5 9 0 0 1 0 0 0
##   MA-OK 0 0 4 4 0 2 0 0
##   CT-HI 1 1 0 0 5 3 0 0
##   NY    0 0 0 1 0 0 0 0
##   MI-AZ 0 0 0 5 0 1 0 4
##   ND-WV 2 1 0 0 0 0 0 0
```

```
##   DC    0 0 0 0 0 0 1 0
##   FL    0 0 0 0 0 0 0 1
```

```r
table(dfclus$grpm8,dfclus$grpk)
```

```
##
##      1  2  3  4  5  6  7  8
##   1  0  6  0  2  0  0  0  0
##   2  0 10  0  0  0  0  0  1
##   3  0  0  0  0  0  1  3  0
##   4  5  0  0  0  5  0  0  0
##   5  0  0  0  0  0  0  0  6
##   6  0  0  0  4  1  1  0  0
##   7  0  0  1  0  0  0  0  0
##   8  1  0  0  0  0  4  0  0
```

```r
orderBy(~ grp + grpk + grpm + grpm8, dfclus[,c(8,9,10,11)])
```

```
##      grp grpk grpm grpm8
## PA ME-MS    2    1     1
## IN ME-MS    2    1     1
## VA ME-MS    2    1     1
## KY ME-MS    2    1     1
## ME ME-MS    2    1     2
## NH ME-MS    2    1     2
## VT ME-MS    2    1     2
## WI ME-MS    2    1     2
## IA ME-MS    2    1     2
## NE ME-MS    2    1     2
## MT ME-MS    2    1     2
## ID ME-MS    2    1     2
## WY ME-MS    2    1     2
## MS ME-MS    4    1     1
## MN ME-MS    8    1     5
## SC MA-OK    4    2     6
## GA MA-OK    5    1     4
## TN MA-OK    5    1     4
## IL MA-OK    5    2     4
## MD MA-OK    5    2     4
## LA MA-OK    5    2     6
## OK MA-OK    6    2     3
## MA MA-OK    7    2     3
## RI MA-OK    7    2     3
## NJ MA-OK    7    2     3
## AR CT-HI    4    1     1
## MO CT-HI    4    1     6
## NC CT-HI    4    1     6
## AL CT-HI    4    1     6
## UT CT-HI    8    1     2
## OH CT-HI    8    1     5
## KS CT-HI    8    1     5
## HI CT-HI    8    1     5
## CT CT-HI    8    2     5
## DE CT-HI    8    2     5
## NY    NY    5    2     4
```

```
## MI MI-AZ    1    2     4
## TX MI-AZ    1    2     4
## NV MI-AZ    1    2     4
## CA MI-AZ    1    2     4
## AK MI-AZ    1    2     4
## NM MI-AZ    6    2     6
## CO MI-AZ    6    2     8
## AZ MI-AZ    6    2     8
## WA MI-AZ    6    2     8
## OR MI-AZ    6    2     8
## SD ND-WV    2    1     1
## WV ND-WV    2    1     1
## ND ND-WV    2    1     2
## DC    DC    3    2     7
## FL    FL    1    2     8
```

2 Análisis de Componentes Principales

Acceder a los datos gironde la librería PCAmixdata. En los siguientes apartados seleccionar los registros completos si hay valores perdidos.

i) Realizar e interpretar un análisis de componentes principales (matriz de correlaciones) para 'gironde$employment'.

ii) Realizar e interpretar un análisis de componentes principales para datos mixtos sobre la unión de 'gironde$employment' y 'gironde$services'.

iii) Aplicar procedimientos de selección de variables para construir modelos de regresión lineal donde income es la variable dependiente, sobre 'gironde$employment'.

2.1 Lectura de datos

```r
library(PCAmixdata)
data(gironde)
```

2.2 Análisis de datos de Empleo

2.2.1 Resumen de datos

```r
summary(gironde$employment)
```

```
##     farmers          tradesmen         managers          workers
##  Min.   : 0.0000   Min.   : 0.000   Min.   : 0.000   Min.   : 0.00
##  1st Qu.: 0.5125   1st Qu.: 2.772   1st Qu.: 2.795   1st Qu.:28.57
##  Median : 1.9700   Median : 3.995   Median : 4.650   Median :33.66
##  Mean   : 3.4650   Mean   : 4.189   Mean   : 5.287   Mean   :33.52
##  3rd Qu.: 4.6875   3rd Qu.: 5.300   3rd Qu.: 7.147   3rd Qu.:38.40
##  Max.   :33.3300   Max.   :16.130   Max.   :22.730   Max.   :57.14
##
##    unemployed      middleempl        retired        employrate
##  Min.   : 0.00   Min.   : 0.000   Min.   : 9.33   Min.   : 75.08
```

```
##  1st Qu.:11.22   1st Qu.: 8.523   1st Qu.:23.25   1st Qu.: 88.35
##  Median :13.55   Median :11.875   Median :27.45   Median : 90.66
##  Mean   :13.38   Mean   :11.993   Mean   :28.17   Mean   : 90.30
##  3rd Qu.:15.59   3rd Qu.:15.440   3rd Qu.:32.14   3rd Qu.: 92.71
##  Max.   :33.33   Max.   :31.580   Max.   :51.28   Max.   :100.00
##
##      income
##  Min.   :12187
##  1st Qu.:18367
##  Median :19990
##  Mean   :21003
##  3rd Qu.:22768
##  Max.   :70062
##  NA's   :2
```

```
cat(' Total de casos: \t',nrow(gironde$employment),
    '\n Casos completos: \t',nrow(na.omit(gironde$employment)))
```

```
##  Total de casos:      542
##  Casos completos:     540
```

2.2.2 Matriz de correlación incluyendo sólo los casos completos.

```
emR = cor(gironde$employment,use="complete.obs")
library(knitr)
kable(round(emR,2), caption="Matriz de Correlación (emR)")
```

Table 3: Matriz de Correlación (emR)

	farmers	tradesmen	managers	workers	unemployed	middleempl	retired	employrate	income
farmers	1.00	-0.06	-0.18	-0.16	-0.23	-0.33	0.04	0.19	-0.06
tradesmen	-0.06	1.00	-0.06	-0.14	-0.10	-0.10	0.00	0.03	0.05
managers	-0.18	-0.06	1.00	-0.36	0.09	0.31	-0.26	0.33	0.48
workers	-0.16	-0.14	-0.36	1.00	-0.19	-0.19	-0.48	-0.07	-0.24
unemployed	-0.23	-0.10	0.09	-0.19	1.00	-0.03	-0.22	-0.18	-0.01
middleempl	-0.33	-0.10	0.31	-0.19	-0.03	1.00	-0.40	0.24	0.38
retired	0.04	0.00	-0.26	-0.48	-0.22	-0.40	1.00	-0.27	-0.23
employrate	0.19	0.03	0.33	-0.07	-0.18	0.24	-0.27	1.00	0.41
income	-0.06	0.05	0.48	-0.24	-0.01	0.38	-0.23	0.41	1.00

2.2.3 Representación gráfica de la Matriz de Correlación.

```
library(corrplot)
corrplot(emR, method="ellipse", addCoef.col='black', number.cex=0.7,
         tl.cex = 0.8,tl.col = 'black')
```

2.2.4 ACP

```r
emACP = princomp(emR, cor = TRUE)
```

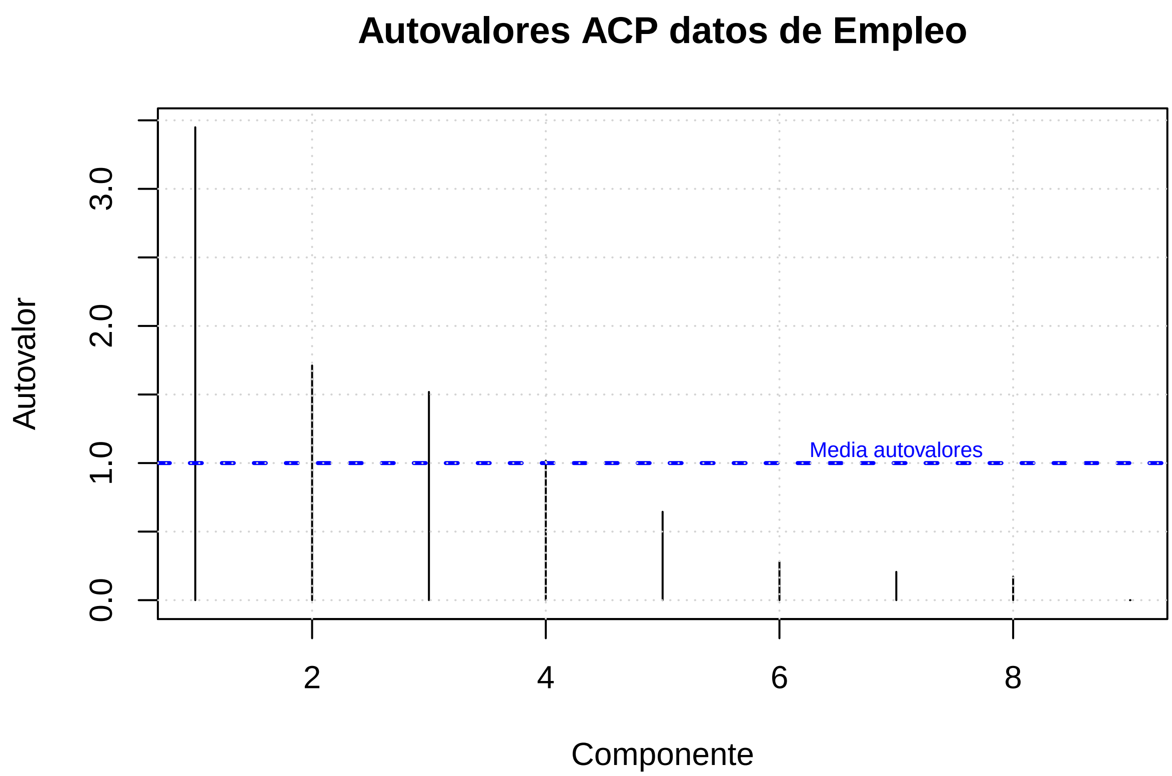
```r
emACPresumen= matrix(NA,nrow=length(emACP$sdev),ncol=3)
emACPresumen[,1]=  emACP$sdev^2
emACPresumen[,2]= 100*emACPresumen[,1]/sum(emACPresumen[,1])
emACPresumen[,3]= cumsum(emACPresumen[,2])
colnames(emACPresumen)= c("Autovalor","Porcentaje","Porcentaje acumulado")
rownames(emACPresumen)= c(1:nrow(emACPresumen))
kable(emACPresumen,caption = "Resumen ACP Empleo",row.names = TRUE)
```

Table 4: Resumen ACP Empleo

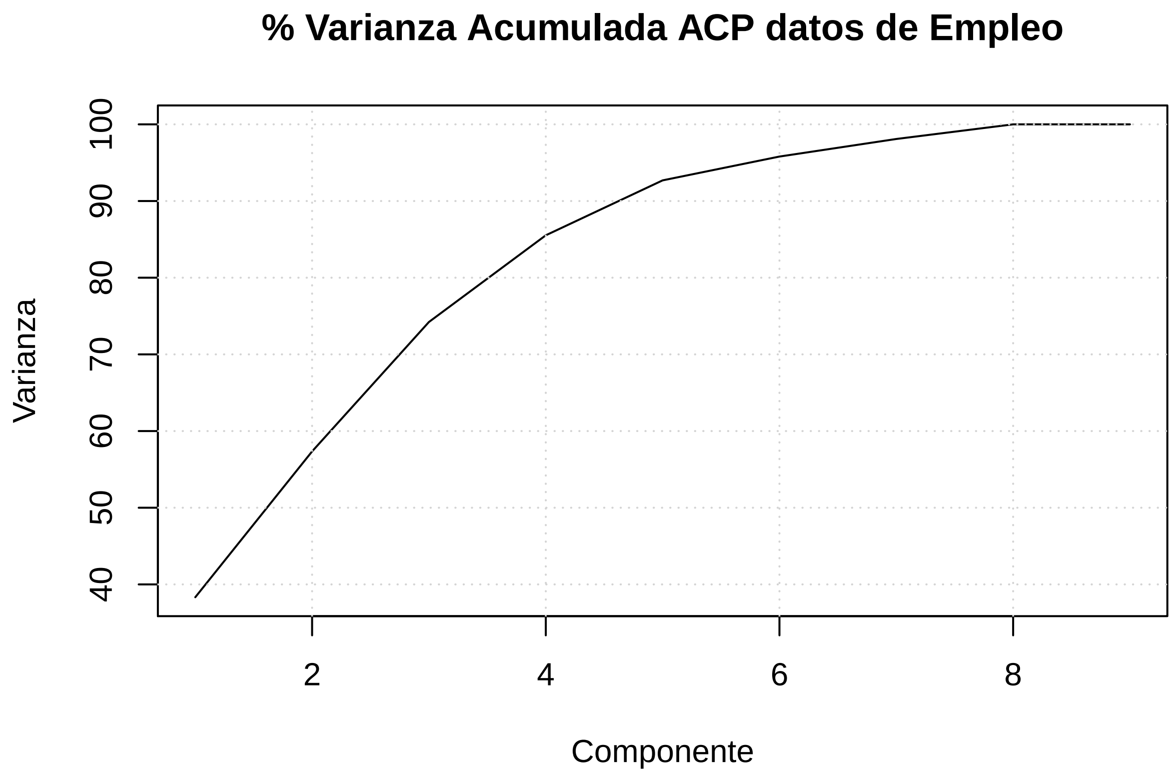
	Autovalor	Porcentaje	Porcentaje acumulado
1	3.4493254	38.325838	38.32584
2	1.7117959	19.019955	57.34579
3	1.5189578	16.877309	74.22310
4	1.0176541	11.307267	85.53037
5	0.6448197	7.164663	92.69503
6	0.2794878	3.105421	95.80045
7	0.2062433	2.291592	98.09204
8	0.1717160	1.907956	100.00000
9	0.0000000	0.000000	100.00000

Las dos primeras componentes de recogen aproximadamente el 57% de la varianza total de los datos, con las tres primeras se supera el 74% y con la cuarta se supera el 85%.

```
plot(emACPresumen[,1],type="h",
     main="Autovalores ACP datos de Empleo",
     xlab='Componente', ylab="Autovalor")
abline(h=mean(emACPresumen[,1]),lwd=2,lty=2,col="blue")
text(x=7,y=1.10*mean(emACPresumen[,1]),
     labels = "Media autovalores",col="blue",cex = 0.7)
grid()
```




```
plot(emACPresumen[,3],type="l",
     main="% Varianza Acumulada ACP datos de Empleo",
     xlab='Componente', ylab="Varianza")
grid()
```



2.2.4.1 Coeficientes de las CP:

```r
loadings(emACP)
```

```
##
## Loadings:
##            Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8 Comp.9
## farmers     0.212 -0.466 -0.318 -0.355  0.469 -0.314  0.280  0.173 -0.288
## tradesmen         -0.209  0.133  0.880  0.359                      -0.148
## managers   -0.477 -0.125  0.173                0.483         0.643 -0.265
## workers     0.123  0.491 -0.568  0.136         0.277  0.106        -0.553
## unemployed         0.417  0.494 -0.240  0.594        -0.150 -0.256 -0.270
## middleempl -0.472                      -0.322 -0.712               -0.374
## retired     0.336 -0.399  0.368        -0.397  0.176 -0.154 -0.262 -0.553
## employrate -0.365 -0.317 -0.387         0.184  0.139 -0.687 -0.303
## income     -0.477 -0.203                       0.171  0.622 -0.559
##
##                Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
## SS loadings     1.000  1.000  1.000  1.000  1.000  1.000  1.000  1.000
## Proportion Var  0.111  0.111  0.111  0.111  0.111  0.111  0.111  0.111
## Cumulative Var  0.111  0.222  0.333  0.444  0.556  0.667  0.778  0.889
##                Comp.9
## SS loadings     1.000
## Proportion Var  0.111
## Cumulative Var  1.000
```

Los coeficientes más altos, en valor absoluto, de la primera componente son los correspondientes a las variables 'manager' e 'income' (-0,477), le siguen 'middelempl' (-0,472) y 'employrate' (-0,365), y todos ellos con signo negativo.

Para la segunda componente principal los coeficientes más altos en valor absoluto son para las variables 'workers' (0,491), 'farmers' (-0.466) y 'unemployed' (0,417) con signo positivo la primera y la última y negativo la segunda.

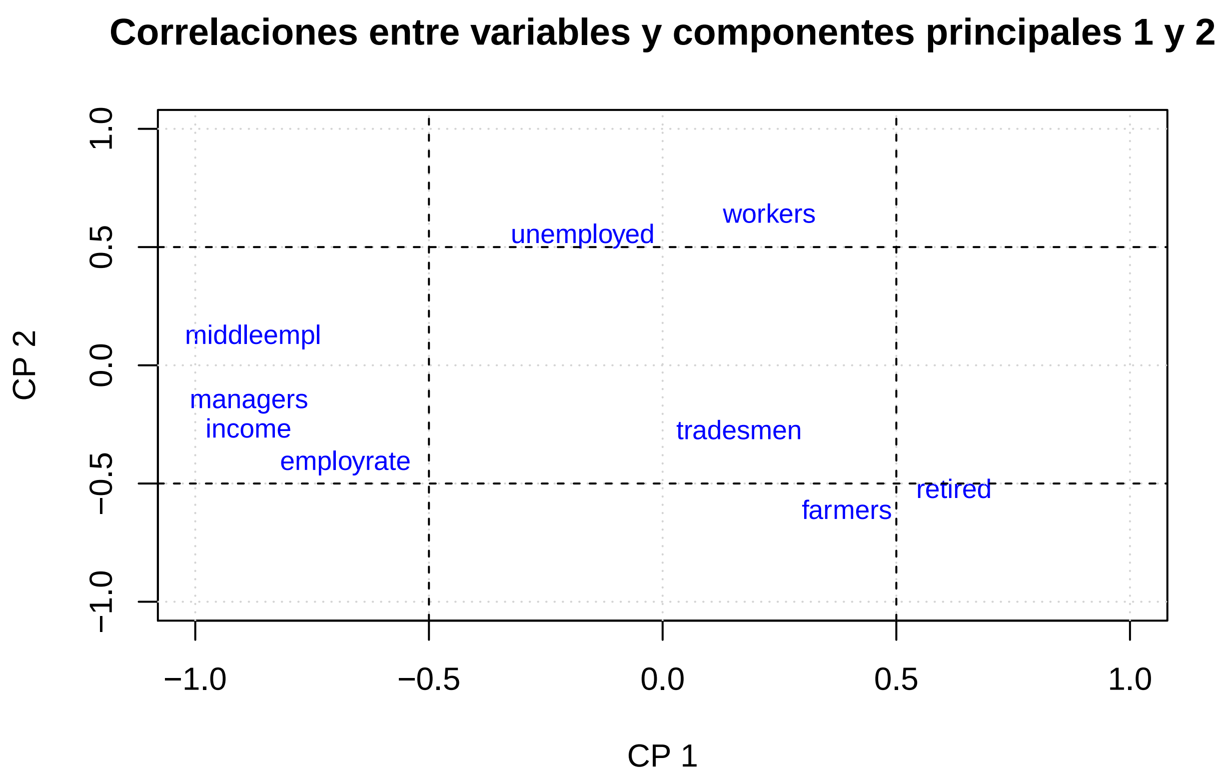
2.2.4.2 Correlaciones entre Variables y CP:

```
emCorVCP = loadings(emACP)%*%diag(emACP$sdev)
kable(round(emCorVCP,2), col.names = c(1:ncol(emCorVCP)),
      caption = 'Correlaciones entre Variables y CP')
```

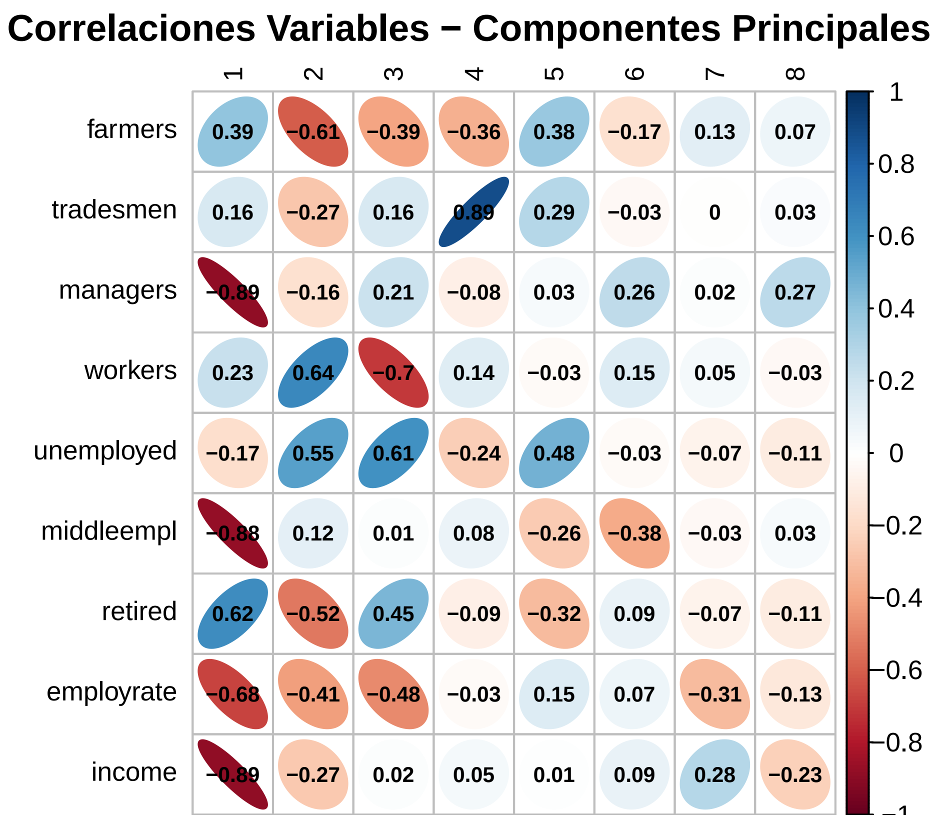
Table 5: Correlaciones entre Variables y CP

	1	2	3	4	5	6	7	8	9
farmers	0.39	-0.61	-0.39	-0.36	0.38	-0.17	0.13	0.07	0
tradesmen	0.16	-0.27	0.16	0.89	0.29	-0.03	0.00	0.03	0
managers	-0.89	-0.16	0.21	-0.08	0.03	0.26	0.02	0.27	0
workers	0.23	0.64	-0.70	0.14	-0.03	0.15	0.05	-0.03	0
unemployed	-0.17	0.55	0.61	-0.24	0.48	-0.03	-0.07	-0.11	0
middleempl	-0.88	0.12	0.01	0.08	-0.26	-0.38	-0.03	0.03	0
retired	0.62	-0.52	0.45	-0.09	-0.32	0.09	-0.07	-0.11	0
employrate	-0.68	-0.41	-0.48	-0.03	0.15	0.07	-0.31	-0.13	0
income	-0.89	-0.27	0.02	0.05	0.01	0.09	0.28	-0.23	0

```
plot(emCorVCP[,1:2],
      main="Correlaciones entre variables y componentes principales 1 y 2",
     xlab="CP 1", ylab="CP 2",type="n",xlim=c(-1,1),ylim=c(-1,1))
text(emCorVCP[,1:2],labels=rownames(emCorVCP),col="blue",cex = 0.8)
grid()
abline(v=0.5,h=-0.5,lty=2)
abline(v=-0.5,h=0.5,lty=2)
```




```
library(corrplot)
corrplot(emCorVCP[,1:8], method="ellipse", addCoef.col='black',
         number.cex=0.7, tl.cex = 0.8,tl.col = 'black',
         title = 'Correlaciones Variables - Componentes Principales',
         mar=c(1,1,2,1))
```



2.2.4.3 Puntuaciones

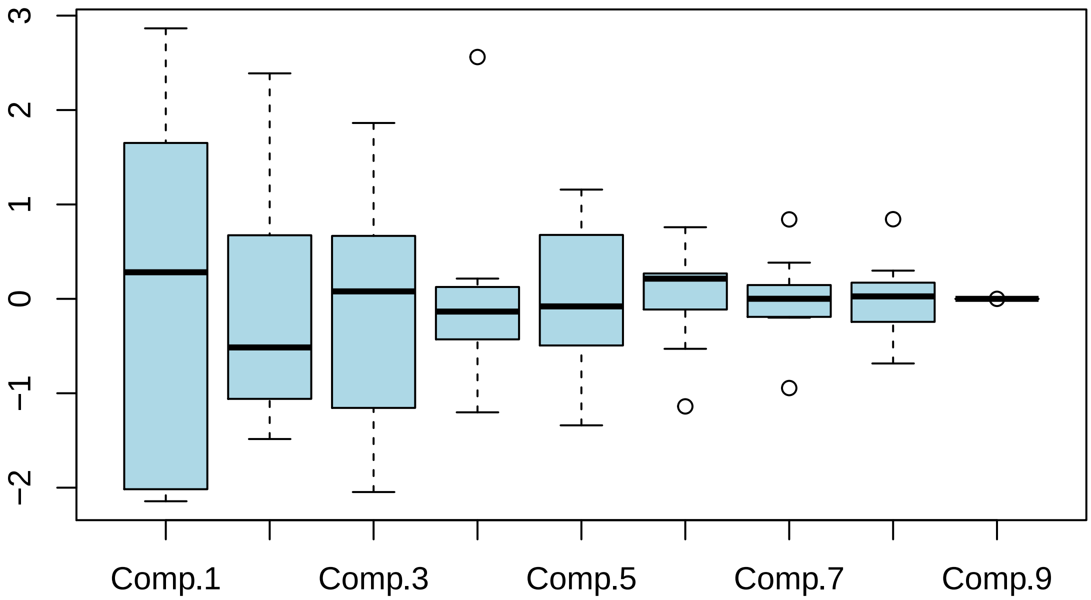
```r
kable(round(emACP$scores,2),
      caption = "Puntuaciones datos Empleo")
```

Table 6: Puntuaciones datos Empleo

	Comp.1	Comp.2	Comp.3	Comp.4	Comp.5	Comp.6	Comp.7	Comp.8	Comp.9
farmers	1.86	-1.48	-1.16	-1.20	0.87	-0.53	0.38	0.30	0
tradesmen	1.10	-0.51	0.51	2.56	0.68	-0.11	0.00	0.14	0
managers	-2.13	-0.24	0.67	-0.36	-0.08	0.76	0.05	0.84	0
workers	1.65	2.39	-2.05	0.21	-0.49	0.41	0.15	0.03	0
unemployed	0.28	1.99	1.86	-0.87	1.16	-0.10	-0.20	-0.24	0
middleempl	-2.02	0.67	0.08	0.13	-1.04	-1.14	-0.09	0.17	0
retired	2.87	-1.12	1.42	-0.43	-1.34	0.24	-0.19	-0.19	0
employrate	-1.47	-1.06	-1.39	-0.13	0.33	0.21	-0.94	-0.36	0
income	-2.14	-0.62	0.05	0.09	-0.08	0.27	0.84	-0.68	0

2.2.4.4 Variabilidad de puntuaciones en cada componente

```r
boxplot(emACP$scores,col="lightblue",notched=TRUE)
```



2.2.4.5 Correlaciones estimadas con k C.P. y sus residuales

```
emDSp=eigen(emR)
emAutoVal=emDSp$values
emAutoVec=emDSp$vectors
#emAutoVec%*%diag(emAutoVal)%*%t(emAutoVec)
#emR

emRe2CP = emAutoVec[,1:2]%*%diag(emAutoVal[1:2])%*%t(emAutoVec[,1:2])
emRr2CP = emR - emRe2CP

emRe3CP = emAutoVec[,1:3]%*%diag(emAutoVal[1:3])%*%t(emAutoVec[,1:3])
emRr3CP = emR - emRe3CP

emRe4CP = emAutoVec[,1:4]%*%diag(emAutoVal[1:4])%*%t(emAutoVec[,1:4])
emRr4CP = emR - emRe4CP

kable(round(emRr2CP,2),
      caption = 'Correlación residual con 2 Componentes')
```

Table 7: Correlación residual con 2 Componentes

	farmers	tradesmen	managers	workers	unemployed	middleempl	retired	employrate	income
farmers	0.77	-0.16	-0.09	0.14	-0.11	-0.10	-0.36	0.24	0.02
tradesmen	-0.16	0.94	-0.09	0.05	-0.04	-0.04	-0.17	0.00	0.01

	farmers	tradesmen	managers	workers	unemployed	middleempl	retired	employrate	income
managers	-0.09	-0.09	0.40	-0.01	0.05	-0.20	0.01	-0.15	-0.13
workers	0.14	0.05	-0.01	0.25	-0.35	-0.15	-0.04	0.22	0.13
unemployed	-0.11	-0.04	0.05	-0.35	0.94	-0.14	-0.01	-0.20	-0.05
middleempl	-0.10	-0.04	-0.20	-0.15	-0.14	0.47	0.09	-0.15	-0.12
retired	-0.36	-0.17	0.01	-0.04	-0.01	0.09	0.26	-0.08	0.03
employrate	0.24	0.00	-0.15	0.22	-0.20	-0.15	-0.08	0.62	-0.08
income	0.02	0.01	-0.13	0.13	-0.05	-0.12	0.03	-0.08	0.38

```
kable(round(emRr3CP,2),
      caption = 'Correlación residual con 3 Componentes')
```

Table 8: Correlación residual con 3 Componentes

	farmers	tradesmen	managers	workers	unemployed	middleempl	retired	employrate	income
farmers	0.42	-0.18	0.01	-0.11	0.30	-0.03	-0.22	-0.09	-0.03
tradesmen	-0.18	0.94	-0.09	0.04	-0.02	-0.04	-0.16	-0.02	0.01
managers	0.01	-0.09	0.37	0.06	-0.06	-0.21	-0.03	-0.06	-0.12
workers	-0.11	0.04	0.06	0.07	-0.06	-0.11	0.06	-0.01	0.09
unemployed	0.30	-0.02	-0.06	-0.06	0.48	-0.22	-0.18	0.17	0.01
middleempl	-0.03	-0.04	-0.21	-0.11	-0.22	0.45	0.06	-0.09	-0.12
retired	-0.22	-0.16	-0.03	0.06	-0.18	0.06	0.20	0.06	0.05
employrate	-0.09	-0.02	-0.06	-0.01	0.17	-0.09	0.06	0.32	-0.12
income	-0.03	0.01	-0.12	0.09	0.01	-0.12	0.05	-0.12	0.37

```
kable(round(emRr4CP,2),
      caption = 'Correlación residual con 4 Componentes')
```

Table 9: Correlación residual con 4 Componentes

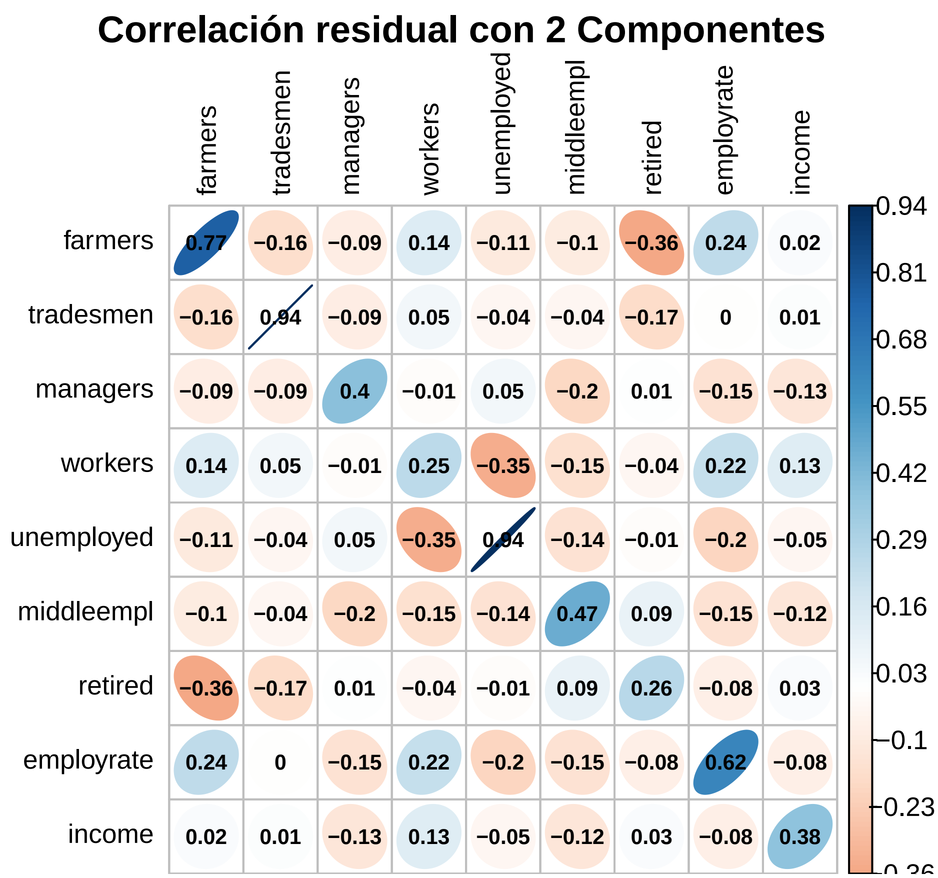
	farmers	tradesmen	managers	workers	unemployed	middleempl	retired	employrate	income
farmers	0.24	0.18	-0.04	-0.08	0.16	0.02	-0.21	-0.11	0.00
tradesmen	0.18	0.21	0.01	-0.04	0.25	-0.13	-0.19	0.03	-0.03
managers	-0.04	0.01	0.35	0.07	-0.10	-0.20	-0.03	-0.06	-0.11
workers	-0.08	-0.04	0.07	0.07	-0.04	-0.12	0.06	0.00	0.09
unemployed	0.16	0.25	-0.10	-0.04	0.38	-0.19	-0.17	0.15	0.03
middleempl	0.02	-0.13	-0.20	-0.12	-0.19	0.44	0.06	-0.08	-0.12
retired	-0.21	-0.19	-0.03	0.06	-0.17	0.06	0.20	0.06	0.05
employrate	-0.11	0.03	-0.06	0.00	0.15	-0.08	0.06	0.31	-0.12
income	0.00	-0.03	-0.11	0.09	0.03	-0.12	0.05	-0.12	0.37

```
kable(round (data.frame('Dos CP'=mean(emRr2CP^2),
                 'Tres CP'=mean(emRr3CP^2),
                 'Cuatro CP'=mean(emRr4CP^2)),4),
      caption = 'Correlación residual cuadrática media según número de componentes')
```

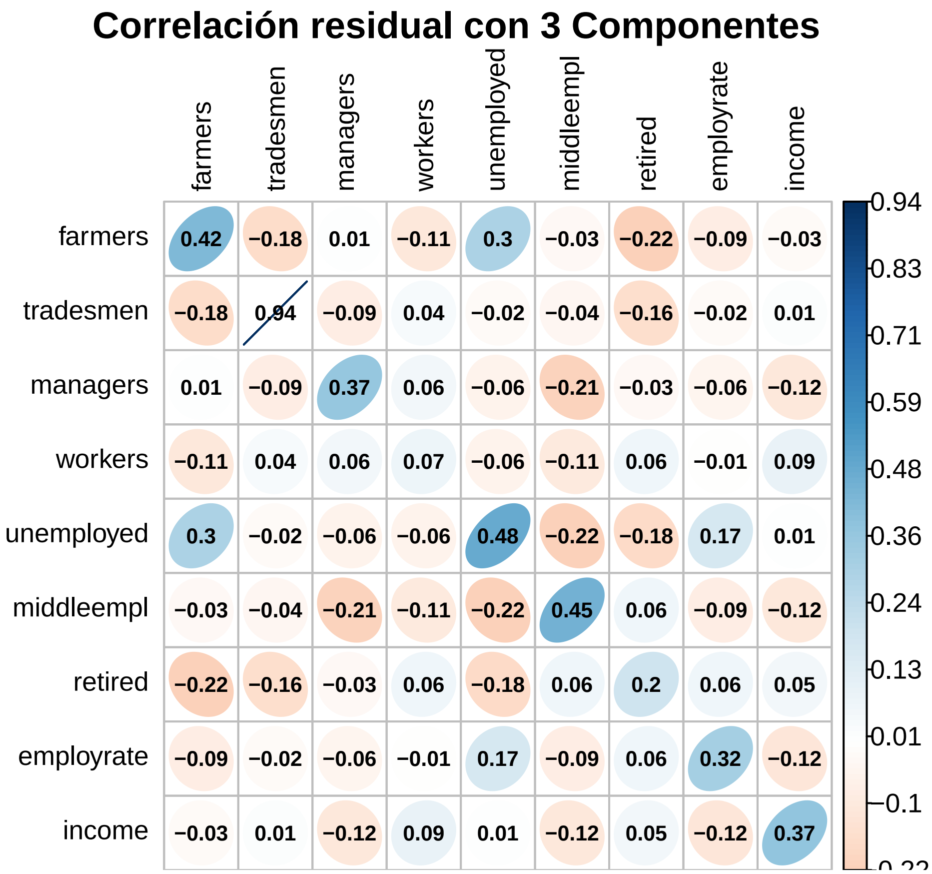
Table 10: Correlación residual cuadrática media según número de componentes

Dos.CP	Tres.CP	Cuatro.CP
0.0598	0.0355	0.022

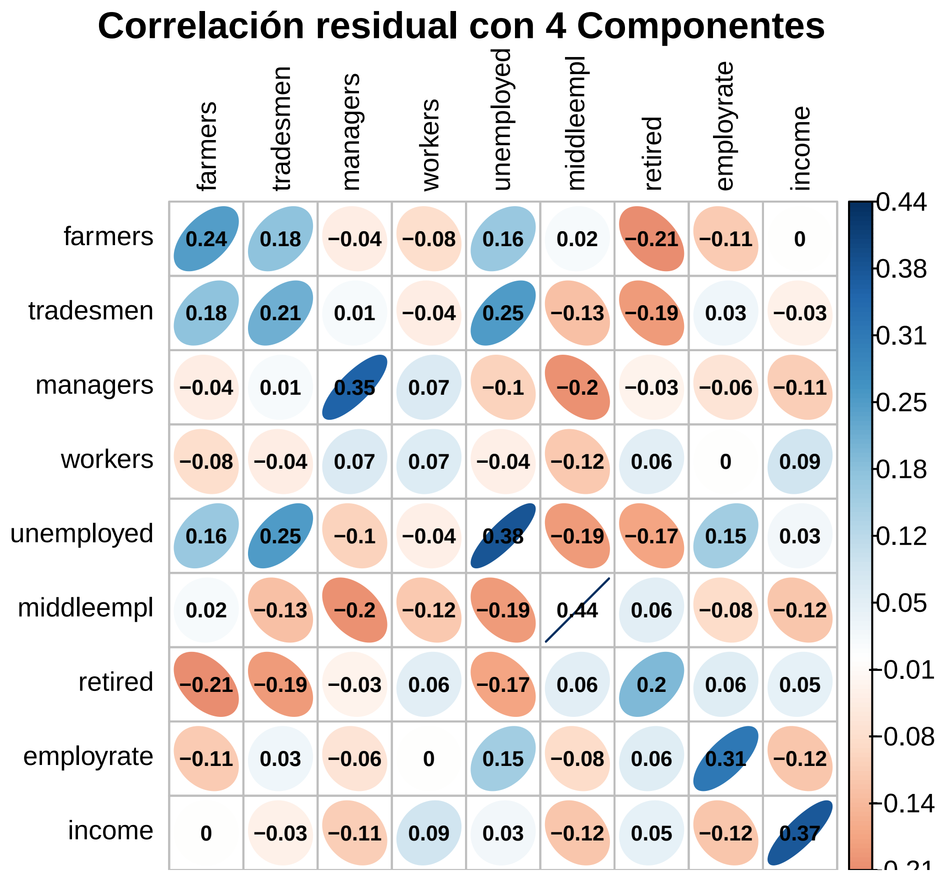
```r
library(corrplot)
corrplot(emRr2CP, method="ellipse", addCoef.col='black',
         number.cex=0.7, tl.cex = 0.8,tl.col = 'black',
         title = 'Correlación residual con 2 Componentes',
         is.corr = FALSE,mar = c(0, 0, 1, 0))
```




```r
corrplot(emRr3CP, method="ellipse", addCoef.col='black',
         number.cex=0.7, tl.cex = 0.8,tl.col = 'black',
         title = 'Correlación residual con 3 Componentes',
         is.corr = FALSE,mar = c(0, 0, 1, 0))
```




```r
corrplot(emRr4CP, method="ellipse", addCoef.col='black',
         number.cex=0.7, tl.cex = 0.8,tl.col = 'black',
         title = 'Correlación residual con 4 Componentes',
         is.corr = FALSE,mar = c(0, 0, 1, 0))
```



2.3 Análisis de datos mixtos; empleo y servicios

2.3.1 Resumen de datos

```
str(gironde$employment)
```

```
## 'data.frame':    542 obs. of  9 variables:
##  $ farmers    : num  1.98 5.23 0.1 0.18 0.3 ...
##  $ tradesmen : num  3.68 5.23 4.38 2.29 3.8 5.63 4.21 1.75 4.61 2.3 ...
##  $ managers  : num  3.97 1.96 5.56 3.7 8.19 1.25 4.21 3.51 5.8 0 ...
##  $ workers   : num  38.2 21.6 36 42.4 18.6 ...
##  $ unemployed: num  13.6 15 18.2 15.1 13 ...
##  $ middleempl: num  9.63 14.38 15.48 8.98 12.07 ...
##  $ retired   : num  28.9 36.6 20.3 27.3 44 ...
##  $ employrate: num  89.3 90.9 90.2 87.4 89.4 ...
##  $ income    : num  17671 19422 21047 18015 27147 ...
```

```
str(gironde$services)
```

```
## 'data.frame':    542 obs. of  9 variables:
##  $ butcher   : Factor w/ 3 levels "0","1","2 or +": 1 1 2 1 3 1 1 1 3 1 ...
##  $ baker     : Factor w/ 3 levels "0","1","2 or +": 3 1 3 2 3 2 1 1 3 2 ...
##  $ postoffice: Factor w/ 2 levels "0","1 or +": 2 1 2 2 2 1 1 1 2 1 ...
##  $ dentist   : Factor w/ 3 levels "0","1 to 2","3 or +": 1 1 3 2 3 1 1 1 3 1 ...
##  $ grocery   : Factor w/ 2 levels "0","1 or +": 1 2 2 2 2 2 2 2 2 1 ...
```

```
##  $ nursery   : Factor w/ 2 levels "0","1 or +": 1 1 2 1 1 1 1 1 2 1 ...
##  $ doctor    : Factor w/ 3 levels "0","1 to 2","3 or +": 1 3 3 3 3 1 1 1 3 1 ...
##  $ chemist   : Factor w/ 3 levels "0","1","2 or +": 2 1 3 2 3 1 1 1 3 1 ...
##  $ restaurant: Factor w/ 4 levels "0","1","2","3 or +": 2 2 4 4 4 3 3 1 4 3 ...
```

```r
summary(gironde$services)
```

```
##    butcher       baker      postoffice    dentist      grocery
##  0     :371   0     :291   0     :346   0     :380   0     :365
##  1     : 95   1     :128   1 or +:196   1 to 2: 90   1 or +:177
##  2 or +: 76   2 or +:123                3 or +: 72
##
##    nursery       doctor      chemist     restaurant
##  0     :520   0     :326   0     :357   0     :247
##  1 or +: 22   1 to 2: 92   1     :107   1     :122
##               3 or +:124   2 or +: 78   2     : 52
##                                         3 or +:121
```

```r
cat(' Total de casos: \t',nrow(gironde$services),
    '\n Casos completos: \t',nrow(na.omit(gironde$services)))
```

```
##  Total de casos:      542
##  Casos completos:     542
```

Los datos de empleo ya han sido analizados previamente. Los datos de servicios corresponden en todos los casos a variables cualitativas codificadas como factores con entre 2 y 4 niveles y sin datos faltantes.

2.3.2 Análisis con PCAmix

Como paso previo normalizamos los datos de empleo y descartamos los casos con datos faltantes.

```r
zem = as.data.frame(scale(gironde$employment))
#zem
es.df = data.frame(zem,gironde$services)
es.df = na.omit(es.df)
nrow(es.df)
```

```
## [1] 540
```

```r
str(es.df)
```

```
## 'data.frame':    540 obs. of  18 variables:
##  $ farmers   : num  -0.328 0.39 -0.743 -0.725 -0.698 ...
##  $ tradesmen : num  -0.2135 0.4372 0.0804 -0.797 -0.1631 ...
##  $ managers  : num  -0.3731 -0.9426 0.0775 -0.4496 0.8227 ...
##  $ workers   : num  0.618 -1.563 0.322 1.164 -1.945 ...
##  $ unemployed: num  0.054 0.4042 1.188 0.4238 -0.0832 ...
##  $ middleempl: num  -0.4837 0.4886 0.7138 -0.6168 0.0157 ...
##  $ retired   : num  0.103 1.183 -1.106 -0.117 2.217 ...
##  $ employrate: num  -0.3064 0.1705 -0.0149 -0.8598 -0.2563 ...
##  $ income    : num  -0.73067 -0.34651 0.00974 -0.65525 1.34746 ...
##  $ butcher   : Factor w/ 3 levels "0","1","2 or +": 1 1 2 1 3 1 1 1 3 1 ...
##  $ baker     : Factor w/ 3 levels "0","1","2 or +": 3 1 3 2 3 2 1 1 3 2 ...
##  $ postoffice: Factor w/ 2 levels "0","1 or +": 2 1 2 2 2 1 1 1 2 1 ...
##  $ dentist   : Factor w/ 3 levels "0","1 to 2","3 or +": 1 1 3 2 3 1 1 1 3 1 ...
##  $ grocery   : Factor w/ 2 levels "0","1 or +": 1 2 2 2 2 2 2 2 2 1 ...
##  $ nursery   : Factor w/ 2 levels "0","1 or +": 1 1 2 1 1 1 1 1 2 1 ...
```

```
##  $ doctor    : Factor w/ 3 levels "0","1 to 2","3 or +": 1 3 3 3 3 1 1 1 3 1 ...
##  $ chemist   : Factor w/ 3 levels "0","1","2 or +": 2 1 3 2 3 1 1 1 3 1 ...
##  $ restaurant: Factor w/ 4 levels "0","1","2","3 or +": 2 2 4 4 4 3 3 1 4 3 ...
##  - attr(*, "na.action")=Class 'omit'  Named int [1:2] 63 369
##   .. ..- attr(*, "names")= chr [1:2] "BOSSUGAN" "SAINT-AVIT-DE-SOULEGE"
```

```
es.df = splitmix(es.df)
str(es.df)
```

```
## List of 3
##  $ X.quanti :'data.frame':   540 obs. of  9 variables:
##   ..$ farmers   : num [1:540] -0.328 0.39 -0.743 -0.725 -0.698 ...
##   ..$ tradesmen : num [1:540] -0.2135 0.4372 0.0804 -0.797 -0.1631 ...
##   ..$ managers  : num [1:540] -0.3731 -0.9426 0.0775 -0.4496 0.8227 ...
##   ..$ workers   : num [1:540] 0.618 -1.563 0.322 1.164 -1.945 ...
##   ..$ unemployed: num [1:540] 0.054 0.4042 1.188 0.4238 -0.0832 ...
##   ..$ middleempl: num [1:540] -0.4837 0.4886 0.7138 -0.6168 0.0157 ...
##   ..$ retired   : num [1:540] 0.103 1.183 -1.106 -0.117 2.217 ...
##   ..$ employrate: num [1:540] -0.3064 0.1705 -0.0149 -0.8598 -0.2563 ...
##   ..$ income    : num [1:540] -0.73067 -0.34651 0.00974 -0.65525 1.34746 ...
##  $ X.quali  :'data.frame':   540 obs. of  9 variables:
##   ..$ butcher   : Factor w/ 3 levels "0","1","2 or +": 1 1 2 1 3 1 1 1 3 1 ...
##   ..$ baker     : Factor w/ 3 levels "0","1","2 or +": 3 1 3 2 3 2 1 1 3 2 ...
##   ..$ postoffice: Factor w/ 2 levels "0","1 or +": 2 1 2 2 2 1 1 1 2 1 ...
##   ..$ dentist   : Factor w/ 3 levels "0","1 to 2","3 or +": 1 1 3 2 3 1 1 1 3 1 ...
##   ..$ grocery   : Factor w/ 2 levels "0","1 or +": 1 2 2 2 2 2 2 2 2 1 ...
##   ..$ nursery   : Factor w/ 2 levels "0","1 or +": 1 1 2 1 1 1 1 1 2 1 ...
##   ..$ doctor    : Factor w/ 3 levels "0","1 to 2","3 or +": 1 3 3 3 3 1 1 1 3 1 ...
##   ..$ chemist   : Factor w/ 3 levels "0","1","2 or +": 2 1 3 2 3 1 1 1 3 1 ...
##   ..$ restaurant: Factor w/ 4 levels "0","1","2","3 or +": 2 2 4 4 4 3 3 1 4 3 ...
##  $ typ.group: chr "MIX"
```

Aplicamos PCAmix:

```
es.pcamix=PCAmix(X.quanti = es.df$X.quanti,
                 X.quali = es.df$X.quali,
                 rename.level = TRUE, graph = FALSE)
es.pcamix
```

```
##
## Call:
## PCAmix(X.quanti = es.df$X.quanti, X.quali = es.df$X.quali, rename.level = TRUE,     graph = FALSE)
##
## Method = Principal Component of mixed data (PCAmix)
##
##
##      name
## [1,] "$eig"
## [2,] "$ind"
## [3,] "$quanti"
## [4,] "$levels"
## [5,] "$quali"
## [6,] "$sqload"
## [7,] "$coef"
##      description
## [1,] "eigenvalues of the principal components (PC) "
```

```
## [2,] "results for the individuals (coord,contrib,cos2)"
## [3,] "results for the quantitative variables (coord,contrib,cos2)"
## [4,] "results for the levels of the qualitative variables (coord,contrib,cos2)"
## [5,] "results for the qualitative variables (coord,contrib,cos2)"
## [6,] "squared loadings"
## [7,] "coef of the linear combinations defining the PC"
```

2.3.2.1 Autovalores e inercia parcial y acumulada

```r
es.pcamix$eig
```

```
##          Eigenvalue   Proportion Cumulative
## dim 1  6.310191e+00 2.524076e+01   25.24076
## dim 2  2.697311e+00 1.078924e+01   36.03001
## dim 3  2.337837e+00 9.351347e+00   45.38135
## dim 4  1.560155e+00 6.240620e+00   51.62197
## dim 5  1.179731e+00 4.718922e+00   56.34090
## dim 6  1.050822e+00 4.203289e+00   60.54418
## dim 7  1.024263e+00 4.097051e+00   64.64124
## dim 8  9.791605e-01 3.916642e+00   68.55788
## dim 9  9.391575e-01 3.756630e+00   72.31451
## dim 10 8.660783e-01 3.464313e+00   75.77882
## dim 11 7.734111e-01 3.093644e+00   78.87247
## dim 12 7.289646e-01 2.915859e+00   81.78832
## dim 13 6.871324e-01 2.748530e+00   84.53685
## dim 14 6.030891e-01 2.412356e+00   86.94921
## dim 15 5.577486e-01 2.230994e+00   89.18020
## dim 16 4.835794e-01 1.934318e+00   91.11452
## dim 17 4.750533e-01 1.900213e+00   93.01474
## dim 18 3.966466e-01 1.586586e+00   94.60132
## dim 19 3.446142e-01 1.378457e+00   95.97978
## dim 20 3.109675e-01 1.243870e+00   97.22365
## dim 21 2.622891e-01 1.049157e+00   98.27280
## dim 22 2.163303e-01 8.653213e-01   99.13813
## dim 23 1.320954e-01 5.283815e-01   99.66651
## dim 24 8.337272e-02 3.334909e-01  100.00000
## dim 25 3.459173e-07 1.383669e-06  100.00000
```

El crecimiento de la inercia acumulada al aumentar el número de componentes es notablemente bajo, con el límite habitual de 5 CP no se alcanza el 60% de la varianza y es necesario considerar 12 CP para superar el 80%.

2.3.2.2 Resumen PCAmix

```r
summary(es.pcamix)
```

```
##
## Call:
## PCAmix(X.quanti = es.df$X.quanti, X.quali = es.df$X.quali, rename.level = TRUE,     graph = FALSE)
##
## Method = Factor Analysis of mixed data (FAmix)
##
## Data:
##    number of observations:  540
##    number of  variables:  18
```

```
##         number of numerical variables:  9
##         number of categorical variables:  9
##
## Squared loadings :
##            dim1 dim2 dim3 dim4 dim5
## farmers    0.22 0.02 0.01 0.09 0.07
## tradesmen  0.01 0.00 0.00 0.07 0.05
## managers   0.11 0.10 0.37 0.02 0.02
## workers    0.03 0.12 0.01 0.58 0.16
## unemployed 0.09 0.00 0.00 0.06 0.25
## middleempl 0.07 0.01 0.41 0.02 0.01
## retired    0.00 0.00 0.32 0.47 0.01
## employrate 0.06 0.04 0.48 0.02 0.06
## income     0.05 0.08 0.46 0.03 0.00
## butcher    0.62 0.13 0.03 0.00 0.01
## baker      0.76 0.35 0.01 0.00 0.08
## postoffice 0.67 0.08 0.00 0.00 0.01
## dentist    0.81 0.39 0.04 0.04 0.06
## grocery    0.19 0.01 0.04 0.01 0.06
## nursery    0.23 0.15 0.01 0.04 0.03
## doctor     0.84 0.42 0.02 0.01 0.04
## chemist    0.87 0.52 0.07 0.06 0.01
## restaurant 0.68 0.26 0.05 0.02 0.28
```

La varianza explicada por cada variable para cada componente (Squared Loadings) pone de manifiesto que las dos primeras CP están preferentemente relacionadas con los servicios, sobre todo, sanitarios. La tercera CP tiene una mayor relación con tasa de empleo, ingresos, etc.

2.3.2.3 Coeficientes

```r
#str(es.pcamix$coef)
es.pcamix.coef =cbind(es.pcamix$coef$dim1,
                es.pcamix$coef$dim2,es.pcamix$coef$dim3,
                es.pcamix$coef$dim4,es.pcamix$coef$dim5)

kable(round(es.pcamix.coef,4),
      col.names = c('dim1','dim2','dim3','dim4','dim5'),
      caption = 'Coeficientes de las componentes')
```

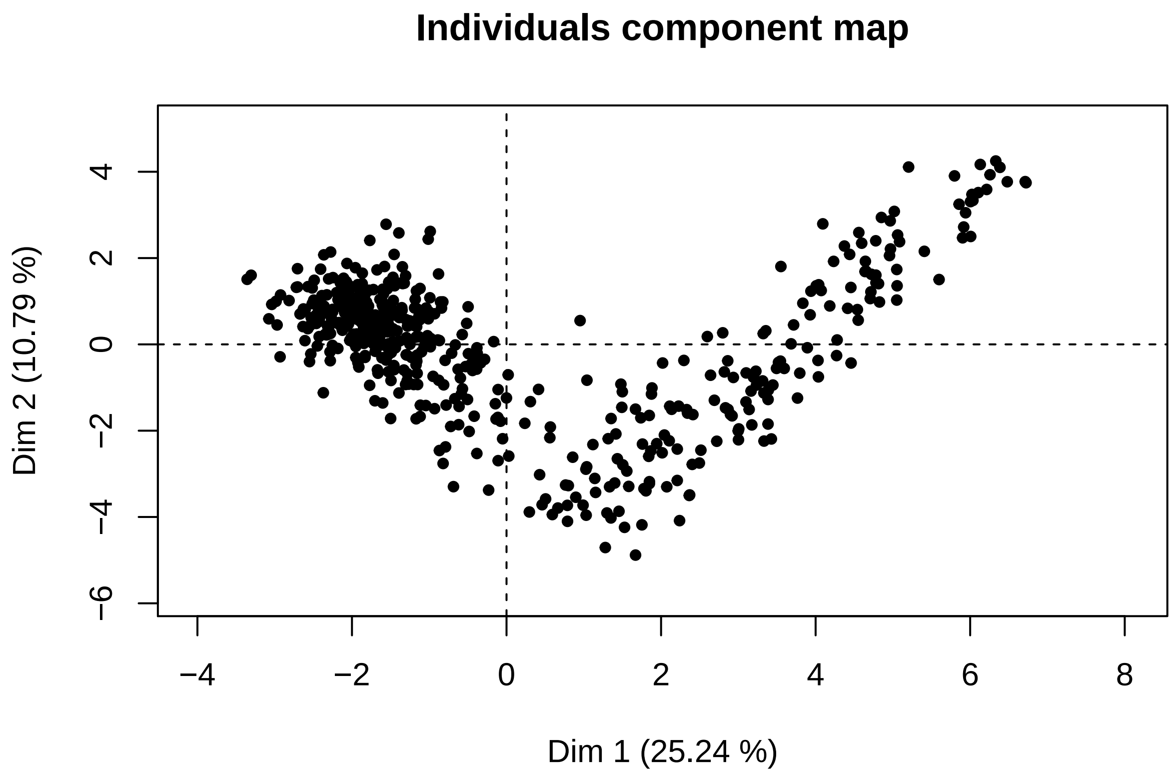
Table 11: Coeficientes de las componentes

	dim1	dim2	dim3	dim4	dim5
const	-0.0027	0.0061	-0.0032	0.0142	0.0026
farmers	-0.2043	0.0970	-0.0614	0.2652	-0.2612
tradesmen	-0.0370	-0.0288	0.0201	0.2178	0.2110
managers	0.1310	0.1904	0.3971	0.1200	0.1133
workers	-0.0692	-0.2196	-0.0636	-0.6302	-0.3788
unemployed	0.1198	0.0429	-0.0365	-0.2072	0.4660
middleempl	0.1056	0.0693	0.4191	-0.1167	0.0765
retired	-0.0120	0.0123	-0.3723	0.5492	0.0842
employrate	-0.0952	0.1285	0.4554	0.1049	-0.2283
income	0.0881	0.1699	0.4451	0.1398	0.0060
butcher=0	-0.1919	0.0305	0.0437	-0.0133	0.0223
butcher=1	0.1942	-0.4122	0.0610	0.0552	0.1071

	dim1	dim2	dim3	dim4	dim5
butcher=2 or +	0.6889	0.3670	-0.2886	-0.0045	-0.2421
baker=0	-0.2723	0.2051	-0.0016	-0.0136	-0.0388
baker=1	0.0416	-0.6476	0.0998	-0.0446	0.4129
baker=2 or +	0.5966	0.1920	-0.1002	0.0783	-0.3384
postoffice=0	-0.2456	0.1294	0.0042	-0.0138	0.0537
postoffice=1 or +	0.4310	-0.2272	-0.0074	0.0242	-0.0942
dentist=0	-0.2258	0.0548	-0.0245	-0.0395	0.1096
dentist=1 to 2	0.3727	-0.7320	0.2744	0.3246	-0.4795
dentist=3 or +	0.7196	0.6273	-0.2145	-0.1985	0.0237
grocery=0	-0.1206	0.0472	0.0943	0.0602	-0.1582
grocery=1 or +	0.2474	-0.0969	-0.1935	-0.1235	0.3244
nursery=0	-0.0395	-0.0483	0.0098	0.0318	-0.0306
nursery=1 or +	0.9300	1.1377	-0.2301	-0.7495	0.7206
doctor=0	-0.2696	0.1812	-0.0370	-0.0667	0.0441
doctor=1 to 2	0.1160	-0.8758	0.2121	0.1921	0.2420
doctor=3 or +	0.6184	0.1765	-0.0608	0.0317	-0.2947
chemist=0	-0.2535	0.1225	-0.0126	-0.0796	0.0790
chemist=1	0.3071	-0.8263	0.2817	0.4003	-0.1463
chemist=2 or +	0.7326	0.5759	-0.3289	-0.1870	-0.1590
restaurant=0	-0.2684	0.1925	-0.0621	0.0259	-0.2031
restaurant=1	-0.0777	-0.3116	0.2424	-0.2149	0.8482
restaurant=2	0.1211	-0.6893	0.0656	0.1186	-0.7104
restaurant=3 or +	0.5697	0.2206	-0.1468	0.1132	-0.1386

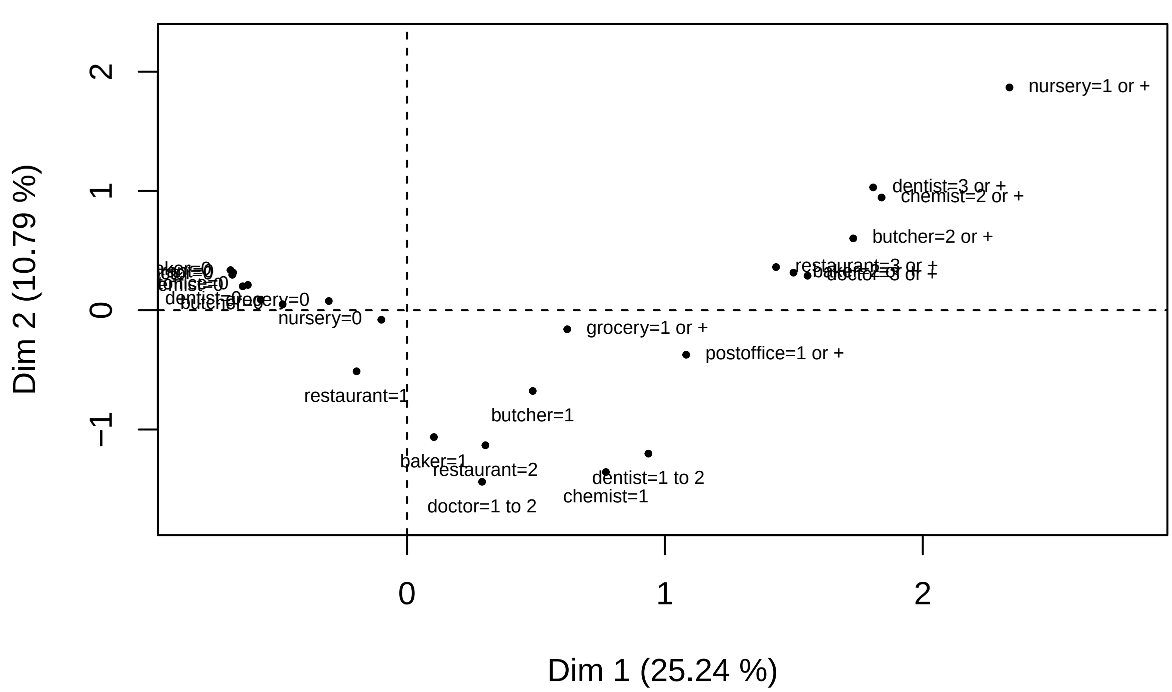
2.3.2.4 Gráficos

```
plot(es.pcamix,choice="ind",axes=c(1,2),label=FALSE)
```

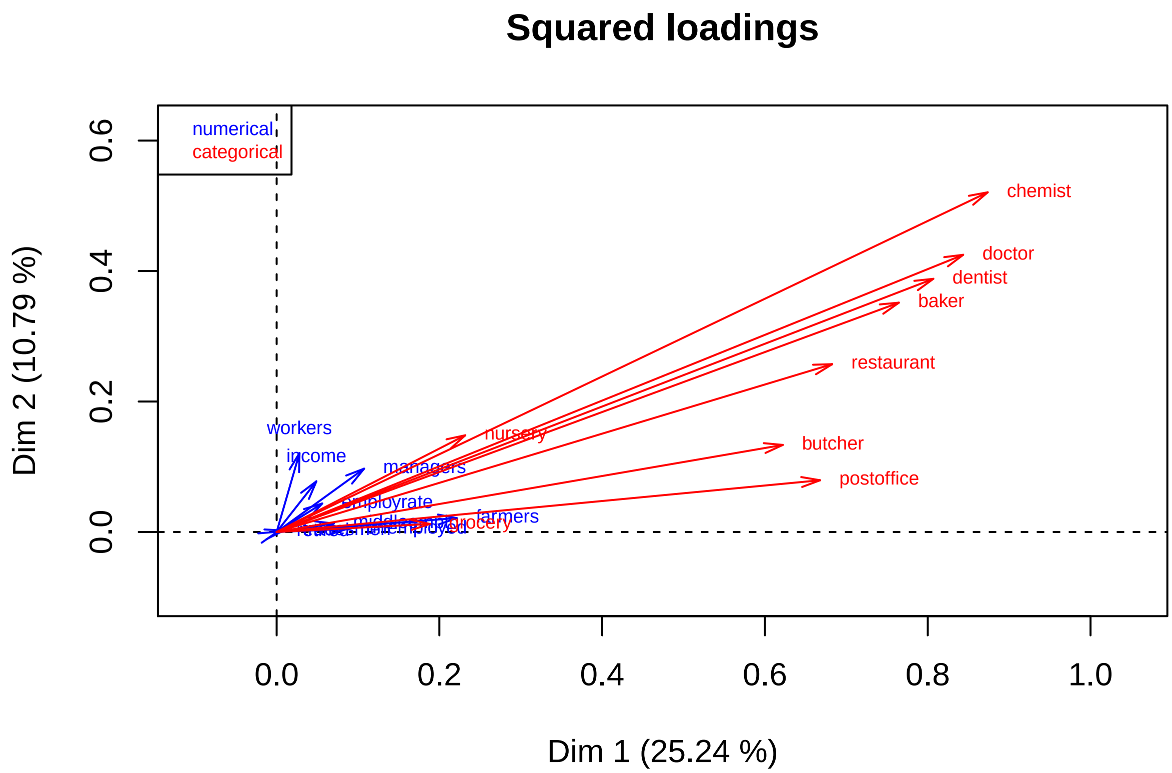



```r
plot(es.pcamix,choice="levels",axes=c(1,2),label=TRUE,cex = 0.6)
```

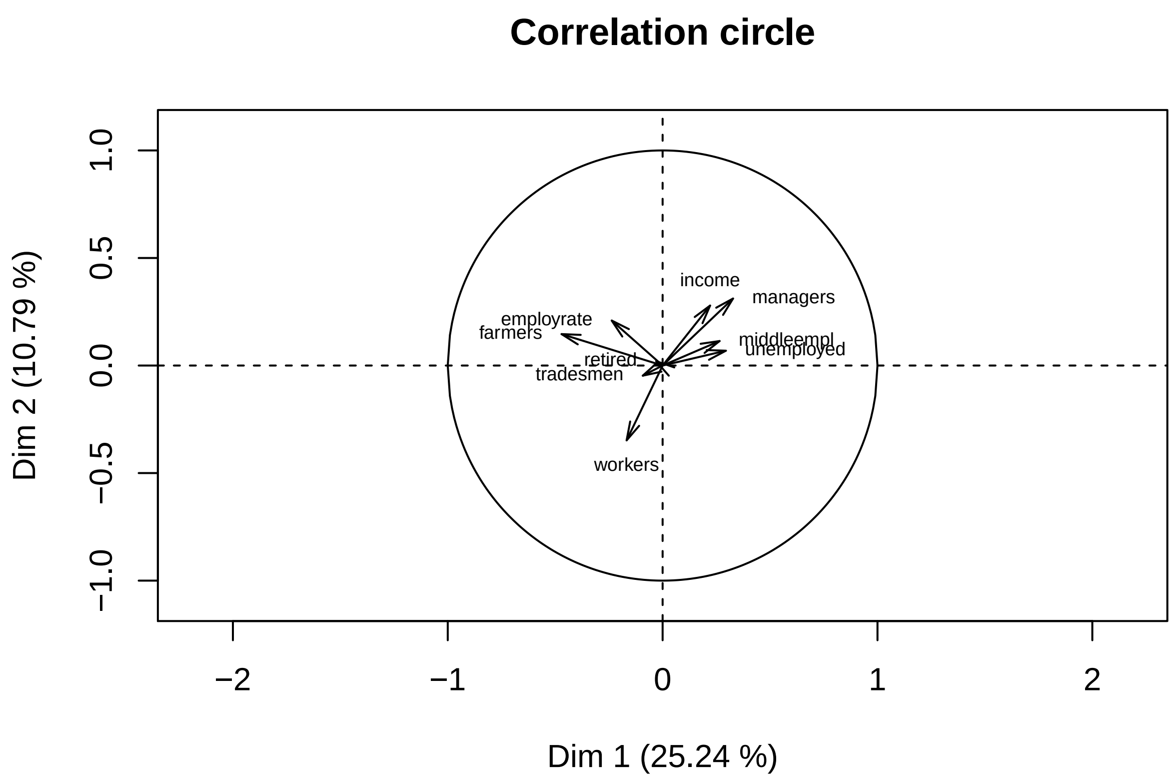
Levels component map




```
plot(es.pcamix,choice="sqload",axes=c(1,2),label=TRUE,cex = 0.6, coloring.var = "type",cex.leg = 0.6)
```




```r
plot(es.pcamix,choice="cor",axes=c(1,2),label=TRUE,cex = 0.6)
```



2.4 Regresión datos de empleo

2.4.1 Selección de casos completos, resumen, transformación y outliers

```
em.df = na.omit(gironde$employment)
str(em.df)
```

```
## 'data.frame':    540 obs. of  9 variables:
##  $ farmers   : num  1.98 5.23 0.1 0.18 0.3 ...
##  $ tradesmen : num  3.68 5.23 4.38 2.29 3.8 5.63 4.21 1.75 4.61 2.3 ...
##  $ managers  : num  3.97 1.96 5.56 3.7 8.19 1.25 4.21 3.51 5.8 0 ...
##  $ workers   : num  38.2 21.6 36 42.4 18.6 ...
##  $ unemployed: num  13.6 15 18.2 15.1 13 ...
##  $ middleempl: num  9.63 14.38 15.48 8.98 12.07 ...
##  $ retired   : num  28.9 36.6 20.3 27.3 44 ...
##  $ employrate: num  89.3 90.9 90.2 87.4 89.4 ...
##  $ income    : num  17671 19422 21047 18015 27147 ...
##  - attr(*, "na.action")=Class 'omit'  Named int [1:2] 63 369
##   .. ..- attr(*, "names")= chr [1:2] "BOSSUGAN" "SAINT-AVIT-DE-SOULEGE"
```

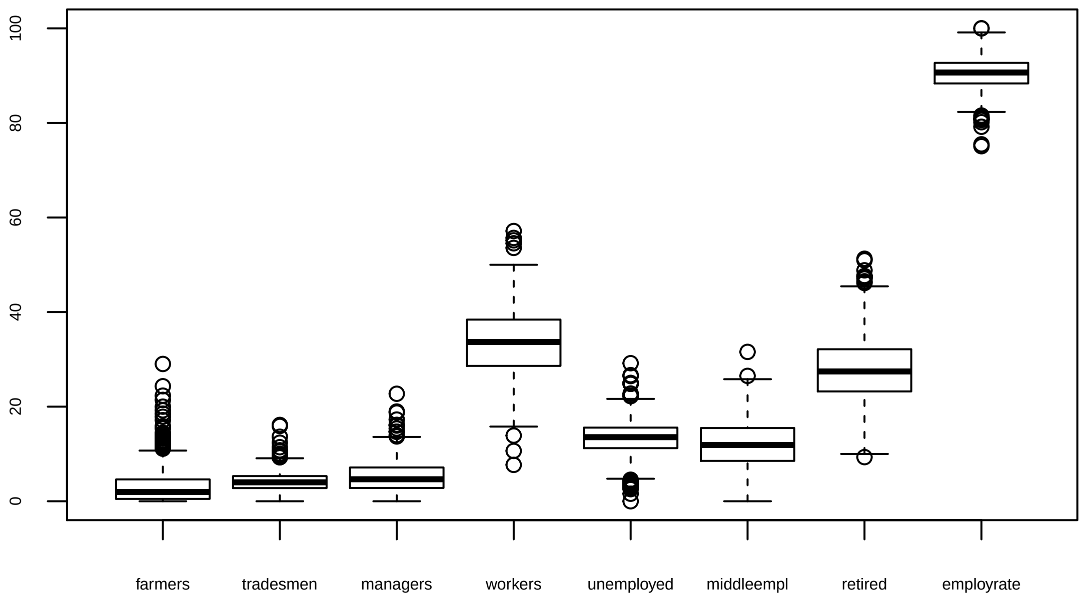
```
summary(em.df)
```

```
##     farmers          tradesmen         managers         workers
##  Min.   : 0.0000   Min.   : 0.000   Min.   : 0.000   Min.   : 7.69
##  1st Qu.: 0.5025   1st Qu.: 2.780   1st Qu.: 2.825   1st Qu.:28.64
```

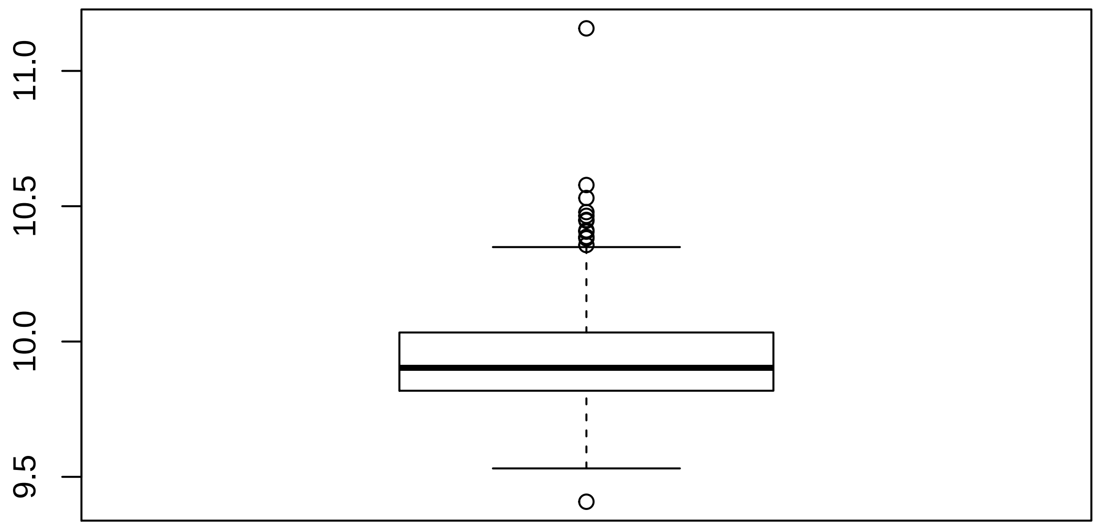
```
##  Median : 1.9550   Median : 4.000   Median : 4.650   Median :33.67
##  Mean   : 3.3544   Mean   : 4.204   Mean   : 5.286   Mean   :33.65
##  3rd Qu.: 4.6125   3rd Qu.: 5.312   3rd Qu.: 7.143   3rd Qu.:38.41
##  Max.   :29.0300   Max.   :16.130   Max.   :22.730   Max.   :57.14
##    unemployed      middleempl         retired         employrate
##  Min.   : 0.00   Min.   : 0.000   Min.   : 9.33   Min.   : 75.08
##  1st Qu.:11.23   1st Qu.: 8.547   1st Qu.:23.23   1st Qu.: 88.35
##  Median :13.55   Median :11.905   Median :27.45   Median : 90.66
##  Mean   :13.35   Mean   :12.005   Mean   :28.16   Mean   : 90.31
##  3rd Qu.:15.55   3rd Qu.:15.465   3rd Qu.:32.14   3rd Qu.: 92.70
##  Max.   :29.19   Max.   :31.580   Max.   :51.28   Max.   :100.00
##      income
##  Min.   :12187
##  1st Qu.:18367
##  Median :19990
##  Mean   :21003
##  3rd Qu.:22768
##  Max.   :70062
```

```r
boxplot(em.df)
```


```r
# Transformación logaritmica de variable income
em.df$income = log(em.df$income)
#colnames(em.df)[9] = 'lincome'

boxplot(em.df[,-9],cex.axis=0.5)
```




```r
boxplot(em.df[,9])
```



Hay una observación claramente muy alejada de las demás respecto a la variable income. Se elimina esta observación para los análisis que siguen.

```
iout = which.max(em.df$income)
em.df[iout,]
```

```
##          farmers tradesmen managers workers unemployed middleempl retired
## DOULEZON    9.43      7.55     3.77    28.3      20.75     11.32   18.87
##          employrate   income
## DOULEZON      94.68 11.15713
```

```
# Excluimos el caso
em.df = em.df[-iout]
```

2.4.2 Segregación en conjunto entrenamiento y test

```
set.seed(12345)
n=nrow(em.df)
ind=1:n
itest=sample(ind,trunc(n*0.25)+1)
ient=setdiff(ind,itest)
```

2.4.3 Función auxiliar Ajuste

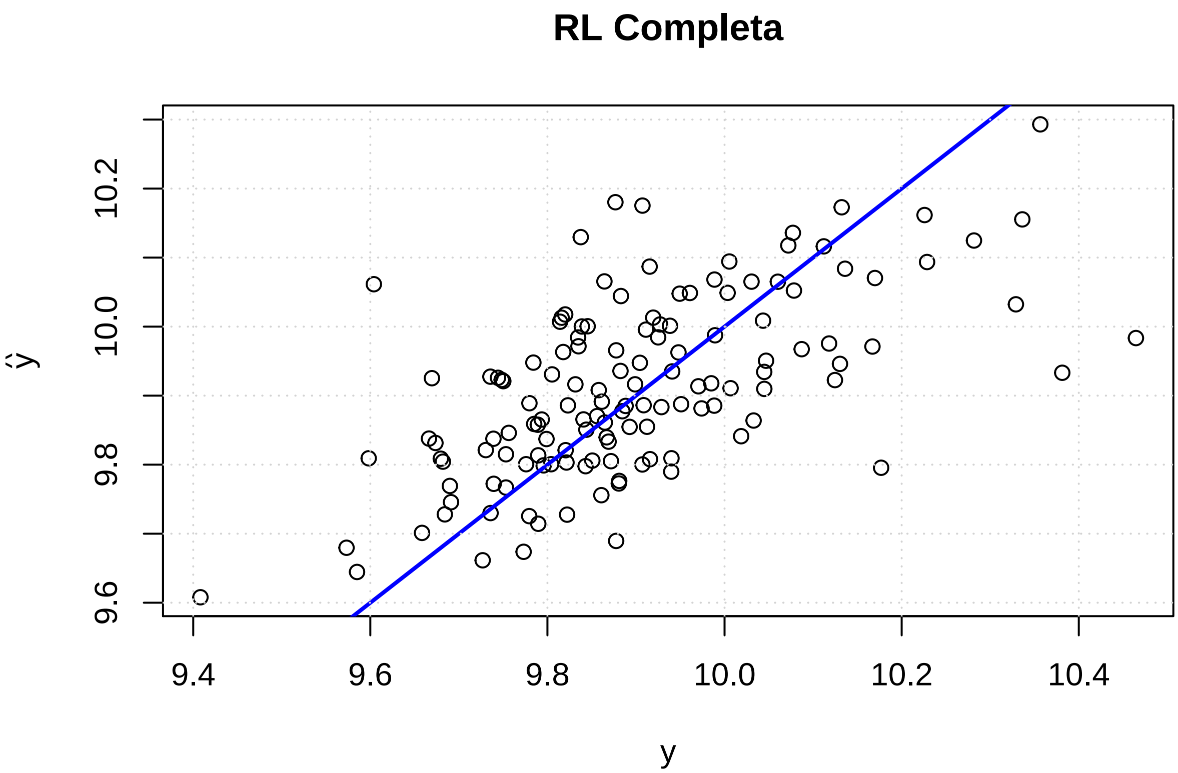
```
Ajuste<- function(y,pred,titulo)
{
  residuos=y-pred
  plot(y,pred,main=titulo,ylab=expression(hat(y)))
  abline(a=0,b=1,col="blue",lwd=2)
  grid()
  MSE= mean(residuos^2)
  RMSE= sqrt(MSE)
  R2= cor(y,pred)^2
  return(list(MSE=MSE,RMSE=RMSE,R2=R2))
}
```

2.4.4 Regresión lineal completa

```
em.df.all = lm(income~.,data=em.df,subset=ient)
summary(em.df.all)
```

```
##
## Call:
## lm(formula = income ~ ., data = em.df, subset = ient)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.62050 -0.07591 -0.00989  0.06974  1.17730
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.317e+02  9.364e+01  -2.475   0.0138 *
## farmers      2.402e+00  9.363e-01   2.565   0.0107 *
## tradesmen    2.408e+00  9.364e-01   2.572   0.0105 *
## managers     2.420e+00  9.362e-01   2.585   0.0101 *
## workers      2.400e+00  9.363e-01   2.564   0.0107 *
## unemployed   2.401e+00  9.363e-01   2.564   0.0107 *
## middleempl   2.411e+00  9.363e-01   2.575   0.0104 *
## retired      2.400e+00  9.363e-01   2.564   0.0107 *
## employrate   1.478e-02  2.659e-03   5.559 5.01e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1468 on 395 degrees of freedom
## Multiple R-squared:  0.4281, Adjusted R-squared:  0.4165
## F-statistic: 36.95 on 8 and 395 DF,  p-value: < 2.2e-16
```

```
em.df.all.pred.test=predict(em.df.all,newdata=em.df[itest,])
Ajuste(em.df[itest,9],em.df.all.pred.test,"RL Completa")
```




```
## $MSE
## [1] 0.0192036
##
## $RMSE
## [1] 0.138577
##
## $R2
## [1] 0.3792036
```

2.4.5 Regresión lineal con mejor subconjunto (leaps)

```
library(leaps)
em.df.best = regsubsets(income~.,data=em.df[ient,],nvmax=8)
summary(em.df.best)
```

```
## Subset selection object
## Call: regsubsets.formula(income ~ ., data = em.df[ient, ], nvmax = 8)
## 8 Variables  (and intercept)
##            Forced in Forced out
## farmers        FALSE      FALSE
## tradesmen      FALSE      FALSE
## managers       FALSE      FALSE
## workers        FALSE      FALSE
## unemployed     FALSE      FALSE
## middleempl     FALSE      FALSE
```

```
## retired        FALSE      FALSE
## employrate     FALSE      FALSE
## 1 subsets of each size up to 8
## Selection Algorithm: exhaustive
##          farmers tradesmen managers workers unemployed middleempl retired
## 1  ( 1 ) " "     " "       "*"      " "     " "        " "        " "
## 2  ( 1 ) " "     " "       "*"      " "     " "        " "        " "
## 3  ( 1 ) " "     " "       "*"      " "     " "        "*"        " "
## 4  ( 1 ) " "     "*"       "*"      " "     " "        "*"        " "
## 5  ( 1 ) "*"     "*"       "*"      " "     " "        "*"        " "
## 6  ( 1 ) "*"     "*"       "*"      " "     "*"        "*"        " "
## 7  ( 1 ) "*"     "*"       "*"      "*"     "*"        "*"        " "
## 8  ( 1 ) "*"     "*"       "*"      "*"     "*"        "*"        "*"
##          employrate
## 1  ( 1 ) " "
## 2  ( 1 ) "*"
## 3  ( 1 ) "*"
## 4  ( 1 ) "*"
## 5  ( 1 ) "*"
## 6  ( 1 ) "*"
## 7  ( 1 ) "*"
## 8  ( 1 ) "*"
```

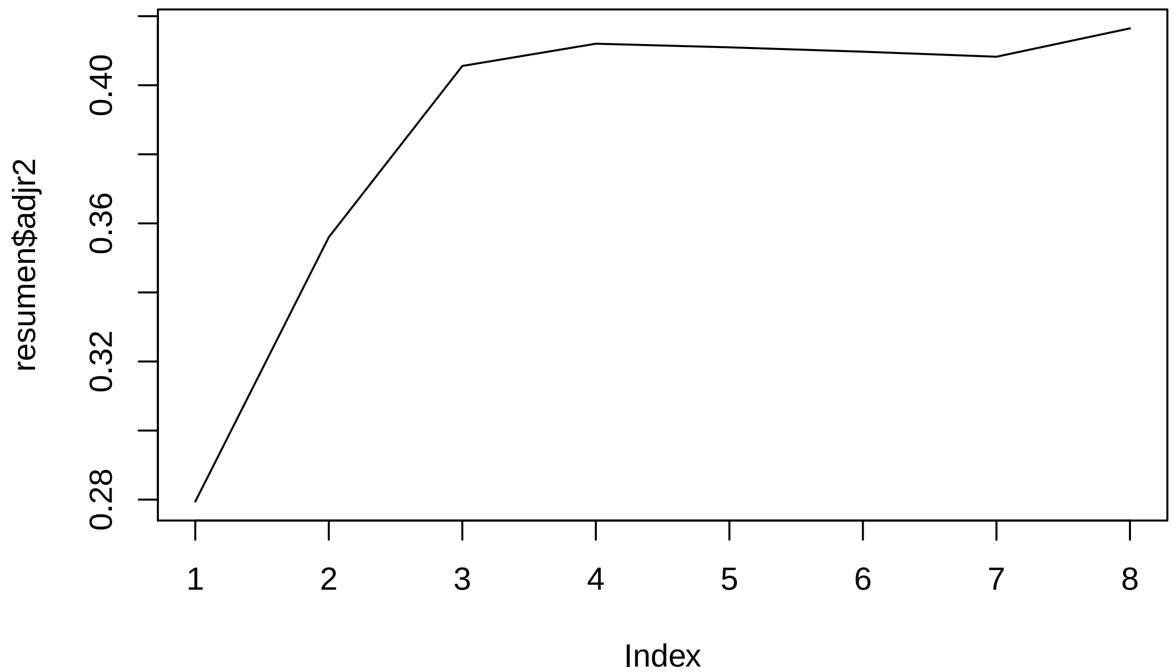
```
resumen=summary(em.df.best)
names(resumen)
```

```
## [1] "which"  "rsq"    "rss"    "adjr2"  "cp"     "bic"    "outmat" "obj"
```

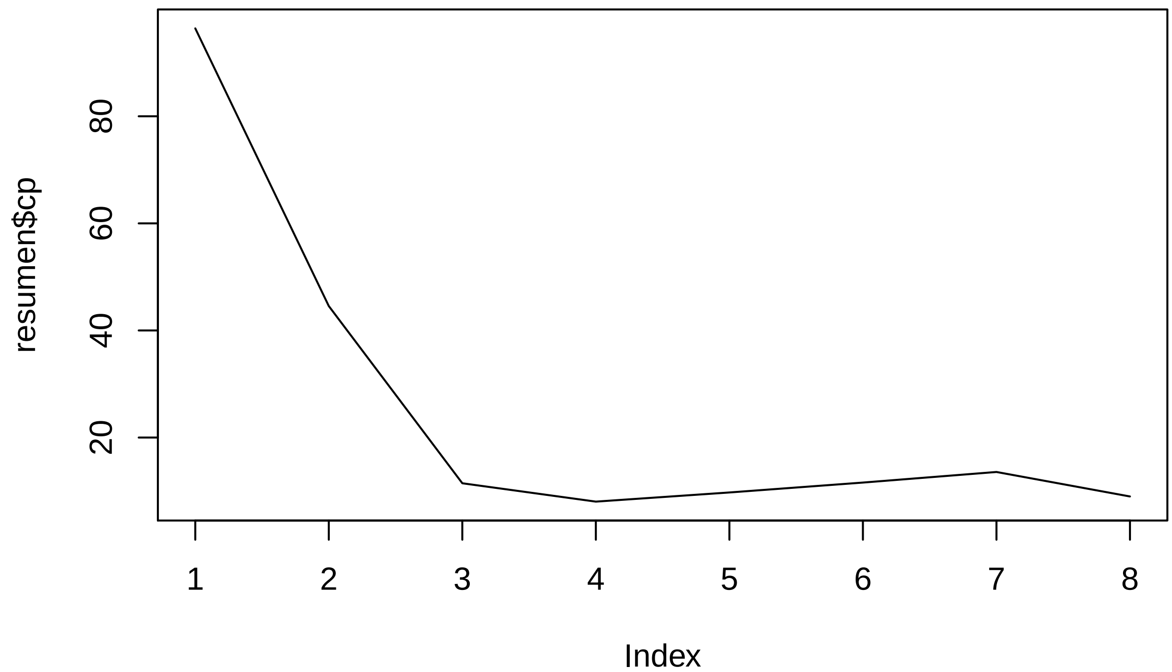
```
resumen$rsq #R2 aumenta con el número de predictores
```

```
## [1] 0.2812109 0.3591887 0.4100028 0.4178610 0.4182872 0.4185024 0.4185361
## [8] 0.4280517
```

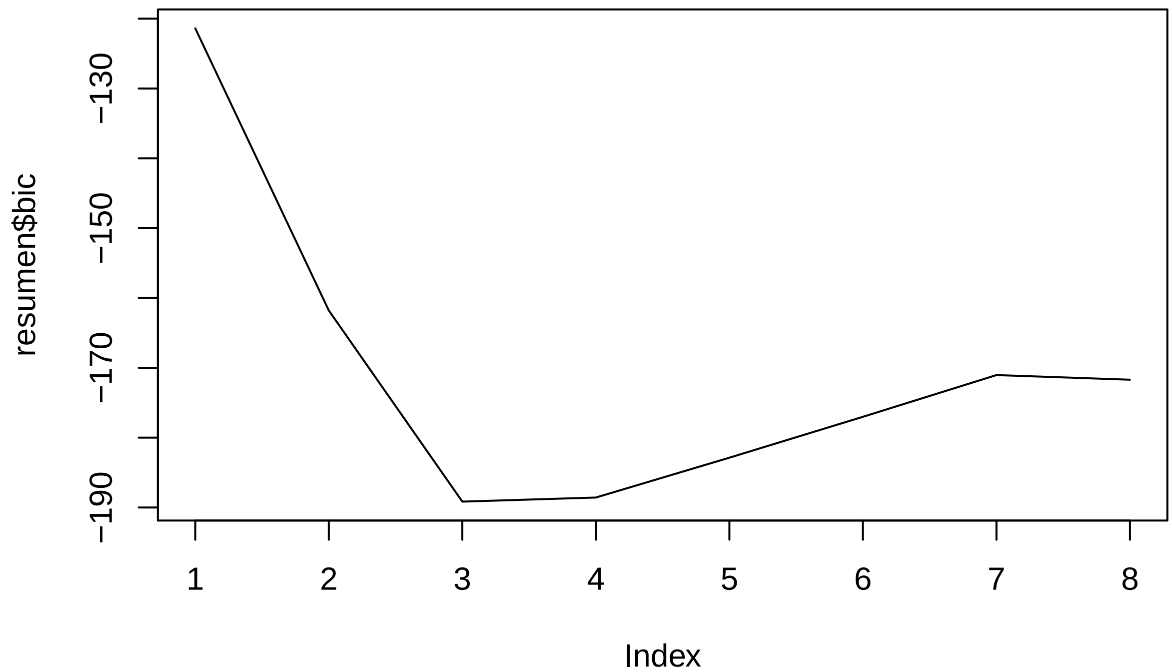
```
plot(resumen$adjr2,type="l")
```




```r
plot(resumen$cp,type="l")
```




```
plot(resumen$bic,type="l")
```




```r
which.min(resumen$cp)
```

```
## [1] 4
```

```r
which.min(resumen$bic)
```

```
## [1] 3
```

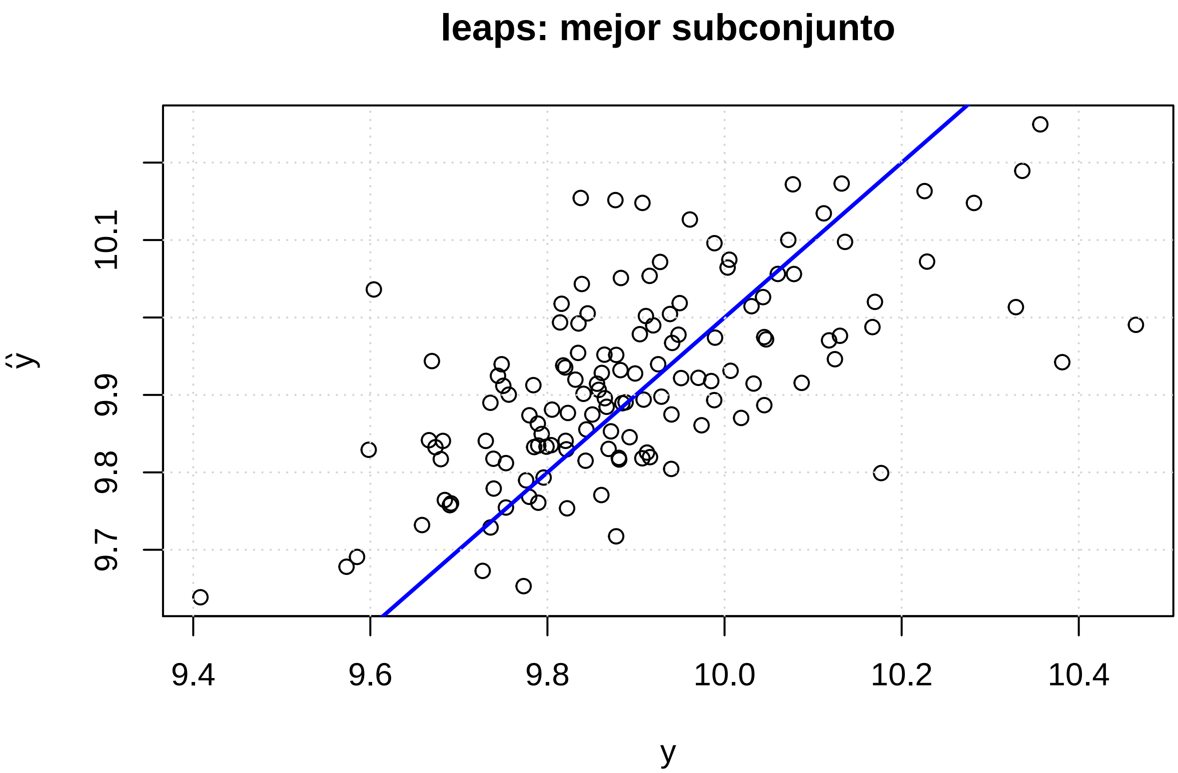
```r
compos<- which.min(resumen$bic)
vsel<- colnames(resumen$which)[resumen$which[compos,]]
vsel
```

```
## [1] "(Intercept)" "managers"    "middleempl"  "employrate"
```

```r
#quitamos (Intercept)
vsel=vsel[-1]
fmla <- as.formula(paste("income ~ ", paste(vsel, collapse= "+")))
fmla
```

```
## income ~ managers + middleempl + employrate
```

```r
em.df.best1<- lm(fmla,data=em.df[ient,])

em.df.best1.pred.test=predict(em.df.best1,newdata=em.df[itest,])
Ajuste(em.df[itest,9],em.df.best1.pred.test,"leaps: mejor subconjunto")
```




```
## $MSE
## [1] 0.01831894
##
## $RMSE
## [1] 0.1353475
##
## $R2
## [1] 0.4037347
```

2.4.6 Regresión lineal secuencial (seqrep)

```r
library(leaps)
em.df.seq = regsubsets(income~.,data=em.df[ient,],nvmax=8,method = "seqrep")
summary(em.df.seq)
```

```
## Subset selection object
## Call: regsubsets.formula(income ~ ., data = em.df[ient, ], nvmax = 8,
##     method = "seqrep")
## 8 Variables  (and intercept)
##            Forced in Forced out
## farmers        FALSE      FALSE
## tradesmen      FALSE      FALSE
## managers       FALSE      FALSE
## workers        FALSE      FALSE
## unemployed     FALSE      FALSE
```

```
## middleempl     FALSE      FALSE
## retired        FALSE      FALSE
## employrate     FALSE      FALSE
## 1 subsets of each size up to 8
## Selection Algorithm: 'sequential replacement'
##          farmers tradesmen managers workers unemployed middleempl retired
## 1  ( 1 ) " "     " "       "*"      " "     " "        " "        " "
## 2  ( 1 ) " "     " "       "*"      " "     " "        " "        " "
## 3  ( 1 ) " "     " "       "*"      " "     " "        "*"        " "
## 4  ( 1 ) " "     "*"       "*"      " "     " "        "*"        " "
## 5  ( 1 ) "*"     "*"       "*"      "*"     "*"        " "        " "
## 6  ( 1 ) "*"     "*"       "*"      " "     "*"        "*"        " "
## 7  ( 1 ) "*"     "*"       "*"      "*"     "*"        "*"        " "
## 8  ( 1 ) "*"     "*"       "*"      "*"     "*"        "*"        "*"
##          employrate
## 1  ( 1 ) " "
## 2  ( 1 ) "*"
## 3  ( 1 ) "*"
## 4  ( 1 ) "*"
## 5  ( 1 ) " "
## 6  ( 1 ) "*"
## 7  ( 1 ) "*"
## 8  ( 1 ) "*"
```

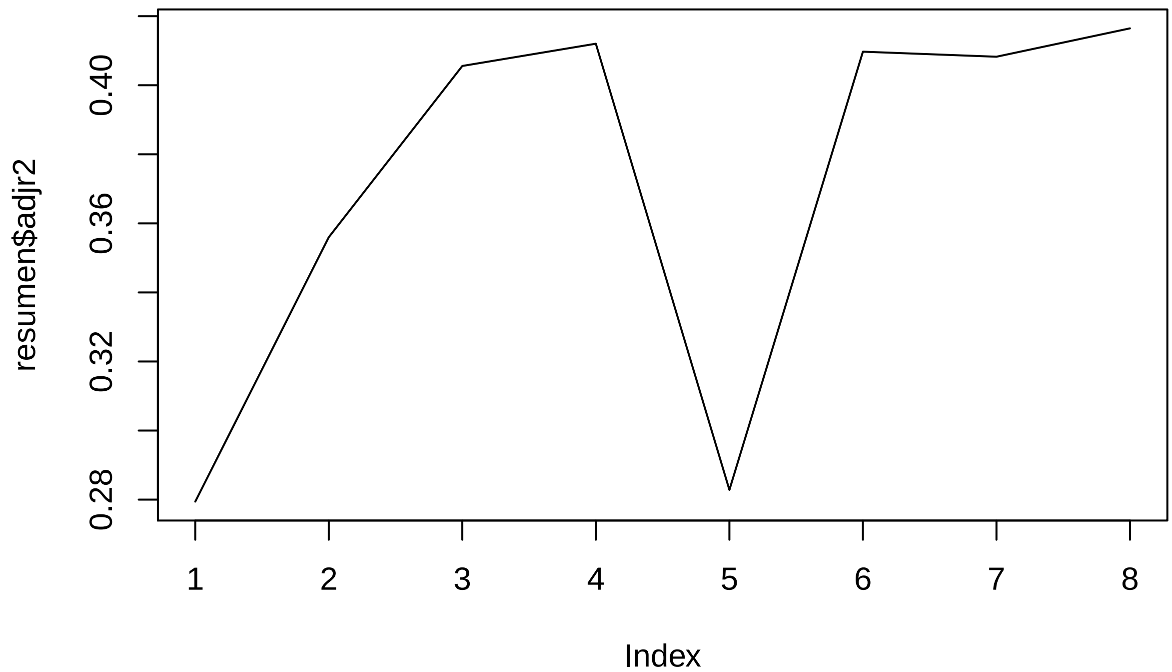
```r
resumen=summary(em.df.seq)
names(resumen)
```

```
## [1] "which"  "rsq"    "rss"    "adjr2"  "cp"     "bic"    "outmat" "obj"
```

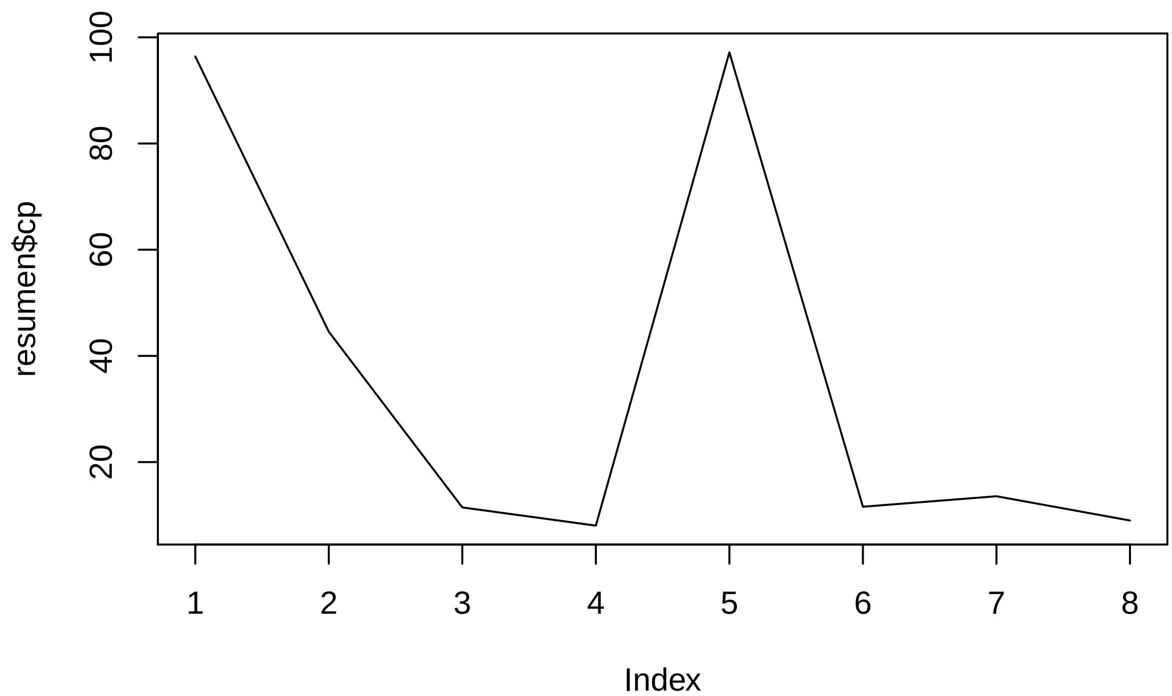
```r
resumen$rsq #R2 aumenta con el número de predictores
```

```
## [1] 0.2812109 0.3591887 0.4100028 0.4178610 0.2917160 0.4185024 0.4185361
## [8] 0.4280517
```

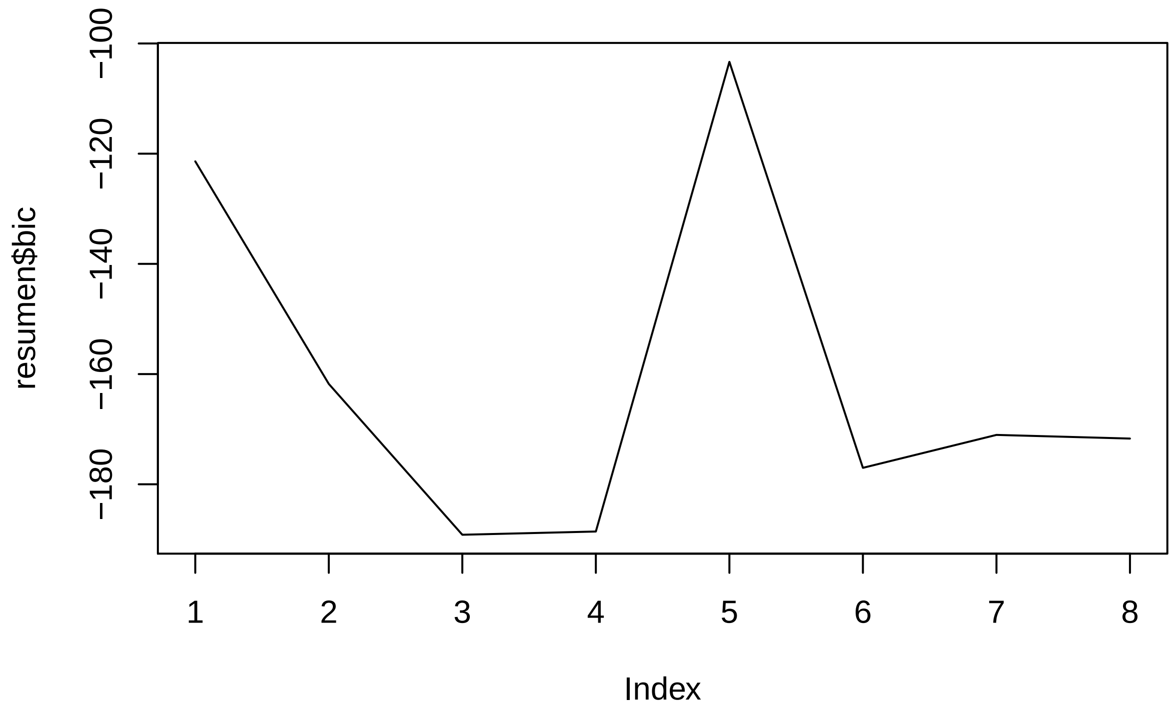
```r
plot(resumen$adjr2,type="l")
```




```
plot(resumen$cp,type="l")
```




```
plot(resumen$bic,type="l")
```




```
which.min(resumen$cp)
```

```
## [1] 4
```

```
which.min(resumen$bic)
```

```
## [1] 3
```

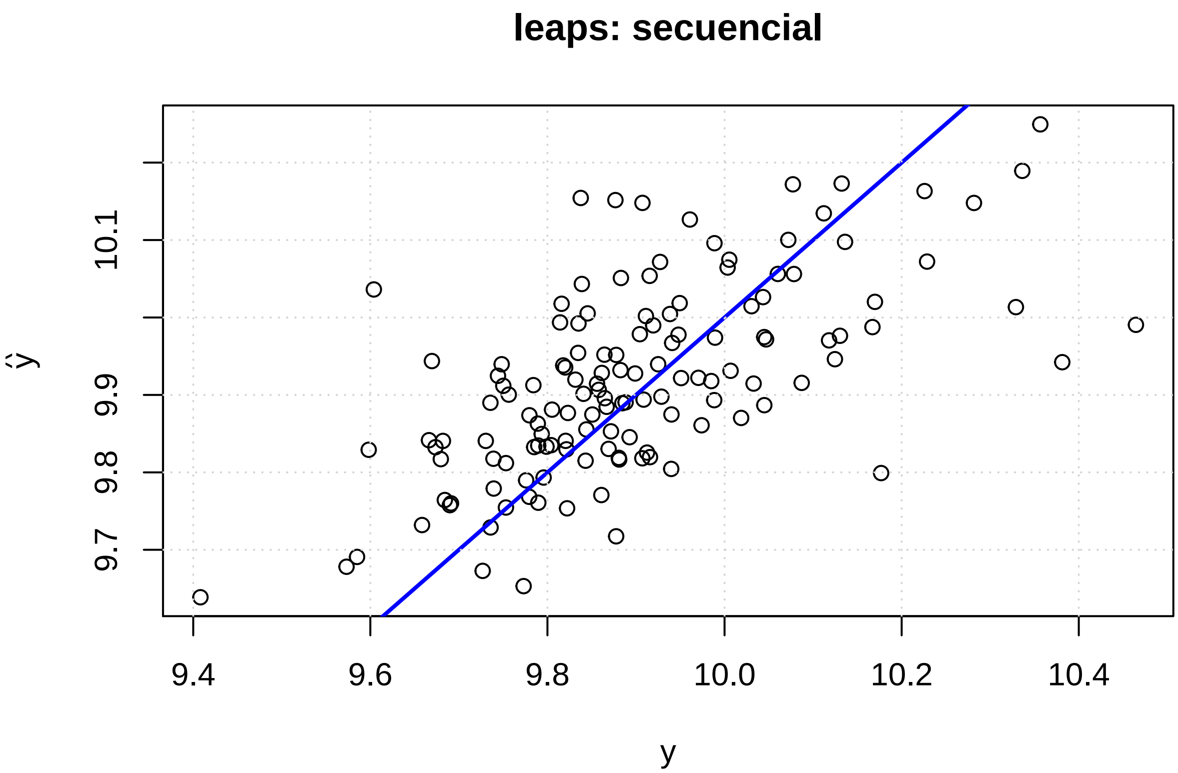
```
compos<- which.min(resumen$bic)
vsel<- colnames(resumen$which)[resumen$which[compos,]]
vsel
```

```
## [1] "(Intercept)" "managers"    "middleempl"  "employrate"
```

```
#quitamos (Intercept)
vsel=vsel[-1]
fmla <- as.formula(paste("income ~ ", paste(vsel, collapse= "+")))
fmla
```

```
## income ~ managers + middleempl + employrate
```

```
em.df.seq1<- lm(fmla,data=em.df[ient,])

em.df.seq1.pred.test=predict(em.df.seq1,newdata=em.df[itest,])
Ajuste(em.df[itest,9],em.df.seq1.pred.test,"leaps: secuencial")
```




```
## $MSE
## [1] 0.01831894
##
## $RMSE
## [1] 0.1353475
##
## $R2
## [1] 0.4037347
```

2.4.7 Algoritmos genéticos

```r
library(GA)
```

```
## Loading required package: foreach
## Loading required package: iterators
## Package 'GA' version 3.0.2
## Type 'citation("GA")' for citing this R package in publications.
```

```r
#Matrices x e y, datos entrenamiento:
xent <- model.matrix(em.df.all)[,-1]
yent <- model.response(model.frame(em.df.all))

#String: vector con 0-1 (1:la variable se usa)
#la función fitness devuelve -AIC del modelo de regresión
```

```r
#lineal múltiple definido por las variables cuya
#posición en string sea 1

fitness <- function(string)
{
  inc <- which(string==1)
  X <- cbind(1, xent[,inc])
  mod <- lm.fit(X, yent)
  class(mod) <- "lm"
  -AIC(mod)   #ga es para maximizar
}

em.df.AG <- ga("binary",
               fitness = fitness, nBits = ncol(xent),
               names = colnames(xent), monitor = FALSE,
               popSize=100)

summary(em.df.AG)
```

```
## +-----------------------------------+
## |         Genetic Algorithm         |
## +-----------------------------------+
##
## GA settings:
## Type                  =  binary
## Population size       =  100
## Number of generations =  100
## Elitism               =  5
## Crossover probability =  0.8
## Mutation probability  =  0.1
##
## GA results:
## Iterations             = 100
## Fitness function value = 393.7289
## Solution =
##      farmers tradesmen managers workers unemployed middleempl retired
## [1,]       0         1        1       0          0          1       0
##      employrate
## [1,]          1
```

```r
#Modelo con las variables seleccionadas
vsel=colnames(em.df.AG@solution)[em.df.AG@solution==1]
fmla <- as.formula(paste("income ~ ", paste(vsel, collapse= "+")))
fmla
```

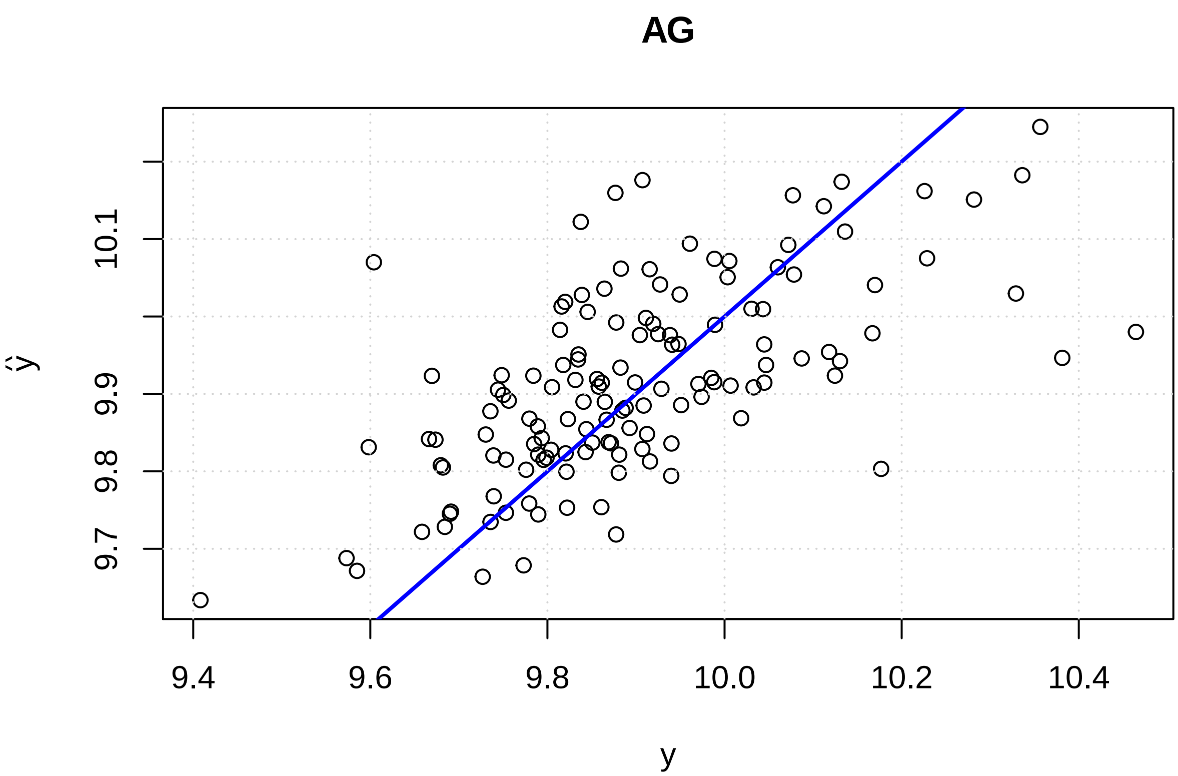
```
## income ~ tradesmen + managers + middleempl + employrate
```

```r
em.df.AG1<- lm(fmla,data=em.df[ient,])
summary(em.df.AG1)
```

```
##
## Call:
## lm(formula = fmla, data = em.df[ient, ])
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -0.63154 -0.07900 -0.00871  0.06925  1.16625
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 8.330910   0.212145  39.270  < 2e-16 ***
## tradesmen   0.007523   0.003242   2.321   0.0208 *
## managers    0.020109   0.002232   9.009  < 2e-16 ***
## middleempl  0.009888   0.001609   6.144 1.94e-09 ***
## employrate  0.014950   0.002413   6.197 1.44e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1474 on 399 degrees of freedom
## Multiple R-squared:  0.4179, Adjusted R-squared:  0.412
## F-statistic:  71.6 on 4 and 399 DF,  p-value: < 2.2e-16
```

```r
em.df.AG1.pred.test=predict(em.df.AG1,newdata=em.df[itest,])
Ajuste(em.df[itest,9],em.df.AG1.pred.test,"AG")
```




```
## $MSE
## [1] 0.01832133
##
## $RMSE
## [1] 0.1353563
##
## $R2
```

```
## [1] 0.4017796
```

3 Árbol de clasificación

```r
library(rpart)
library(rpart.plot)
```

3.1 Lectura de datos, partición entrenamiento / test

```r
#LEER LOS DATOS, PARTICIÓN ENTRENAMIENTO/TEST
################################################
#VARIABLES:
#default (No/Yes): el cliente presenta números
#         rojos en la tarjeta de crédito
#student (No/Yes)
#balance:saldo medio tras el pago mensual
#income: ingresos
Default=read.table(file="Default.txt",header=TRUE)

str(Default)
```

```
## 'data.frame':    673 obs. of  4 variables:
##  $ default: Factor w/ 2 levels "No","Yes": 1 1 1 1 2 1 2 1 2 1 ...
##  $ student: Factor w/ 2 levels "No","Yes": 2 1 2 1 1 2 1 1 2 1 ...
##  $ balance: num  700 1095 256 1717 2064 ...
##  $ income : num  15905 26465 15628 51057 37373 ...
```

```r
n = nrow(Default)

ind=1:n
itest=sample(ind,trunc(n*0.25)+1)
ient=setdiff(ind,itest)

cat(' Observaciones a entrenamiento: \t', length(ient),'\n',
    'Observaciones a test:         \t', length(itest),'\n')
```

```
##  Observaciones a entrenamiento:   504
##  Observaciones a test:            169
```

3.2 Matriz de costes

```r
#EL BANCO PREFIERE EVITAR TARJETAS "DEUDORAS"
#SE VA A CONSIDERAR UNA MATRIZ DE COSTES
#COSTE DE CLASIFICAR NO COMO YES ES 5 VECES SUPERIOR
#A CLASIFICAR YES COMO NO
L=matrix(c(0,1,5,0),2,2)
rownames(L)=colnames(L)=levels(Default$default)
L
```

```
##     No Yes
```

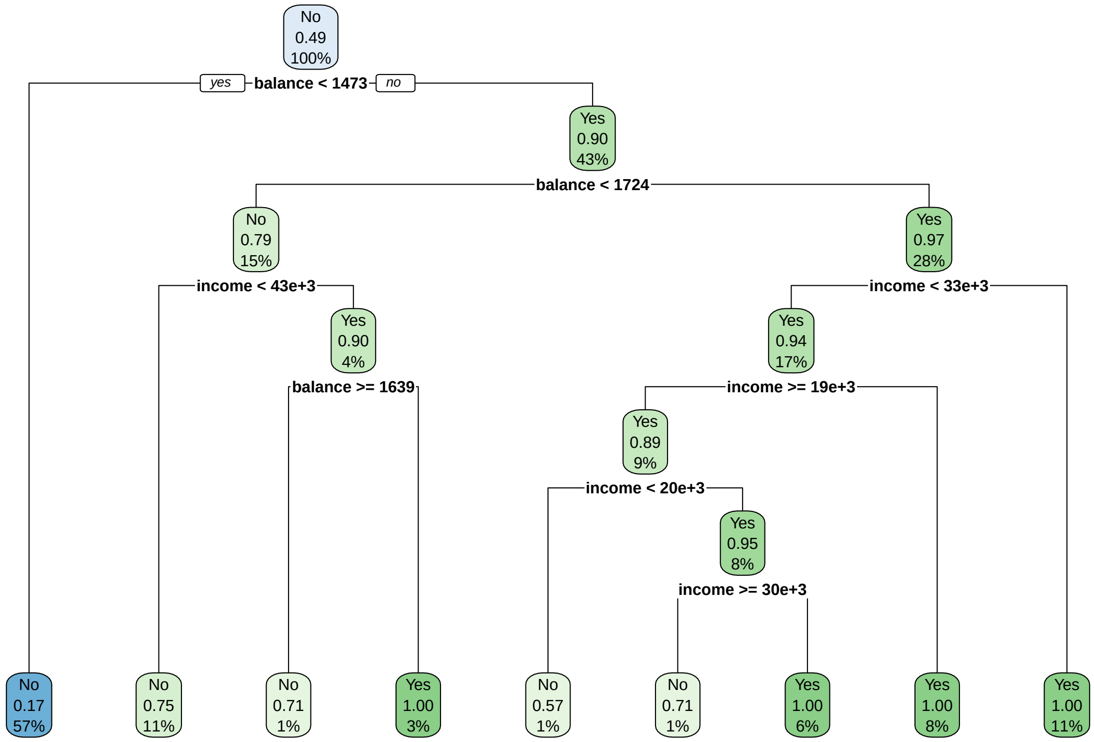
```
## No    0   5
## Yes   1   0
```

3.3 Definición del Árbol de clasificación

```
#CONSTRUIR UN ÁRBOL DE CLASIFICACIÓN CONSIDERANDO
#LOS COSTES DEFINIDOS EN LA MATRIZ L Y
#APLICANDO EL PROCEDIMIENTO DE RECORTE 1-ES
#EVALUAR EL MODELO (ACIERTO, SENSITIVIDAD, ESPECIFICDAD)

Default.rpart = rpart(
  default~., data = Default, subset = ient, method = 'class',
  parms = list(loss = L, split = "gini"))

Default.rpart
```

```
## n= 504
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##   1) root 504 247 No (0.50992063 0.49007937)
##     2) balance< 1472.992 285  49 No (0.82807018 0.17192982) *
##     3) balance>=1472.992 219 105 Yes (0.09589041 0.90410959)
##       6) balance< 1723.545 76  60 No (0.21052632 0.78947368)
##        12) income< 42621.59 56  42 No (0.25000000 0.75000000) *
##        13) income>=42621.59 20  10 Yes (0.10000000 0.90000000)
##          26) balance>=1638.795 7   5 No (0.28571429 0.71428571) *
##          27) balance< 1638.795 13   0 Yes (0.00000000 1.00000000) *
##       7) balance>=1723.545 143  25 Yes (0.03496503 0.96503497)
##        14) income< 33379.21 86  25 Yes (0.05813953 0.94186047)
##          28) income>=19038.19 46  25 Yes (0.10869565 0.89130435)
##            56) income< 19729.35 7   4 No (0.42857143 0.57142857) *
##            57) income>=19729.35 39  10 Yes (0.05128205 0.94871795)
##             114) income>=29809.3 7   5 No (0.28571429 0.71428571) *
##             115) income< 29809.3 32   0 Yes (0.00000000 1.00000000) *
##          29) income< 19038.19 40   0 Yes (0.00000000 1.00000000) *
##        15) income>=33379.21 57   0 Yes (0.00000000 1.00000000) *
```

```
# summary(Default.rpart)

rpart.plot(Default.rpart)
```

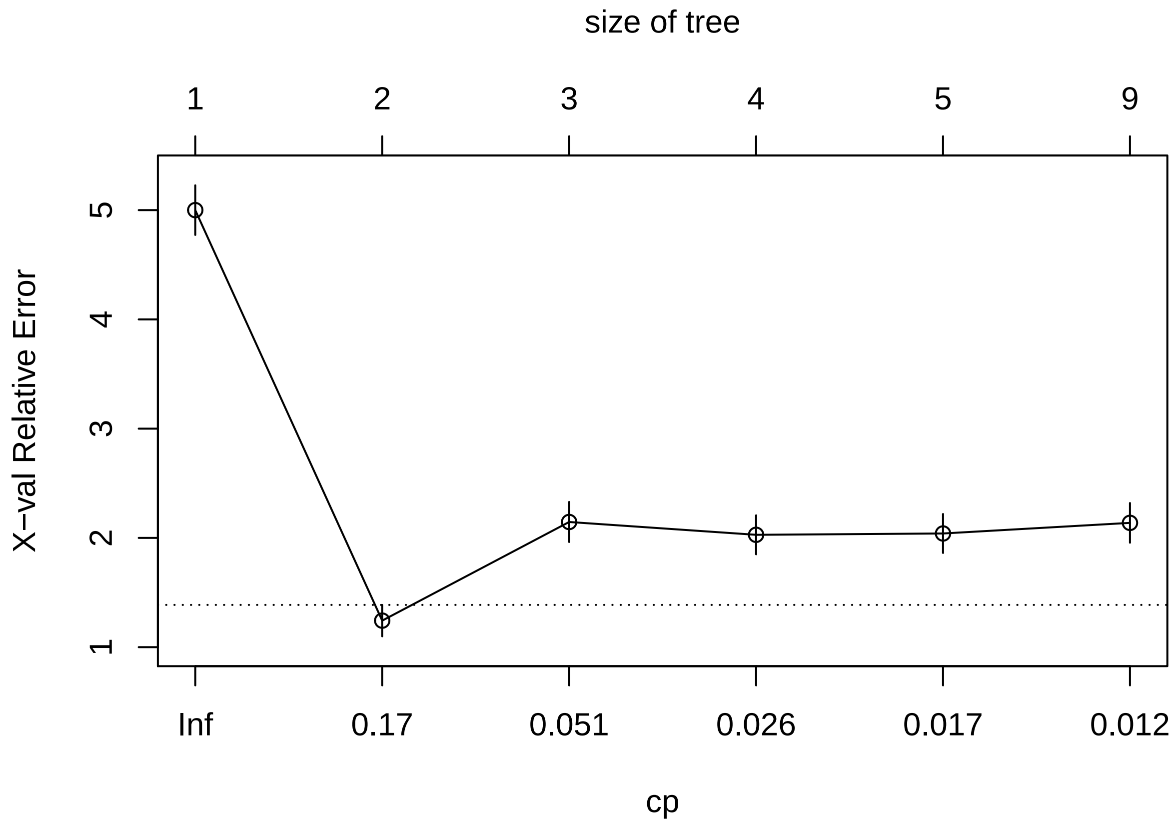


3.4 Recorte 1-ES

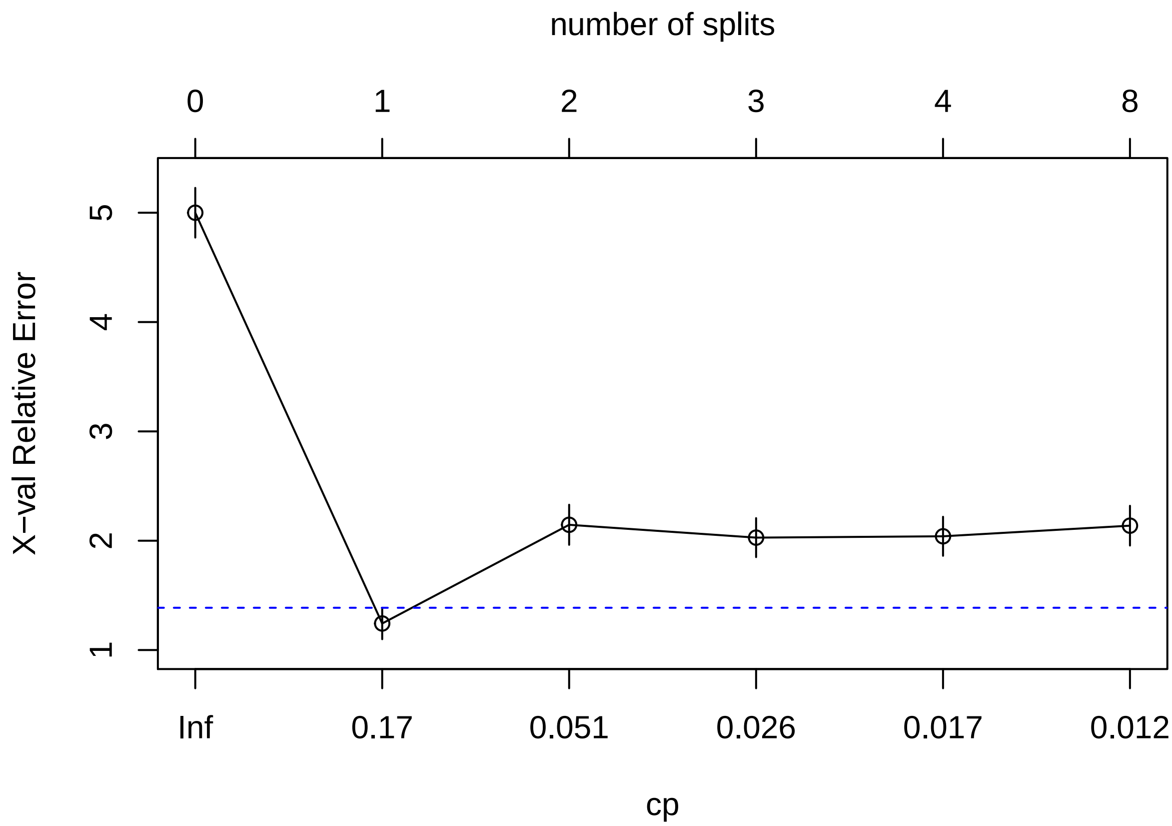
```
printcp(Default.rpart,digits=3)
```

```
##
## Classification tree:
## rpart(formula = default ~ ., data = Default, subset = ient, method = "class",
##     parms = list(loss = L, split = "gini"))
##
## Variables actually used in tree construction:
## [1] balance income
##
## Root node error: 247/504 = 0.49
##
## n= 504
##
##       CP nsplit rel error xerror  xstd
## 1 0.3765      0     1.000   5.00 0.227
## 2 0.0810      1     0.623   1.24 0.144
## 3 0.0324      2     0.543   2.15 0.183
## 4 0.0202      3     0.510   2.03 0.178
## 5 0.0148      4     0.490   2.04 0.178
## 6 0.0100      8     0.425   2.14 0.182
```

```r
plotcp(Default.rpart)
```




```r
plotcp(Default.rpart,lty=2,upper="splits",col="blue")
```




```
#Tabla
cptab=Default.rpart$cptable

#Regla 1-ES
CP1ES=min(cptab[,4])+cptab[which.min(cptab[,4]),5]
CP1ES
```

```
## [1] 1.386625
```

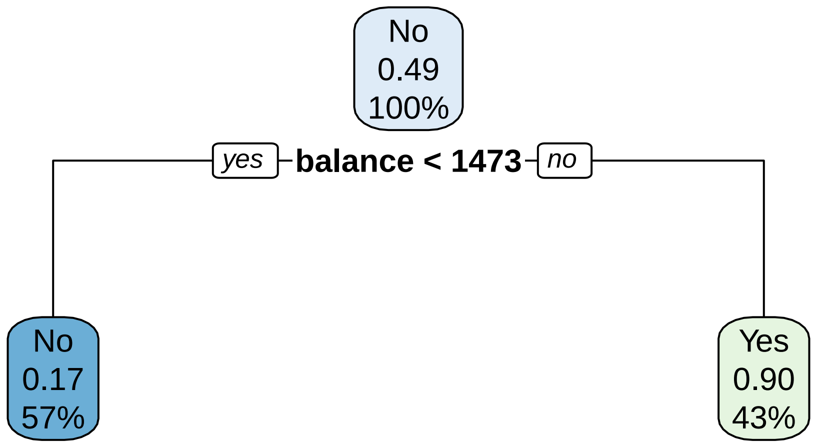
```
#cprecorte=cptab[cptab[,4]<CP1ES,][1,1]
cprecorte=cptab[cptab[,4]<CP1ES,][1]
cprecorte
```

```
##         CP
## 0.08097166
```

```
Default.rpart2=prune.rpart(Default.rpart, cp=cprecorte)
Default.rpart2
```

```
## n= 504
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 504 247 No (0.50992063 0.49007937)
##   2) balance< 1472.992 285  49 No (0.82807018 0.17192982) *
##   3) balance>=1472.992 219 105 Yes (0.09589041 0.90410959) *
```

```r
rpart.plot(Default.rpart2)
```



Finalmente tras el recorte es únicamente la variable ‘balance’ la utilizada en la clasificación.

3.5 Evaluación

```r
library(knitr)

ct = table(Default[itest,]$default,
           predict(Default.rpart2,Default[itest,],
                   type="class"))
ctm = addmargins(ct)
kable(ctm, caption = 'Matriz de confusión')
```

Table 12: Matriz de confusión

	No	Yes	Sum
No	76	7	83
Yes	15	71	86
Sum	91	78	169

```r
# Porcentaje de acierto por grupo
# Sensibilidad: % Verdaderos positivos (Yes)
```

```r
# Especificidad: % Verdaderos negativos (No)
100*diag(prop.table(ct, 1))
```

```
##       No      Yes
## 91.56627 82.55814
```

```r
# Porcentaje de acierto global
100*sum(diag(prop.table(ct)))
```

```
## [1] 86.98225
```

3.6 Área bajo la curva operativa característica

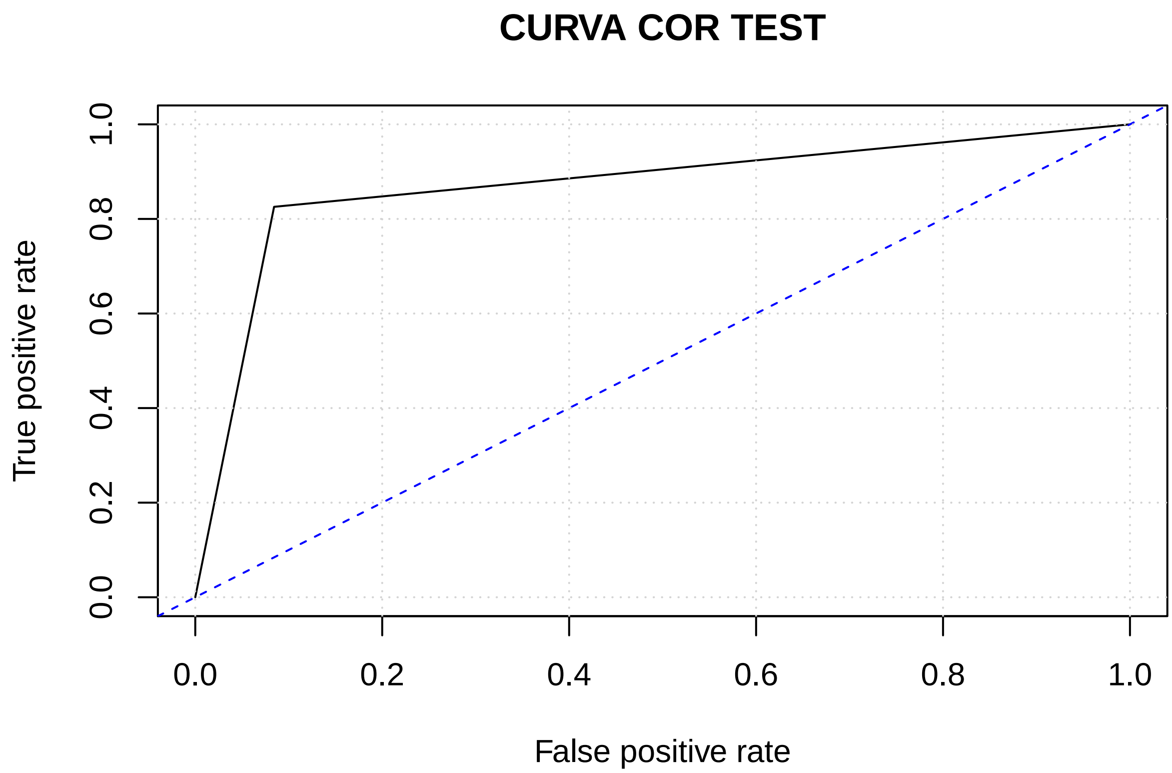
```r
#AREA BAJO LA CURVA OPERATIVA CARACTERISTICA
library(ROCR)
```

```
## Loading required package: gplots
```

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

```r
probYes = predict(Default.rpart2, Default[itest,],
                  type="prob")[,2] #Prob. yes

predobj = prediction(probYes, Default[itest,]$default)

plot(performance(predobj, "tpr","fpr"),
main="CURVA COR TEST")
abline(a=0,b=1,col="blue",lty=2)
grid()
```




```
Default.auc=as.numeric(performance(predobj,"auc")@y.values)
cat("AUC test= ",Default.auc ,"\n")
```

```
## AUC test=  0.870622
```

3.7 Coste esperado de clasificación errónea (EMC)

```
#CALCULAR EN EL CONJUNTO TEST EL INDICADOR EMC:
#EXPECTED MISCLASSIFICATION COST=
#P[NO]P[YES/NO]COSTE[YES/NO]+P[YES]P[NO/YES]COSTE[NO/YES]
ctm
```

```
##
##          No Yes Sum
##    No    76   7  83
##    Yes   15  71  86
##    Sum   91  78 169
```

```
L
```

```
##      No Yes
## No    0   5
## Yes   1   0
```

```
(P_NO = ctm[1,3]/ctm[3,3])
```

```
## [1] 0.4911243
```

```r
(P_YES = ctm[2,3]/ctm[3,3])
```

```
## [1] 0.5088757
```

```r
(P_YES_NO = ctm[1,2]/ctm[1,3])
```

```
## [1] 0.08433735
```

```r
(P_NO_YES = ctm[2,1]/ctm[2,3])
```

```
## [1] 0.1744186
```

```r
(EMC = P_NO*P_YES_NO*L[1,2]+P_YES*P_NO_YES*L[2,1])
```

```
## [1] 0.295858
```