Extra-Análisis factorial

Itzel Teodocio Olivares

2022-06-02

Se instalan las paqueteria

```
install.packages("datos")
library(datos)
```

1.- Lectura de la matriz de datos

```
C<-data.frame(datos::mtautos)
C<-as.data.frame(mtautos)
```

1.1.- Exploración de matriz

```
dim(mtautos)
```

```
## [1] 32 11
```

```
colnames(mtautos)
```

```
##  [1] "millas"       "cilindros"    "cilindrada"   "caballos"     "eje"
##  [6] "peso"         "velocidad"    "forma"        "transmision"  "cambios"
## [11] "carburadores"
```

```
str(mtautos)
```

```
## 'data.frame':    32 obs. of  11 variables:
##  $ millas      : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
##  $ cilindros   : num  6 6 4 6 8 6 8 4 4 6 ...
##  $ cilindrada  : num  160 160 108 258 360 ...
##  $ caballos    : num  110 110 93 110 175 105 245 62 95 123 ...
##  $ eje         : num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
##  $ peso        : num  2.62 2.88 2.32 3.21 3.44 ...
##  $ velocidad   : num  16.5 17 18.6 19.4 17 ...
##  $ forma       : num  0 0 1 1 0 1 0 1 1 1 ...
##  $ transmision : num  1 1 1 0 0 0 0 0 0 0 ...
##  $ cambios     : num  4 4 4 3 3 3 3 4 4 4 ...
##  $ carburadores: num  4 4 1 1 2 1 4 2 2 4 ...
```

#2.- Quitar los espacios de los nombres

```
#colnames(C)[1]="Life.Exp"
#colnames(C)[11]= "HS.Grad"
```

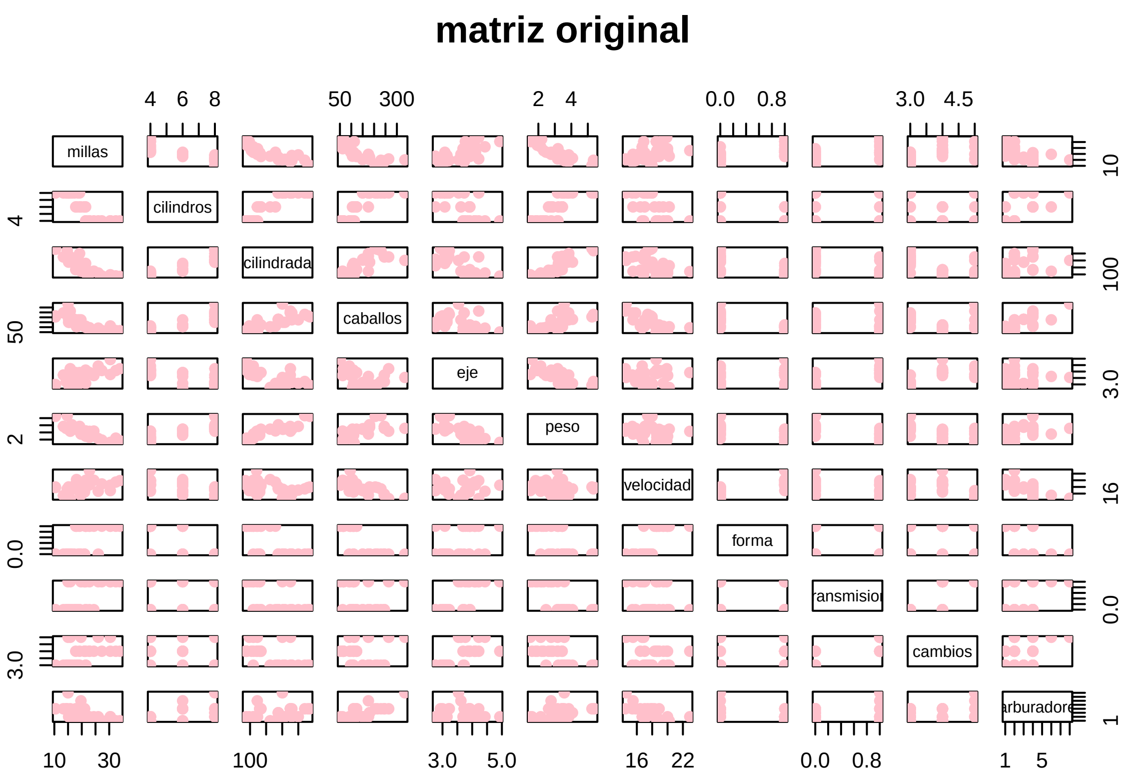
esto no va tus nombres estan bien

#3.- Separa n (autos) y p (variables)

```
n<-dim(C)[1]
p<-dim(C)[11]
```

4.- Generacion de un scater plot para la visualización de variables originales

```
pairs(C, col="pink", pch=19, main="matriz original")
```



Transformación de alguna varibles

#1.- Aplicamos logaritmo para las columnas 1,3 y 8

```
#C[,1]<-log(C[,1])
#colnames(C)[1]<-"Log-Population"
#C[,3]<-log(C[,3])
#colnames(C)[3]<-"Log-Illiteracy"
#C[,8]<-log(C[,8])
#colnames(C)[8]<-"Log-Area"
```

las cantidades en tus variables son pequeñas no hay necesidad de sacarles logartimo; el logarimo lo utilizamos cuando las cantidades en las variables son muy grandes. # Grafico scater para la visualizacion de la # matriz original con 3 variables que se incluyeron

```
#pairs(C,col="red", pch=19, main="Matriz original")
```

ya no seria necesaari esto por lo antes mencionado. # Nota: Como las variables tiene diferentes unidades # de medida, se va a implementar la matriz de # correlaciones para estimar la matriz de carga

Reduccion de la dimensionalidad

Análsis Factorial de componentes principales (PCFA)

#1.- Calcular la matriz de medias y de correlaciones # Matriz de medias

```
mu<-colMeans(C)
mu
```

```
##      millas    cilindros   cilindrada     caballos          eje         peso 
##   20.090625     6.187500   230.721875   146.687500     3.596563     3.217250 
##   velocidad        forma  transmision      cambios carburadores 
##   17.848750     0.437500     0.406250     3.687500     2.812500
```

#Matriz de correlaciones

```
R<-cor(C)
R
```

```
##                  millas  cilindros cilindrada   caballos         eje       peso
## millas        1.0000000 -0.8521620 -0.8475514 -0.7761684  0.68117191 -0.8676594
## cilindros    -0.8521620  1.0000000  0.9020329  0.8324475 -0.69993811  0.7824958
## cilindrada   -0.8475514  0.9020329  1.0000000  0.7909486 -0.71021393  0.8879799
## caballos     -0.7761684  0.8324475  0.7909486  1.0000000 -0.44875912  0.6587479
## eje           0.6811719 -0.6999381 -0.7102139 -0.4487591  1.00000000 -0.7124406
## peso         -0.8676594  0.7824958  0.8879799  0.6587479 -0.71244065  1.0000000
## velocidad     0.4186840 -0.5912421 -0.4336979 -0.7082234  0.09120476 -0.1747159
## forma         0.6640389 -0.8108118 -0.7104159 -0.7230967  0.44027846 -0.5549157
## transmision   0.5998324 -0.5226070 -0.5912270 -0.2432043  0.71271113 -0.6924953
## cambios       0.4802848 -0.4926866 -0.5555692 -0.1257043  0.69961013 -0.5832870
## carburadores -0.5509251  0.5269883  0.3949769  0.7498125 -0.09078980  0.4276059
##                velocidad      forma transmision    cambios carburadores
## millas        0.41868403  0.6640389  0.59983243  0.4802848  -0.55092507
## cilindros    -0.59124207 -0.8108118 -0.52260705 -0.4926866   0.52698829
## cilindrada   -0.43369788 -0.7104159 -0.59122704 -0.5555692   0.39497686
## caballos     -0.70822339 -0.7230967 -0.24320426 -0.1257043   0.74981247
## eje           0.09120476  0.4402785  0.71271113  0.6996101  -0.09078980
## peso         -0.17471588 -0.5549157 -0.69249526 -0.5832870   0.42760594
## velocidad     1.00000000  0.7445354 -0.22986086 -0.2126822  -0.65624923
## forma         0.74453544  1.0000000  0.16834512  0.2060233  -0.56960714
## transmision  -0.22986086  0.1683451  1.00000000  0.7940588   0.05753435
## cambios      -0.21268223  0.2060233  0.79405876  1.0000000   0.27407284
## carburadores -0.65624923 -0.5696071  0.05753435  0.2740728   1.00000000
```

2.- Reducción de la dimensionalidad mediante

Análisis factorial de componentes principales (PCFA).

1.- Calcular los valores y vectores propios.

```
eR<-eigen(R)
```

2.- Valores propios

```
eigen.val<-eR$values
eigen.val
```

```
##  [1] 6.60840025 2.65046789 0.62719727 0.26959744 0.22345110 0.21159612
##  [7] 0.13526199 0.12290143 0.07704665 0.05203544 0.02204441
```

3.- Vectores propios

```
eigen.vec<-eR$vectors
eigen.vec
```

```
##             [,1]        [,2]        [,3]         [,4]        [,5]        [,6]
##  [1,]  0.3625305 -0.01612440 -0.22574419 -0.022540255 -0.10284468  0.10879743
##  [2,] -0.3739160 -0.04374371 -0.17531118 -0.002591838 -0.05848381 -0.16855369
##  [3,] -0.3681852  0.04932413 -0.06148414  0.256607885 -0.39399530  0.33616451
##  [4,] -0.3300569 -0.24878402  0.14001476 -0.067676157 -0.54004744 -0.07143563
##  [5,]  0.2941514 -0.27469408  0.16118879  0.854828743 -0.07732727 -0.24449705
##  [6,] -0.3461033  0.14303825  0.34181851  0.245899314  0.07502912  0.46493964
##  [7,]  0.2004563  0.46337482  0.40316904  0.068076532  0.16466591  0.33048032
##  [8,]  0.3065113  0.23164699  0.42881517 -0.214848616 -0.59953955 -0.19401702
##  [9,]  0.2349429 -0.42941765 -0.20576657 -0.030462908 -0.08978128  0.57081745
## [10,]  0.2069162 -0.46234863  0.28977993 -0.264690521 -0.04832960  0.24356284
## [11,] -0.2140177 -0.41357106  0.52854459 -0.126789179  0.36131875 -0.18352168
##               [,7]         [,8]         [,9]       [,10]        [,11]
##  [1,]  0.367723810  0.754091423  0.235701617  0.13928524  0.124895628
##  [2,]  0.057277736  0.230824925  0.054035270 -0.84641949  0.140695441
##  [3,]  0.214303077 -0.001142134  0.198427848  0.04937979 -0.660606481
##  [4,] -0.001495989  0.222358441 -0.575830072  0.24782351  0.256492062
##  [5,]  0.021119857 -0.032193501 -0.046901228 -0.10149369  0.039530246
##  [6,] -0.020668302  0.008571929  0.359498251  0.09439426  0.567448697
##  [7,]  0.050010522  0.231840021 -0.528377185 -0.27067295 -0.181361780
##  [8,] -0.265780836 -0.025935128  0.358582624 -0.15903909 -0.008414634
##  [9,] -0.587305101  0.059746952 -0.047403982 -0.17778541 -0.029823537
## [10,]  0.605097617 -0.336150240 -0.001735039 -0.21382515  0.053507085
## [11,] -0.174603192  0.395629107  0.170640677  0.07225950 -0.319594676
```

4.- Calcular la proporcion de variabilidad

```
prop.var<-eigen.val/sum(eigen.val)
prop.var
```

```
##  [1] 0.600763659 0.240951627 0.057017934 0.024508858 0.020313737 0.019236011
##  [7] 0.012296544 0.011172858 0.007004241 0.004730495 0.002004037
```

5.- Calcular la proporcion de variabilidad acumulada

```
prop.var.acum<-cumsum(eigen.val)/sum(eigen.val)
prop.var.acum
```

```
##  [1] 0.6007637 0.8417153 0.8987332 0.9232421 0.9435558 0.9627918 0.9750884
##  [8] 0.9862612 0.9932655 0.9979960 1.0000000
```

Estimacion de la matriz de carga

Nota: se estima la matriz de carga usando los

autovalores y autovectores.

se aplica la rotación varimax

Primera estimación de Lamda mayuscula

se calcula multiplicando la matriz de los

3 primeros autovectores por la matriz diagonal

formada por la raiz cuadrada de los primeros

3 autovalores.

```
L.est.1<-eigen.vec[,1:3] %*% diag(sqrt(eigen.val[1:3]))
L.est.1
```

```
##             [,1]        [,2]        [,3]
##  [1,]  0.9319502 -0.02625094 -0.17877989
##  [2,] -0.9612188 -0.07121589 -0.13883907
##  [3,] -0.9464866  0.08030095 -0.04869285
##  [4,] -0.8484710 -0.40502680  0.11088579
##  [5,]  0.7561693 -0.44720905  0.12765473
##  [6,] -0.8897212  0.23286996  0.27070586
##  [7,]  0.5153093  0.75438614  0.31929289
##  [8,]  0.7879428  0.37712727  0.33960355
##  [9,]  0.6039632 -0.69910300 -0.16295845
## [10,]  0.5319156 -0.75271549  0.22949350
## [11,] -0.5501711 -0.67330434  0.41858505
```

Rotación varimax

```
L.est.1.var<-varimax(L.est.1)
L.est.1.var
```

```
## $loadings
##
## Loadings:
##       [,1]   [,2]   [,3]
##  [1,]  0.542 -0.664  0.408
##  [2,] -0.342  0.617 -0.671
##  [3,] -0.345  0.715 -0.523
##  [4,] -0.629  0.296 -0.642
##  [5,]        -0.848  0.258
##  [6,] -0.512  0.782 -0.213
##  [7,]  0.284  0.182  0.907
##  [8,]  0.232 -0.279  0.864
##  [9,]  0.108 -0.921 -0.145
## [10,] -0.259 -0.914
## [11,] -0.854 -0.107 -0.437
##
##                 [,1]  [,2]  [,3]
## SS loadings    2.132 4.557 3.197
## Proportion Var 0.194 0.414 0.291
## Cumulative Var 0.194 0.608 0.899
##
## $rotmat
##            [,1]        [,2]      [,3]
## [1,]  0.4405412 -0.69407713 0.5693684
## [2,]  0.4125491  0.71981583 0.5582727
## [3,] -0.7973247 -0.01104967 0.6034494
```

Estimación de la matriz de los errores

#1.- Estimación de la matriz de perturbaciones

```
Psi.est.1<-diag(diag(R-as.matrix(L.est.1.var$loadings)%*% t(as.matrix(L.est.1.var$loadings))))
Psi.est.1
```

```
##             [,1]       [,2]       [,3]      [,4]      [,5]       [,6]
##  [1,] 0.09881739 0.00000000 0.00000000 0.0000000 0.0000000 0.00000000
##  [2,] 0.00000000 0.05171045 0.00000000 0.0000000 0.0000000 0.00000000
##  [3,] 0.00000000 0.00000000 0.09534379 0.0000000 0.0000000 0.00000000
##  [4,] 0.00000000 0.00000000 0.00000000 0.1037545 0.0000000 0.00000000
##  [5,] 0.00000000 0.00000000 0.00000000 0.0000000 0.2119163 0.00000000
##  [6,] 0.00000000 0.00000000 0.00000000 0.0000000 0.0000000 0.08088614
##  [7,] 0.00000000 0.00000000 0.00000000 0.0000000 0.0000000 0.00000000
##  [8,] 0.00000000 0.00000000 0.00000000 0.0000000 0.0000000 0.00000000
##  [9,] 0.00000000 0.00000000 0.00000000 0.0000000 0.0000000 0.00000000
## [10,] 0.00000000 0.00000000 0.00000000 0.0000000 0.0000000 0.00000000
## [11,] 0.00000000 0.00000000 0.00000000 0.0000000 0.0000000 0.00000000
##             [,7]      [,8]     [,9]     [,10]      [,11]
##  [1,] 0.00000000 0.0000000 0.000000 0.0000000 0.00000000
##  [2,] 0.00000000 0.0000000 0.000000 0.0000000 0.00000000
##  [3,] 0.00000000 0.0000000 0.000000 0.0000000 0.00000000
##  [4,] 0.00000000 0.0000000 0.000000 0.0000000 0.00000000
##  [5,] 0.00000000 0.0000000 0.000000 0.0000000 0.00000000
##  [6,] 0.00000000 0.0000000 0.000000 0.0000000 0.00000000
##  [7,] 0.06340992 0.0000000 0.000000 0.0000000 0.00000000
```

```
##  [8,] 0.00000000 0.1215906 0.000000 0.0000000 0.00000000
##  [9,] 0.00000000 0.0000000 0.119928 0.0000000 0.00000000
## [10,] 0.00000000 0.0000000 0.000000 0.0978179 0.00000000
## [11,] 0.00000000 0.0000000 0.000000 0.0000000 0.06875959
```

2.- Se utiliza el método Análisis de factor principal (PFA) para estimación de autovalores y autovectores

```r
RP<-R-Psi.est.1
RP
```

```
##                 millas  cilindros cilindrada   caballos         eje       peso
## millas       0.9011826 -0.8521620 -0.8475514 -0.7761684  0.68117191 -0.8676594
## cilindros   -0.8521620  0.9482895  0.9020329  0.8324475 -0.69993811  0.7824958
## cilindrada  -0.8475514  0.9020329  0.9046562  0.7909486 -0.71021393  0.8879799
## caballos    -0.7761684  0.8324475  0.7909486  0.8962455 -0.44875912  0.6587479
## eje          0.6811719 -0.6999381 -0.7102139 -0.4487591  0.78808373 -0.7124406
## peso        -0.8676594  0.7824958  0.8879799  0.6587479 -0.71244065  0.9191139
## velocidad    0.4186840 -0.5912421 -0.4336979 -0.7082234  0.09120476 -0.1747159
## forma        0.6640389 -0.8108118 -0.7104159 -0.7230967  0.44027846 -0.5549157
## transmision  0.5998324 -0.5226070 -0.5912270 -0.2432043  0.71271113 -0.6924953
## cambios      0.4802848 -0.4926866 -0.5555692 -0.1257043  0.69961013 -0.5832870
## carburadores -0.5509251  0.5269883  0.3949769  0.7498125 -0.09078980  0.4276059
##               velocidad      forma transmision    cambios carburadores
## millas       0.41868403  0.6640389  0.59983243  0.4802848  -0.55092507
## cilindros   -0.59124207 -0.8108118 -0.52260705 -0.4926866   0.52698829
## cilindrada  -0.43369788 -0.7104159 -0.59122704 -0.5555692   0.39497686
## caballos    -0.70822339 -0.7230967 -0.24320426 -0.1257043   0.74981247
## eje          0.09120476  0.4402785  0.71271113  0.6996101  -0.09078980
## peso        -0.17471588 -0.5549157 -0.69249526 -0.5832870   0.42760594
## velocidad    0.93659008  0.7445354 -0.22986086 -0.2126822  -0.65624923
## forma        0.74453544  0.8784094  0.16834512  0.2060233  -0.56960714
## transmision -0.22986086  0.1683451  0.88007201  0.7940588   0.05753435
## cambios     -0.21268223  0.2060233  0.79405876  0.9021821   0.27407284
## carburadores -0.65624923 -0.5696071  0.05753435  0.2740728   0.93124041
```

Calculo de la matriz de autovalores y autovectores

```r
eRP<-eigen(RP)
```

Autovalores

```r
eigen.val.RP<-eRP$values
eigen.val.RP
```

```
##  [1]  6.50827377  2.55158506  0.53989628  0.12815610  0.12047514  0.07098103
##  [7]  0.03536869  0.02770690 -0.01082465 -0.01918291 -0.06636999
```

Autovectores

```
eigen.vec.RP<-eRP$vectors
eigen.val.RP
```

```
##  [1]  6.50827377  2.55158506  0.53989628  0.12815610  0.12047514  0.07098103
##  [7]  0.03536869  0.02770690 -0.01082465 -0.01918291 -0.06636999
```

Proporcion de variabilidad

```
prop.var.RP<-eigen.val.RP/ sum(eigen.val.RP)
prop.var.RP
```

```
##  [1]  0.658328010  0.258099148  0.054611846  0.012963307  0.012186358
##  [6]  0.007179907  0.003577631  0.002802622 -0.001094940 -0.001940399
## [11] -0.006713489
```

Proporcion de variabilidad acumulada

```
prop.var.RP.acum<-cumsum(eigen.val.RP)/ sum(eigen.val.RP)
prop.var.RP.acum
```

```
##  [1] 0.6583280 0.9164272 0.9710390 0.9840023 0.9961887 1.0033686 1.0069462
##  [8] 1.0097488 1.0086539 1.0067135 1.0000000
```

Estimación de la matriz de cargas

con rotación varimax

```
L.est.2<-eigen.vec.RP[,1:3] %*% diag(sqrt(eigen.val.RP[1:3]))
L.est.2
```

```
##             [,1]        [,2]        [,3]
##  [1,]  0.9250245 -0.03126684 -0.15888375
##  [2,] -0.9611279 -0.06563034 -0.14100963
##  [3,] -0.9398412  0.08492069 -0.05055728
##  [4,] -0.8425167 -0.39170851  0.09440621
##  [5,]  0.7364339 -0.42327193  0.09226618
##  [6,] -0.8851139  0.23587577  0.25271063
##  [7,]  0.5159435  0.74772017  0.31233902
##  [8,]  0.7801603  0.36202748  0.29373198
##  [9,]  0.5960270 -0.68373849 -0.13672189
## [10,]  0.5263604 -0.74209587  0.21354024
## [11,] -0.5502616 -0.66520866  0.40348209
```

Rotacion varimax

```
L.est.2.var<-varimax(L.est.2)
```

Estimación de la matriz de covarianzas de los errores.

```
Psi.est.2<-diag(diag(R-as.matrix(L.est.2.var$loadings)%*% t(as.matrix(L.est.2.var$loadings))))
Psi.est.2
```

```
##          [,1]       [,2]     [,3]      [,4]      [,5]       [,6]       [,7]
##  [1,] 0.118108 0.00000000 0.000000 0.0000000 0.0000000 0.00000000 0.00000000
##  [2,] 0.000000 0.05204206 0.000000 0.0000000 0.0000000 0.00000000 0.00000000
##  [3,] 0.000000 0.00000000 0.106931 0.0000000 0.0000000 0.00000000 0.00000000
##  [4,] 0.000000 0.00000000 0.000000 0.1278176 0.0000000 0.00000000 0.00000000
##  [5,] 0.000000 0.00000000 0.000000 0.0000000 0.2699929 0.00000000 0.00000000
##  [6,] 0.000000 0.00000000 0.000000 0.0000000 0.0000000 0.09707329 0.00000000
##  [7,] 0.000000 0.00000000 0.000000 0.0000000 0.0000000 0.00000000 0.07716116
##  [8,] 0.000000 0.00000000 0.000000 0.0000000 0.0000000 0.00000000 0.00000000
##  [9,] 0.000000 0.00000000 0.000000 0.0000000 0.0000000 0.00000000 0.00000000
## [10,] 0.000000 0.00000000 0.000000 0.0000000 0.0000000 0.00000000 0.00000000
## [11,] 0.000000 0.00000000 0.000000 0.0000000 0.0000000 0.00000000 0.00000000
##            [,8]      [,9]    [,10]     [,11]
##  [1,] 0.0000000 0.0000000 0.000000 0.0000000
##  [2,] 0.0000000 0.0000000 0.000000 0.0000000
##  [3,] 0.0000000 0.0000000 0.000000 0.0000000
##  [4,] 0.0000000 0.0000000 0.000000 0.0000000
##  [5,] 0.0000000 0.0000000 0.000000 0.0000000
##  [6,] 0.0000000 0.0000000 0.000000 0.0000000
##  [7,] 0.0000000 0.0000000 0.000000 0.0000000
##  [8,] 0.1740075 0.0000000 0.000000 0.0000000
##  [9,] 0.0000000 0.1585607 0.000000 0.0000000
## [10,] 0.0000000 0.0000000 0.126639 0.0000000
## [11,] 0.0000000 0.0000000 0.000000 0.0919118
```

Obtencion de los scores de ambos métodos

PCFA

```
FS.est.1<-scale(C)%*% as.matrix(L.est.1.var$loadings)
FS.est.1
```

```
##                          [,1]       [,2]       [,3]
## Mazda RX4          -0.0410256 -3.1506899 -0.8884768
## Mazda RX4 Wag      -0.0854961 -2.8898642 -0.6597321
## Datsun 710          3.3448166 -4.7100357  4.0215781
## Hornet 4 Drive      1.9939958  2.1792130  2.4991646
## Hornet Sportabout  -0.9971326  4.3474154 -2.7032108
## Valiant             1.8024768  3.1134384  2.6501998
## Duster 360         -3.3428720  4.8900060 -4.7959068
## Merc 240D           2.6729260 -1.9276237  4.7156274
## Merc 230            2.7439166 -1.7446688  5.9150571
## Merc 280           -0.2339269 -0.7503930  1.6054121
## Merc 280C          -0.2646498 -0.5351551  1.8152385
## Merc 450SE         -1.8204128  4.7390614 -2.8033838
## Merc 450SL         -1.5298686  4.3885003 -2.5668152
## Merc 450SLC        -1.6814718  4.7004501 -2.5168213
## Cadillac Fleetwood -4.2018891  7.7980184 -4.5729608
```

```
## Lincoln Continental -4.3705337  7.7837280 -4.7013419
## Chrysler Imperial   -4.0676656  6.7908190 -4.5411618
## Fiat 128             4.7573865 -6.4262929  5.6219890
## Honda Civic          4.4152463 -8.2629727  5.4152887
## Toyota Corolla       5.1941016 -7.1097770  6.1303991
## Toyota Corona        3.4263878 -0.9002863  4.7132634
## Dodge Challenger    -1.0384992  5.0168149 -2.7900049
## AMC Javelin         -0.8796376  4.3262084 -2.3262352
## Camaro Z28          -3.5689167  4.2896344 -4.8477017
## Pontiac Firebird    -1.2766789  4.9610423 -2.9450836
## Fiat X1-9            4.3457889 -6.1327204  5.0437596
## Porsche 914-2        2.1185070 -7.0091824  1.5982447
## Lotus Europa         3.1446320 -7.5320084  3.4307529
## Ford Pantera L      -3.7592309 -1.6246832 -5.1731174
## Ferrari Dino        -2.3768080 -3.7181529 -2.6891172
## Maserati Bora       -6.7112287 -0.3758850 -7.1267213
## Volvo 142E           2.2877629 -4.5239583  3.4718175
```

PFA

```
FS.est.2<-scale(C)%*% as.matrix (L.est.2.var$loadings)
FS.est.2
```

```
##                           [,1]       [,2]       [,3]
## Mazda RX4           -0.1077303 -3.0993659 -0.8315649
## Mazda RX4 Wag       -0.1509015 -2.8387206 -0.6096052
## Datsun 710           3.0480740 -4.7362985  4.1391518
## Hornet 4 Drive       1.9041708  2.0776248  2.5340651
## Hornet Sportabout   -0.8333251  4.3124349 -2.7060136
## Valiant              1.7203014  2.9941992  2.6759485
## Duster 360          -3.0203632  4.9082231 -4.8995476
## Merc 240D            2.4439804 -1.9854245  4.7510058
## Merc 230             2.5026052 -1.7917167  5.9315159
## Merc 280            -0.2332308 -0.7345388  1.5135711
## Merc 280C           -0.2624221 -0.5199431  1.7176870
## Merc 450SE          -1.6214524  4.7169260 -2.8517014
## Merc 450SL          -1.3565535  4.3620207 -2.6003060
## Merc 450SLC         -1.4951107  4.6752591 -2.5611735
## Cadillac Fleetwood  -3.8072152  7.8032273 -4.7306551
## Lincoln Continental -3.9612694  7.7967285 -4.8688271
## Chrysler Imperial   -3.6844879  6.8158259 -4.6958614
## Fiat 128             4.3300998 -6.4606071  5.7966780
## Honda Civic          4.0105895 -8.2370654  5.5567925
## Toyota Corolla       4.7287845 -7.1424163  6.3213950
## Toyota Corona        3.1931881 -0.9877093  4.7885040
## Dodge Challenger    -0.8782235  4.9578913 -2.7882747
## AMC Javelin         -0.7296103  4.2877292 -2.3312551
## Camaro Z28          -3.2183860  4.3426408 -4.9770702
## Pontiac Firebird    -1.0873987  4.9252398 -2.9593636
## Fiat X1-9            3.9602386 -6.1610267  5.2012300
## Porsche 914-2        1.8677790 -6.9526900  1.7349393
## Lotus Europa         2.8053883 -7.5327158  3.5596593
## Ford Pantera L      -3.4921201 -1.4704866 -5.2743876
```

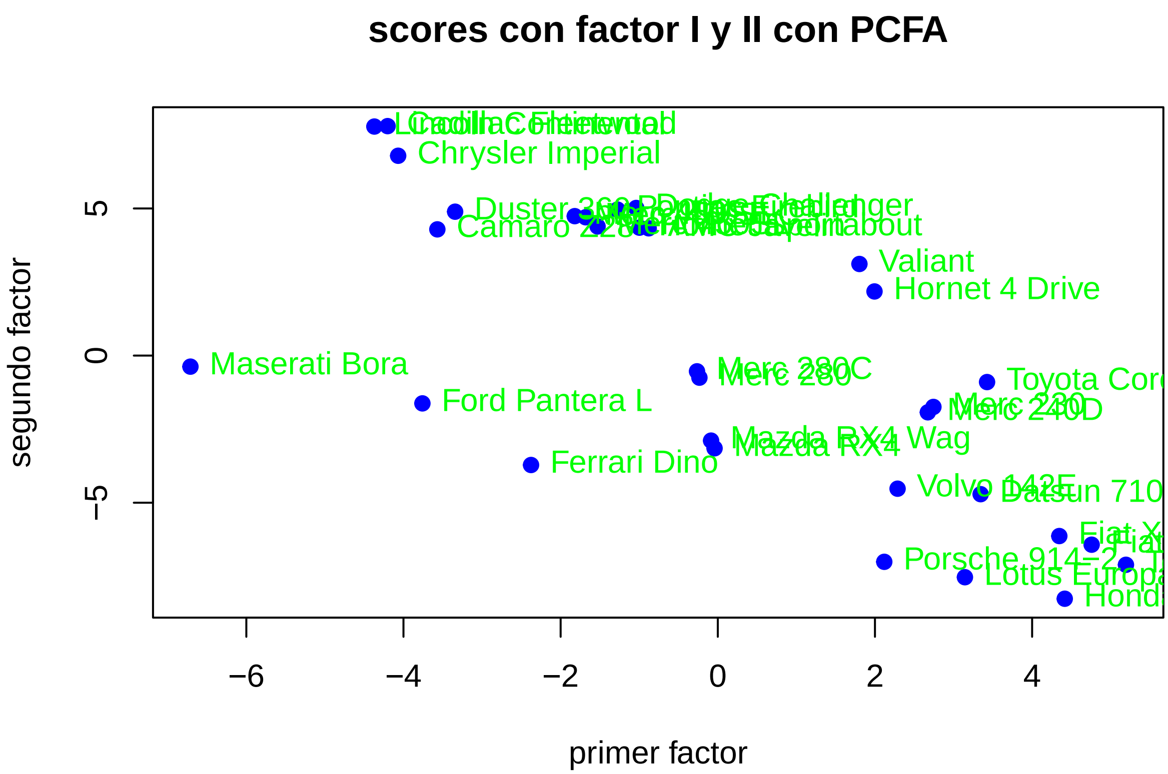
```
## Ferrari Dino        -2.3292993 -3.6166195 -2.7255218
## Maserati Bora       -6.3077997 -0.1945857 -7.3427546
## Volvo 142E           2.0617005 -4.5140402  3.5317402
```

graficamos ambos scores

```r
par(mfrow=c(2,1))
```

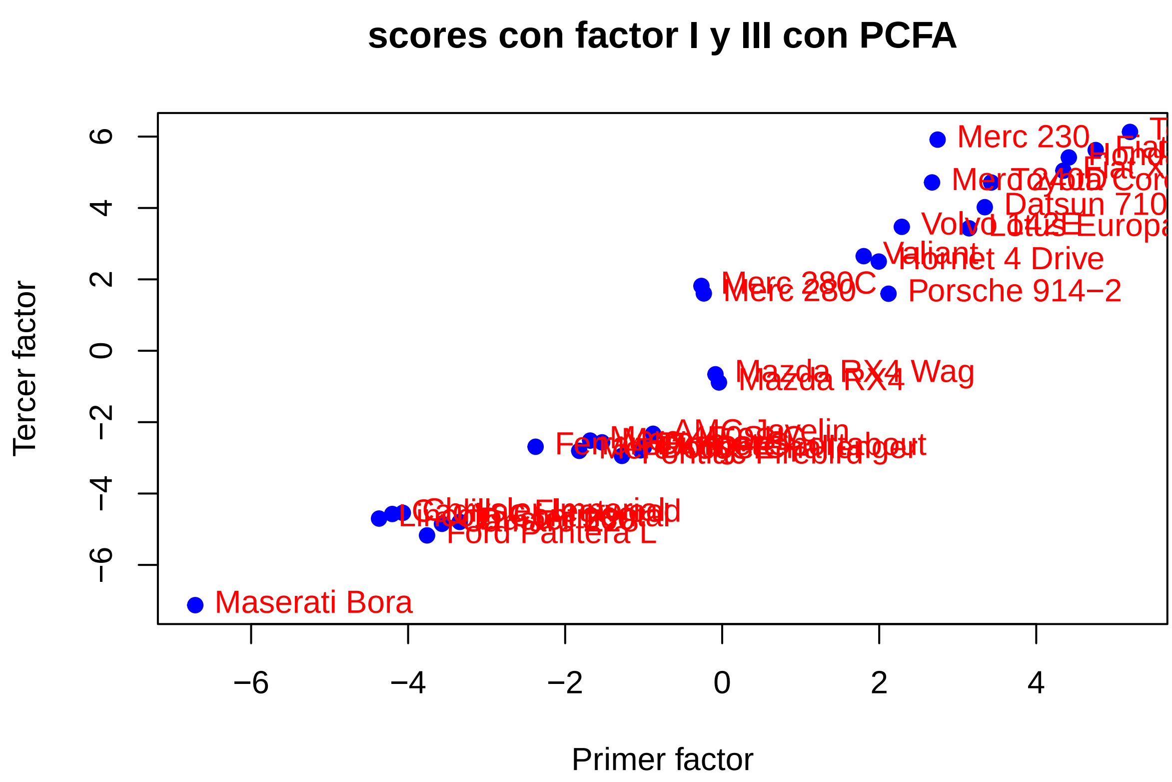
Factor I y II

```r
pl1<-plot(FS.est.1[,1], FS.est.1[,2], xlab="primer factor",
          ylab="segundo factor", main="scores con factor I y II con PCFA",
          pch=19, col="blue")
text(FS.est.1[,1], FS.est.1[,2], labels = rownames(C), pos=4, col="green")
```



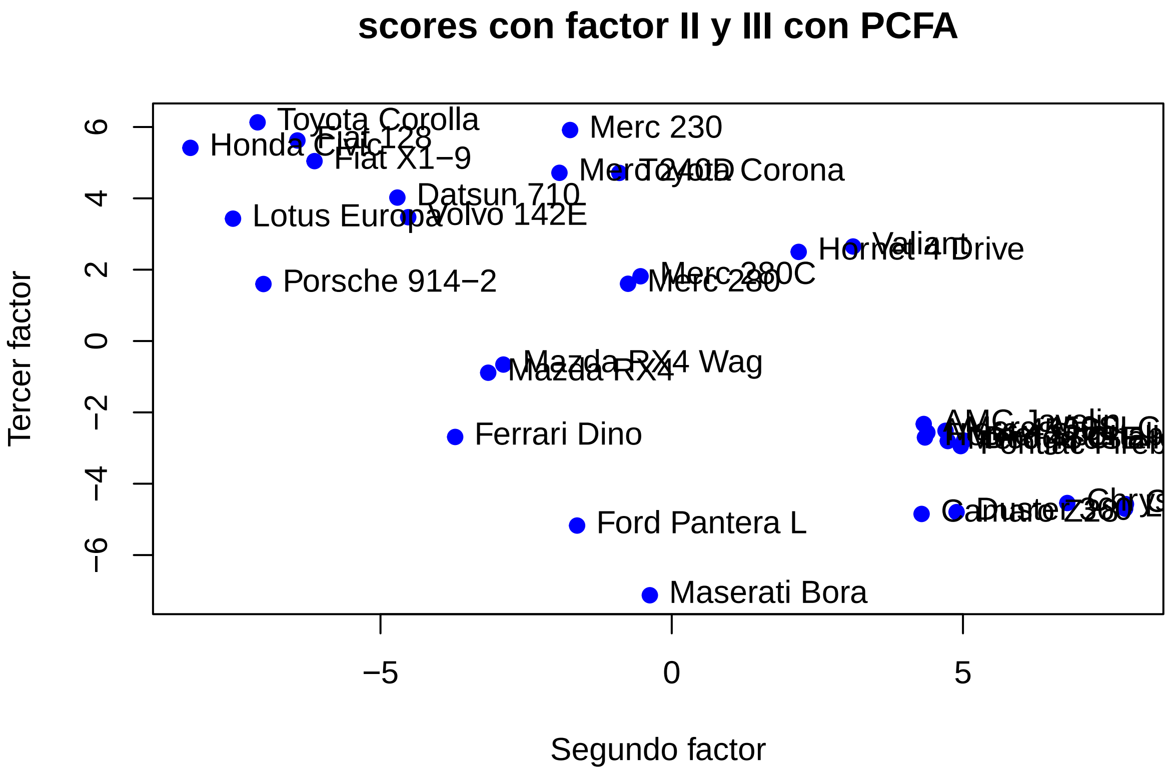
Factor I y III

```r
pl2<-plot(FS.est.1[,1], FS.est.1[,3], xlab="Primer factor",
          ylab="Tercer factor", main="scores con factor I y III con PCFA",
          pch=19, col="blue")
text(FS.est.1[,1], FS.est.1[,3], labels = rownames(C), pos=4, col="red")
```



Factor II y III

```
pl3<-plot(FS.est.1[,2], FS.est.1[,3], xlab="Segundo factor",
          ylab="Tercer factor", main="scores con factor II y III con PCFA",
          pch=19, col="blue")
text(FS.est.1[,2], FS.est.1[,3], labels = rownames(C), pos=4, col="black")
```


scores con factor II y III con PCFA
Tercer factor
Segundo factor
Toyota Corolla
Honda Civic
Fiat 128
Fiat X1−9
Merc 230
Merc 240D
Toyota Corona
Datsun 710
Lotus Europa
Volvo 142E
Valiant
Hornet 4 Drive
Porsche 914−2
Merc 280C
Merc 280
Mazda RX4 Wag
Mazda RX4
AMC Javelin
Ferrari Dino
Pontiac Firebird
Ford Pantera L
Camaro Z28
Duster 360
Maserati Bora