# Análisis Factorial, matriz precargada en R_punto extra

Medel Colorado Yoselin Merari

2022-06-05

## Análisis factorial matriz precargada en R_Punto extra

#Instalar paquete

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

## Selección de datos

```
datos::mtautos
```

```
##                     millas cilindros cilindrada caballos  eje  peso velocidad
## Mazda RX4             21.0         6      160.0      110 3.90 2.620     16.46
## Mazda RX4 Wag         21.0         6      160.0      110 3.90 2.875     17.02
## Datsun 710            22.8         4      108.0       93 3.85 2.320     18.61
## Hornet 4 Drive        21.4         6      258.0      110 3.08 3.215     19.44
## Hornet Sportabout     18.7         8      360.0      175 3.15 3.440     17.02
## Valiant               18.1         6      225.0      105 2.76 3.460     20.22
## Duster 360            14.3         8      360.0      245 3.21 3.570     15.84
## Merc 240D             24.4         4      146.7       62 3.69 3.190     20.00
## Merc 230              22.8         4      140.8       95 3.92 3.150     22.90
## Merc 280              19.2         6      167.6      123 3.92 3.440     18.30
## Merc 280C             17.8         6      167.6      123 3.92 3.440     18.90
## Merc 450SE            16.4         8      275.8      180 3.07 4.070     17.40
## Merc 450SL            17.3         8      275.8      180 3.07 3.730     17.60
## Merc 450SLC           15.2         8      275.8      180 3.07 3.780     18.00
## Cadillac Fleetwood    10.4         8      472.0      205 2.93 5.250     17.98
## Lincoln Continental   10.4         8      460.0      215 3.00 5.424     17.82
## Chrysler Imperial     14.7         8      440.0      230 3.23 5.345     17.42
## Fiat 128              32.4         4       78.7       66 4.08 2.200     19.47
## Honda Civic           30.4         4       75.7       52 4.93 1.615     18.52
## Toyota Corolla        33.9         4       71.1       65 4.22 1.835     19.90
## Toyota Corona         21.5         4      120.1       97 3.70 2.465     20.01
## Dodge Challenger      15.5         8      318.0      150 2.76 3.520     16.87
## AMC Javelin           15.2         8      304.0      150 3.15 3.435     17.30
## Camaro Z28            13.3         8      350.0      245 3.73 3.840     15.41
## Pontiac Firebird      19.2         8      400.0      175 3.08 3.845     17.05
## Fiat X1-9             27.3         4       79.0       66 4.08 1.935     18.90
## Porsche 914-2         26.0         4      120.3       91 4.43 2.140     16.70
## Lotus Europa          30.4         4       95.1      113 3.77 1.513     16.90
## Ford Pantera L        15.8         8      351.0      264 4.22 3.170     14.50
## Ferrari Dino          19.7         6      145.0      175 3.62 2.770     15.50
```

```
## Maserati Bora         15.0         8      301.0      335 3.54 3.570     14.60
## Volvo 142E            21.4         4      121.0      109 4.11 2.780     18.60
##                     forma transmision cambios carburadores
## Mazda RX4               0            1       4            4
## Mazda RX4 Wag           0            1       4            4
## Datsun 710              1            1       4            1
## Hornet 4 Drive          1            0       3            1
## Hornet Sportabout       0            0       3            2
## Valiant                 1            0       3            1
## Duster 360              0            0       3            4
## Merc 240D               1            0       4            2
## Merc 230                1            0       4            2
## Merc 280                1            0       4            4
## Merc 280C               1            0       4            4
## Merc 450SE              0            0       3            3
## Merc 450SL              0            0       3            3
## Merc 450SLC             0            0       3            3
## Cadillac Fleetwood      0            0       3            4
## Lincoln Continental     0            0       3            4
## Chrysler Imperial       0            0       3            4
## Fiat 128                1            1       4            1
## Honda Civic             1            1       4            2
## Toyota Corolla          1            1       4            1
## Toyota Corona           1            0       3            1
## Dodge Challenger        0            0       3            2
## AMC Javelin             0            0       3            2
## Camaro Z28              0            0       3            4
## Pontiac Firebird        0            0       3            2
## Fiat X1-9               1            1       4            1
## Porsche 914-2           0            1       5            2
## Lotus Europa            1            1       5            2
## Ford Pantera L          0            1       5            4
## Ferrari Dino            0            1       5            6
## Maserati Bora           0            1       5            8
## Volvo 142E              1            1       4            2
```

```
autos <- mtautos
```

# Tratamineto de la matriz

```
dim(autos)
```

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

```
str(autos)
```

```
## '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 ...
```

```
anyNA(autos)
```

```
## [1] FALSE
```

# Instalar libreria

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

# Preparación de la matriz

```
autos1 <- select(autos, -forma, -transmision) # Forma simple 1
```

# Análisis factorial

#1.- Lectura de la matriz de datos

```
x<-as.data.frame(autos1)
head(x)
```

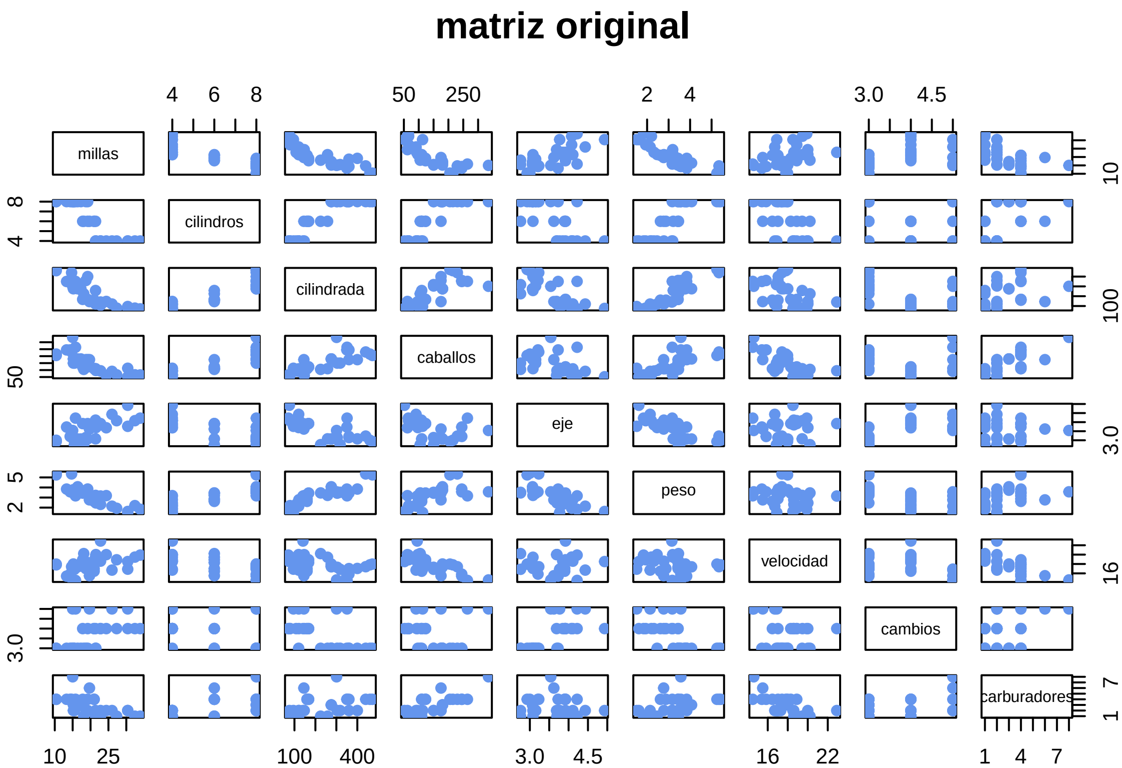
```
##                   millas cilindros cilindrada caballos  eje  peso velocidad
## Mazda RX4           21.0         6        160      110 3.90 2.620     16.46
## Mazda RX4 Wag       21.0         6        160      110 3.90 2.875     17.02
## Datsun 710          22.8         4        108       93 3.85 2.320     18.61
## Hornet 4 Drive      21.4         6        258      110 3.08 3.215     19.44
## Hornet Sportabout   18.7         8        360      175 3.15 3.440     17.02
## Valiant             18.1         6        225      105 2.76 3.460     20.22
##                   cambios carburadores
## Mazda RX4               4            4
## Mazda RX4 Wag           4            4
## Datsun 710              4            1
## Hornet 4 Drive          3            1
## Hornet Sportabout       3            2
## Valiant                 3            1
```

## Separa n (automoviles) y p (variables)

```
n<-dim(x)[1]
p<-dim(x)[2]
```

## Scater plot para la visualización de variables originales

```
pairs(x, col="cornflowerblue", pch=19, main="matriz original")
```




#———————————— # Transformación de alguna varibles #————————————-

## 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) #————————————

## Calcular la matriz de medias y de correlaciones

## Matriz de medias

```
mu<-colMeans(x)
mu
```

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

#Matriz de correlaciones

```
R<-cor(x)
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
## 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    cambios carburadores
## millas        0.41868403  0.4802848   -0.5509251
## cilindros    -0.59124207 -0.4926866    0.5269883
## cilindrada   -0.43369788 -0.5555692    0.3949769
## caballos     -0.70822339 -0.1257043    0.7498125
## eje           0.09120476  0.6996101   -0.0907898
## peso         -0.17471588 -0.5832870    0.4276059
## velocidad     1.00000000 -0.2126822   -0.6562492
## cambios      -0.21268223  1.0000000    0.2740728
## carburadores -0.65624923  0.2740728    1.0000000
```

```r
head(R)
```

```
##                millas  cilindros cilindrada   caballos        eje       peso
## millas      1.0000000 -0.8521620 -0.8475514 -0.7761684  0.6811719 -0.8676594
## cilindros  -0.8521620  1.0000000  0.9020329  0.8324475 -0.6999381  0.7824958
## cilindrada -0.8475514  0.9020329  1.0000000  0.7909486 -0.7102139  0.8879799
## caballos   -0.7761684  0.8324475  0.7909486  1.0000000 -0.4487591  0.6587479
## eje         0.6811719 -0.6999381 -0.7102139 -0.4487591  1.0000000 -0.7124406
## peso       -0.8676594  0.7824958  0.8879799  0.6587479 -0.7124406  1.0000000
##              velocidad    cambios carburadores
## millas      0.41868403  0.4802848   -0.5509251
## cilindros  -0.59124207 -0.4926866    0.5269883
## cilindrada -0.43369788 -0.5555692    0.3949769
## caballos   -0.70822339 -0.1257043    0.7498125
## eje         0.09120476  0.6996101   -0.0907898
## peso       -0.17471588 -0.5832870    0.4276059
```

# Reducción de la dimensionalidad mediante Análisis factorial de componentes principales (PCFA).

## Calcular los valores y vectores propios.

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

### Valores propios

```r
eR$values
```

```
## [1] 5.65593947 2.08210029 0.50421482 0.26502753 0.18315864 0.12379319 0.10506192
## [8] 0.05851375 0.02219038
```

### Vectores propios

```
eR$vectors
```

```
##             [,1]        [,2]        [,3]         [,4]       [,5]        [,6]
##  [1,]  0.3931477 -0.02753861 -0.22119309 -0.006126378  0.3207620  0.72015586
##  [2,] -0.4025537 -0.01570975 -0.25231615  0.040700251 -0.1171397  0.22432550
##  [3,] -0.3973528  0.08888469 -0.07825139  0.339493732  0.4867849 -0.01967516
##  [4,] -0.3670814 -0.26941371 -0.01721159  0.068300993  0.2947317  0.35394225
##  [5,]  0.3118165 -0.34165268  0.14995507  0.845658485 -0.1619259 -0.01536794
##  [6,] -0.3734771  0.17194306  0.45373418  0.191260029  0.1874822 -0.08377237
##  [7,]  0.2243508  0.48404435  0.62812782 -0.030329127  0.1482495  0.25752940
##  [8,]  0.2094749 -0.55078264  0.20658376 -0.282381831  0.5624860 -0.32298239
##  [9,] -0.2445807 -0.48431310  0.46412069 -0.214492216 -0.3997820  0.35706914
##              [,7]        [,8]        [,9]
##  [1,]  0.38138068  0.12465987 -0.11492862
##  [2,]  0.15893251 -0.81032177 -0.16266295
##  [3,]  0.18233095  0.06416707  0.66190812
##  [4,] -0.69620751  0.16573993 -0.25177306
##  [5,] -0.04767957 -0.13505066 -0.03809096
##  [6,]  0.42777608  0.19839375 -0.56918844
##  [7,] -0.27622581 -0.35613350  0.16873731
##  [8,]  0.08555707 -0.31636479 -0.04719694
##  [9,]  0.20604210  0.10832772  0.32045892
```

## Valores propios

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

```
## [1] 5.65593947 2.08210029 0.50421482 0.26502753 0.18315864 0.12379319 0.10506192
## [8] 0.05851375 0.02219038
```

## Vectores propios

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


```
##             [,1]        [,2]        [,3]         [,4]       [,5]        [,6]
##  [1,]  0.3931477 -0.02753861 -0.22119309 -0.006126378  0.3207620  0.72015586
##  [2,] -0.4025537 -0.01570975 -0.25231615  0.040700251 -0.1171397  0.22432550
##  [3,] -0.3973528  0.08888469 -0.07825139  0.339493732  0.4867849 -0.01967516
##  [4,] -0.3670814 -0.26941371 -0.01721159  0.068300993  0.2947317  0.35394225
##  [5,]  0.3118165 -0.34165268  0.14995507  0.845658485 -0.1619259 -0.01536794
##  [6,] -0.3734771  0.17194306  0.45373418  0.191260029  0.1874822 -0.08377237
##  [7,]  0.2243508  0.48404435  0.62812782 -0.030329127  0.1482495  0.25752940
##  [8,]  0.2094749 -0.55078264  0.20658376 -0.282381831  0.5624860 -0.32298239
##  [9,] -0.2445807 -0.48431310  0.46412069 -0.214492216 -0.3997820  0.35706914
##              [,7]        [,8]        [,9]
##  [1,]  0.38138068  0.12465987 -0.11492862
##  [2,]  0.15893251 -0.81032177 -0.16266295
##  [3,]  0.18233095  0.06416707  0.66190812
##  [4,] -0.69620751  0.16573993 -0.25177306
```

```
##  [5,] -0.04767957 -0.13505066 -0.03809096
##  [6,]  0.42777608  0.19839375 -0.56918844
##  [7,] -0.27622581 -0.35613350  0.16873731
##  [8,]  0.08555707 -0.31636479 -0.04719694
##  [9,]  0.20604210  0.10832772  0.32045892
```

# Calcular la proporcion de variabilidad

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

```
## [1] 0.628437719 0.231344477 0.056023869 0.029447503 0.020350960 0.013754799
## [7] 0.011673547 0.006501528 0.002465598
```

# Calcular la proporción de variabilidad acumulada

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

```
## [1] 0.6284377 0.8597822 0.9158061 0.9452536 0.9656045 0.9793593 0.9910329
## [8] 0.9975344 1.0000000
```

#——————————————- # Estimación 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

# onal 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,]  0.9349924 -0.03973679 -0.15706498
##  [2,] -0.9573620 -0.02266837 -0.17916487
##  [3,] -0.9449932  0.12825603 -0.05556481
##  [4,] -0.8730011 -0.38875010 -0.01222162
##  [5,]  0.7415688 -0.49298721  0.10648022
##  [6,] -0.8882114  0.24810498  0.32218796
##  [7,]  0.5335561  0.69845105  0.44602154
##  [8,]  0.4981777 -0.79475097  0.14669117
##  [9,] -0.5816671 -0.69883885  0.32956322
```

# Rotación varimax

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

```
## $loadings
##
## Loadings:
##       [,1]   [,2]   [,3]
##  [1,]  0.708  0.564  0.284
##  [2,] -0.717 -0.355 -0.555
##  [3,] -0.795 -0.372 -0.377
##  [4,] -0.397 -0.602 -0.628
##  [5,]  0.888         0.101
##  [6,] -0.794 -0.569
##  [7,]         0.252  0.953
##  [8,]  0.910 -0.219 -0.161
##  [9,]        -0.850 -0.458
##
##                  [,1]  [,2]  [,3]
## SS loadings    4.056 2.106 2.080
## Proportion Var 0.451 0.234 0.231
## Cumulative Var 0.451 0.685 0.916
##
## $rotmat
##            [,1]       [,2]      [,3]
## [1,]  0.7461406  0.4983919 0.4414520
## [2,] -0.6581439  0.4519395 0.6021605
## [3,]  0.1006023 -0.7398353 0.6652240
```

#—————————————————- # 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]       [,7]
##  [1,] 0.0995408 0.00000000 0.00000000 0.000000 0.0000000 0.0000000 0.00000000
##  [2,] 0.0000000 0.05084412 0.00000000 0.000000 0.0000000 0.0000000 0.00000000
##  [3,] 0.0000000 0.00000000 0.08745088 0.000000 0.0000000 0.0000000 0.00000000
##  [4,] 0.0000000 0.00000000 0.00000000 0.086593 0.0000000 0.0000000 0.00000000
##  [5,] 0.0000000 0.00000000 0.00000000 0.000000 0.1957012 0.0000000 0.00000000
##  [6,] 0.0000000 0.00000000 0.00000000 0.000000 0.0000000 0.0457193 0.00000000
##  [7,] 0.0000000 0.00000000 0.00000000 0.000000 0.0000000 0.0000000 0.02854884
##  [8,] 0.0000000 0.00000000 0.00000000 0.000000 0.0000000 0.0000000 0.00000000
##  [9,] 0.0000000 0.00000000 0.00000000 0.000000 0.0000000 0.0000000 0.00000000
##             [,8]       [,9]
##  [1,] 0.00000000 0.00000000
##  [2,] 0.00000000 0.00000000
##  [3,] 0.00000000 0.00000000
##  [4,] 0.00000000 0.00000000
##  [5,] 0.00000000 0.00000000
##  [6,] 0.00000000 0.00000000
##  [7,] 0.00000000 0.00000000
##  [8,] 0.09867157 0.00000000
##  [9,] 0.00000000 0.06467568
```

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

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

```
##                 millas  cilindros cilindrada   caballos         eje       peso
## millas       0.9004592 -0.8521620 -0.8475514 -0.7761684  0.68117191 -0.8676594
## cilindros   -0.8521620  0.9491559  0.9020329  0.8324475 -0.69993811  0.7824958
## cilindrada  -0.8475514  0.9020329  0.9125491  0.7909486 -0.71021393  0.8879799
## caballos    -0.7761684  0.8324475  0.7909486  0.9134070 -0.44875912  0.6587479
## eje          0.6811719 -0.6999381 -0.7102139 -0.4487591  0.80429876 -0.7124406
## peso        -0.8676594  0.7824958  0.8879799  0.6587479 -0.71244065  0.9542807
## velocidad    0.4186840 -0.5912421 -0.4336979 -0.7082234  0.09120476 -0.1747159
## 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    cambios carburadores
## millas       0.41868403  0.4802848   -0.5509251
## cilindros   -0.59124207 -0.4926866    0.5269883
## cilindrada  -0.43369788 -0.5555692    0.3949769
## caballos    -0.70822339 -0.1257043    0.7498125
## eje          0.09120476  0.6996101   -0.0907898
## peso        -0.17471588 -0.5832870    0.4276059
## velocidad    0.97145116 -0.2126822   -0.6562492
## cambios     -0.21268223  0.9013284    0.2740728
## carburadores -0.65624923  0.2740728    0.9353243
```

### Cálculo de la matriz de autovalores y autovectores

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

### Autovalores

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

```
## [1]  5.572165151  2.000038362  0.453056211  0.119348645  0.080093612
## [6]  0.045648492  0.026571256 -0.005172587 -0.049494554
```

### Autovectores

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

```
## [1]  5.572165151  2.000038362  0.453056211  0.119348645  0.080093612
## [6]  0.045648492  0.026571256 -0.005172587 -0.049494554
```

## Proporción de variabilidad

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

```
## [1]  0.6760486577  0.2426567076  0.0549675100  0.0144800969  0.0097174397
## [6]  0.0055383502  0.0032237850 -0.0006275695 -0.0060049775
```

## Proporción de variabilidad acumulada

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

```
## [1] 0.6760487 0.9187054 0.9736729 0.9881530 0.9978704 1.0034088 1.0066325
## [8] 1.0060050 1.0000000
```

## Estimación de la matriz de cargas con rotación varimax

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

```
##             [,1]        [,2]          [,3]
##  [1,]  0.9252825 -0.04770735 -0.1292947970
##  [2,] -0.9559189 -0.01514217 -0.1700268077
##  [3,] -0.9370706  0.13428909 -0.0550328186
##  [4,] -0.8672733 -0.37209492  0.0006824083
##  [5,]  0.7196227 -0.46095442  0.0796155549
##  [6,] -0.8869184  0.25806723  0.2987270323
##  [7,]  0.5371042  0.70107682  0.4249817837
##  [8,]  0.4907391 -0.77625561  0.1512985675
##  [9,] -0.5815371 -0.68402174  0.3245360152
```

## Rotación varimax

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

```
## $loadings
##
## Loadings:
##       [,1]   [,2]   [,3]
##  [1,]  0.738  0.497  0.290
##  [2,] -0.740 -0.319 -0.541
##  [3,] -0.813 -0.323 -0.366
##  [4,] -0.436 -0.580 -0.604
##  [5,]  0.852
##  [6,] -0.830 -0.503
##  [7,]         0.273  0.941
##  [8,]  0.879 -0.266 -0.153
##  [9,]        -0.842 -0.451
##
##                 [,1]  [,2]  [,3]
```

```
## SS loadings    4.130 1.898 1.997
## Proportion Var 0.459 0.211 0.222
## Cumulative Var 0.459 0.670 0.892
##
## $rotmat
##             [,1]       [,2]      [,3]
## [1,]  0.77336100  0.4585853 0.4377354
## [2,] -0.63083151  0.4880770 0.6031852
## [3,]  0.06296326 -0.7426172 0.6667498
```

## 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,] 0.1248591 0.00000000 0.0000000 0.000000 0.0000000 0.0000000 0.00000000
##  [2,] 0.0000000 0.05708058 0.0000000 0.000000 0.0000000 0.0000000 0.00000000
##  [3,] 0.0000000 0.00000000 0.1008365 0.000000 0.0000000 0.0000000 0.00000000
##  [4,] 0.0000000 0.00000000 0.0000000 0.109382 0.0000000 0.0000000 0.00000000
##  [5,] 0.0000000 0.00000000 0.0000000 0.000000 0.2633256 0.0000000 0.00000000
##  [6,] 0.0000000 0.00000000 0.0000000 0.000000 0.0000000 0.0575393 0.00000000
##  [7,] 0.0000000 0.00000000 0.0000000 0.000000 0.0000000 0.0000000 0.03940087
##  [8,] 0.0000000 0.00000000 0.0000000 0.000000 0.0000000 0.0000000 0.00000000
##  [9,] 0.0000000 0.00000000 0.0000000 0.000000 0.0000000 0.0000000 0.00000000
##            [,8]       [,9]
##  [1,] 0.0000000 0.00000000
##  [2,] 0.0000000 0.00000000
##  [3,] 0.0000000 0.00000000
##  [4,] 0.0000000 0.00000000
##  [5,] 0.0000000 0.00000000
##  [6,] 0.0000000 0.00000000
##  [7,] 0.0000000 0.00000000
##  [8,] 0.1337111 0.00000000
##  [9,] 0.0000000 0.08860526
```

#——————————- # Obtención de los scores de ambos métodos #———————————–

# PCFA

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

```
##                           [,1]        [,2]       [,3]
## Mazda RX4           2.27844633  0.12939676 -0.4187558
## Mazda RX4 Wag       2.06638675  0.05997537 -0.1275513
## Datsun 710          3.75569001  3.04922364  2.5989277
## Hornet 4 Drive     -1.51880734  1.71740014  1.7880288
## Hornet Sportabout  -3.67687551 -0.79790863 -1.4339938
## Valiant            -2.40357558  1.47802676  2.1266113
## Duster 360         -4.52036408 -3.11023420 -3.4714888
## Merc 240D           2.92683738  2.49848641  3.2423244
## Merc 230            2.97330246  2.53759808  4.4737814
```

```
## Merc 280             1.29396657 -0.39118982  0.3153355
## Merc 280C            1.12389994 -0.43777844  0.5692139
## Merc 450SE          -4.04671408 -1.65341065 -1.4468090
## Merc 450SL          -3.66709332 -1.34330455 -1.2878450
## Merc 450SLC         -3.95807367 -1.51265770 -1.1750845
## Cadillac Fleetwood  -7.31343197 -4.17149833 -2.5906455
## Lincoln Continental -7.31762883 -4.33804918 -2.7227834
## Chrysler Imperial   -6.32099474 -3.98824982 -2.7638017
## Fiat 128             5.69909966  4.49296780  3.8937342
## Honda Civic          7.49681907  4.22614001  3.3243961
## Toyota Corolla       6.45451244  4.95555504  4.2631707
## Toyota Corona        1.88950656  3.24722418  3.3963488
## Dodge Challenger    -4.35022314 -0.86928977 -1.3846206
## AMC Javelin         -3.58265957 -0.69579075 -1.0505321
## Camaro Z28          -3.92456250 -3.32523095 -3.6268088
## Pontiac Firebird    -4.31983816 -1.11129150 -1.5410626
## Fiat X1-9            5.31834057  4.08843908  3.3559644
## Porsche 914-2        6.46123425  2.41559661  1.3254373
## Lotus Europa         6.42240480  3.01882074  1.4080048
## Ford Pantera L       0.08441992 -3.53195747 -4.4953470
## Ferrari Dino         2.57376835 -2.12511459 -2.3846307
## Maserati Bora       -1.40866147 -6.48884560 -6.2532326
## Volvo 142E           3.51086890  1.97695133  2.0937138
```

# PFA

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

```
##                          [,1]         [,2]       [,3]
## Mazda RX4            2.2503095 -0.007388463 -0.4237748
## Mazda RX4 Wag        2.0339867 -0.052856971 -0.1374758
## Datsun 710           3.9194092  2.819466334  2.5434606
## Hornet 4 Drive      -1.3506087  1.772825496  1.7615711
## Hornet Sportabout   -3.6701835 -0.623580325 -1.3973205
## Valiant             -2.2241753  1.594832408  2.0928410
## Duster 360          -4.6655130 -2.862306712 -3.3994468
## Merc 240D            3.0667931  2.343251734  3.1756554
## Merc 230             3.1002297  2.430766638  4.3933369
## Merc 280             1.2337881 -0.424111156  0.2973881
## Merc 280C            1.0624240 -0.447943959  0.5460880
## Merc 450SE          -4.0917307 -1.429370436 -1.4173333
## Merc 450SL          -3.6932328 -1.149953280 -1.2570579
## Merc 450SLC         -3.9926517 -1.287742636 -1.1491101
## Cadillac Fleetwood  -7.4944240 -3.698047616 -2.5427780
## Lincoln Continental -7.5152865 -3.859117773 -2.6738287
## Chrysler Imperial   -6.5196121 -3.579929788 -2.7091940
## Fiat 128             5.9263861  4.128687804  3.8255323
## Honda Civic          7.6429164  3.797350174  3.2449839
## Toyota Corolla       6.6988719  4.545855969  4.1915157
## Toyota Corona        2.1023254  3.134075282  3.3239981
## Dodge Challenger    -4.3168910 -0.663983833 -1.3546629
## AMC Javelin         -3.5680231 -0.509392748 -1.0326682
```

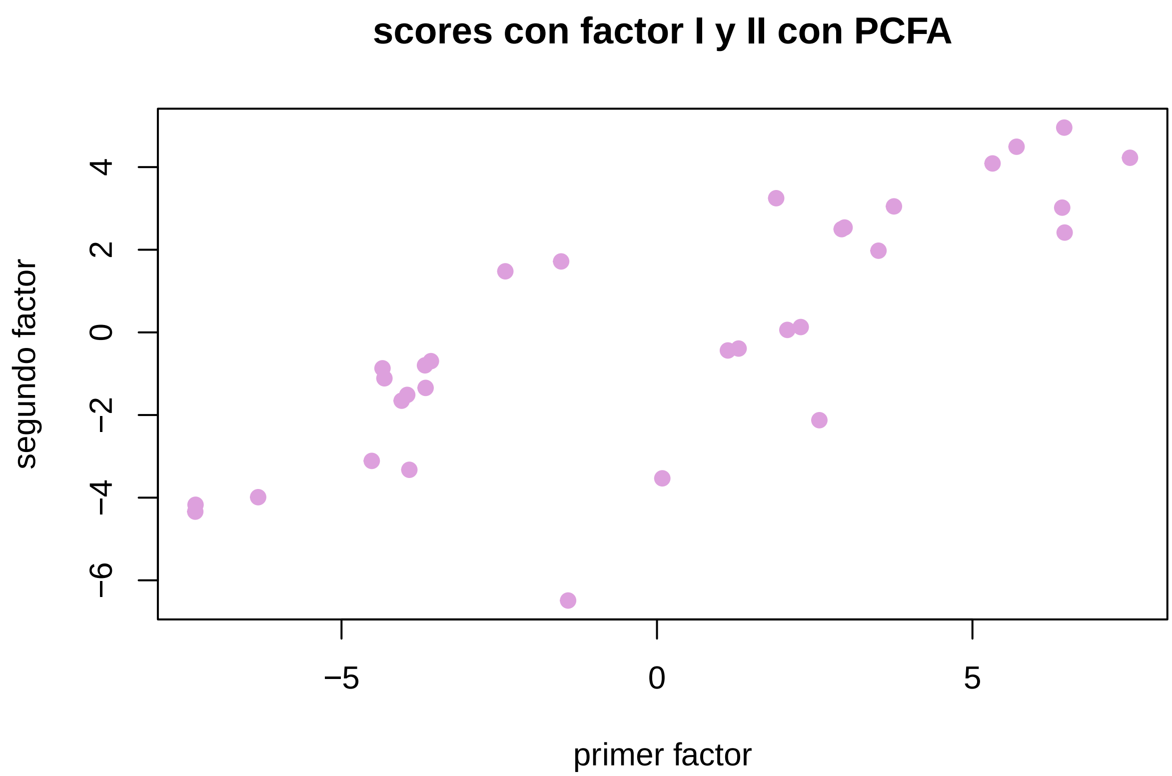
```
## Camaro Z28          -4.1223469 -3.078407847 -3.5662112
## Pontiac Firebird    -4.3264404 -0.896158600 -1.5011515
## Fiat X1-9            5.5251247  3.756377691  3.2879849
## Porsche 914-2        6.5127941  2.036983264  1.2905917
## Lotus Europa         6.5563165  2.575498652  1.3986533
## Ford Pantera L      -0.2123233 -3.509814603 -4.4035949
## Ferrari Dino         2.3965454 -2.275682280 -2.3372024
## Maserati Bora       -1.8515380 -6.378450263 -6.1113761
## Volvo 142E           3.5867601  1.798267843  2.0405861
```

## Graficamos ambos scores

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

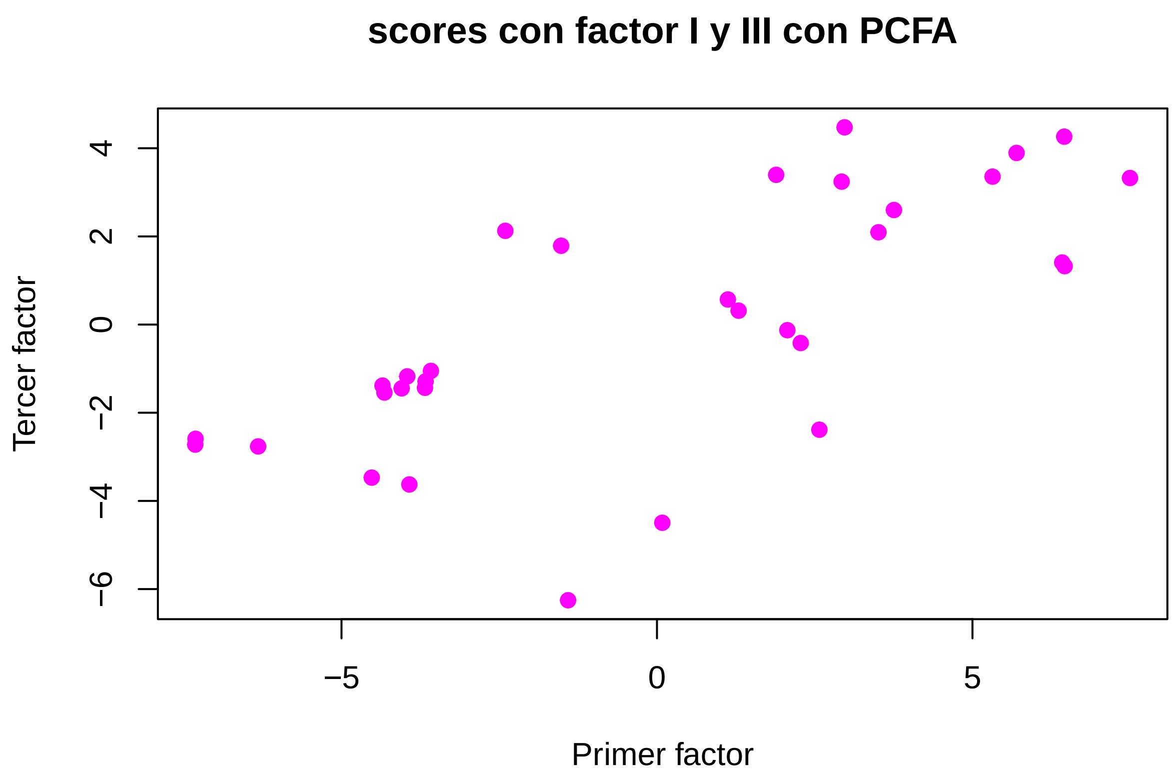
## Factor I y II

```
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="plum")
```



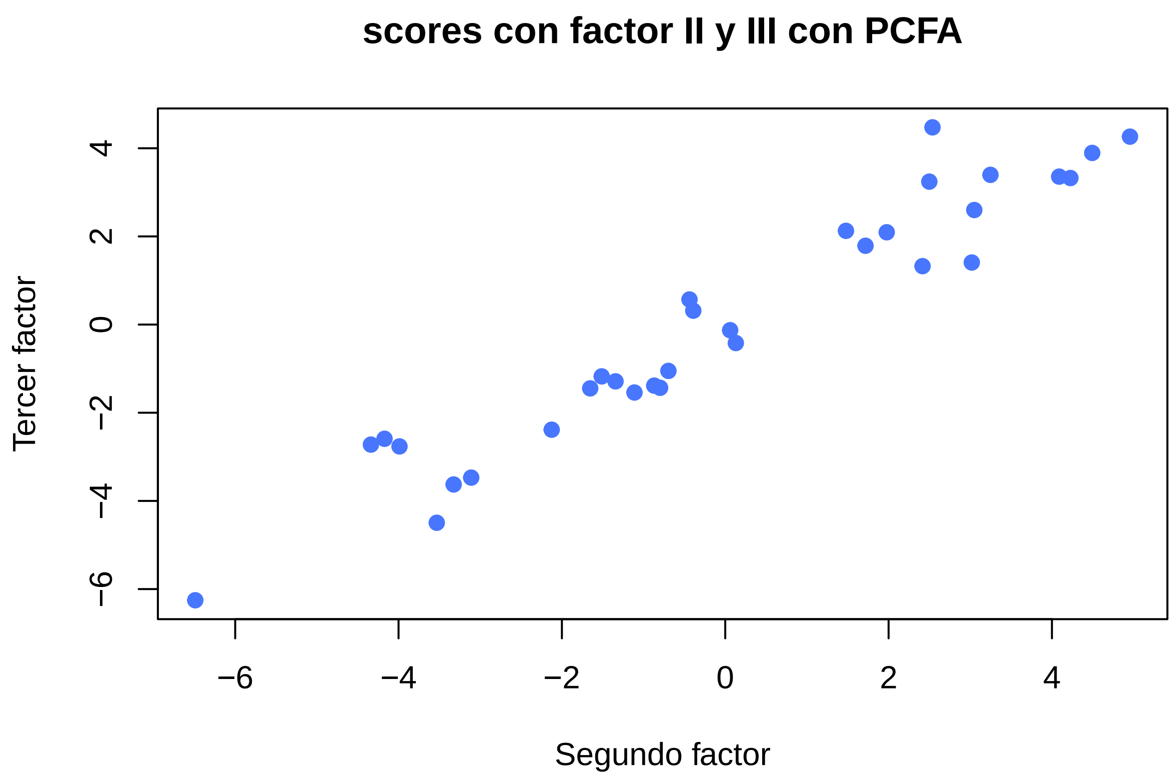

## Factor I y III

```
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="magenta1")
```




## 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="royalblue1")
```

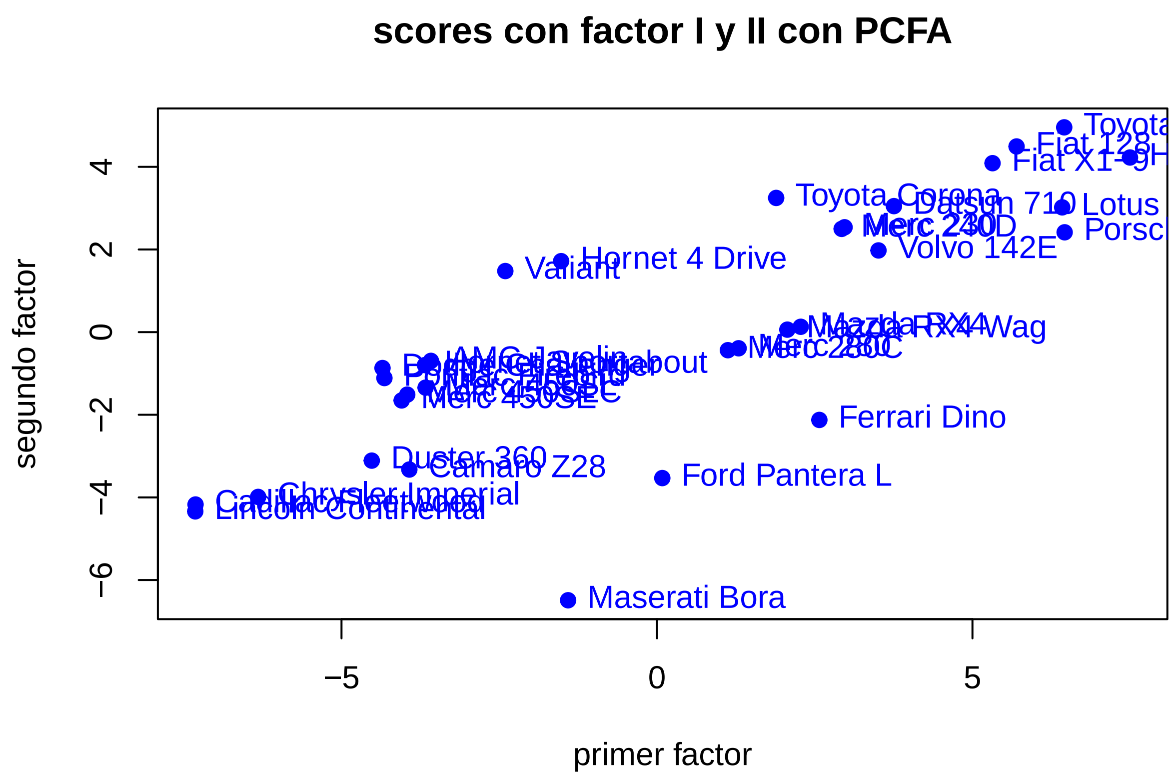



### Graficamos ambos scores con nombres de nuestras variables

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

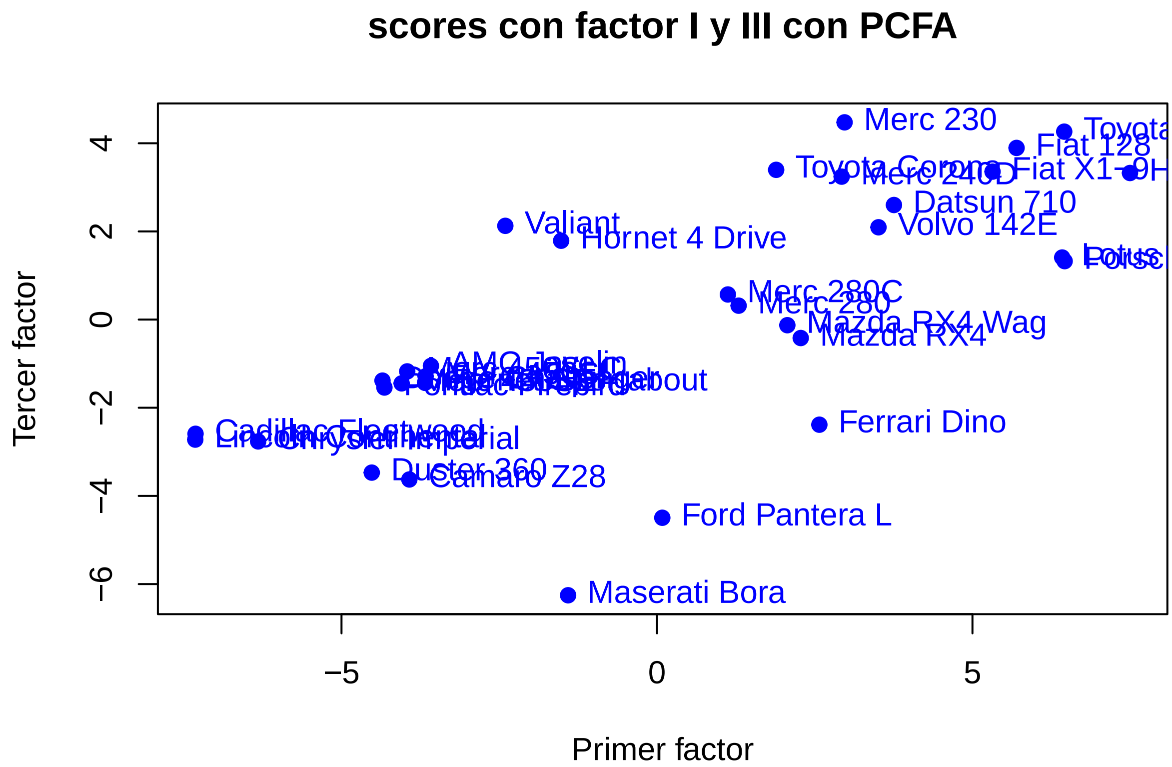
### Factor I y II

```
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(x), pos=4, col="blue")
```




## Factor I y III

```
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(x), pos=4, col="blue")
```


scores con factor I y III con PCFA
Tercer factor
Primer factor
Merc 230
Toyota
Fiat 128
Toyota Corona
Merc 240D
Fiat X1-9
Datsun 710
Volvo 142E
Valiant
Hornet 4 Drive
Lotus
Merc 280C
Merc 280
Mazda RX4 Wag
Mazda RX4
Pontiac Firebird
Ferrari Dino
Cadillac Fleetwood
Chrysler Imperial
Duster 360
Camaro Z28
Ford Pantera L
Maserati Bora