PCA Revealed

Part 7: PCA with R

Gaston **S**anchez

August 2014

Readme

PCA

Principal Components Analysis (PCA) allows us to study and explore a set of quantitative variables measured on a set of objects

Core Idea

With PCA we seek to reduce the dimensionality (reduce the number of variables) of a data set while retaining as much as possible of the variation present in the data

Toy Dataset
`cars2004`

Data cars2004

```
cars2004
```

```
##                          Cylinders Horsepower Speed Weight Width Length
## Citroen C2                    1124         61   158    932  1659   3666
## Smart Fortwo                   698         52   135    730  1515   2500
## Mini 1.6 170                  1598        170   218   1215  1690   3625
## Nissan Micra 1.2              1240         65   154    965  1660   3715
## Renault Clio 3.0 V6           2946        255   245   1400  1810   3812
## Audi A3 1.9                   1896        105   187   1295  1765   4203
## Peugeot 307 1.4               1398         70   160   1179  1746   4202
## Peugeot 407 V6                2946        211   229   1640  1811   4676
## Mercedes Classe C             2685        170   230   1600  1728   4528
## BMW 530d                      2993        218   245   1595  1846   4841
## Jaguar S-Type 2.7 V6          2720        207   230   1722  1818   4905
## BMW  745i                     4398        333   250   1870  1902   5029
## Mercedes Classe S 400         3966        260   250   1915  2092   5038
## Citroen C3 Pluriel 1.6i       1587        110   185   1177  1700   3934
## BMW Z4 2.5i                   2494        192   235   1260  1781   4091
## Audi TT 1.8T 180              1781        180   228   1280  1764   4041
## Aston Martin Vanquish         5935        460   306   1835  1923   4665
## Bentley Continental GT        5998        560   318   2385  1918   4804
## Ferrari Enzo                  5998        660   350   1365  2650   4700
## Renault Scenic 1.9            1870        120   188   1430  1805   4259
## VW Touran 1.9 TDI             1896        105   180   1498  1794   4391
## LandRover Defender            2495        122   135   1695  1790   3883
## LandRover Discovery           2495        138   157   2175  2190   4705
## Nissan X-Trail 2.2            2184        136   180   1520  1765   4455
```

Toy Data Example: cars2004

The data consists of 24 cars measured on the following six variables:

`Cylinders`	Number of cylinders (cm^3)
`Horsepower`	Horsepower (hp)
`Speed`	Maximum speed (km/h)
`Weight`	weight of the car (kg)
`Length`	length of the car (mm)
`Width`	width of the car (mm)

Example from Michel Tenenhaus (2007) book:

Statistique: Methodes pour decrire, expliquer et prevoir

Data

Download the data in R using the package `RCurl` as follows:

```r
# load package RCurl
library(RCurl)

# google docs spreadsheets url
google_docs = "https://docs.google.com/spreadsheet/"

# public key of data 'cars'
cars_key = "pub?key=0AjoVnZ9iB261dHRfQlVuWDRUSHdZQ1A4N294TEstc0E&output=csv"

# download URL of data file
cars_csv = getURL(paste(google_docs, cars_key, sep = ""))

# import data in R (through a text connection)
cars2004 = read.csv(textConnection(cars_csv), row.names = 1, header = TRUE)
```


Data

Use the function `head()` to take a peek of the data contained in `cars2004`:

```
# take a peek
head(cars2004)

##                      Cylinders Horsepower Speed Weight Width Length
## Citroen C2                1124         61   158    932  1659   3666
## Smart Fortwo               698         52   135    730  1515   2500
## Mini 1.6 170              1598        170   218   1215  1690   3625
## Nissan Micra 1.2          1240         65   154    965  1660   3715
## Renault Clio 3.0 V6       2946        255   245   1400  1810   3812
## Audi A3 1.9               1896        105   187   1295  1765   4203
```


Before applying PCA

Preliminaries

Before performing a PCA (or any other multivariate method) we should start with some preliminary explorations

- Descriptive statistics
- Basic graphical displays
- Distribution of variables
- Pair-wise correlations among variables
- Perhaps transforming some variables
- ETC

Descriptive Statistics

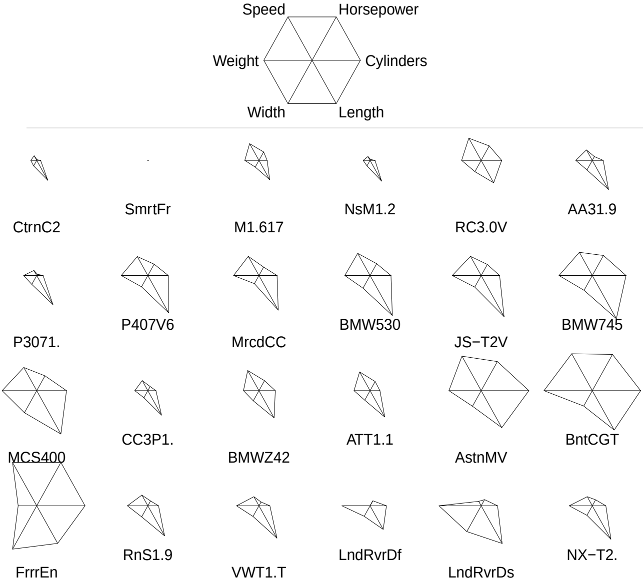
Let's get some summary statistics:

```
# descriptive statistics
cars_stats = data.frame(
  Minimum = apply(cars2004, 2, min),
  Maximum = apply(cars2004, 2, max),
  Mean = apply(cars2004, 2, mean),
  Std_Dev = apply(cars2004, 2, sd))

print(cars_stats, print.gap = 3)

##                Minimum   Maximum     Mean   Std_Dev
## Cylinders          698      5998   2722.5   1516.44
## Horsepower          52       660    206.7    155.72
## Speed              135       350    214.7     56.57
## Weight             730      2385   1486.6    387.51
## Width             1515      2650   1838.4    220.84
## Length            2500      5038   4277.8    581.50
```

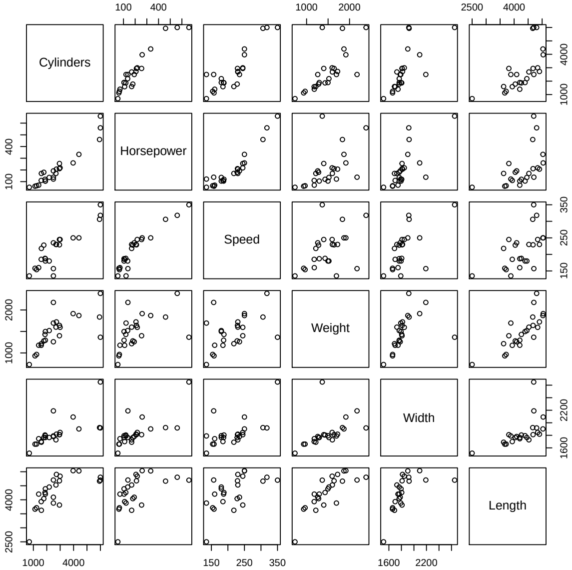

Stars plot

Since we have a small number of observations (24 cars), we can use the function `stars()` to get an idea of the (dis)similarities between the cars:

```
# star plot
stars(cars2004, labels = abbreviate(rownames(cars2004), 6),
      nrow = 4, key.loc = c(8, 11.2))
abline(h = 9.85, col = "gray90")
```



Speed
Horsepower
Weight
Cylinders
Width
Length
CtrnC2
SmrtFr
M1.617
NsM1.2
RC3.0V
AA31.9
P3071.
P407V6
MrcdCC
BMW530
JS−T2V
BMW745
MCS400
CC3P1.
BMWZ42
ATT1.1
AstnMV
BntCGT
FrrrEn
RnS1.9
VWT1.T
LndRvrDf
LndRvrDs
NX−T2.

Scatter-plot matrix



Scatter-plot matrix

The previous graphic is obtained with the following command:

```
# scatterplot to inspect pair-wise relations
pairs(cars2004)
```

Look at the pair-wise scatterplot:

- What kind of patterns do you see?
- What variables seem to be correlated with each other?
- Are there any points (objects) that stand out?
- Is there anything in particular that calls your attention?

Matrix of Correlations

We can also examine the correlations among variables:

```
# show lower triangular part of matrix of correlations
as.dist(round(cor(cars2004), 3))

##            Cylinders Horsepower Speed Weight Width
## Horsepower     0.954
## Speed          0.885      0.934
## Weight         0.692      0.529 0.466
## Width          0.706      0.730 0.619  0.477
## Length         0.664      0.527 0.578  0.795 0.591
```

Notice how all variables are positively correlated

PCA in R

PCA in R

Several functions (and packages) to perform PCA in R

PCA functions in R

Function	Package	Author
`prcomp()`	`stats`	R Core Team
`princomp()`	`stats`	R Core Team
`PCA()`	`FactoMineR`	Husson, Josse, Le, Mazet
`dudi.pca()`	`ade4`	Chessel, Dufour, Dray
`acp()`	`amap`	Lucas
`nipals()`	`plsdepot`	Sanchez
`rda()`	`vegan`	Oksanen *et al*
`pca()`	`pcaMethods` *	Stacklies, Redestig, Wright

*See `http://www.bioconductor.org/packages/release/bioc/html/pcaMethods.html`
The default PCA functions in R are `prcomp()` and `princomp()`

Eigenvalues, Scores, Loadings

The minimal output from any PCA should contain 3 things:

- **Eigenvalues** provide information about the amount of variability captured by each principal component
- **Scores** or PCs that provide coordinates to graphically represent objects in a lower dimensional space
- **Loadings** provide information to determine what variables characterize each principal component

PCA with `prcomp()`

PCA with prcomp()

One of the default PCA functions in R is `prcomp()`:

```
# PCA with prcomp()
cars_prcomp = prcomp(cars2004, scale. = TRUE)

# what does prcomp() provide?
names(cars_prcomp)

## [1] "sdev"     "rotation" "center"   "scale"    "x"

# eigenvalues
cars_prcomp$sdev^2

## [1] 4.41127 0.85341 0.43566 0.23587 0.05144 0.01235
```

`scale.= TRUE` indicates that PCA is performed on standardized data (mean = 0, variance = 1)

PCA with prcomp() con't

```
# scores
round(head(cars_prcomp$x, 5), 2)

##                      PC1   PC2   PC3   PC4   PC5   PC6
## Citroen C2         -2.54 -0.50 -0.18  0.16 -0.20  0.03
## Smart Fortwo       -4.06 -1.63  0.27 -0.90 -0.03 -0.03
## Mini 1.6 170       -1.35 -0.80  0.36 -0.05  0.45  0.05
## Nissan Micra 1.2   -2.46 -0.40 -0.17  0.12 -0.28  0.05
## Renault Clio 3.0 V6  0.00 -0.90  0.38 -0.27  0.28 -0.13

# loadings
round(head(cars_prcomp$rotation, 5), 2)

##             PC1   PC2   PC3   PC4   PC5   PC6
## Cylinders  0.46 -0.14  0.21 -0.23 -0.65 -0.50
## Horsepower 0.44 -0.38  0.14 -0.17 -0.09  0.78
## Speed      0.42 -0.37  0.31  0.41  0.57 -0.31
## Weight     0.36  0.62  0.22 -0.53  0.39 -0.01
## Width      0.38 -0.12 -0.88 -0.14  0.15 -0.13
```

PCA with `princomp()`

PCA with princomp()

The other default PCA function is princomp()

```
# PCA with princomp()
cars_princomp = princomp(cars2004, cor = TRUE)

# what does princomp() provide?
names(cars_princomp)

## [1] "sdev"     "loadings" "center"   "scale"    "n.obs"    "scores"

# eigenvalues
cars_princomp$sdev^2

##  Comp.1  Comp.2  Comp.3  Comp.4  Comp.5  Comp.6
## 4.41127 0.85341 0.43566 0.23587 0.05144 0.01235
```

cor = TRUE indicates that PCA is performed on standardized data (mean = 0, variance = 1)

PCA with princomp() con't

```r
# scores
round(head(cars_princomp$scores, 5), 3)

##                     Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6
## Citroen C2           2.596  0.510  0.179  0.166  0.207  0.032
## Smart Fortwo         4.150  1.666 -0.274 -0.924  0.029 -0.034
## Mini 1.6 170         1.382  0.816 -0.372 -0.051 -0.464  0.051
## Nissan Micra 1.2     2.513  0.404  0.174  0.124  0.289  0.051
## Renault Clio 3.0 V6  0.003  0.916 -0.385 -0.274 -0.286 -0.134

# loadings
round(head(unclass(cars_princomp$loadings), 5), 3)

##            Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6
## Cylinders  -0.458  0.137 -0.214 -0.232  0.652 -0.496
## Horsepower -0.440  0.382 -0.137 -0.173  0.094  0.777
## Speed      -0.422  0.367 -0.313  0.410 -0.569 -0.314
## Weight     -0.360 -0.623 -0.223 -0.529 -0.389 -0.008
## Width      -0.381  0.120  0.883 -0.140 -0.153 -0.132
```

PCA with "FactoMineR"

A richer and nicer PCA() with FactoMineR

```r
# load FactoMineR
library(FactoMineR)

# nice PCA
cars_pca = PCA(cars2004, graph = FALSE)

# what does PCA provide?
cars_pca
```

```
## **Results for the Principal Component Analysis (PCA)**
## The analysis was performed on 24 individuals, described by 6 variables
## *The results are available in the following objects:
##
##    name               description
## 1  "$eig"             "eigenvalues"
## 2  "$var"             "results for the variables"
## 3  "$var$coord"       "coord. for the variables"
## 4  "$var$cor"         "correlations variables - dimensions"
## 5  "$var$cos2"        "cos2 for the variables"
## 6  "$var$contrib"     "contributions of the variables"
## 7  "$ind"             "results for the individuals"
## 8  "$ind$coord"       "coord. for the individuals"
## 9  "$ind$cos2"        "cos2 for the individuals"
## 10 "$ind$contrib"     "contributions of the individuals"
## 11 "$call"            "summary statistics"
## 12 "$call$centre"     "mean of the variables"
## 13 "$call$ecart.type" "standard error of the variables"
## 14 "$call$row.w"      "weights for the individuals"
## 15 "$call$col.w"      "weights for the variables"
```


About PCA()

Extensive output

As you can tell, there is an extensive list of results provided in the output of `PCA()` —by `FactoMineR`

Reading results

We'll discuss how to interpret the main results from `PCA()` and what things we should pay attention to

Graphical Examination

With the obtained scores and loadings we can get several graphical displays

Some graphics

- correlations between scores and variables
- relationships among variables
- positions of objects on the score plots
- (dis)siminalirities among objects
- relationships between objects and variables

What do we care about in PCA?

Some questions to keep in mind

- How many PCs should be retained?
- How good (or bad) is the data approximation with the reatined PCs?
- What variables characterize each PC?
- Which variables are influential, and how are they correlated?
- Which variables are responsible for the patterns among objects?
- Are there any *outlier* objects?

How many PCs to retain?

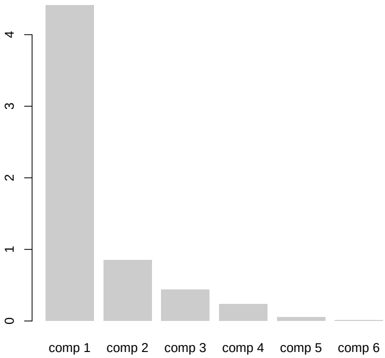
There is no universal criterion to determine the number of PCs to retain. But we must look at the eigenvalues and see what's the percentage of variance captured by each dimension:

```
# table of eigenvalues
cars_pca$eig

##        eigenvalue percentage of variance cumulative percentage of variance
## comp 1    4.41127                73.5211                             73.52
## comp 2    0.85341                14.2235                             87.74
## comp 3    0.43566                 7.2611                             95.01
## comp 4    0.23587                 3.9312                             98.94
## comp 5    0.05144                 0.8573                             99.79
## comp 6    0.01235                 0.2059                            100.00
```

In this example we get 87% of variance explained with the first two PCs

Screeplot of eigenvalues

```
# screeplot of eigenvalues
barplot(cars_pca$eig[,"eigenvalue"], border = NA, col = "gray80",
        names.arg = rownames(cars_pca$eig))
```



What variables characterize each PC?

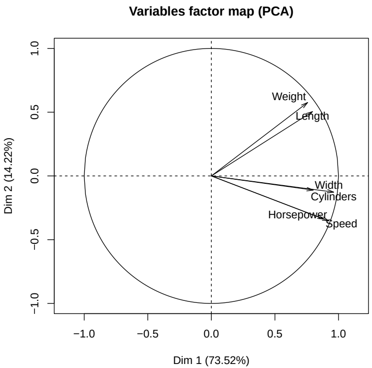
To see how each PC is characterized, we either check the loadings or the correlations between the variables and the PCs:

```
# correlations between variables and PCs
round(cars_pca$var$coord[,1:2], 4)

##              Dim.1   Dim.2
## Cylinders  0.9624 -0.1269
## Horsepower 0.9233 -0.3527
## Speed      0.8861 -0.3387
## Weight     0.7569  0.5757
## Width      0.8012 -0.1110
## Length     0.7953  0.5044
```

Notice that PC1 is positively correlated with all the variables. In turn, PC2 opposes `Weight` and `Length` against `Cylinders`, `Horsepower`, `Speed` and `Weight`

Circle of Correlations

```r
# plot circle of correlations
plot(cars_pca, choix = "var")
```



- We can read this plot as a *radar*.
- The closer an arrow is to the circumference of the circle, the better its representation on the given axes.
- Also note how the variables are grouped.

Influence of variables on each PC?

We can also examine the contributions of the variables

```
# Contribution of variables
print(rbind(cars_pca$var$contrib,
            TOTAL = colSums(cars_pca$var$contrib)), print.gap = 3)

##                Dim.1     Dim.2      Dim.3     Dim.4      Dim.5
## Cylinders      21.00     1.888     4.5964     5.385    42.5754
## Horsepower     19.33    14.573     1.8738     2.986     0.8921
## Speed          17.80    13.446     9.7721    16.808    32.3352
## Weight         12.99    38.837     4.9896    28.030    15.1508
## Width          14.55     1.444    77.9636     1.958     2.3415
## Length         14.34    29.812     0.8045    44.832     6.7051
## TOTAL         100.00   100.000   100.0000   100.000   100.0000
```

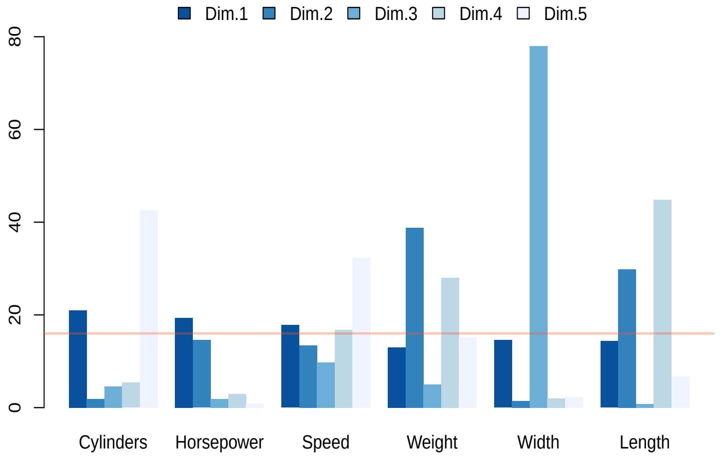
If all variables were to contribute uniformly, they would have a contribution of 1/6 or 16.67%.

Influence of variables on each PC (con't)

To inspect what variables are above and below 16.67 we can create a barplot of variable contributions in the following form:

```
library(RColorBrewer)
# color palette
colpal = brewer.pal(n = 5, name = "Blues")[5:1]

# Contribution of variables
barplot(t(cars_pca$var$contrib), beside = TRUE,
        border = NA, ylim = c(0, 90), col = colpal,
        legend.text = colnames(cars_pca$var$contrib),
        args.legend = list(x = "top", ncol = 5, bty = 'n'))
abline(h = 16, col = "#ff572255", lwd = 2)
```

Influence of each variable on the obtained PCs



PC scores

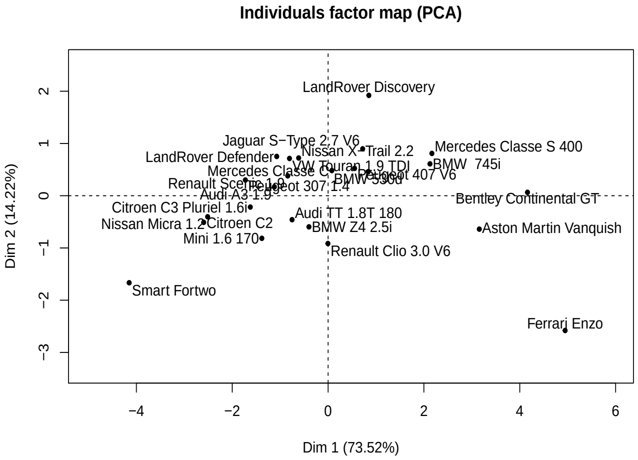
```r
# PC scores (first 2 dimesions)
print(round(cars_pca$ind$coord[,1:2], 3),
      print.gap = 3)
```

```
##                            Dim.1    Dim.2
## Citroen C2                -2.596   -0.510
## Smart Fortwo              -4.150   -1.666
## Mini 1.6 170              -1.382   -0.816
## Nissan Micra 1.2          -2.513   -0.404
## Renault Clio 3.0 V6       -0.003   -0.916
## Audi A3 1.9               -1.121    0.169
## Peugeot 307 1.4           -1.725    0.300
## Peugeot 407 V6             0.553    0.523
## Mercedes Classe C          0.078    0.482
## BMW 530d                   0.838    0.460
## Jaguar S-Type 2.7 V6       0.721    0.898
## BMW  745i                  2.126    0.610
## Mercedes Classe S 400      2.167    0.810
## Citroen C3 Pluriel 1.6i   -1.623   -0.218
## BMW Z4 2.5i               -0.399   -0.596
## Audi TT 1.8T 180          -0.751   -0.459
## Aston Martin Vanquish      3.155   -0.639
## Bentley Continental GT     4.161    0.064
## Ferrari Enzo               4.946   -2.580
## Renault Scenic 1.9        -0.842    0.380
## VW Touran 1.9 TDI         -0.805    0.713
## LandRover Defender        -1.072    0.751
## LandRover Discovery        0.851    1.920
## Nissan X-Trail 2.2        -0.614    0.722
```

We can use the scores as coordinates to plot the objects in a scatterplot

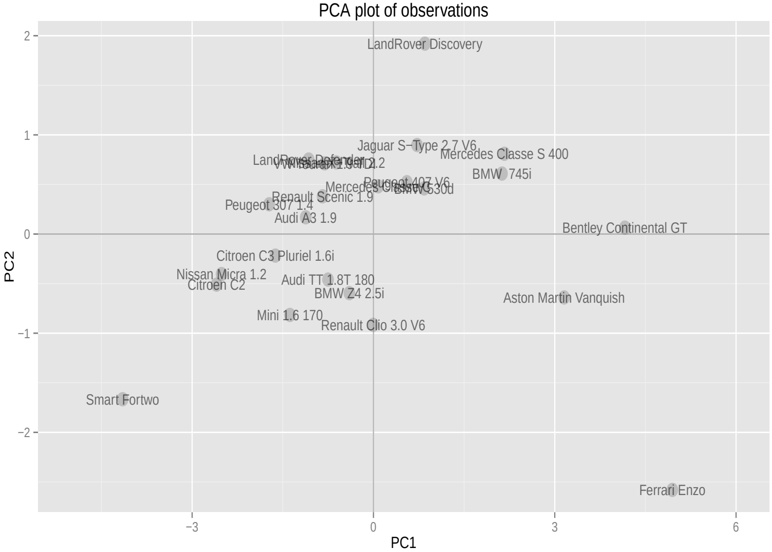
Default plot of objects in FactoMineR

```
# plot of scores
plot(cars_pca, choix = "ind")
```



Alternative plot of objects with ggplot2

```r
# load ggplot2
library(ggplot2)

# data frame with observations from PCA results
cars_pca_obs = data.frame(cars_pca$ind$coord[,1:3])

# PCA plots of observations
ggplot(cars_pca_obs, aes(x = Dim.1, y = Dim.2, label = rownames(cars2004))) +
  geom_hline(yintercept = 0, color = "gray70") +
  geom_vline(xintercept = 0, color = "gray70") +
  geom_point(color = "#55555544", size = 5) +
  geom_text(alpha = 0.55, size = 4) +
  xlab("PC1") +
  ylab("PC2") +
  xlim(-5, 6) +
  ggtitle("PCA plot of observations")
```


PCA plot of observations
LandRover Discovery
Jaguar S−Type 2.7 V6
Mercedes Classe S 400
BMW 745i
Renault Scenic 1.9
Peugeot 307 1.4
Audi A3 1.9
Bentley Continental GT
Citroen C3 Pluriel 1.6i
Nissan Micra 1.2
Citroen C2
Audi TT 1.8T 180
BMW Z4 2.5i
Aston Martin Vanquish
Mini 1.6 170
Renault Clio 3.0 V6
Smart Fortwo
Ferrari Enzo
PC1
PC2
2
1
0
−1
−2
−3
0
3
6

Contributions of objects to PCs

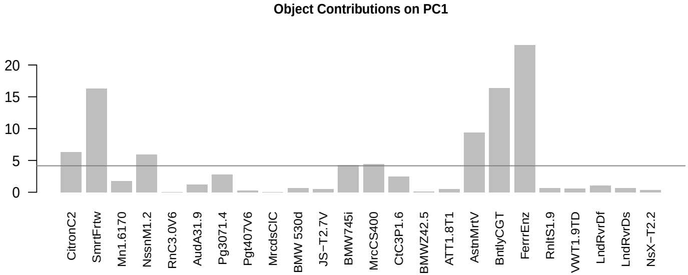
```
# Contributions on PCs (first 2 dimesions)
print(round(cars_pca$ind$contrib[,1:2], 3),
      print.gap = 3)
```

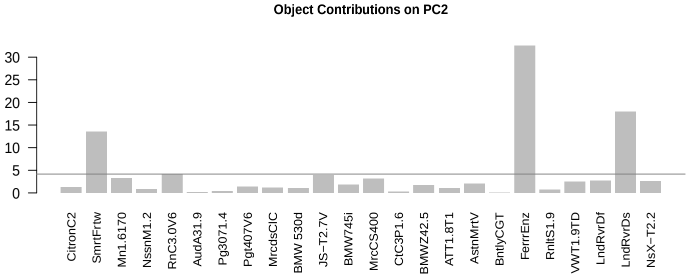
```
##                            Dim.1    Dim.2
## Citroen C2                 6.365    1.270
## Smart Fortwo              16.269   13.550
## Mini 1.6 170               1.804    3.249
## Nissan Micra 1.2           5.967    0.795
## Renault Clio 3.0 V6        0.000    4.092
## Audi A3 1.9                1.186    0.139
## Peugeot 307 1.4            2.812    0.441
## Peugeot 407 V6             0.288    1.336
## Mercedes Classe C          0.006    1.133
## BMW 530d                   0.663    1.033
## Jaguar S-Type 2.7 V6       0.492    3.935
## BMW  745i                  4.271    1.816
## Mercedes Classe S 400      4.434    3.201
## Citroen C3 Pluriel 1.6i    2.487    0.231
## BMW Z4 2.5i                0.150    1.734
## Audi TT 1.8T 180           0.533    1.030
## Aston Martin Vanquish      9.404    1.997
## Bentley Continental GT    16.352    0.020
## Ferrari Enzo              23.110   32.503
## Renault Scenic 1.9         0.669    0.706
## VW Touran 1.9 TDI          0.612    2.481
## LandRover Defender         1.085    2.757
## LandRover Discovery        0.683   18.004
## Nissan X-Trail 2.2         0.357    2.547
```

The contributions (in percentage) reflect the influence that each object has on the formation of the PCs.
If all objects had the same contribution on each PC, they would contribute with a value of $4.16 = 100/24$

Barplots of object contributions to PCs

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

# barplot of object contributions for PC1
barplot(cars_pca$ind$contrib[,1], border = NA, las = 2,
        names.arg = abbreviate(rownames(cars2004), 8), cex.names = 0.8)
title("Object Contributions on PC1", cex.main = 0.9)
abline(h = 4.16, col = "gray50")

# barplot of object contributions for PC2
barplot(cars_pca$ind$contrib[,2], border = NA, las = 2,
        names.arg = abbreviate(rownames(cars2004), 8), cex.names = 0.8)
title("Object Contributions on PC2", cex.main = 0.9)
abline(h = 4.16, col = "gray50")

par(op)
```


Object Contributions on PC1
0
5
10
15
20
CitronC2
SmrtFrtw
Mn1.6170
NssnM1.2
RnC3.0V6
AudA31.9
Pg3071.4
Pgt407V6
MrcdsClC
BMW 530d
JS−T2.7V
BMW745i
MrcCS400
CtC3P1.6
BMWZ42.5
ATT1.8T1
AstnMrtV
BntlyCGT
FerrrEnz
RnltS1.9
VWT1.9TD
LndRvrDf
LndRvrDs
NsX−T2.2


Object Contributions on PC2
0
5
10
15
20
25
30
CitronC2
SmrtFrtw
Mn1.6170
NssnM1.2
RnC3.0V6
AudA31.9
Pg3071.4
Pgt407V6
MrcdsClC
BMW 530d
JS−T2.7V
BMW745i
MrcCS400
CtC3P1.6
BMWZ42.5
ATT1.8T1
AstnMrtV
BntlyCGT
FerrrEnz
RnltS1.9
VWT1.9TD
LndRvrDf
LndRvrDs
NsX−T2.2

PCA with Clustering

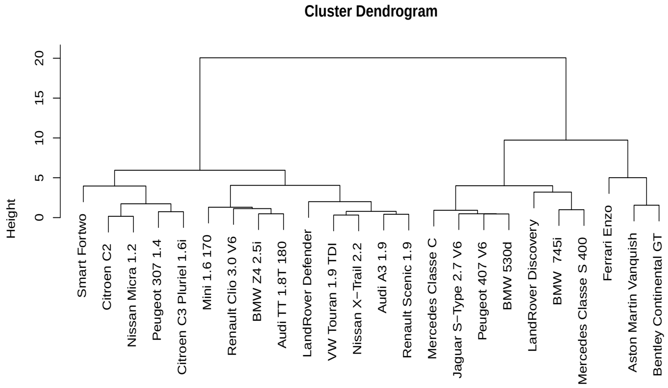
We can gain some insight by combining PCA and Clustering

- Is there a typology of objects?
- How could they be clustered?

One option is to apply a hierarchical clustering to the obtained scores, and then add the clustered groups to the Scores scatterplot

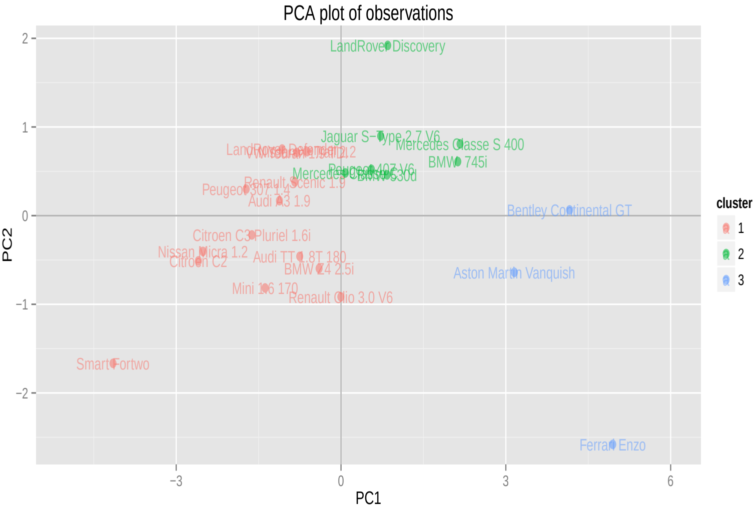
Hierarchical Clustering

```
# clustering
cars_clustering = hclust(dist(cars_pca$ind$coord), method = "ward")
plot(cars_clustering, xlab = "", sub = "")
```



PC plot with clustering partition

```r
# get 3 cluster
cars_clusters = cutree(cars_clustering, k = 3)

# add cluster to data frame of scores
cars_pca_obs$cluster = as.factor(cars_clusters)

# ggplot
ggplot(cars_pca_obs, aes(x=Dim.1, y=Dim.2, label=rownames(cars2004))) +
  geom_hline(yintercept = 0, color = "gray70") +
  geom_vline(xintercept = 0, color = "gray70") +
  geom_point(aes(color = cluster), alpha = 0.55, size = 3) +
  geom_text(aes(color = cluster), alpha = 0.55, size = 4) +
  xlab("PC1") +
  ylab("PC2") +
  xlim(-5, 6) +
  ggtitle("PCA plot of observations")
```



PCA plot of observations
PC2
PC1
2
1
0
−1
−2
−3
0
3
6
LandRover Discovery
Jaguar S−Type 2.7 V6
Mercedes Classe S 400
BMW 745i
LandRover Defender
Peugeot 407 V6
BMW 530d
Renault Scenic 1.9
Peugeot 307 1.4
Audi A3 1.9
Bentley Continental GT
Citroen C3 Pluriel 1.6i
Nissan Micra 1.2
Citroen C2
Audi TT 1.8T 180
BMW Z4 2.5i
Aston Martin Vanquish
Mini 1.6 170
Renault Clio 3.0 V6
Smart Fortwo
Ferrari Enzo
cluster
1
2
3