Joint Dimension Reduction and Clustering in Practice

Multivariate Data Analysis Group Workshop - SASA 2021

Angelos Markos, Democritus University of Thrace, Greece

29 Nov 2021

Introduction

Cluster analysis classifies observations (e.g., respondents, products, or other entities), on a set of user selected characteristics (clustering variables). The common desired characteristics of the resulting clusters are high internal (within-cluster) homogeneity and high external (between-cluster) heterogeneity. Similarity between observations may be defined in various ways depending on data specificities (e.g., measurement scales) and corresponding distance/similarity measures. If the set of variables is large, it is highly unlikely that all variables relate to the same cluster structure and the high dimensionality influences the calculation of distances between observations (a problem known as the "curse of dimensionality"). To overcome these problems, a common strategy is to reduce the set of variables prior to clustering, i.e., perform dimension reduction and then perform clustering in the reduced space. This strategy is known as *tandem analysis* and can be summarized as follows:

Step 1. Carry out PCA and obtain the first few components.

Step 2. Perform K-means clustering for the principal scores on the first few principal components, which are obtained in Step 1.

Intuitive and straightforward as the tandem approach may be, it may not yield an optimal cluster allocation as the two methods optimize different criteria. Dimension reduction typically aims to retain as much variance as possible in as few dimensions as possible, whereas cluster analysis aims to find similar and dissimilar observations in the dataset and allocate the observations accordingly to clusters. This problem is well-known (e.g., Bock 1987; De Soete and Carroll 1994; van Buuren and Heiser 1989; Vichi and Kiers 2001) and several solutions have been proposed that combine dimension reduction with clustering in a single step, i.e., via optimisation of a single objective function. In this tutorial, we first illustrate tandem analysis on a toy data set and then we move on to apply joint dimension reduction and clustering methods.

Tandem Analysis: Dimension Reduction and Clustering

We illustrate dimension reduction followed by clustering in the reduced space using a toy data set. The data set refers to 26 James Bond films produced up to 2021, based on 10 attributes of films: 7 continuous (year of release, production budget (in millions USD), box-office gross in the USA and box-office gross worldwide (both in millions USD), running time in minutes, IMDB rating, Rotten Tomatoes or Tomatometer score) and 3 categorical (Bond actor, native country of the villain, native country of the Bond girl). All figures in USD are adjusted for inflation.

We load the data set from the `clustrd` package and examine the available variables.

```
# Install necessary packages
list.of.packages <- c('devtools','clustrd','leaflet', 'ggfortify', 'rio', 'dplyr',
                      'geojson', 'geojsonio')
new.packages <- list.of.packages[!(list.of.packages %in% installed.packages()[,"Package"])]
if(length(new.packages)) install.packages(new.packages)
if (!("UBbipl" %in% installed.packages()[,"Package"]))
```

```
{
  install.packages("https://www.wiley.com/legacy/wileychi/gower/supp/UBbipl_3.0.6.tar.gz",
                   repos = NULL, type = "source")
}

# Load clustrd package and mybond data set
library("clustrd")
```

```
## Loading required package: ggplot2
```

```
## Loading required package: grid
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
# Load mybond data set
data(mybond)
# Show first lines of the data frame
head(mybond)
```

```
##                                year budget grossusa grosswrld rtime IMDB
## Dr. No                         1962    8.8    128.0     474.9   110  7.2
## From Russia With Love          1963   15.7    195.0     620.5   115  7.4
## Goldfinger                     1964   23.3    396.7     969.6   110  7.7
## Thunderball                    1965   68.8    485.9    1078.7   130  7.0
## You Only Live Twice            1967   68.4    310.5     310.5   117  6.9
## On Her Majesty's Secret Service 1969  45.9    149.5     573.1   140  6.7
##                                rottentomatoes          actor villaincnt
## Dr. No                                     95   Sean Connery     Canada
## From Russia With Love                      95   Sean Connery    Austria
## Goldfinger                                 99   Sean Connery    Germany
## Thunderball                                87   Sean Connery      Italy
## You Only Live Twice                        73   Sean Connery         UK
## On Her Majesty's Secret Service            81 George Lazenby         UK
##                                bondgirlcnt
## Dr. No                         Switzerland
## From Russia With Love                Italy
## Goldfinger                              UK
## Thunderball                         France
## You Only Live Twice                  Japan
## On Her Majesty's Secret Service         UK
```

```
# Show structure of the data frame
str(mybond)
```

```
## 'data.frame':    26 obs. of  10 variables:
##  $ year          : num  1962 1963 1964 1965 1967 ...
##  $ budget        : num  8.8 15.7 23.3 68.8 68.4 45.9 42.8 37.9 63.5 55.6 ...
##  $ grossusa      : num  128 195 397 486 310 ...
##  $ grosswrld     : num  475 620 970 1079 310 ...
##  $ rtime         : num  110 115 110 130 117 140 120 121 125 125 ...
##  $ IMDB          : num  7.2 7.4 7.7 7 6.9 6.7 6.6 6.8 6.7 7.1 ...
##  $ rottentomatoes: num  95 95 99 87 73 81 64 65 39 80 ...
##  $ actor         : Factor w/ 6 levels "Daniel Craig",..: 5 5 5 5 5 2 5 4 4 4 ...
##  $ villaincnt    : Factor w/ 10 levels "Austria","Canada",..: 2 1 5 6 9 9 9 10 9 5 ...
```

```
##  $ bondgirlcnt   : Factor w/ 9 levels "France","Italy",..: 7 2 8 1 3 8 9 8 6 9 ...
```

```
# Show data description
help(mybond)
```

Notice that the first 7 columns of the `mybond` data frame are numeric and the last 3 are factors. We'll ignore for a moment the 3 categorical variables - as if they were never there - and we'll focus on the 7 continuous only. Our **aim** is to find whether different films separate from each other based on these variables.

Let's examine Pearson correlations:

```
round(cor(mybond[,1:7]),2)
```

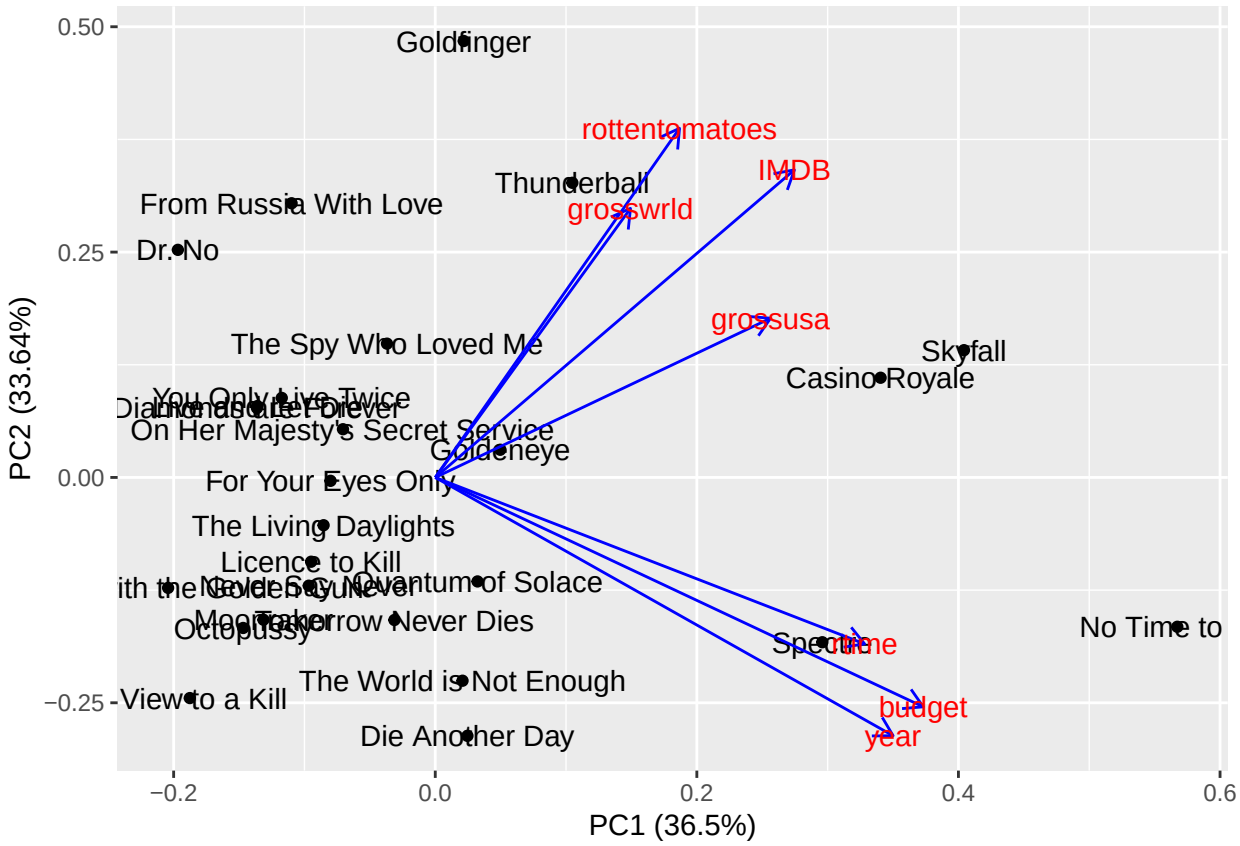
```
##                 year budget grossusa grosswrld rtime IMDB rottentomatoes
## year            1.00   0.93     0.04     -0.04  0.62 0.07          -0.14
## budget          0.93   1.00     0.28     -0.01  0.57 0.07          -0.11
## grossusa        0.04   0.28     1.00      0.25  0.19 0.40           0.38
## grosswrld      -0.04  -0.01     0.25      1.00 -0.09 0.50           0.41
## rtime           0.62   0.57     0.19     -0.09  1.00 0.16           0.02
## IMDB            0.07   0.07     0.40      0.50  0.16 1.00           0.79
## rottentomatoes -0.14  -0.11     0.38      0.41  0.02 0.79           1.00
```

We notice that some of the variables are strongly correlated, such as year and budget, and IMDB and Tomatometer ratings. Correlated variables represent the same characteristic of a cluster and irrelevant variables can conceal information about clusters.

To reduce the number of variables prior to clustering, we apply Principal Component Analysis (PCA). PCA allows us to create linear combinations of our original variables, principal components, which explain in our data as much variance as possible.

We apply PCA on the continuous variables of the James Bond films data set using the `prcomp()` R function. Because some of the variables are in different units, we set `center = TRUE` and `scale = TRUE` to standardize each input variable by subtracting its mean and dividing by its standard deviation.

```
# Apply Principal Component Analysis on the correlation matrix
outPCA <- prcomp(mybond[,1:7], center = TRUE, scale = TRUE)
# Create the PCA biplot with labeled observations
library("ggfortify")
autoplot(outPCA, data = mybond[,1:7],
         loadings = TRUE, loadings.colour = 'blue',
         loadings.label = TRUE, loadings.label.size = 4,label = TRUE)
```



We use the `autoplot()` function of the `ggfortify` package to represent both the films and their attributes on the same plot, the PCA biplot. The first two principal components explain around 70% of the total variance in the data. We observe that IMDB rating, Tomatometer score and box office gross in the USA and worldwide are positively correlated. Another group of correlated variables can be also detected: year of release, movie budget and running time. For instance, James Bond films located on right of the map are more recent, have higher budget and longer runtime, on average, than films on the left of the map.

Alternatively, we can also use the `PCAbipl()` function of the `UBbipl` package.

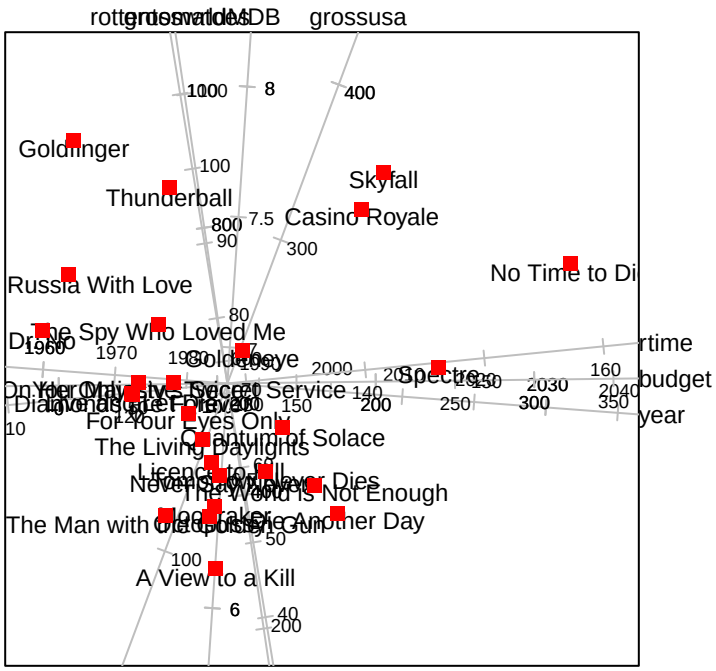
```
# Alternatively
library(UBbipl)
```

```
## Loading required package: rgl
```

```
## Loading required package: MASS
```

```
## Loading required package: numDeriv
```

```
outPCAbipl <- PCAbipl(mybond[,c(1:7)],scaled.mat = TRUE,colours = "red",pch.samples = 15,
                      pos = "Hor",label=TRUE, rotate.degrees = 35)
```



So far, PCA was used to project the observed variables onto a reduced subspace, which contains most of the variability in the data and a clustering algorithm can now be used to allocate observations to homogeneous clusters on the basis of principal components (PCs). Since PCs are uncorrelated and ordered, the first few PCs, which contain most of the variability in the data, are used in cluster analysis. A reasonable choice for a clustering algorithm is K-means. The aim of K-means is to find K disjoint clusters of observations, such that the distance to K cluster centers is minimized.

We apply K-means on the film scores on the first two PCs using the `kmeans()` function, setting the desired number of clusters (`centers`) to 3. The output of K-means is typically highly dependent on the initialization of the cluster centers: the algorithm can converge to local minima. For this reason, the algorithm should be run several times with many different random initializations (via the `nstart` argument), and the clustering with the smallest sum-of-squared distances between cluster centroids and observations is returned as the final result.

```
# Apply K-means on the PCA scores (2 dimensions) with K = 3 and 100 random starts
outK <- kmeans(outPCA$x[, 1:2], centers = 3, nstart = 100)
outK
```

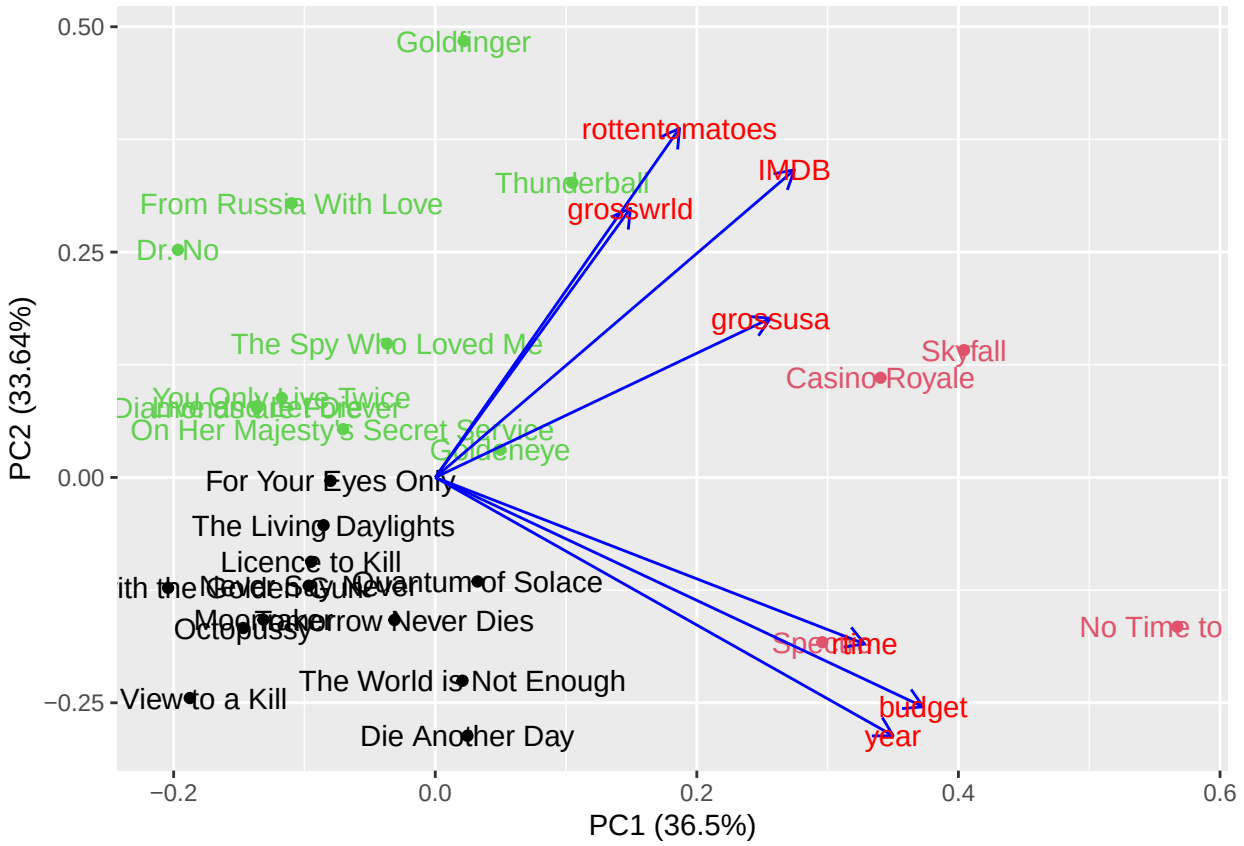
```
## K-means clustering with 3 clusters of sizes 12, 4, 10
##
## Cluster means:
##          PC1        PC2
## 1 -0.6660716 -1.1398830
## 2  3.2765314 -0.1891677
## 3 -0.5113267  1.4435267
##
## Clustering vector:
##                            Dr. No           From Russia With Love
##                                 3                               3
##                         Goldfinger                     Thunderball
##                                 3                               3
##              You Only Live Twice On Her Majesty's Secret Service
##                                 3                               3
##             Diamonds are Forever                Live and Let Die
```

```
##                               3                               3
##     The Man with the Golden Gun            The Spy Who Loved Me
##                               1                               3
##                        Moonraker              For Your Eyes Only
##                               1                               1
##                 Never Say Never                       Octopussy
##                               1                               1
##                A View to a Kill            The Living Daylights
##                               1                               1
##                 Licence to Kill                       Goldeneye
##                               1                               3
##             Tomorrow Never Dies         The World is Not Enough
##                               1                               1
##                 Die Another Day                   Casino Royale
##                               1                               2
##               Quantum of Solace                         Skyfall
##                               1                               2
##                         Spectre                 No Time to Die
##                               2                               2
##
## Within cluster sum of squares by cluster:
## [1]  9.009456  8.368220 17.911631
##  (between_SS / total_SS =  71.2 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

K-means resulted in three clusters of sizes 4, 10 and 12. `outK$cluster` is the cluster membership vector. The smallest cluster consists of *Casino Royale*, *Skyfall*, *Spectre* and *No time to Die*, four of the most recent James Bond films, which were located on the right of the PCA biplot.

Let's plot the three clusters of films using different colours in the two-dimensional space.

```
# Show the clusters on the PCA bibplot (with different colour)
autoplot(outPCA, data = mybond[,1:7], colour = outK$cluster,
         loadings = TRUE, loadings.colour = 'blue',
         loadings.label = TRUE, loadings.label.size = 4, label = TRUE)
```



Consequently, PCA followed by K-means allowed us to visually inspect the shape of the clusters and to assess the degree to which each variable contributes to the cluster structure in the data.

Tandem analysis, however, may not be always optimal, as the dimension reduction step optimises a different objective function from the clustering step. As a result, dimension reduction may mask the clustering structure. The ineffectiveness of tandem analysis has been argued by Arabie and Hubert (1994), Milligan (1996) and van de Velden et al. (2017), among others.

Joint Dimension Reduction and Clustering

Joint dimension reduction and clustering techniques aim to simultaneously find a low-dimensional representation of the data such that the cluster structure in the data is maximally revealed. Here's a (non-exhaustive) list based on the measurement scale of the input variables:

- Continuous variables: combine PCA with K-means
  - Reduced K-means (Bock 1987; De Soete & Carroll 1994)
  - Factorial K-means (Vichi & Kiers, 2001)
- Categorical variables: combine MCA with K-means
  - MCA K-means (Hwang, Dillon & Takane, 2006)
  - iFCB (Iodice D'Enza & Palumbo, 2013)
  - Cluster Correspondence Analysis (van Buuren & Heiser, 1989; van de Velden, Iodice D'Enza & Palumbo 2017)
- Mixture of continuous and categorical: combine PCAMIX/FAMD with K-means
  - Mixed Reduced K-means / Mixed Factorial K-means (van de Velden, Iodice D'Enza and Markos 2019; Vichi, Vicari and Kiers, 2019)

Implemented in R via the `clustrd` package (Markos, Iodice D'Enza, & van de Velden, 2019).

Reduced K-means

Reduced K-means searches a partitioning of the objects that minimizes the sum of the squared distances between the observed data and the 'quasi' centroids, that is centroids that are located in the reduced space. Let's apply Reduced K-means on our data using the `cluspca()` function.

```
# Apply Reduced K-means with 3 clusters and 2 dimensions
outRKM <- cluspca(data = mybond[, 1:7], nclus = 3, ndim = 2, nstart = 100, seed = 1234)
```

```
##   |                                                                      |
```

```
outRKM
```

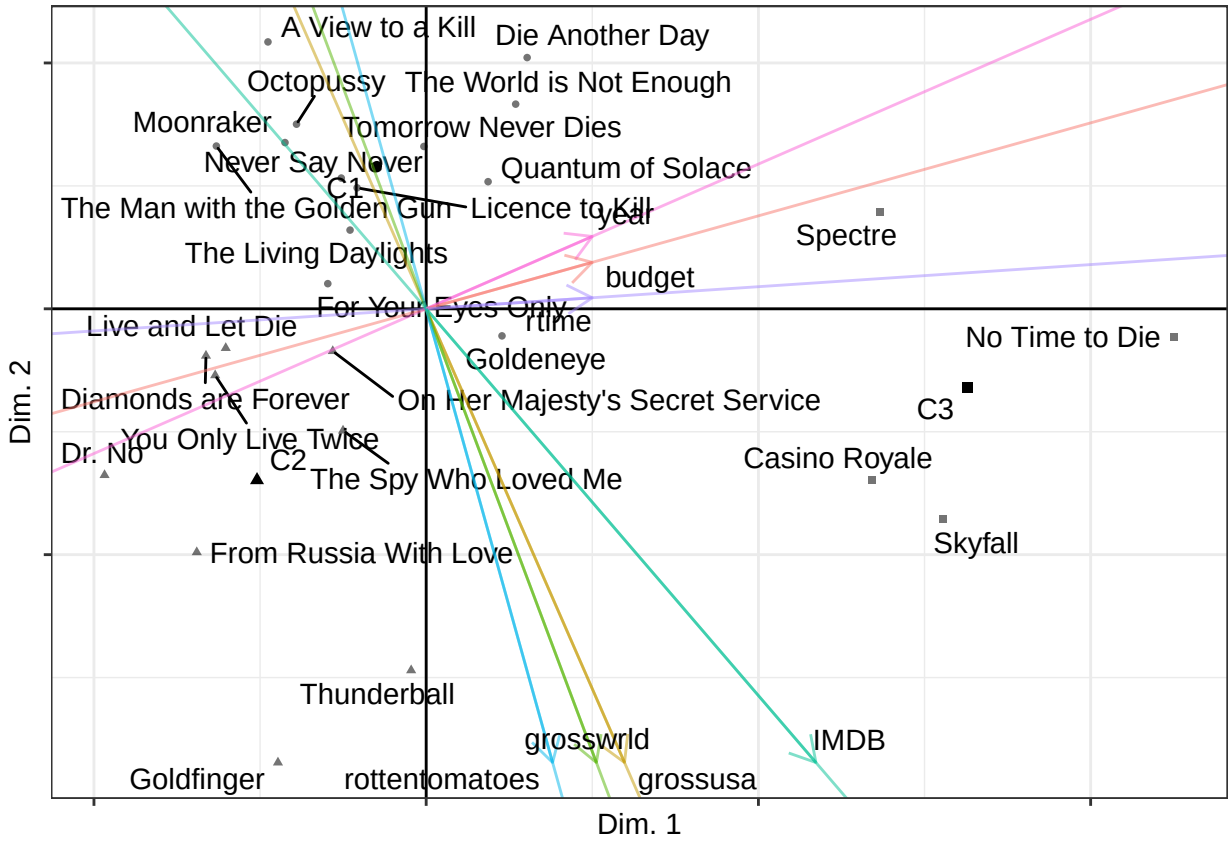
```
## Solution with 3 clusters of sizes 13 (50%), 9 (34.6%), 4 (15.4%) in 2 dimensions. Variables were mea
##
## Cluster centroids:
##             Dim.1   Dim.2
## Cluster 1 -0.2971  1.1605
## Cluster 2 -1.0184 -1.3935
## Cluster 3  3.2568 -0.6363
##
## Variable scores:
##                  Dim.1   Dim.2
## year            0.5522  0.3251
## budget          0.5328  0.2015
## grossusa        0.1325 -0.4097
## grosswrld       0.1137 -0.4096
## rtime           0.5208  0.0468
## IMDB            0.3129 -0.4922
## rottentomatoes 0.1079 -0.5232
##
## Within cluster sum of squares by cluster:
## [1] 10.0981 14.7080  6.1082
##  (between_SS / total_SS =  74.33 %)
##
## Objective criterion value: 42.7435
##
## Available output:
##
##  [1] "obscoord"  "attcoord"  "centroid"  "cluster"   "criterion" "size"
##  [7] "odata"     "scale"     "center"    "nstart"
```

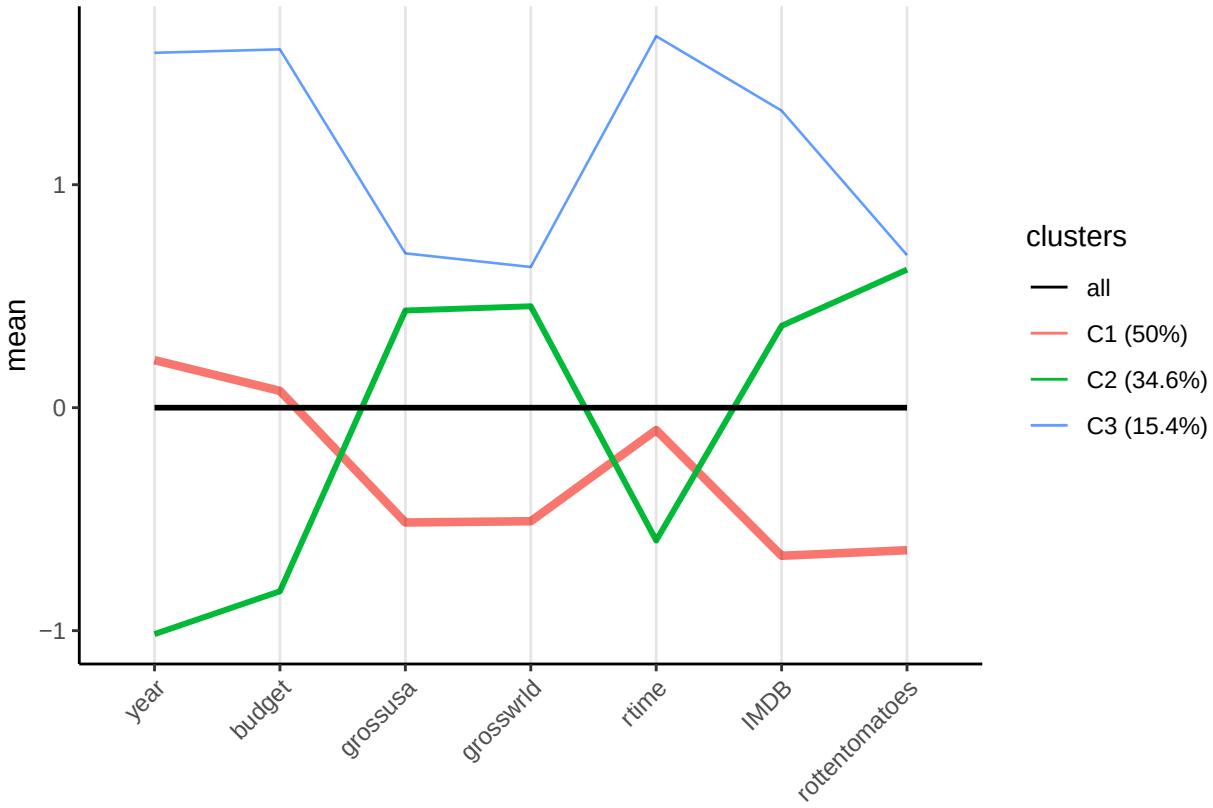
The `cluspca()` function has three arguments that should be provided: a data frame (`data`), the number of clusters (`nclus`) and the number of dimensions (`ndim`). Optional arguments include center, scale (as in `prcomp()`), `nstart` (the number of random starts), `smartStart` (a user-defined cluster membership vector) and `seed` (used for offsetting the random number generator).

We set the number of clusters to 3 and the number of dimensions to 2 (setting the number of dimensions to the number of clusters minus 1 is recommended in the literature) and plot the solution using the `plot()` function with the first argument being the output object of `cluspca()`. When cluster description argument `cludesc` is set to `TRUE`, it additionally creates a parallel coordinate plot to facilitate cluster interpretation. The argument `dim` controls which dimensions to plot. The default value is `dim = c(1, 2)`, i.e., the first two dimensions are plotted. The argument `what` specifies whether to visualize observations, variables or both; a different map is produced in each case. When `what = c(TRUE, FALSE)` a scatterplot of the observations and cluster centroids is obtained. A biplot is given when `what = c(FALSE, TRUE)`. The default option is `what = c(TRUE, TRUE)` and leads to a biplot where biplot axes (lines with arrows drawn from the origin) are used to represent the variables. Observation labels are plotted with `objlabs = TRUE` (default is `FALSE`). A vector of

custom attribute labels, `lbl`, can be provided via `attlabs = lbl`.

```
# Plot Reduced K-means solution with object labels
# Show parallel coordinate plot to facilitate cluster description
plot(outRKM, cludesc = TRUE, objlabs = TRUE)
```





The parallel coordinate plot (shown above) depicts cluster means for each variable within each cluster and provides quick insights into the differences between clusters. Note that the mean values, depicted on the vertical axis, correspond to mean centered and standardized variables.

- Cluster 1 (C1, 50%) contains James Bond films that were rated lower than average both from the audience and professional critics, and were less successful at the box-office.
- Cluster 2 (C2, 34.6%) consists of films that were released before 1977, had lower budget than average, but were quite successful at the box-office and received higher than average ratings, especially from critics.
- Cluster 3 (C3, 15.4%) contains four films that are among the most recent ones, had higher budget, runtime and got higher than average IMDB ratings.

Choosing the number of clusters and dimensions

The problem of deciding the number of clusters is not uniquely defined, and there is no unique "true" number of clusters. A large number of indices are available for evaluating the quality of a clustering based on the clustered data alone. To facilitate a quantitative appraisal of solutions corresponding to different parameter settings, the `clustrd` package provides two well-established distance-based internal validation indices via the the function `tuneclus()` and the `criterion` argument; the overall average silhouette width (ASW) index (`criterion = "asw"`) and the Calinski-Harabasz (CH) index (`criterion = "ch"`). The ASW index, which ranges from -1 to 1, reflects the compactness of the clusters and indicates whether a cluster structure is well separated or not. The CH index is the ratio of between-cluster variance to within-cluster variance, corrected according to the number of clusters, and takes values between 0 and infinity. In general, the higher the ASW and CH values, the better the cluster separation. The `tuneclus()` function requires a dataset (`data` argument), the range of clusters (`nclusrange`) and dimensions (`ndimrange`), as well as the desired method, through the `method` argument. Also, the function allows the specification of the data matrix used to compute the distances between observations, via the `dst` argument. When `dst = "full"` (default) the appropriate distance measure is computed on the original data. When the option is set to `"low"`, the distance is computed

between the low-dimensional object scores.

To demonstrate parameter tuning, we apply Reduced K-means to the James Bond films data for a range of clusters between 2 and 6 and dimensions ranging from 1 to 5. The average silhouette width was used as a cluster quality assessment criterion (`criterion = "asw"`). As shown below, the best solution was obtained for 2 clusters and a single dimension. This solution cannot be plotted, but in the clustering vector we identify the small cluster with the four more recent Bond films that was present in the three-cluster solution.

```
# Cluster quality assessment based on Average Silhouette Width
bestRKM <- tuneclus(data = mybond[, 1:7], nclusrange = 2:6, ndimrange = 1:5, method = "RKM",
criterion = "asw", seed = 1234)
```

```
## [1] "Running for 2 clusters and 1 dimensions..."
##   |                                                                      |
##   |                                                                      |
##   |                                                                      |
##   |                                                                      |
##   |                                                                      |
##   |                                                                      |
##   |                                                                      |
##   |                                                                      |
##   |                                                                      |
##   |                                                                      |
##   |                                                                      |
##   |                                                                      |
##   |                                                                      |
##   |                                                                      |
##   |                                                                      |
```

```
bestRKM
```

```
##
## The best solution was obtained for 2 clusters of sizes 22 (84.6%), 4 (15.4%) in 1 dimensions, for an
##
## Cluster quality criterion values across the specified range of clusters (rows) and dimensions (colum
##      X1    X2    X3    X4    X5
## 2 0.394
## 3 0.158  0.27
## 4 0.186 0.195 0.281
## 5 0.081 0.187 0.254 0.261
## 6 0.038 0.218 0.231 0.253 0.253
##
## The average Silhouette width values of each cluster are:
## [1] 0.42 0.24
##
## Cluster centroids:
##             Dim.1
## Cluster 1  0.6033
## Cluster 2 -3.3184
##
## Within cluster sum of squares by cluster:
## [1] 8.6471 2.1086
##  (between_SS / total_SS =  82.88 %)
##
## Objective criterion value: 61.4719
##
```

```
## Available output:
##
## [1] "clusobjbest" "nclusbest"   "ndimbest"    "critbest"    "critgrid"
## [6] "crit"        "cluasw"
```

```
# Show the cluster membership vector of the best solution
bestRKM$clusobjbest$cluster
```

```
##                          Dr. No           From Russia With Love
##                               1                               1
##                      Goldfinger                     Thunderball
##                               1                               1
##             You Only Live Twice On Her Majesty's Secret Service
##                               1                               1
##            Diamonds are Forever                Live and Let Die
##                               1                               1
##     The Man with the Golden Gun            The Spy Who Loved Me
##                               1                               1
##                        Moonraker             For Your Eyes Only
##                               1                               1
##                 Never Say Never                       Octopussy
##                               1                               1
##                A View to a Kill           The Living Daylights
##                               1                               1
##                 Licence to Kill                      Goldeneye
##                               1                               1
##             Tomorrow Never Dies         The World is Not Enough
##                               1                               1
##                Die Another Day                  Casino Royale
##                               1                               2
##               Quantum of Solace                        Skyfall
##                               1                               2
##                         Spectre                 No Time to Die
##                               2                               2
```

Cluster Correspondence Analysis

Cluster Correspondence Analysis (Cluster CA) combines MCA with K-means. Let's focus on the three categorical variables of the `mybond` data frame, ignoring the continuous variables.

```
# Show first lines of the categorical variables
head(mybond[, 8:10])
```

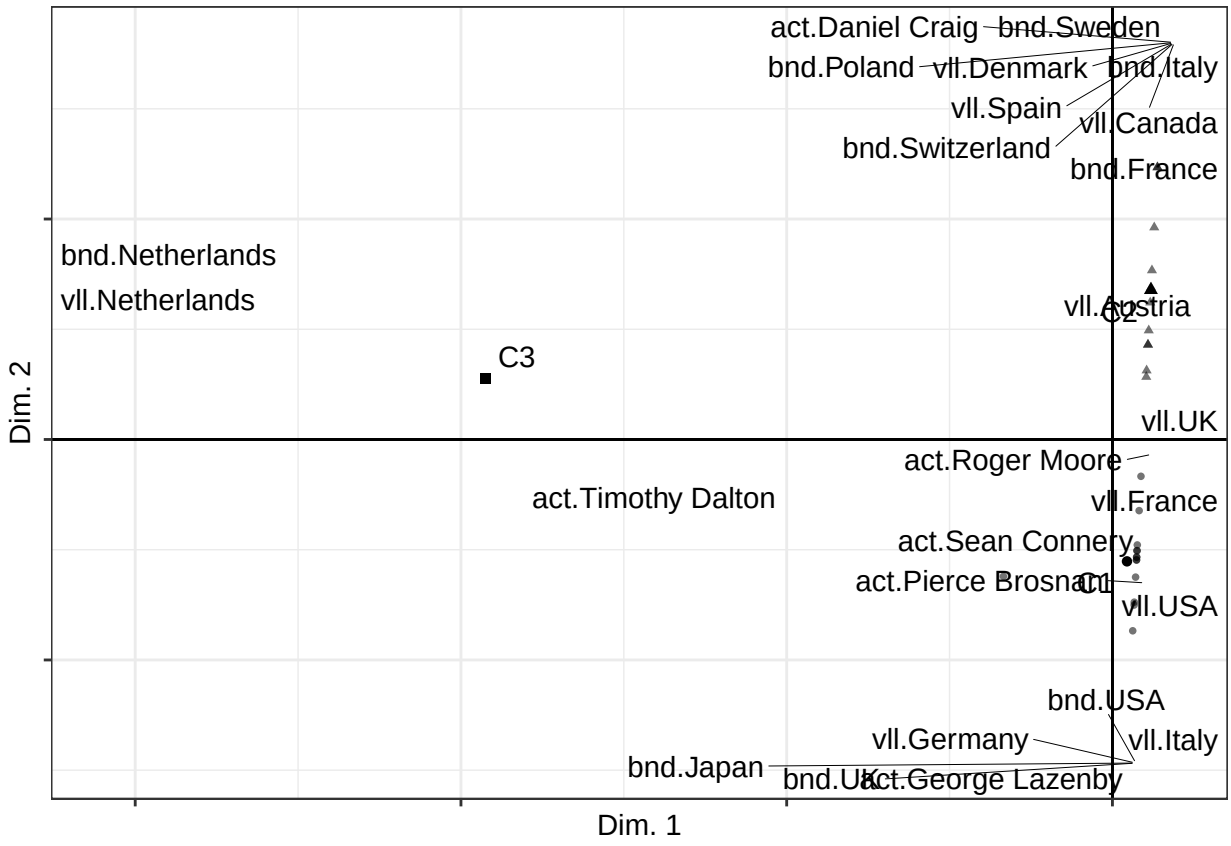
```
##                                      actor villaincnt bondgirlcnt
## Dr. No                         Sean Connery     Canada Switzerland
## From Russia With Love          Sean Connery    Austria       Italy
## Goldfinger                     Sean Connery    Germany          UK
## Thunderball                    Sean Connery      Italy      France
## You Only Live Twice            Sean Connery         UK       Japan
## On Her Majesty's Secret Service George Lazenby         UK          UK
```

We apply Cluster CA on this data with 3 clusters in 2 dimensions and plot the solution with the cluster description argument set to `TRUE` and the `max.overlaps` argument set to 20 so as to allow for more overlapping labels than 10 to appear on the map.

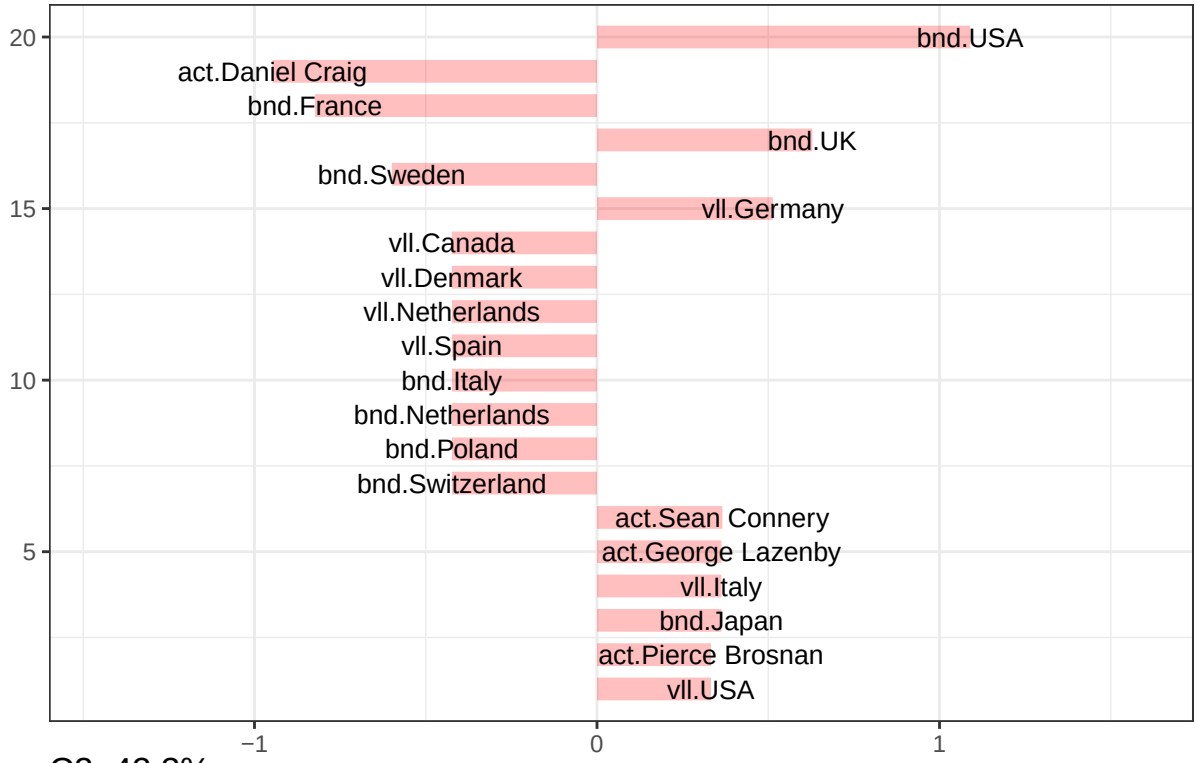
```
# Apply Cluster Correspondence Analysis with 3 clusters and 2 dimensions
outClusCA <- clusmca(mybond[, 8:10], 3, 2, method = "clusCA", seed = 1234)
```

```
##   |                                                                      |
```

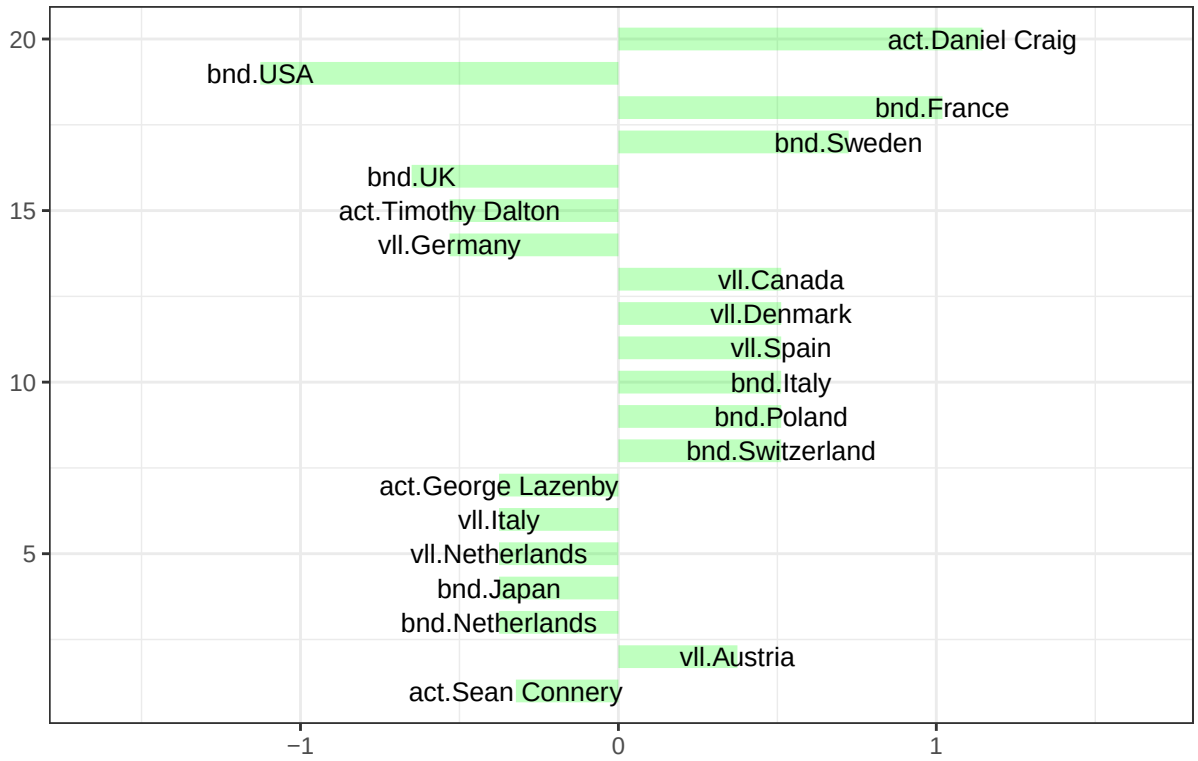
```
# Plot ClusCA solution (allow for overlapping labels)
# Show barplots with standardized residuals to facilitate cluster description
plot(outClusCA, cludesc = TRUE, max.overlaps = 20)
```

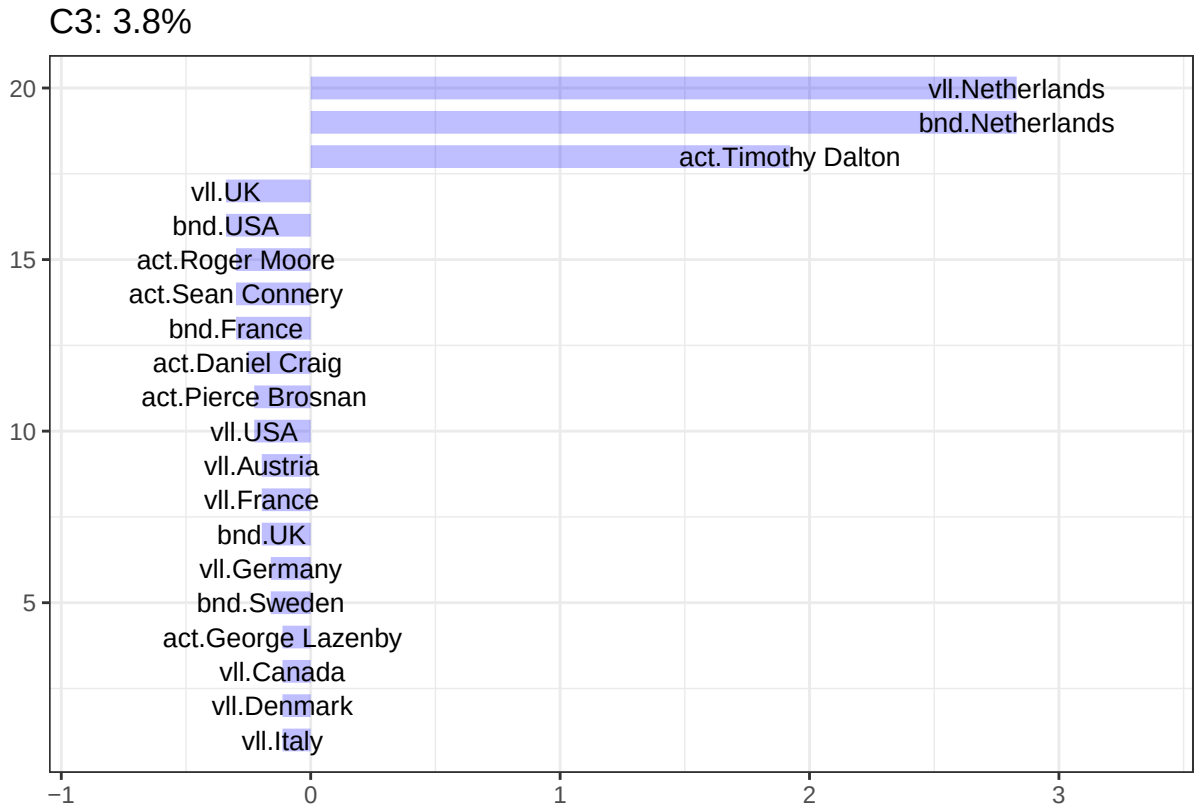


C1: 53.8%


bnd.USA
act.Daniel Craig
bnd.France
bnd.UK
bnd.Sweden
vll.Germany
vll.Canada
vll.Denmark
vll.Netherlands
vll.Spain
bnd.Italy
bnd.Netherlands
bnd.Poland
bnd.Switzerland
act.Sean Connery
act.George Lazenby
vll.Italy
bnd.Japan
act.Pierce Brosnan
vll.USA
20
15
10
5
−1
0
1

C2: 42.3%


act.Daniel Craig
bnd.USA
bnd.France
bnd.Sweden
bnd.UK
act.Timothy Dalton
vll.Germany
vll.Canada
vll.Denmark
vll.Spain
bnd.Italy
bnd.Poland
bnd.Switzerland
act.George Lazenby
vll.Italy
vll.Netherlands
bnd.Japan
bnd.Netherlands
vll.Austria
act.Sean Connery
20
15
10
5
−1
0
1



The map of cluster centroids and categories is displayed in the first Figure above. The method resulted in three clusters with 14, 11 and 1 films, respectively. To interpret the solution we consider individual categories (labelled as a combination of the abbreviated variable's name and the name of the category) and the positions of the cluster mean points relative to these. Note that this map has also a biplot interpretation. However, due to the existence of several categories (6 for Bond actor, 10 for villain country and 9 for Bond girl country), interpretation baseon on the map alone is not an easy task.

To facilitate interpretation, the three barplots show for each cluster the 20 categories with the highest standardized residuals (positive or negative). A positive (negative) residual means that the attribute has an above (below) average frequency within the cluster. We see that cluster 1 (53.8%) consists of films where the actor playing the Bond girl comes from USA or the UK and the actor playing the villain from Germany. Cluster 2 (42.3%) consists of films starring Daniel Craig and the Bond girl comes from France or Sweden. Cluster 3 (3.8%) contains a single film with Timothy Dalton.

Mixed Reduced K-means

Since our original data set had a mixture of continuous and categorical variables, the best strategy would be to apply Mixed Reduced K-means. Mixed Reduced K-means is the default method of choice in function `cluspcamix()`.

```
outMixedRKM <- cluspcamix(mybond, 3, 2, seed = 1234)
```

```
##   |                                                                      |
```

```
outMixedRKM
```

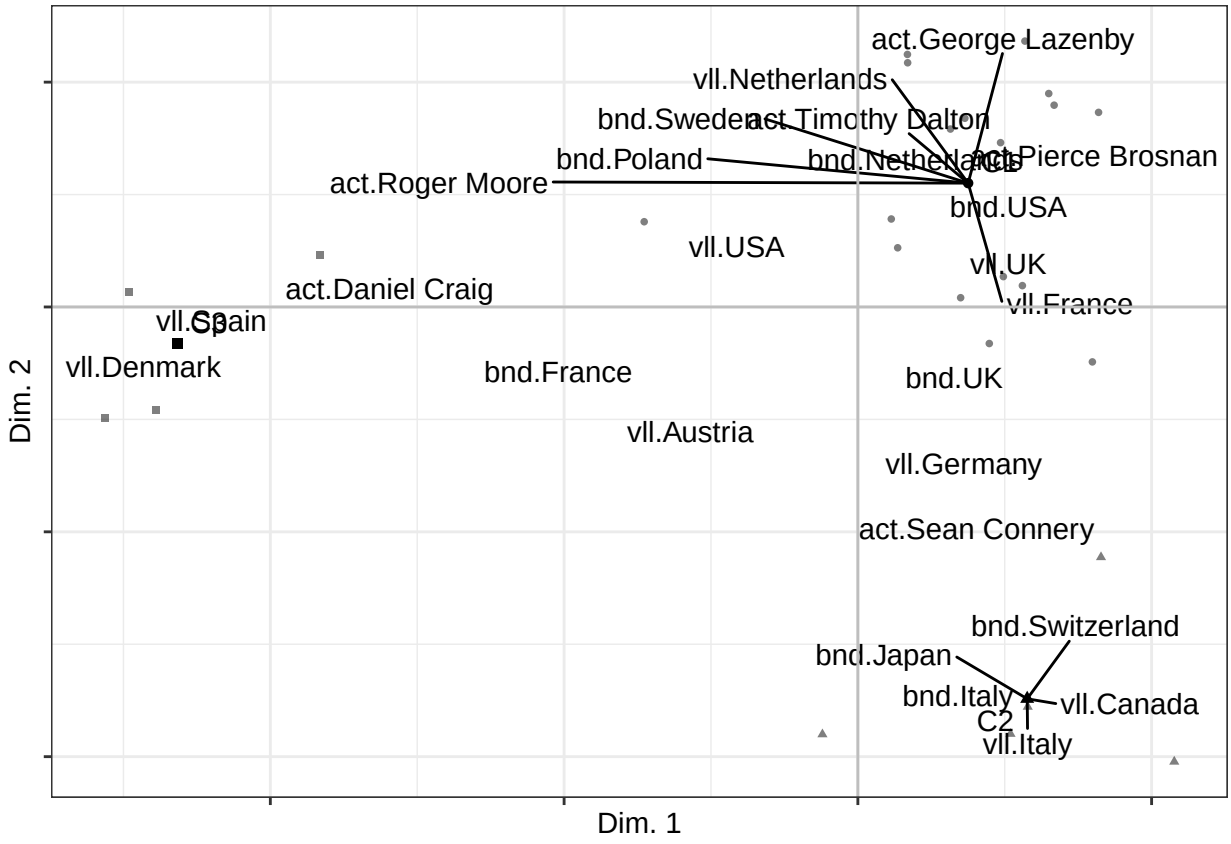
```
## Solution with 3 clusters of sizes 17 (65.4%), 5 (19.2%), 4 (15.4%) in 2 dimensions.
##
## Cluster centroids:
##             Dim.1   Dim.2
## Cluster 1  0.7509  1.1016
```

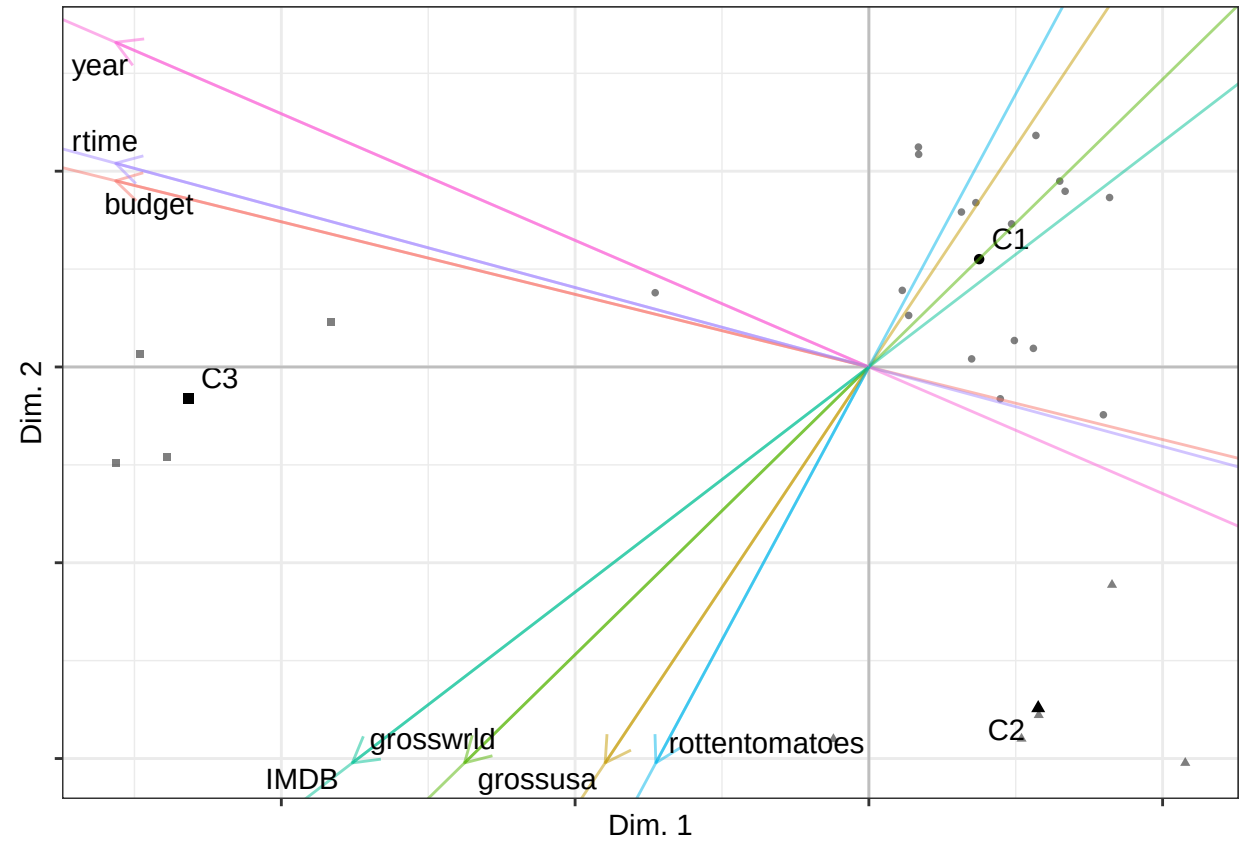
```
## Cluster 2  1.1529 -3.4853
## Cluster 3 -4.6324 -0.3253
##
##  Continuous variables:
##                  Dim.1   Dim.2
## year           -0.3671  0.2374
## budget         -0.3632  0.1347
## grossusa       -0.1316 -0.2965
## grosswrld      -0.1259 -0.1850
## rtime          -0.3776  0.1532
## IMDB           -0.2713 -0.3121
## rottentomatoes -0.1260 -0.3513
##
##  Categories:
##                       Dim.1   Dim.2
## act.Daniel Craig   -3.5557 -0.0399
## act.George Lazenby  0.7509  1.1016
## act.Pierce Brosnan  0.7509  1.1016
## act.Roger Moore     0.7509  1.1016
## act.Sean Connery    1.0380 -2.1747
## act.Timothy Dalton  0.7509  1.1016
## vll.Austria        -0.9095 -0.9030
## vll.Canada          1.1529 -3.4853
## vll.Denmark        -4.6324 -0.3253
## vll.France          0.7509  1.1016
## vll.Germany         0.9519 -1.1918
## vll.Italy           1.1529 -3.4853
## vll.Netherlands     0.7509  1.1016
## vll.Spain          -4.6324 -0.3253
## vll.UK              0.7955  0.5920
## vll.USA            -0.5949  0.7449
## bnd.France         -2.2678 -0.3691
## bnd.Italy           1.1529 -3.4853
## bnd.Japan           1.1529 -3.4853
## bnd.Netherlands     0.7509  1.1016
## bnd.Poland          0.7509  1.1016
## bnd.Sweden          0.7509  1.1016
## bnd.Switzerland     1.1529 -3.4853
## bnd.UK              0.8849 -0.4273
## bnd.USA             0.7509  1.1016
##
## Within cluster sum of squares by cluster:
## [1] 22.2219  5.3134  2.9284
##  (between_SS / total_SS =  85.79 %)
##
## Objective criterion value: 285.0722
##
## Available output:
##
##  [1] "obscoord"    "attcoord"    "attcatcoord" "centroid"    "cluster"
##  [6] "criterion"   "size"        "odata"       "scale"       "center"
## [11] "nstart"
```

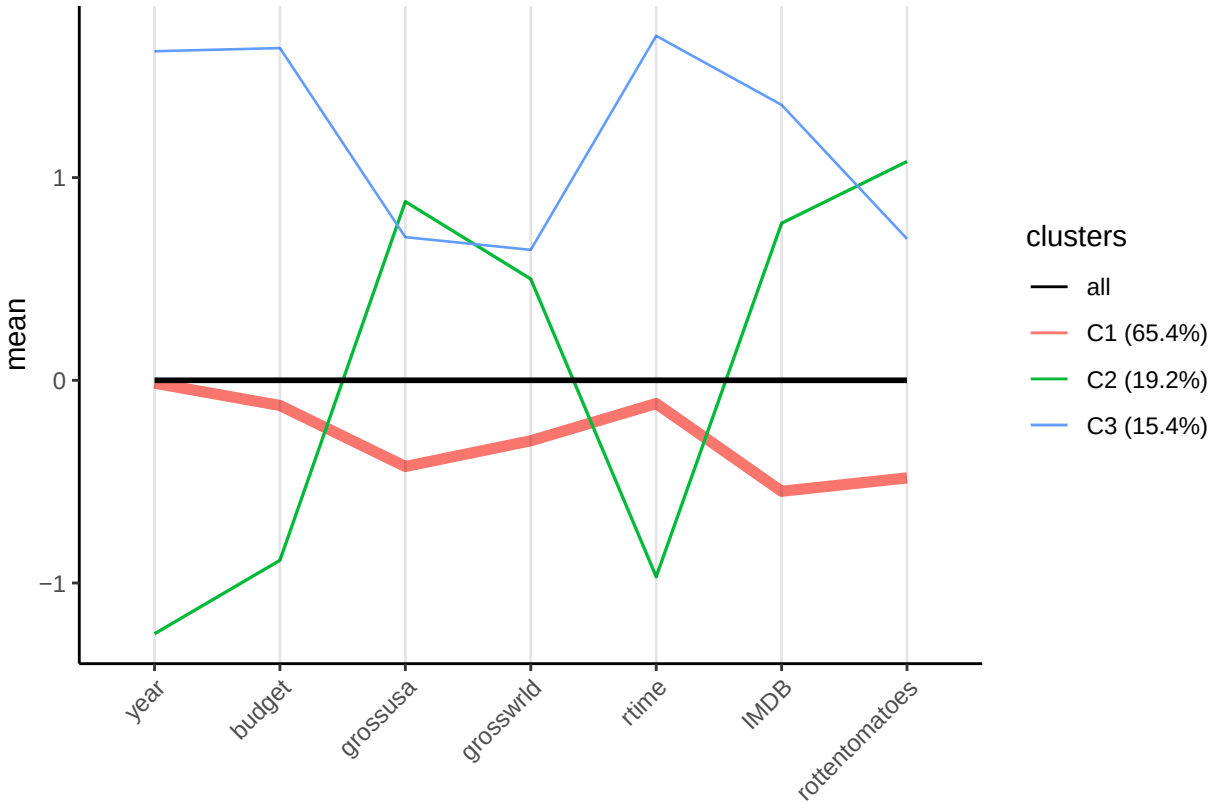
Mixed RKM resulted in 3 clusters of sizes 17 (65.4%), 5 (19.2%), 4 (15.4%). The output object contains the

variable coordinates and the coordinates of the categories. We subsequently plot the solution with `cludesc = TRUE`. The output contains a series of plots: a biplot of the continuous variables and observations (similar to that of Reduced K-means), a biplot of the categories (similar to that of Cluster CA), a parallel coordinate plot for interpreting the clusters with regard to the continuous variables and three standardized residual plots for interpreting the clusters with regard to the categorical variables.

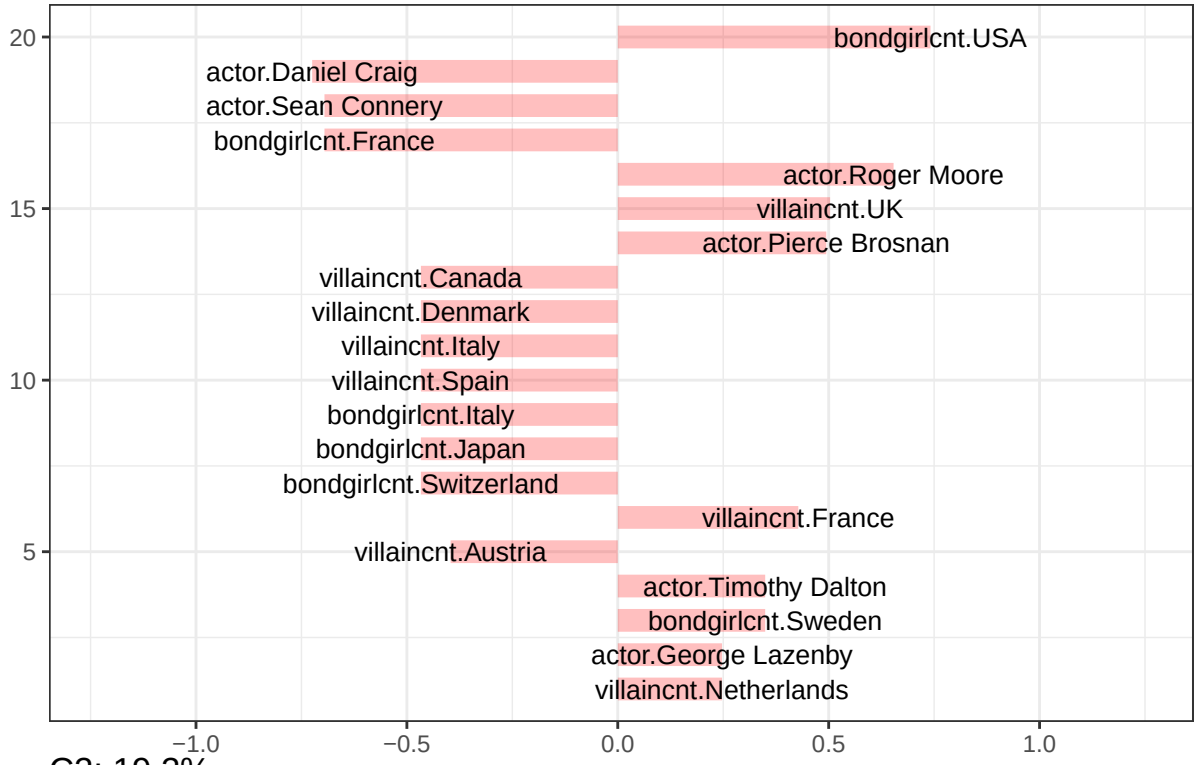
```
# Plot Mixed Reduced K-means solution (allow for overlapping labels)
plot(outMixedRKM, cludesc = TRUE, max.overlaps = 20)
```



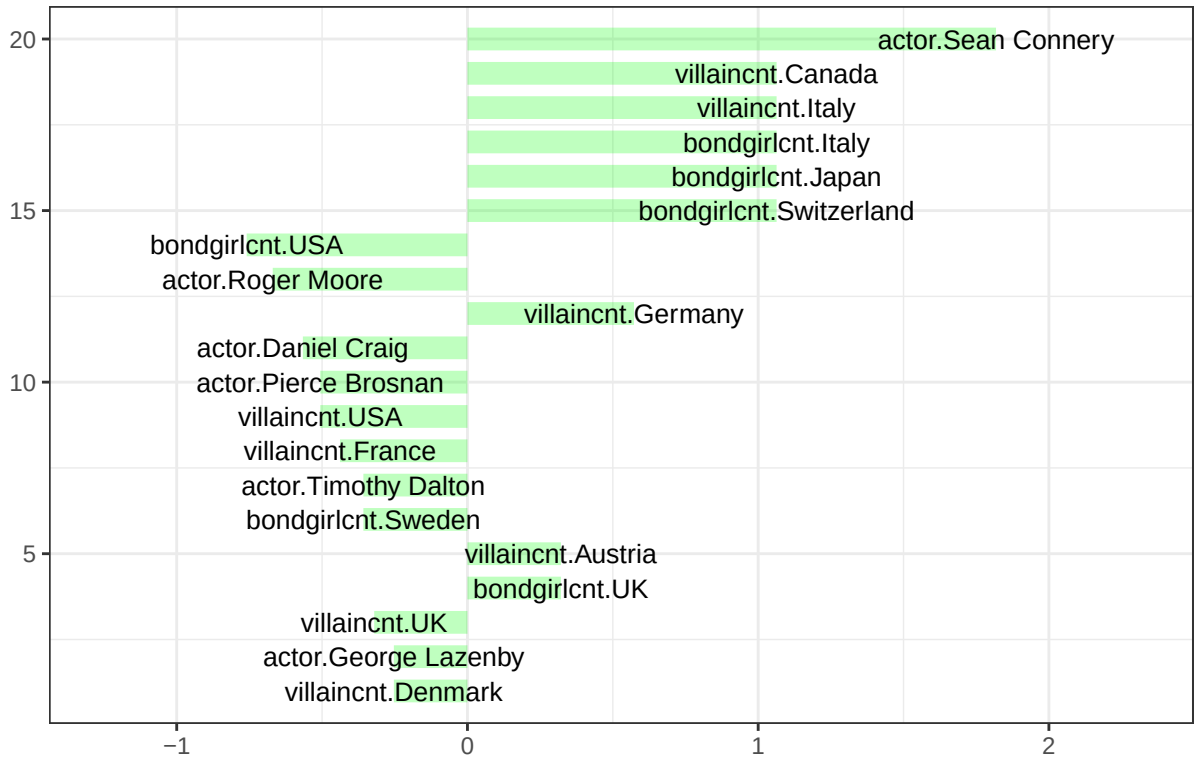

year
rtime
budget
C1
C3
Dim. 2
C2
grosswrld
rottentomatoes
IMDB
grossusa
Dim. 1

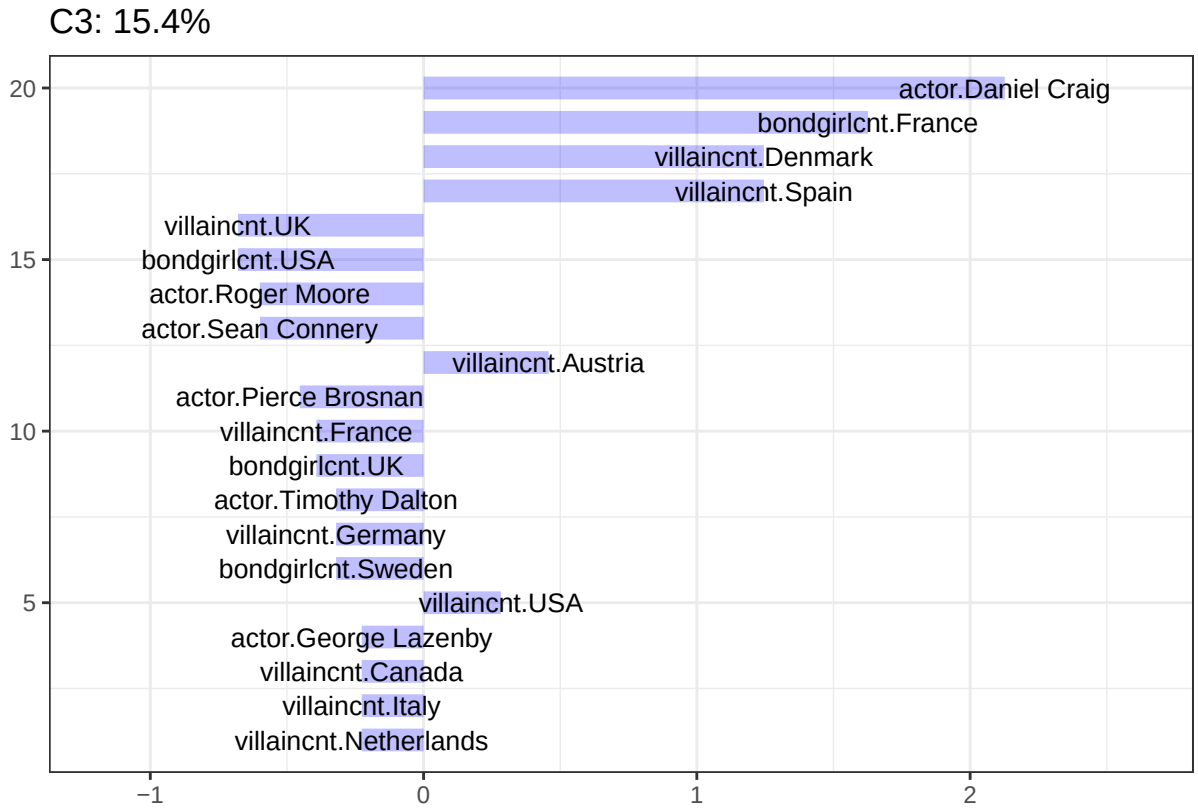

mean
1
0
−1
year
budget
grossusa
grosswrld
rtime
IMDB
rottentomatoes
clusters
all
C1 (65.4%)
C2 (19.2%)
C3 (15.4%)

C1: 65.4%


bondgirlcnt.USA
actor.Daniel Craig
actor.Sean Connery
bondgirlcnt.France
actor.Roger Moore
villaincnt.UK
actor.Pierce Brosnan
villaincnt.Canada
villaincnt.Denmark
villaincnt.Italy
villaincnt.Spain
bondgirlcnt.Italy
bondgirlcnt.Japan
bondgirlcnt.Switzerland
villaincnt.France
villaincnt.Austria
actor.Timothy Dalton
bondgirlcnt.Sweden
actor.George Lazenby
villaincnt.Netherlands
20
15
10
5
−1.0
−0.5
0.0
0.5
1.0

C2: 19.2%


actor.Sean Connery
villaincnt.Canada
villaincnt.Italy
bondgirlcnt.Italy
bondgirlcnt.Japan
bondgirlcnt.Switzerland
bondgirlcnt.USA
actor.Roger Moore
villaincnt.Germany
actor.Daniel Craig
actor.Pierce Brosnan
villaincnt.USA
villaincnt.France
actor.Timothy Dalton
bondgirlcnt.Sweden
villaincnt.Austria
bondgirlcnt.UK
villaincnt.UK
actor.George Lazenby
villaincnt.Denmark
20
15
10
5
−1
0
1
2



The interpretation is now more inclusive: Cluster 1 (C1, 65.4%) contains James Bond films that were rated lower than average both from the audience and professional critics, and were less successful at the box-office. These are films starring Roger Moore or Pierce Brosnan, the Bond girl comes from the USA and the villain from the UK. Cluster 2 (C2, 19.2%) consists of films that were released from 1962 to 1967, had lower budget than average, but were quite successful at the box-office and received higher than average ratings, from both the audience and critics. These are films starring Sean Connery, the Bond girl comes from Italy, Japan or Switzerland and the villain from Canada or Italy. Cluster 3 (C3, 15.4%) contains four of the most recently released films (2006 or later), had higher budget and runtime, and got a higher than average IMDB rating.

Assessing Cluster Stability

Global stability

Cluster stability is an important aspect of cluster validation. Our focus is on stability assessment of a partition, that is, given a new sample from the same population, how likely is it to obtain a similar clustering? Stability can also be used to select the number of clusters because if true clusters exist, the corresponding partition should have a high stability. Numbers of clusters that allow the clustering solution in its entirety to be reproduced in a stable manner across repeated calculations are more attractive than numbers of clusters leading to different clustering solutions across replications. Resampling approaches (that is, bootstrapping, subsetting, replacement of points by noise) provide an elegant framework to computationally derive the distribution of interesting quantities describing the quality of a partition (Hennig, 2007; Dolnicar and Leisch, 2017).

The function `global_bootclus()` can be used for assessing global cluster stability via bootstrapping. "Global" here refers to the stability of each partition over a range of clustering number choices, e.g., from 2 to 5 clusters. The algorithm implemented in `global_bootclus()` is similar to that of Dolnicar and Leisch (2010) and can be summarized as follows: two bootstrap samples of size n, the number of observations, are drawn with replacement from the original data set. Then, for a prespecified number of clusters and a number of dimensions, a joint dimension reduction and clustering method is applied to each sample and two clustering

partitions are obtained. Subsequently, each one of the original observations is assigned to the closest center of each one of the obtained clustering partitions, using the Euclidean distance, and two new partitions are obtained. Stability is defined as the degree of agreement between these two final partitions and can be evaluated via the Adjusted Rand Index (ARI; Hubert & Arabie, 1985) or the Measure of Concordance (MOC; Pfitzner, Leibbrandt, & Powers, 2009) as measure of agreement and stability. This procedure is repeated many times (e.g., 100) and the distributions of ARI and MOC can be inspected to assess the global reproducibility of different solutions.

To illustrate global cluster stability we apply Reduced K-Means (`method = RKM`) on the first seven variables of the James Bond films data set, with clusters ranging from 2 to 6 (`nclusrange = 2:6`), 10 bootstrap samples (`nboot = 10`) and 10 random starts (50 or 100 bootstrap samples work well in practice). The number of dimensions is internally set to the number of clusters minus 1, unless given using the argument `ndim`.

```
# Assess global cluster stability
# with 100 bootstrapping samples (this will take some time...)
boot_RKM <- global_bootclus(mybond[, 1:7], nclusrange = 2:6,method = "RKM",
           nboot = 10, nstart = 10, seed = 1234)
```

```
##
## [1] "nboot = 1"
##
## [1] "Running for 2 clusters and 1 dimensions."
##   |                                                                      |
## [1] "Running for 3 clusters and 2 dimensions."
##   |                                                                      |
## [1] "Running for 4 clusters and 3 dimensions."
##   |                                                                      |
## [1] "Running for 5 clusters and 4 dimensions."
##   |                                                                      |
## [1] "Running for 6 clusters and 5 dimensions."
##   |                                                                      |
## [1] "nboot = 2"
##
## [1] "Running for 2 clusters and 1 dimensions."
##   |                                                                      |
## [1] "Running for 3 clusters and 2 dimensions."
##   |                                                                      |
## [1] "Running for 4 clusters and 3 dimensions."
##   |                                                                      |
## [1] "Running for 5 clusters and 4 dimensions."
##   |                                                                      |
## [1] "Running for 6 clusters and 5 dimensions."
##   |                                                                      |
## [1] "nboot = 3"
##
## [1] "Running for 2 clusters and 1 dimensions."
##   |                                                                      |
## [1] "Running for 3 clusters and 2 dimensions."
##   |                                                                      |
## [1] "Running for 4 clusters and 3 dimensions."
##   |                                                                      |
## [1] "Running for 5 clusters and 4 dimensions."
##   |                                                                      |
## [1] "Running for 6 clusters and 5 dimensions."
##   |                                                                      |
```

```
## [1] "nboot = 4"
##
## [1] "Running for 2 clusters and 1 dimensions."
##   |                                                                      |
## [1] "Running for 3 clusters and 2 dimensions."
##   |                                                                      |
## [1] "Running for 4 clusters and 3 dimensions."
##   |                                                                      |
## [1] "Running for 5 clusters and 4 dimensions."
##   |                                                                      |
## [1] "Running for 6 clusters and 5 dimensions."
##   |                                                                      |
## [1] "nboot = 5"
##
## [1] "Running for 2 clusters and 1 dimensions."
##   |                                                                      |
## [1] "Running for 3 clusters and 2 dimensions."
##   |                                                                      |
## [1] "Running for 4 clusters and 3 dimensions."
##   |                                                                      |
## [1] "Running for 5 clusters and 4 dimensions."
##   |                                                                      |
## [1] "Running for 6 clusters and 5 dimensions."
##   |                                                                      |
## [1] "nboot = 6"
##
## [1] "Running for 2 clusters and 1 dimensions."
##   |                                                                      |
## [1] "Running for 3 clusters and 2 dimensions."
##   |                                                                      |
## [1] "Running for 4 clusters and 3 dimensions."
##   |                                                                      |
## [1] "Running for 5 clusters and 4 dimensions."
##   |                                                                      |
## [1] "Running for 6 clusters and 5 dimensions."
##   |                                                                      |
## [1] "nboot = 7"
##
## [1] "Running for 2 clusters and 1 dimensions."
##   |                                                                      |
## [1] "Running for 3 clusters and 2 dimensions."
##   |                                                                      |
## [1] "Running for 4 clusters and 3 dimensions."
##   |                                                                      |
## [1] "Running for 5 clusters and 4 dimensions."
##   |                                                                      |
## [1] "Running for 6 clusters and 5 dimensions."
##   |                                                                      |
## [1] "nboot = 8"
##
## [1] "Running for 2 clusters and 1 dimensions."
##   |                                                                      |
## [1] "Running for 3 clusters and 2 dimensions."
##   |                                                                      |
```

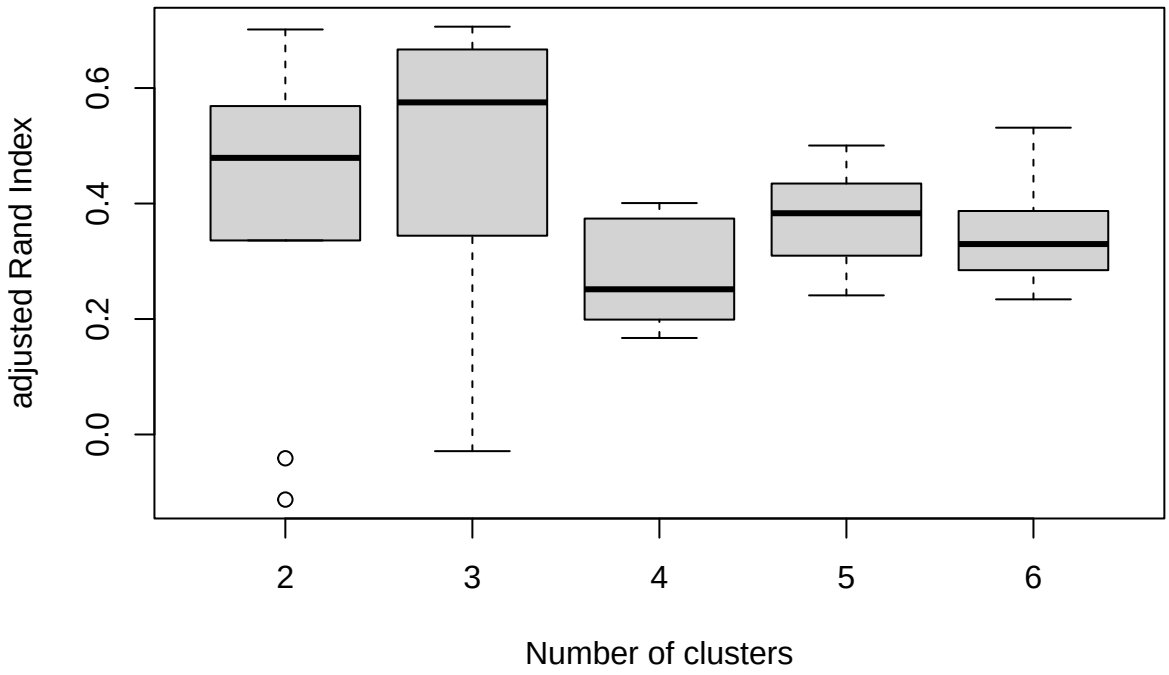
```
## [1] "Running for 4 clusters and 3 dimensions."
##   |                                                                      |
## [1] "Running for 5 clusters and 4 dimensions."
##   |                                                                      |
## [1] "Running for 6 clusters and 5 dimensions."
##   |                                                                      |
## [1] "nboot = 9"
##
## [1] "Running for 2 clusters and 1 dimensions."
##   |                                                                      |
## [1] "Running for 3 clusters and 2 dimensions."
##   |                                                                      |
## [1] "Running for 4 clusters and 3 dimensions."
##   |                                                                      |
## [1] "Running for 5 clusters and 4 dimensions."
##   |                                                                      |
## [1] "Running for 6 clusters and 5 dimensions."
##   |                                                                      |
## [1] "nboot = 10"
##
## [1] "Running for 2 clusters and 1 dimensions."
##   |                                                                      |
## [1] "Running for 3 clusters and 2 dimensions."
##   |                                                                      |
## [1] "Running for 4 clusters and 3 dimensions."
##   |                                                                      |
## [1] "Running for 5 clusters and 4 dimensions."
##   |                                                                      |
## [1] "Running for 6 clusters and 5 dimensions."
##   |                                                                      |
```

```
summary(boot_RKM$rand)
```

```
##        2                 3                 4                5
##  Min.   :-0.1127   Min.   :-0.02884   Min.   :0.1671   Min.   :0.2409
##  1st Qu.: 0.3574   1st Qu.: 0.34779   1st Qu.:0.1999   1st Qu.:0.3245
##  Median : 0.4791   Median : 0.57516   Median :0.2515   Median :0.3832
##  Mean   : 0.3975   Mean   : 0.47445   Mean   :0.2754   Mean   :0.3723
##  3rd Qu.: 0.5688   3rd Qu.: 0.66458   3rd Qu.:0.3663   3rd Qu.:0.4287
##  Max.   : 0.7014   Max.   : 0.70627   Max.   :0.4008   Max.   :0.5004
##        6
##  Min.   :0.2341
##  1st Qu.:0.2897
##  Median :0.3298
##  Mean   :0.3474
##  3rd Qu.:0.3819
##  Max.   :0.5313
```

A summary of the ARI distributions shows relatively unstable solutions (median ARIs range from .25 to .57). The three-cluster solution appears to be most stable. A visual inspection of the corresponding boxplots also confirms this finding.

```
# Create boxplots showing ARI distribution per clustering solution
boxplot(boot_RKM$rand, xlab = "Number of clusters", ylab = "adjusted Rand Index")
```



Local stability

Relying on global stability analysis could lead to selecting a cluster solution with suitable global stability, but without a single highly stable cluster. Dolnicar and Leisch (2017) recommend to also assess stability of clusters contained in those solutions to protect against discarding solutions containing interesting individual clusters from being prematurely discarded. For instance, in the context of market segmentation, most organisations only need one single target cluster, that is all they need to secure their survival and competitive advantage.

The criterion of cluster level stability within solutions is similar to the concept of global stability. The difference is that stability is computed at cluster level. Cluster level stability within solutions measures how often a cluster with the same characteristics is identified across a number of repeated calculations of cluster solutions with the same number of cluster. It is calculated by drawing several bootstrap samples, calculating cluster solutions independently for each of those bootstrap samples, and then determining the maximum agreement across all repeated calculations.

The algorithm implemented in `local_bootclus()` is similar to that of Hennig (2007) and can be summarized as follows: For a prespecified number of clusters and a number of dimensions, a joint dimension reduction and clustering method is applied to the original data. Then, two bootstrap samples of size n, the number of observations, are drawn with replacement from the original data set. The method is applied to each sample and two clustering partitions are obtained. Subsequently, each one of the original observations is assigned to the closest center of each one of the obtained clustering partitions, using the Euclidean distance, and two new partitions are obtained. The maximum Jaccard agreement between each original cluster and each one of the two partitions is calculated. The average of each pair is taken as a measure of "local" or cluster-wise stability. This procedure is repeated many times (e.g., 100) and the distributions of the Jaccard similarity can be inspected to assess the stability of each cluster in the solution.

Here are some guidelines for interpretation. Generally, a valid, stable cluster should yield a mean Jaccard similarity value of 0.75 or more. Between 0.6 and 0.75, clusters may be considered as indicating patterns in the data, but which points exactly should belong to these clusters is highly doubtful. Below average Jaccard values of 0.6, clusters should not be trusted. "Highly stable" clusters should yield average Jaccard similarities of 0.85 and above

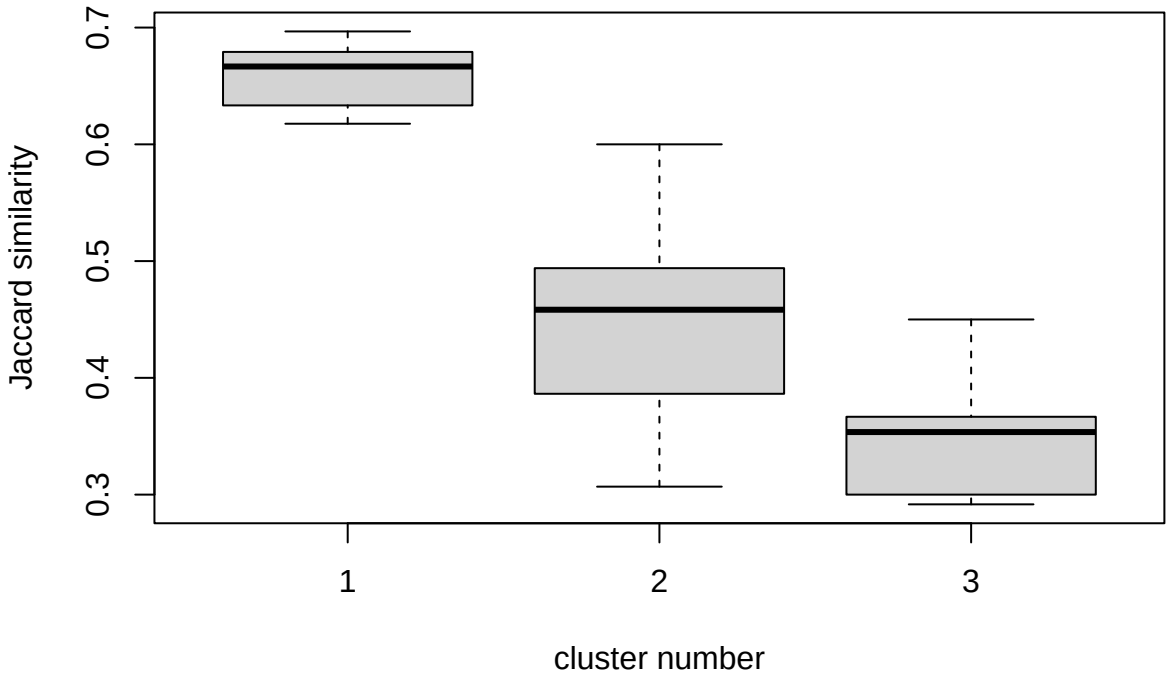
The function `local_bootclus` draws pairs of bootstrap samples, and returns the average agreement measured by the average Jaccard index for each pair. We assess cluster stability for the three-cluster solution of the

James Bond data set setting the number of clusters equal to 3 and the number of bootstrap samples to 10.

```
# Assess local cluster stability of the most stable solution
bootres <- local_bootclus(mybond[, 1:7], nclus = 3, ndim = 2,
                          method = "RKM", nboot = 10, nstart = 10, seed = 1234)
```

```
##
## [1] "nboot = 1"
##   |                                                                      |
## [1] "nboot = 2"
##   |                                                                      |
## [1] "nboot = 3"
##   |                                                                      |
## [1] "nboot = 4"
##   |                                                                      |
## [1] "nboot = 5"
##   |                                                                      |
## [1] "nboot = 6"
##   |                                                                      |
## [1] "nboot = 7"
##   |                                                                      |
## [1] "nboot = 8"
##   |                                                                      |
## [1] "nboot = 9"
##   |                                                                      |
## [1] "nboot = 10"
##   |                                                                      |
```

```
# Create boxplots showing ARI distribution per cluster
boxplot(bootres$Jaccard, xlab = "cluster number", ylab = "Jaccard similarity")
```



The most stable cluster appears to be the first and larger cluster (50%), containing films that were rated lower than average both from the audience and professional critics, and were less successful at the box-office.

Clustering Airbnb listings in Cape Town

To understand the rental landscape in Cape Town, we'll apply Mixed Reduced K-means to an Airbnb listings data set sourced from the Inside Airbnb website.

Data Overview

Airbnb does not release any data on the listings in its marketplace, but a separate group named Inside Airbnb scrapes and compiles publicly available information about many cities Airbnb's listings from the Airbnb web-site. For this project, a cleaned version of their data set scraped on October 26, 2021, on the city of Cape Town, South Africa, is used. It contains information on Cape Town Airbnb listings that were live on the site on that date.

The data set comprises of 11, 833 Airbnb listings and 34 attributes for each of the listings.

27 variables are continuous:

longitude, latitude, host_duration (in days), host_listings_count, accommodates, bathrooms, bedrooms, beds, price_per_guest (log-transformed), number of amenities, minimum nights, maximum_nights, availability within 30/60/90/365 days, instant_bookable, number_of_reviews, reviews_per _month, review_scores (accuracy, cleanliness, checkin, etc.),

and 7 are categorical.

host_response_time, is_superhost, host_verifications, listing_type, host_identity_verified, property_type, room_type,

Longitude and latitude of the location where the listed properties are situated will not be used in clustering, but will be used as external variables to inform the clustering. Mixed Reduced K-means is applied with 3 clusters and 2 dimensions.

```
# Clustering Airbnb listings in Cape Town
# Import the data from URL
library(rio)
airbnb <- import("http://amarkos.gr/datasets/airbnb.Rdata")

# Show data frame structure
str(airbnb)
```

```
## 'data.frame':    11833 obs. of  34 variables:
##  $ latitude                     : num  -34.2 -34.2 -34 -34 -34 ...
##  $ longitude                    : num  18.5 18.5 18.5 18.3 18.3 ...
##  $ host_since                   : num  1819 1819 559 1474 1474 ...
##  $ accommodates                 : num  3 5 2 2 2 2 2 2 2 10 ...
##  $ bathrooms                    : num  1 2 1 1 1 1 1 2 1 3 ...
##  $ bedrooms                     : num  0 2 1 1 1 0 1 1 0 4 ...
##  $ beds                         : num  1 2 1 1 1 1 1 1 0 6 ...
##  $ price_per_guest              : num  583 524 288 300 350 350 375 400 415 85 ...
##  $ minimum_nights               : num  2 2 2 7 7 2 5 1 2 1 ...
##  $ maximum_nights               : num  1125 1125 1125 31 31 ...
##  $ availability_30              : num  0 0 4 11 27 17 0 0 22 28 ...
##  $ availability_60              : num  2 2 4 16 31 38 0 0 22 54 ...
##  $ availability_90              : num  2 2 12 19 31 44 0 0 22 84 ...
##  $ availability_365             : num  217 218 287 109 87 295 0 208 22 359 ...
##  $ number_of_reviews            : num  10 7 7 4 4 79 45 3 45 4 ...
##  $ number_of_reviews_ltm        : num  0 0 7 3 2 19 0 3 4 4 ...
##  $ number_of_reviews_l30d       : num  0 0 0 0 0 2 0 0 2 0 ...
##  $ review_scores_rating         : num  5 5 4.71 4.75 4 4.95 4.98 5 4.49 5 ...
##  $ review_scores_accuracy       : num  4.8 5 4.71 5 3.75 4.91 4.91 5 4.67 5 ...
```

```
##  $ review_scores_cleanliness     : num  5 5 4.57 4.75 3.25 4.96 4.96 5 4.64 5 ...
##  $ review_scores_checkin         : num  5 5 4.57 5 5 4.97 4.89 5 4.89 5 ...
##  $ review_scores_communication   : num  5 5 4.71 5 5 5 4.98 5 4.69 5 ...
##  $ review_scores_location        : num  4.8 4.83 4.86 5 4.25 4.96 4.98 5 4.71 5 ...
##  $ review_scores_value           : num  4.8 4.83 4.86 5 4 4.94 4.91 5 4.62 5 ...
##  $ calculated_host_listings_count: num  2 2 3 2 2 1 1 2 1 2 ...
##  $ reviews_per_month             : num  0.21 0.15 0.7 0.38 0.23 1.81 1.03 0.28 1.19 0.53 ...
##  $ n_amenities                   : num  32 30 21 34 43 54 36 33 29 16 ...
##  $ host.response.time            : Factor w/ 5 levels "host.response.N/A",..: 1 1 4 5 5 5 1 1 5 1 ..
##  $ host_is_superhost             : Factor w/ 2 levels "superhost.no",..: 1 1 1 1 1 2 1 2 1 1 ...
##  $ id_for_verification           : Factor w/ 2 levels "id_no","id_yes": 2 2 2 2 2 2 2 2 2 2 ...
##  $ hoshost_verified              : Factor w/ 2 levels "host_verified_no",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ property_type                 : Factor w/ 70 levels "Barn","Camper/RV",..: 16 16 14 16 10 11 16 1
##  $ room_type                     : Factor w/ 4 levels "Entire home/apt",..: 1 1 1 1 1 1 1 1 3 ...
##  $ instant_bookable              : Factor w/ 2 levels "instant_book_no",..: 1 1 2 1 2 1 1 1 1 1 ...
```

```
# Apply Mixed Reduced 3-means
# Exclude longitude and lattitude (1st and 2nd columns)
outMixedRKM3 <- cluspcamix(airbnb[,-c(1:2)], 3, 2, nstart = 3, seed = 1234)
```

```
##   |                                                                              |
```

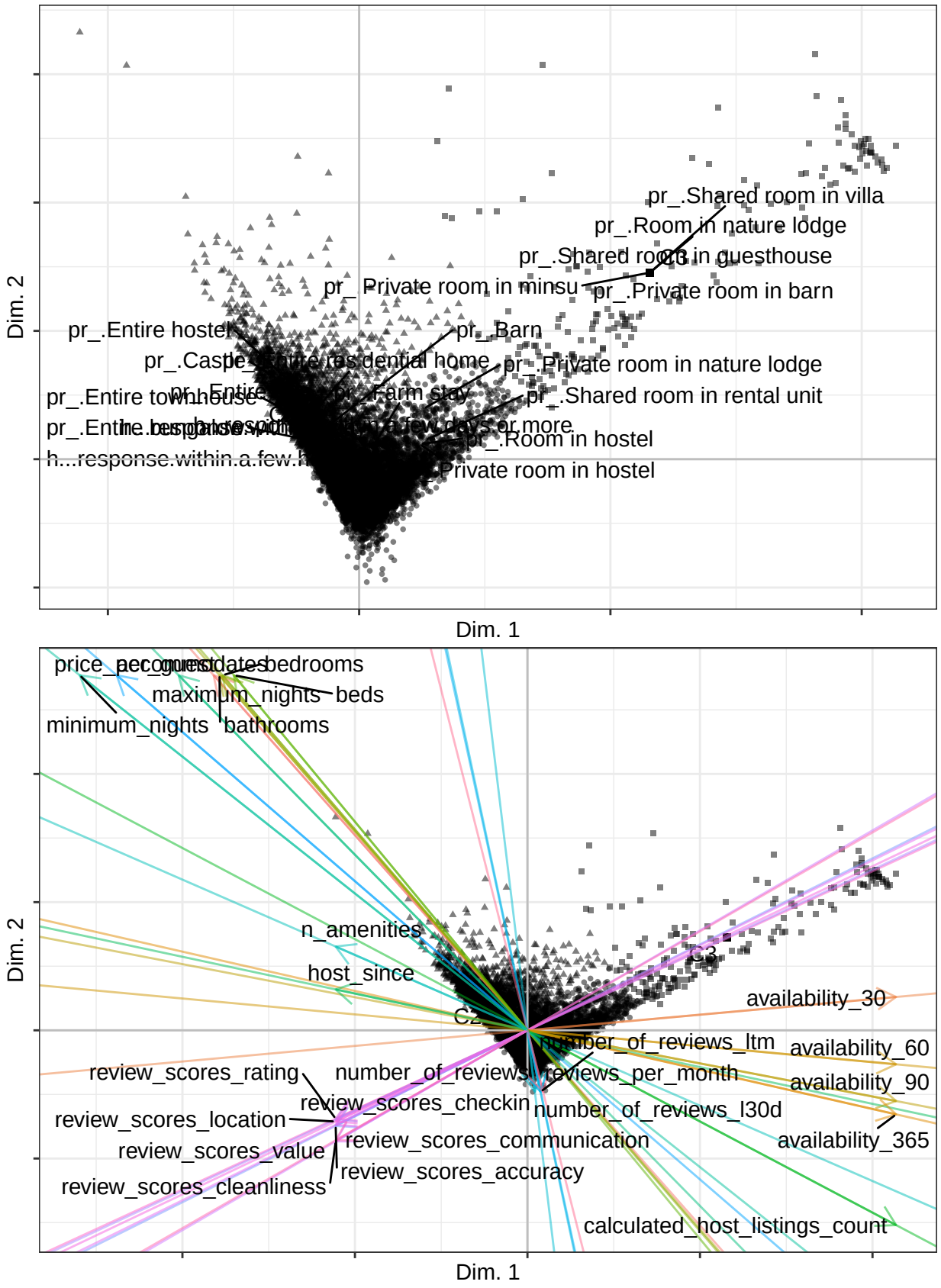
```
table(outMixedRKM3$cluster)
```

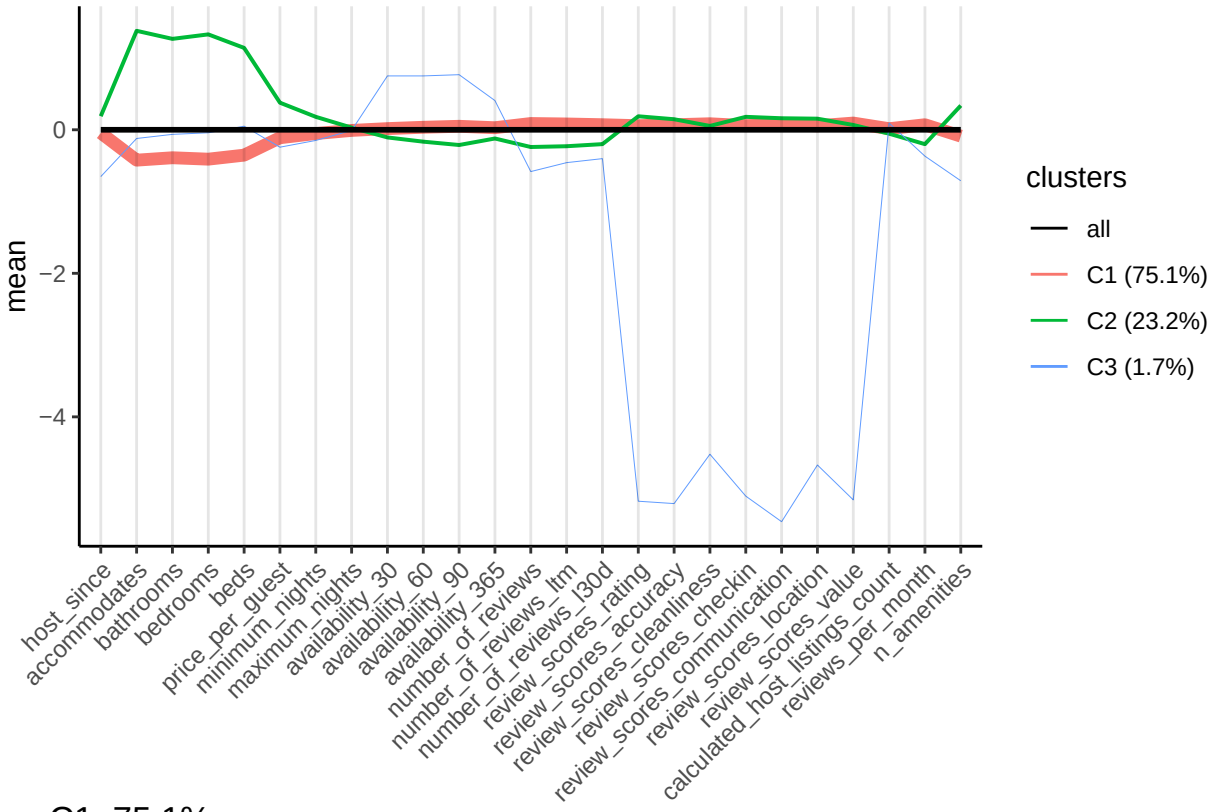
```
##
##    1    2    3
## 8889 2747  197
```

Mixed Reduced K-means partitioned the listings into three clusters of sizes 8889 (75.1%), 2747 (23.2%) and 197 (1.7%) in 2 dimensions. The biplots are expected to be crowded, therefore we'll try to interpret the clusters based on the parallel coordinate plot and the three standardized residual plots.

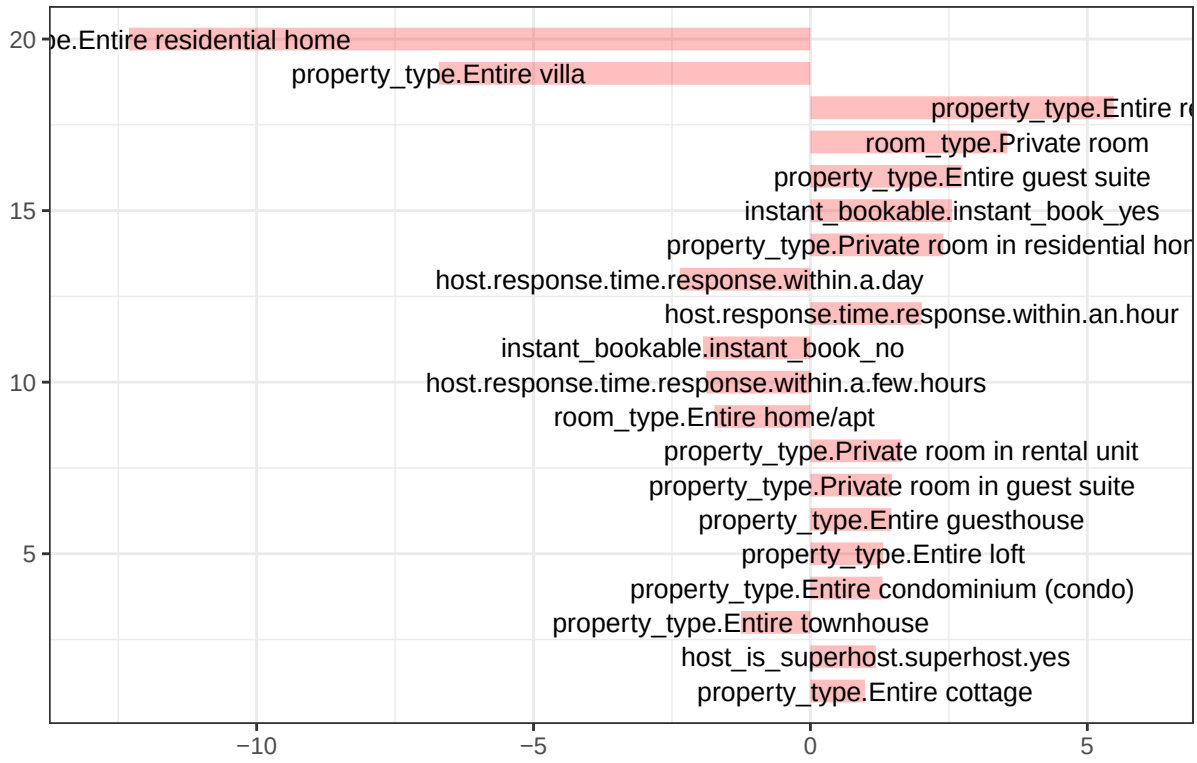
```
plot(outMixedRKM3, cludesc = TRUE, max.overlaps = 80)
```

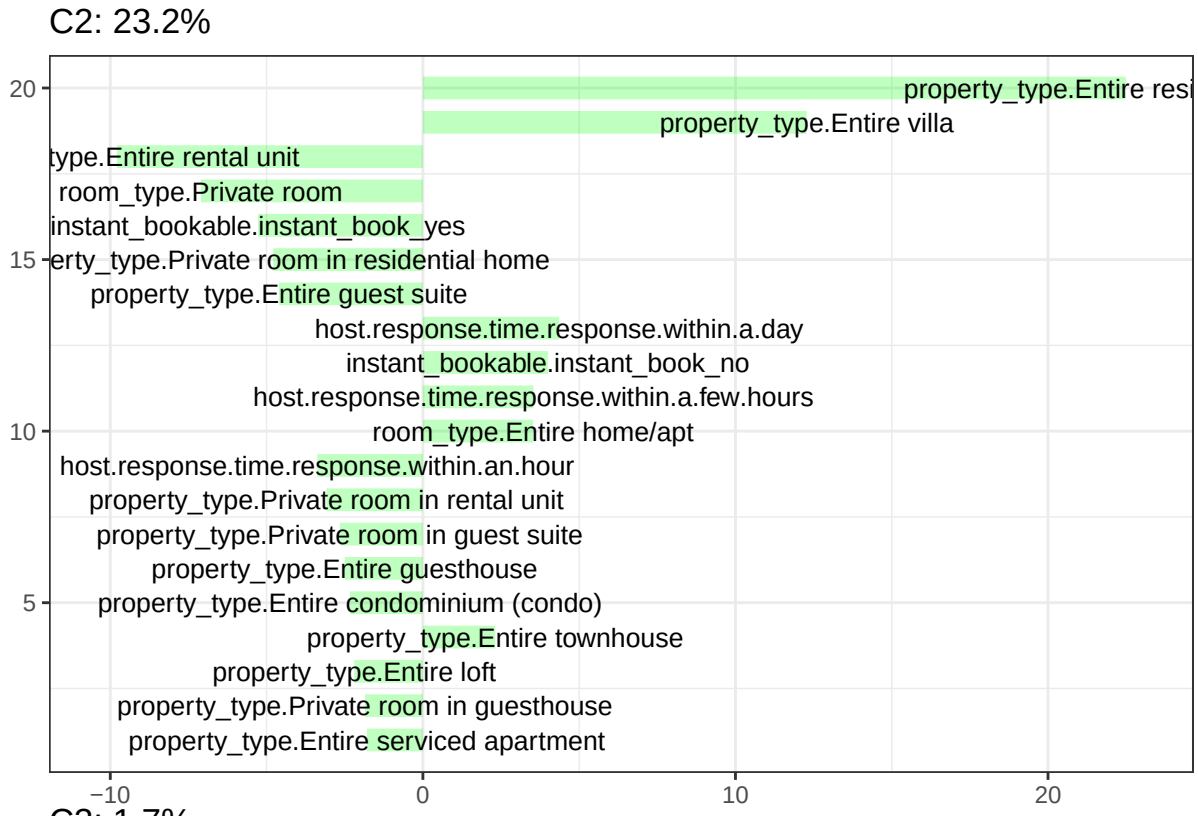
```
## Warning: ggrepel: 67 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

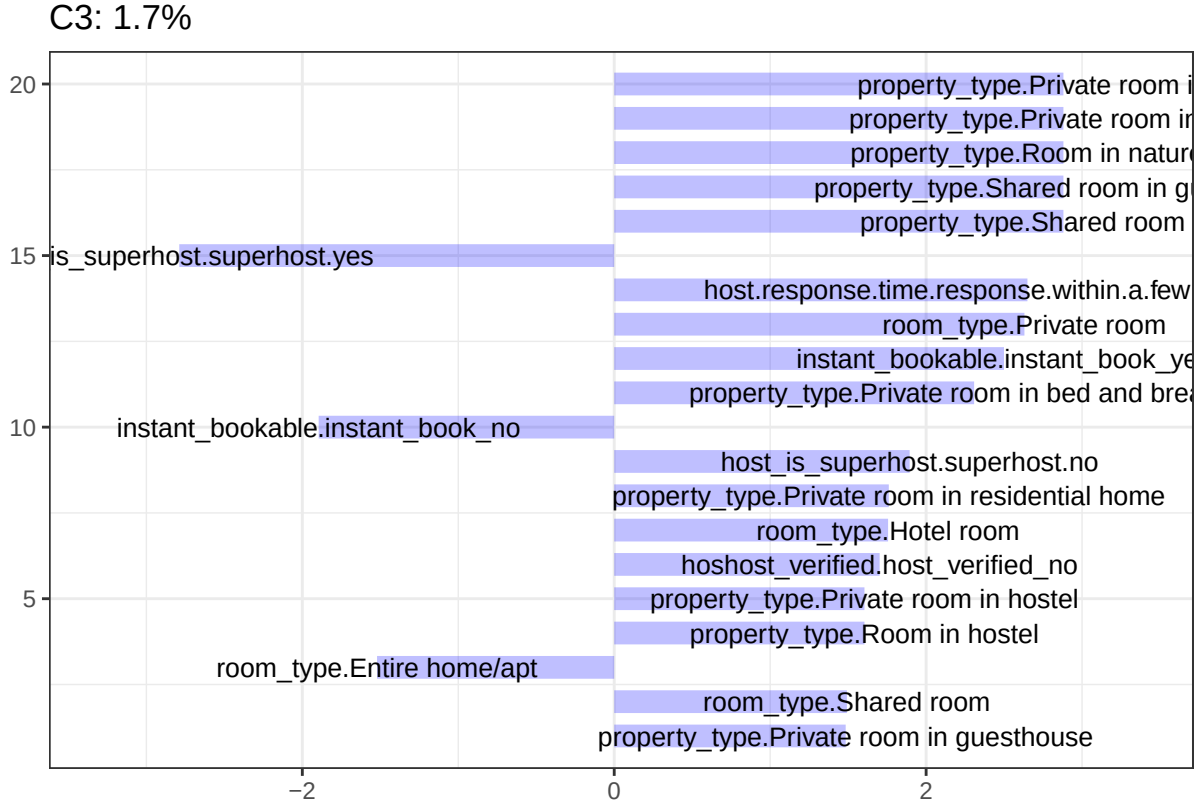

pr_.Shared room in villa
pr_.Room in nature lodge
pr_.Shared room in guesthouse
pr_ Private room in minsu
pr_.Private room in barn
pr_.Entire hostel
pr_ Barn
pr_.Private room in nature lodge
pr_.Shared room in rental unit
pr_.Room in hostel
Private room in hostel
Dim. 2
Dim. 1
price_per_guest
accommodates
bedrooms
maximum_nights
beds
minimum_nights
bathrooms
n_amenities
host_since
availability_30
number_of_reviews_ltm
availability_60
review_scores_rating
number_of_reviews
reviews_per_month
availability_90
review_scores_checkin
review_scores_location
number_of_reviews_l30d
review_scores_communication
availability_365
review_scores_value
review_scores_accuracy
review_scores_cleanliness
calculated_host_listings_count
Dim. 2
Dim. 1


mean
0
−2
−4
host_since
accommodates
bathrooms
bedrooms
beds
price_per_guest
minimum_nights
maximum_nights
availability_30
availability_60
availability_90
availability_365
number_of_reviews
number_of_reviews_ltm
number_of_reviews_l30d
review_scores_rating
review_scores_accuracy
review_scores_cleanliness
review_scores_checkin
review_scores_communication
review_scores_location
review_scores_value
calculated_host_listings_count
reviews_per_month
n_amenities
clusters
all
C1 (75.1%)
C2 (23.2%)
C3 (1.7%)

C1: 75.1%


20
15
10
5
pe.Entire residential home
property_type.Entire villa
property_type.Entire r
room_type.Private room
property_type.Entire guest suite
instant_bookable.instant_book_yes
property_type.Private room in residential hor
host.response.time.response.within.a.day
host.response.time.response.within.an.hour
instant_bookable.instant_book_no
host.response.time.response.within.a.few.hours
room_type.Entire home/apt
property_type.Private room in rental unit
property_type.Private room in guest suite
property_type.Entire guesthouse
property_type.Entire loft
property_type.Entire condominium (condo)
property_type.Entire townhouse
host_is_superhost.superhost.yes
property_type.Entire cottage
−10
−5
0
5





We can also examine and compare the means of the continuous variables for each cluster:

```r
# Compute the mean of each continuous variable for each cluster
round(aggregate(airbnb[,c(3:27)], by = list(outMixedRKM3$cluster), mean), 2)
```

```
##   Group.1 host_since accommodates bathrooms bedrooms beds price_per_guest
## 1       1    1980.97         2.77      1.27     1.26 1.65          410.88
## 2       2    2170.08         7.07      3.09     3.59 4.55          643.60
## 3       3    1487.01         3.49      1.63     1.76 2.43          348.23
##   minimum_nights maximum_nights availability_30 availability_60 availability_90
## 1           3.11         729.69           13.29           28.53           44.98
## 2           5.48        4302.10           11.82           24.12           36.91
## 3           2.13         729.69           22.04           44.26           67.23
##   availability_365 number_of_reviews number_of_reviews_ltm
## 1           218.20             28.61                 5.91
## 2           200.14             15.52                 2.93
## 3           263.62              1.84                 0.74
##   number_of_reviews_l30d review_scores_rating review_scores_accuracy
## 1                   0.67                 4.76                  4.81
## 2                   0.33                 4.82                  4.85
## 3                   0.07                 2.37                  2.45
##   review_scores_cleanliness review_scores_checkin review_scores_communication
## 1                      4.79                  4.87                        4.87
## 2                      4.78                  4.92                        4.91
## 3                      2.73                  2.78                        2.56
##   review_scores_location review_scores_value calculated_host_listings_count
## 1                   4.85                4.77                           7.39
## 2                   4.89                4.75                           6.28
## 3                   3.12                2.33                           8.64
##   reviews_per_month n_amenities
## 1              1.06       27.80
## 2              0.53       33.14
## 3              0.21       20.00
```

Cluster 1 (75.1%) contains the vast majority of properties, mostly entire rental units, guest houses or private rooms (generally every other type of property except entire residential houses, villas and shared rooms), they receive higher than average ratings and more reviews per month. Most of them are instantly bookable and their host responds within an hour via the platform.

Cluster 2 (23.2%) contains mostly entire residential houses and villas, which are more expensive than average (per guest), have more rooms and amenities, allow more accommodates, are available for long-term rentals, and are typically rated 4.8+. Guest verification via id is required. Their hosts have been, on average, more time in the platform, respond within and day and most of them are not superhosts.

The smallest cluster (1.7%) contains mostly private rooms (bed and breakfast, guesthouses etc.), hotel rooms, shared rooms in guesthouses/villas. Guest verification via id is not required. They receive significantly lower ratings and less reviews than average, are usually instantly available for occupation, less expensive with few amenities. The host is not superhost, not verified, is less time in the platform and typically responds within a few days or more.

Other useful functions to describe the clusters are `catdes()` and `plot.catdes()` from the `FactoMineR` package (see the corresponding help files).

The `leaflet` package is then used to visualize the three clusters on the map of Cape Town. To clarify, the latitude and longitude are not precise, it can have a distance variation of 150 meter from the original address. The reason is to protect the Airbnb identifications. Cluster 1 properties are shown in blue, Cluster 2 in green and Cluster 3 in red. If we zoom in and out of the map we observe that there is an association between property location and cluster membership. For instance, most properties in Cluster 2 (green) are located

along the coast (the Atlantic Seaboard, Hout Bay, Camps Bay,...), whereas most properties in Cluster 3 (red) are located in the city centre but also at the Southern and Northern suburbs. Better knowledge of Cape Town's suburbs and neighbourhoods would result in a more thorough interpretation.

```
# Neighbourhoods GEOJSON file contains full list of Cape town neighbourhoods
# with geospatial data that we will use to visualise information on the map.
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following object is masked from 'package:MASS':
##
##     select
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(geojsonio)
```

```
## Registered S3 method overwritten by 'geojsonsf':
##   method        from
##   print.geojson geojson
```

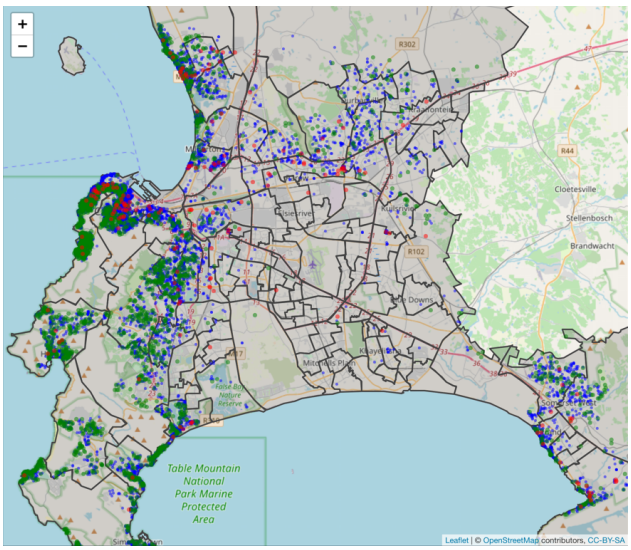
```
##
## Attaching package: 'geojsonio'
```

```
## The following object is masked from 'package:UBbipl':
##
##     projections
```

```
## The following object is masked from 'package:base':
##
##     pretty
```

```
library(leaflet)
nb_geo <- geojson_read('http://data.insideairbnb.com/south-africa/wc/cape-town/2021-10-26/visualisations

# We use the leaflet package to display listings from the three groups
# using lat/long information. This will give us
# an idea of geographical distribution of the clusters.
c1 <- airbnb %>%
  filter(outMixedRKM3$cluster == 1)

c2 <- airbnb %>%
  filter(outMixedRKM3$cluster == 2)

c3 <- airbnb %>%
  filter(outMixedRKM3$cluster == 3)

library(leaflet)
leaflet() %>% setView(lng = 18.423300, lat = -33.918861, zoom = 10) %>%
  addTiles() %>%
  addPolygons(data = nb_geo, color = "#444444", weight = 2, opacity = 1) %>%
```

```
  addCircleMarkers(lng = c1$longitude,
                   lat = c1$latitude,
                   radius = 2,
                   stroke = FALSE,
                   color = "blue",
                   fillOpacity = 0.5,
                   group = "c1"
  ) %>%
  addCircleMarkers(lng = c2$longitude,
                   lat = c2$latitude,
                   radius = 3,
                   stroke = FALSE,
                   color = "green",
                   fillOpacity = 0.5,
                   group = "c2"
  )%>%
  addCircleMarkers(lng = c3$longitude,
                   lat = c3$latitude,
                   radius = 3,
                   stroke = FALSE,
                   color = "red",
                   fillOpacity = 0.5,
                   group = "c3"
  )
```



Cluster 1 is rather large, so let's see if this cluster breaks into smaller clusters if we get the four-cluster solution.

```
# Apply Mixed Reduced 4-means
outMixedRKM4 <- cluspcamix(airbnb[,-c(1:2)], 4, 3, nstart = 3, seed = 1234)
```

```
##   |                                                                      |
```

```
table(outMixedRKM4$cluster)
```

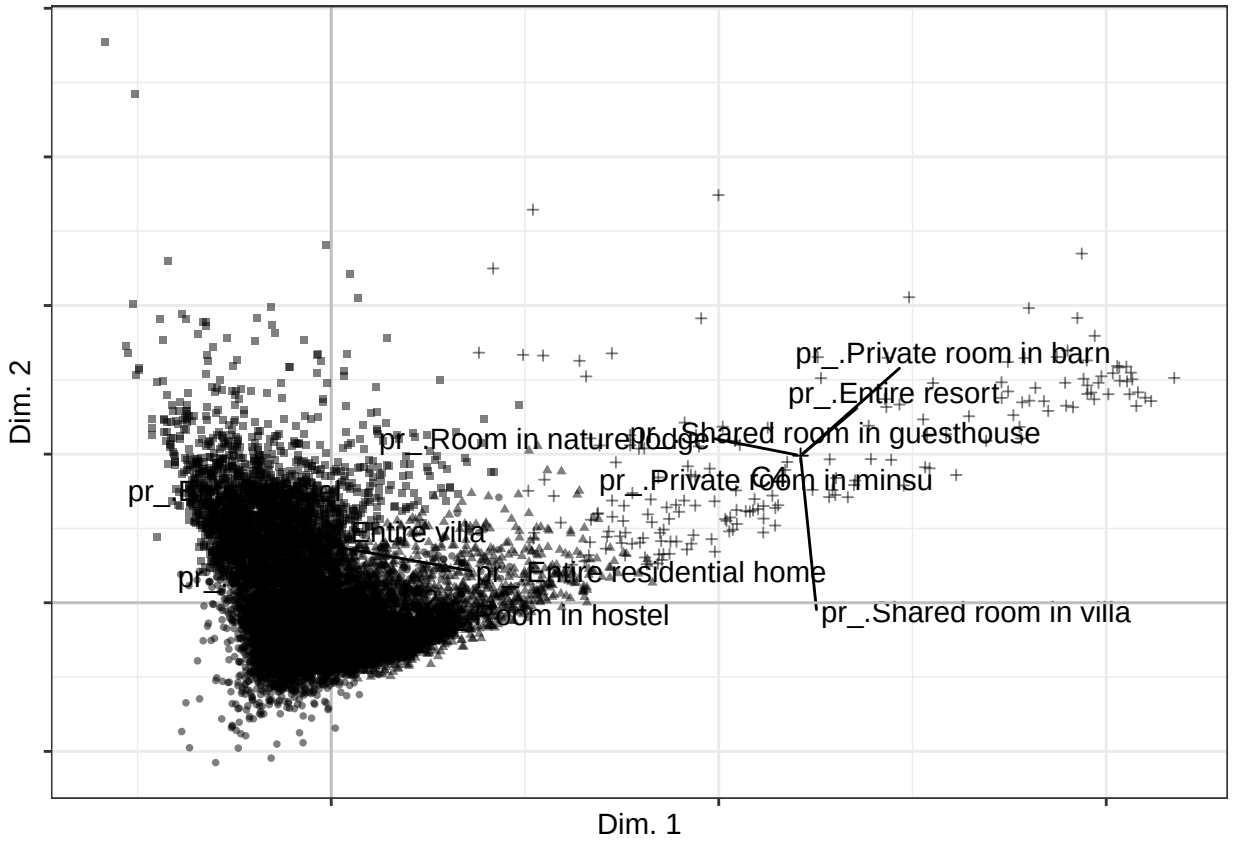
```
##
##    1    2    3    4
```

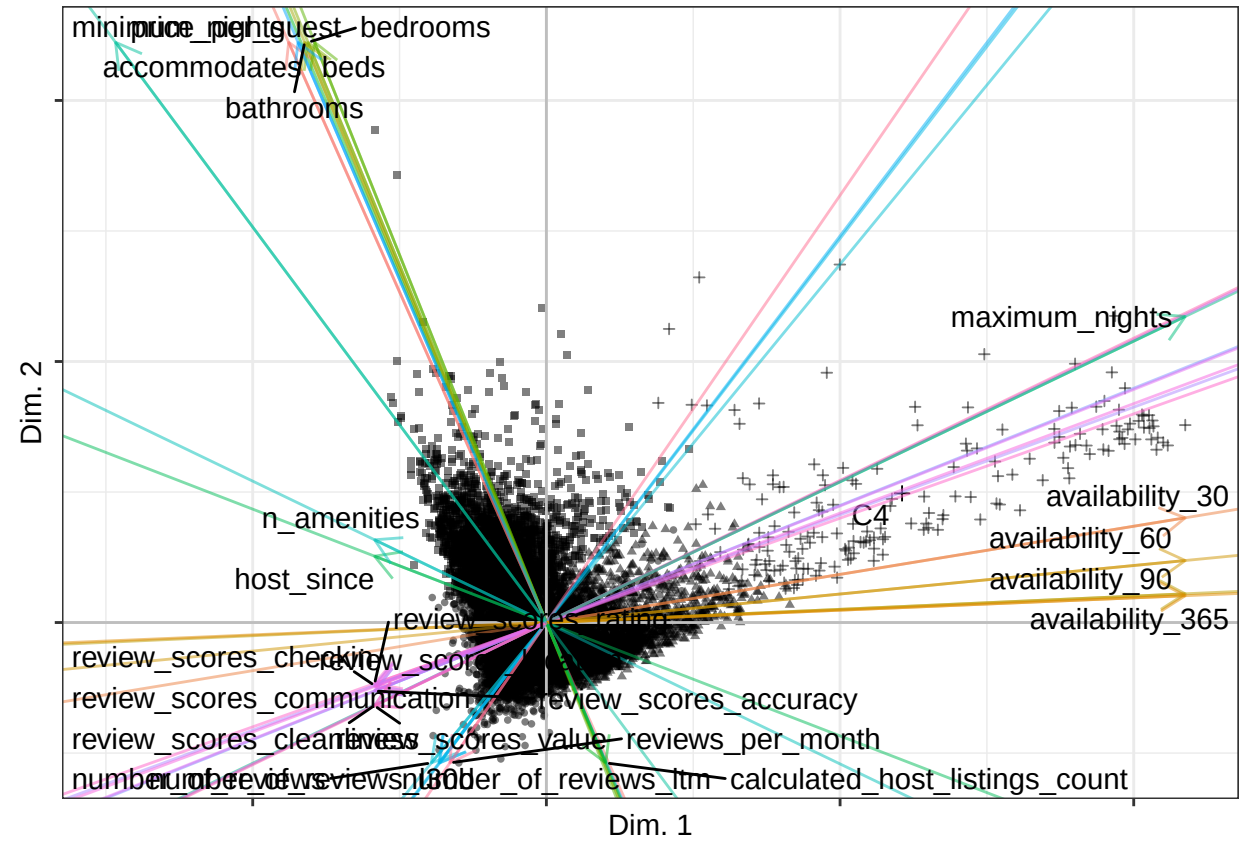
```
## 5537 3756 2359  181
```

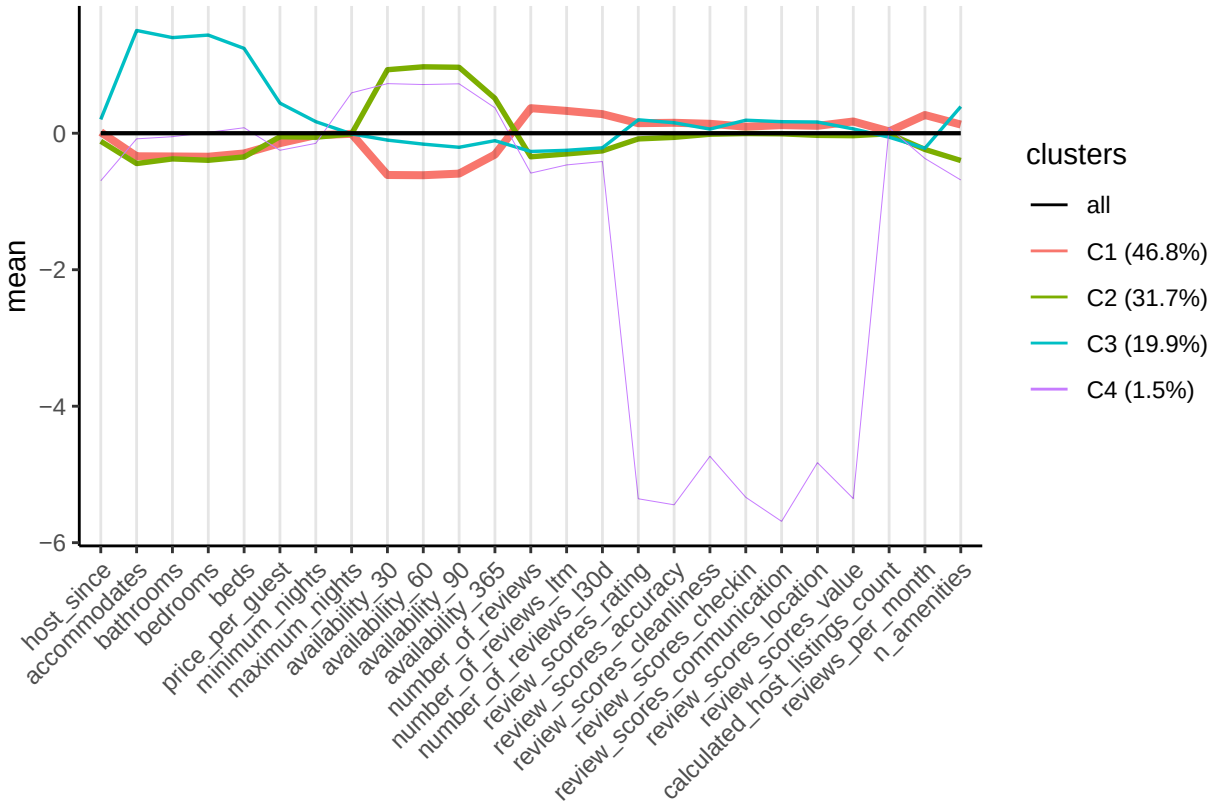
The four clusters have sizes 5537 (46.8%), 3756 (31.7%), 2359 (19.9%) and 181 (1.5%) in 3 dimensions. Let's plot the solution and describe the clusters.

```
plot(outMixedRKM4, cludesc = TRUE, max.overlaps = 80)
```

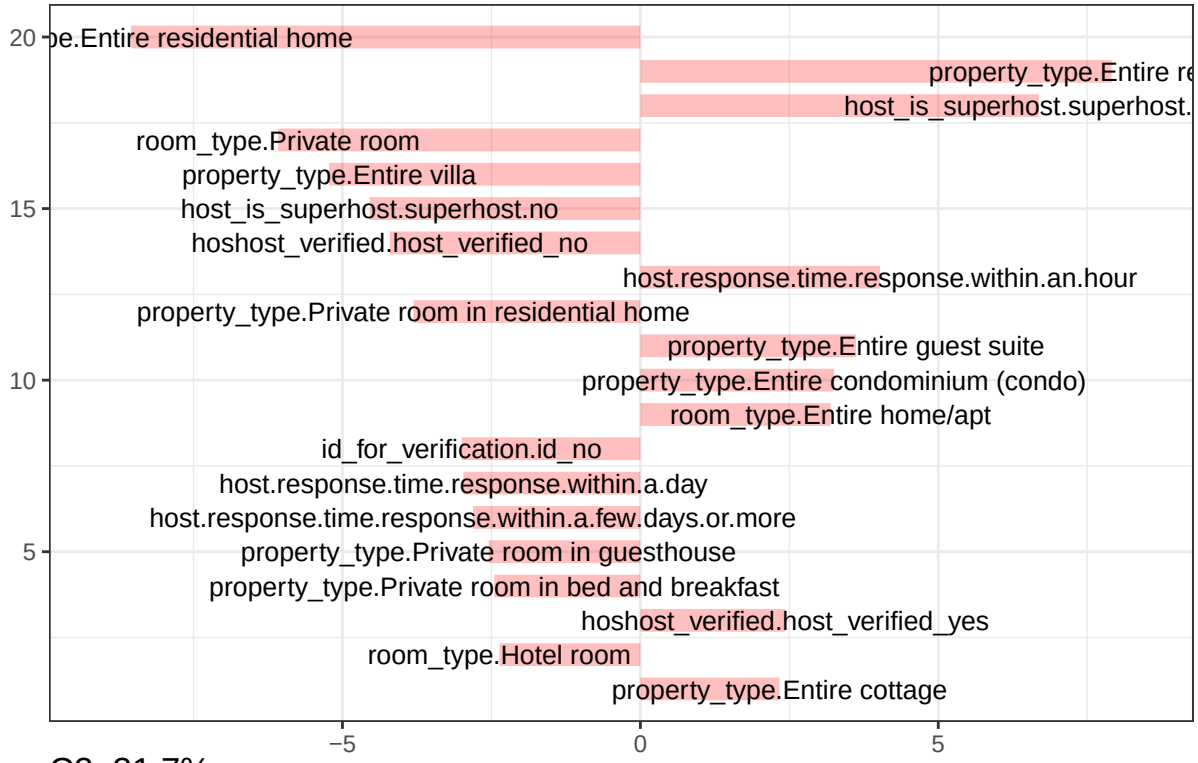
```
## Warning: ggrepel: 76 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



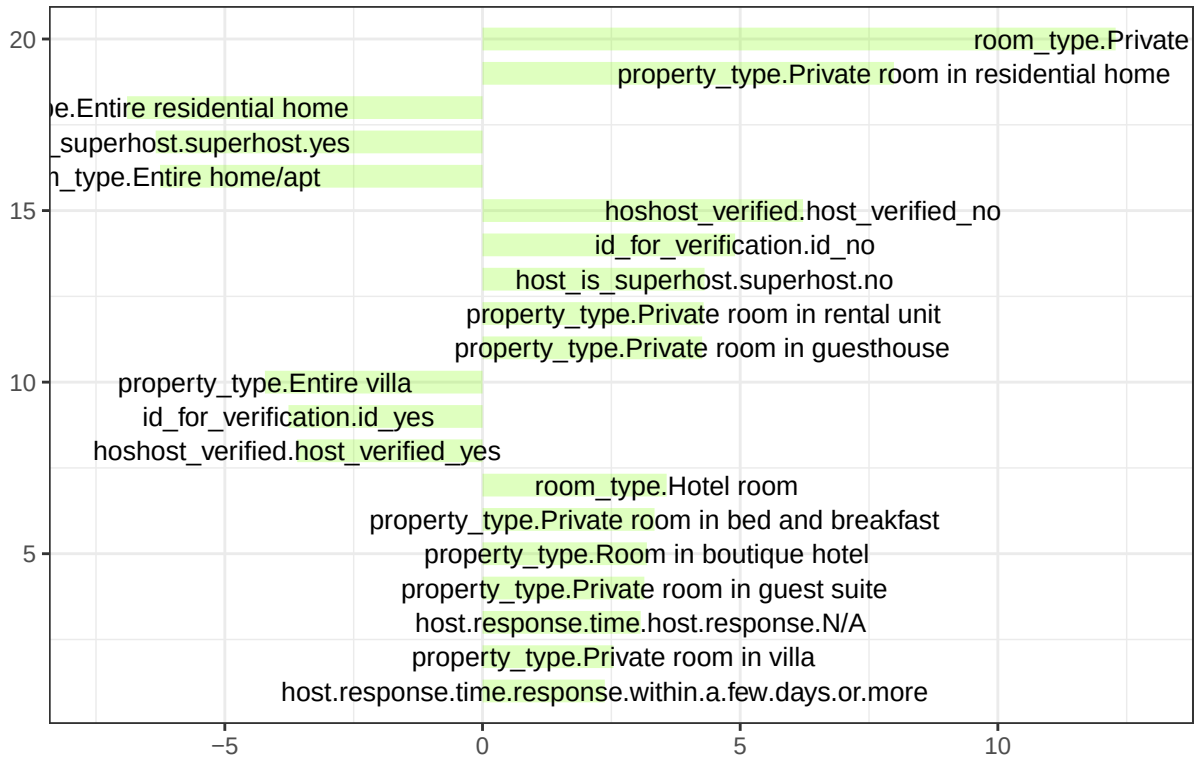

bedrooms
accommodates
beds
bathrooms
maximum_nights
availability_30
availability_60
availability_90
availability_365
C4
n_amenities
host_since
review_scores_accuracy
reviews_per_month
calculated_host_listings_count
Dim. 2
Dim. 1

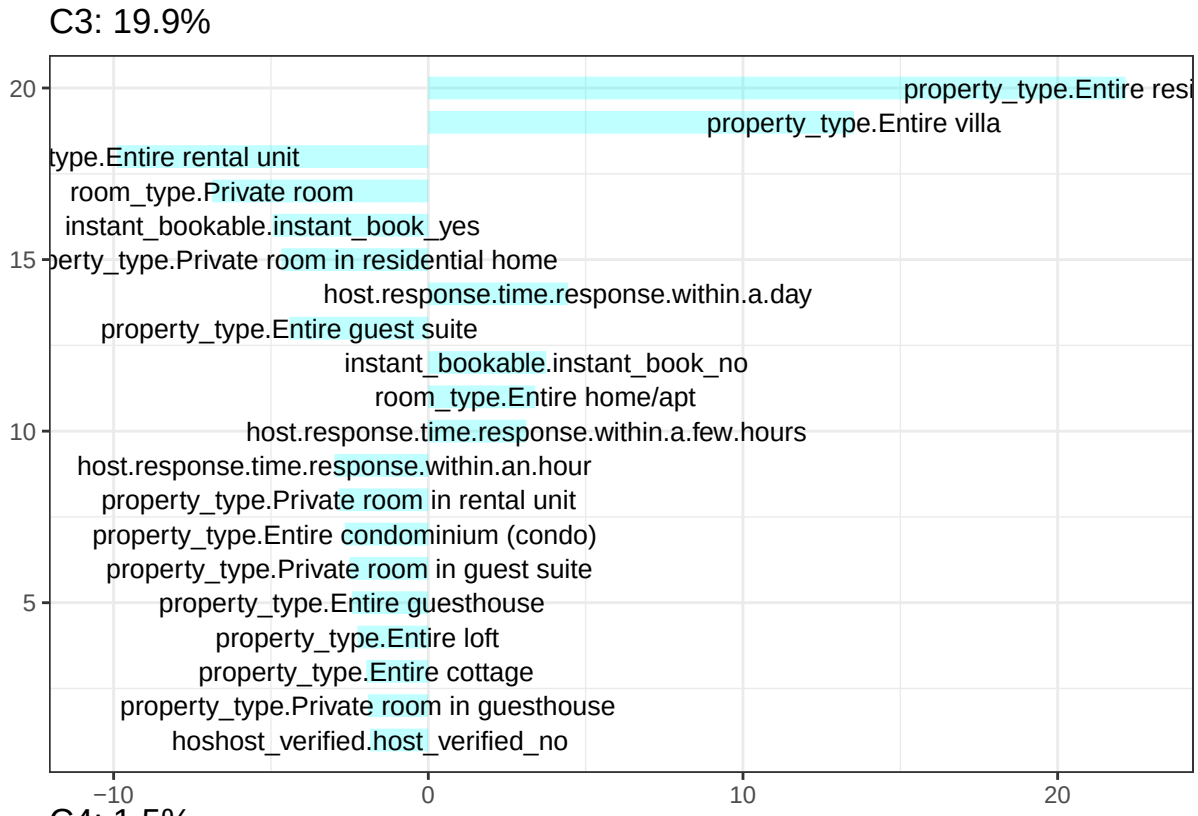

mean
0
−2
−4
−6
host_since
accommodates
bathrooms
bedrooms
beds
price_per_guest
minimum_nights
maximum_nights
availability_30
availability_60
availability_90
availability_365
number_of_reviews
number_of_reviews_ltm
number_of_reviews_l30d
review_scores_rating
review_scores_accuracy
review_scores_cleanliness
review_scores_checkin
review_scores_communication
review_scores_location
review_scores_value
calculated_host_listings_count
reviews_per_month
n_amenities
clusters
all
C1 (46.8%)
C2 (31.7%)
C3 (19.9%)
C4 (1.5%)

C1: 46.8%


20
15
10
5
pe.Entire residential home
property_type.Entire re
host_is_superhost.superhost.
room_type.Private room
property_type.Entire villa
host_is_superhost.superhost.no
hoshost_verified.host_verified_no
host.response.time.response.within.an.hour
property_type.Private room in residential home
property_type.Entire guest suite
property_type.Entire condominium (condo)
room_type.Entire home/apt
id_for_verification.id_no
host.response.time.response.within.a.day
host.response.time.response.within.a.few.days.or.more
property_type.Private room in guesthouse
property_type.Private room in bed and breakfast
hoshost_verified.host_verified_yes
room_type.Hotel room
property_type.Entire cottage
−5
0
5

C2: 31.7%


20
15
10
5
room_type.Private
property_type.Private room in residential home
pe.Entire residential home
_superhost.superhost.yes
n_type.Entire home/apt
hoshost_verified.host_verified_no
id_for_verification.id_no
host_is_superhost.superhost.no
property_type.Private room in rental unit
property_type.Private room in guesthouse
property_type.Entire villa
id_for_verification.id_yes
hoshost_verified.host_verified_yes
room_type.Hotel room
property_type.Private room in bed and breakfast
property_type.Room in boutique hotel
property_type.Private room in guest suite
host.response.time.host.response.N/A
property_type.Private room in villa
host.response.time.response.within.a.few.days.or.more
−5
0
5
10



Cluster 1 (46.8%) contains entire rental units, home/apt, guesthouses, condos, lofts, cottages are instantly bookable, receive more reviews and higher ratings, host is superhost, responds within an hour and guest verification via id is required.

Cluster 2 (31.7%) contains private rooms in residential houses/guesthouses/rental units, hotel rooms, have fewer amenities, receive less reviews and lower ratings, are available for booking, less expensive, require verification via id. Host is not verified and is not superhost.

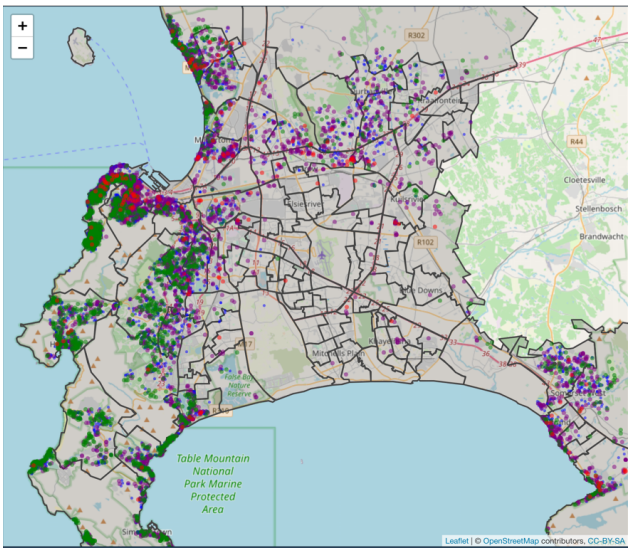
Cluster 3 (19.9%) contains entire residential houses and villas, not instantly bookable, have more rooms and amenities than average, allow more accommodates, are more expensive, available for long-term rentals and typically rated 4.8+. Hosts have been, on average, more time in the platform and respond within a day. Guest verification via id is required.

Cluster 4 (1.5%) contains private rooms (barn, minsu, nature lodge), shared rooms in guesthouses/villas, receive low ratings and less reviews, usually available for occupation, less pricey with fewer amenities and the host is less time in the platform and responds within a few days or more.

All in all, Clusters 3 and 4 in the four-cluster solution are similar to Clusters 2 and 3 in the three-clister solution, whereas Cluster 1 was split into Clusters 1 and 2, distinguishing private rooms from entire homes. On the map below, Cluster 1 is shown in blue and Cluster 2 in purple. Cluster 3 is still in green and Cluster 4 in red. Notice that properties in Clusters 1 and 2 are not well-separated on the map.

```
c1 <- airbnb %>%
  filter(outMixedRKM4$cluster == 1)

c2 <- airbnb %>%
  filter(outMixedRKM4$cluster == 2)

c3 <- airbnb %>%
  filter(outMixedRKM4$cluster == 3)

c4 <- airbnb %>%
  filter(outMixedRKM4$cluster == 4)

leaflet() %>% setView(lng = 18.423300, lat = -33.918861, zoom = 10) %>%
  addTiles() %>%
  addPolygons(data = nb_geo, color = "#444444", weight = 2, opacity = 1) %>%
  addCircleMarkers(  lng = c1$longitude,
                     lat = c1$latitude,
                     radius = 2,
                     stroke = FALSE,
                     color = "blue",
                     fillOpacity = 0.5,
                     group = "c1"
  ) %>%
  addCircleMarkers(  lng = c2$longitude,
                     lat = c2$latitude,
                     radius = 3,
                     stroke = FALSE,
                     color = "purple",
                     fillOpacity = 0.5,
                     group = "c2"
  )%>%
  addCircleMarkers(  lng = c3$longitude,
                     lat = c3$latitude,
                     radius = 3,
                     stroke = FALSE,
                     color = "green",
                     fillOpacity = 0.5,
                     group = "c3"
```

```
)%>%
addCircleMarkers(  lng = c4$longitude,
                   lat = c4$latitude,
                   radius = 3,
                   stroke = FALSE,
                   color = "red",
                   fillOpacity = 0.5,
                   group = "c4"
)
```



References

- Compatibility between PCA and K-means

Ding, C., & He, X. (2004, July). K-means clustering via principal component analysis. In *Proceedings of the twenty-first international conference on Machine learning* (p. 29).

- Methods combining PCA with K-means

De Soete, G., & Carroll, J. D. (1994). K-means clustering in a low-dimensional euclidean space. In E. Diday, Y. Lechevallier, M. Schader, P. Bertrand, & B. Burtschy (Eds.), *New Approaches in Classification and Data Analysis* (pp. 212-219). Springer-Verlag, Berlin.

Vichi, M., & Kiers, H. A. (2001). Factorial k-means analysis for two-way data. *Computational Statistics & Data Analysis*, *37*(1), 49-64.

- Methods combining MCA with K-means

Hwang, H., Dillon, W. R., & Takane, Y. (2006). An Extension of Multiple Correspondence Analysis for Identifying Heterogenous Subgroups of Respondents. *Psychometrika*, *71*, 161-171.

Iodice D'Enza, A., & Palumbo, F. (2013). Iterative factor clustering of binary data. *Computational Statistics*, *28*(2), 789-807.

van de Velden, M., Iodice D'Enza, A., & Palumbo, F. (2017). Cluster Correspondence Analysis. *Psychometrika*, *82*(1), 158-185.

- Methods combining PCAMIX/FAMD with K-means

van de Velden, M., Iodice D'Enza, A., & Markos, A. (2019). Distance-based clustering of mixed data. *WIREs Computational Statistics*, *11*(3), e1456.

Vichi, M., Vicari, D., & Kiers, H. A. (2019). Clustering and dimension reduction for mixed variables. *Behaviormetrika*, *46*(2), 243-269.

- `clustrd` package

Markos, A., Iodice D'Enza, A., & van de Velden, M. (2019). Beyond tandem analysis: Joint dimension reduction and clustering in R. *Journal of Statistical Software* (Online), *91*(10).

- Cluster stability assessment

Dolnicar, S., & Grün, B. (2008). Challenging "factor–cluster segmentation". *Journal of Travel Research*, *47*(1), 63-71.

Dolnicar, S., & Leisch, F. (2010). Evaluation of structure and reproducibility of cluster solutions using the bootstrap. *Marketing Letters*, *21*(1), 83-101.

Dolnicar, S., & Leisch, F. (2017). Using segment level stability to select target segments in data-driven market segmentation studies. *Marketing Letters*, *28*(3), 423-436.

Hennig, C. (2007). Cluster-wise assessment of cluster stability. *Computational Statistics and Data Analysis*, *52*, 258-271.

Pfitzner, D., Leibbrandt, R., & Powers, D. (2009). Characterization and evaluation of similarity measures for pairs of clusterings. *Knowledge and Information Systems*, *19*(3), 361-394.