Package: boxplotcluster (via r-universe)

September 2, 2024

Title Clustering Method Based on Boxplot Statistics

Version 0.3

Description Following Arroyo-Maté-Roque (2006), the function calculates the distance between rows or columns of the dataset using the generalized Minkowski metric as described by Ichino-Yaguchi (1994). The distance measure gives more weight to differences between quartiles than to differences between extremes, making it less sensitive to outliers. Further,the function calculates the silhouette width (Rousseeuw 1987) for different numbers of clusters and selects the number of clusters that maximizes the average silhouette width, unless a specific number of clusters is provided by the user. The approach implemented in this package is based on the following publications: Rousseeuw (1987) <doi:10.1016/0377-0427(87)90125-7>; Ichino-Yaguchi (1994) <doi:10.1109/21.286391>; Arroyo-Maté-Roque (2006) <doi:10.1007/3-540-34416-0_7>.

Depends R (>= 4.0.0)

Imports cluster (>= 2.1.4), graphics (>= 4.0.0), grDevices (>= 4.0.0), stats (>= 4.0.0)

License GPL (>= 2)

Encoding UTF-8

RoxygenNote 7.2.3

NeedsCompilation no

Author Gianmarco Alberti [aut, cre]

Maintainer Gianmarco Alberti `<gianmarcoalberti@gmail.com>`

Date/Publication 2023-12-06 18:30:09 UTC

Repository https://gianmarcoalberti.r-universe.dev

RemoteUrl https://github.com/cran/boxplotcluster

RemoteRef HEAD

RemoteSha 0b74525f194ff753859c80b5f2f5321ceb9e182f

Contents

`boxplotcluster` *Boxplot Cluster Function*

Description

The function `boxplotcluster` implements a special clustering method based on boxplot statistics. Following Arroyo-Maté-Roque (2006), the function calculates the distance between rows or columns of the dataset using the generalized Minkowski metric as described by Ichino and Yaguchi (1994). The distance measure gives more weight to differences between quartiles than to differences between extremes, making it less sensitive to outliers. Further, the function calculates the silhouette width for different numbers of clusters (Rousseeuw 1987) and selects the number of clusters that maximizes the average silhouette width (unless a specific number of clusters is provided by the user).
Visit this LINK to access the package's vignette.

Usage

```
boxplotcluster(
  x,
  target.var = NULL,
  group.var = NULL,
  calc.type = "columns",
  aggl.meth = "ward.D2",
  part = NULL,
  silh.col = TRUE,
  cex.dndr.lab = 0.75,
  cex.sil.lab = 0.75,
  oneplot = TRUE
)
```

Arguments

`x`	A dataframe representing the input dataset (see below and the `Details` section).
`target.var`	A character vector specifying the name of the numerical variable in the input dataset that will be used for analysis. If the input dataset is in `wide` format, this can be left as default (`NULL`). If the dataset is in `long` format, this parameter has to indicate the variable containing the data values that will be grouped by the different levels of the variable fed via the `group.var` parameter.
`group.var`	A character vector specifying the grouping variable. It can be left as default (`NULL`) if the input dataset is in `wide` format.

`calc.type`	A string specifying the units to be clustered if the input dataset is in wide format (either `columns` or `rows`; the former is the default).
`aggl.meth`	A string specifying the agglomeration method to be used in hierarchical clustering. Defaults to "ward.D2". For other methods see `hclust`.
`part`	An optional integer specifying the desired number of clusters. If not provided, the function selects the number of clusters that maximises the average silhouette width.
`silh.col`	A logical value, which takes `TRUE` (default) or `FALSE` if the user wants to give colour to the silhouette plot reflecting the cluster partition.
`cex.dndr.lab`	A numeric specifying the character expansion factor for the labels in the dendrogram plot. Defaults to 0.75.
`cex.sil.lab`	A numeric specifying the character expansion factor for the labels in the silhouette plot. Defaults to 0.75.
`oneplot`	A logical value, which takes `TRUE` (default) or `FALSE` if the user wants or does not want the plots to be visualised in a single window.

Details

The function first calculates the pairwise distance between each unit of the input dataset using the Ichino-Yaguchi dissimilarity measure (equations 7 and 8 in Arroyo-Maté-Roque (2006)). The distance between A and B is defined as:

$$(0.5*(abs(m1-m2)+2*abs(q1-q2)+2*abs(Me1-Me2)+2*abs(Q1-Q2)+abs(M1-M2)))/4$$

where

m1 <- min(A)
m2 <- min(B)
q1 <- quantile(A, probs = 0.25)
q2 <- quantile(B, probs = 0.25)
Q1 <- quantile(A, probs = 0.75)
Q2 <- quantile(B, probs = 0.75)
M1 <- max(A)
M2 <- max(B)
Me1 <- median(A)
Me2 <- median(B)

The distance matrix is then used to perform a hierarchical clustering. Also, the function calculates the silhouette width for different numbers of clusters and selects the number of clusters that maximises the average silhouette width (unless a specific number of clusters is provided by the user).

The silhouette method allows to measure how 'good' is the selected cluster solution. If the parameter `part` is left empty (default), an optimal cluster solution is obtained. The optimal partition is selected via an iterative procedure which identifies at which cluster solution the highest average silhouette width is achieved. If a user-defined partition is needed, the user can input the desired number of clusters using the parameter `part`. In either case, an additional plot is returned besides the cluster dendrogram and the silhouette plot; it displays a scatterplot in which the cluster solution (x-axis) is plotted against the average silhouette width (y-axis). A black dot represents the partition selected either by the iterative procedure or by the user.

In summary, the function generates a series of plots to visualise the results:

(a) boxplots colored by cluster membership,
(b) a dendrogram (where clusters are indicated by rectangles whose color is consistent with the color assigned to the boxplot in the previous plot),
(c) a silhouette plot (with optional by-cluster colours), and
(d) a plot of the average silhouette width vs. the number of clusters.

The silhouette plot is obtained via the `silhouette()` function out from the `cluster` package. For a detailed description of the silhouette plot, its rationale, and its interpretation, see Rousseeuw 1987.

Input dataset format:
When the dataset is in `wide` format, each row corresponds to a distinct unit, and each column corresponds to a different variable. In this format, it is typically assumed that all units have the same number of observations. The element representing the units (either rows or columns) will be clustered.

When the dataset is in `long` format, it consists of rows representing individual observations. One column indicates the variable name (grouping variable) and another column contains the measurements values, or viceversa. If the input dataset comes in this format, the units to be clustered are created by grouping the observations (i.e., rows of the dataframe) by `group.var`. If the input dataset is in `long` format, groups can feature a different number of observations.

For actual examples of both formats, see the `Examples` section below.

Value

The function returns a list storing the following components

- `distance.matrix`: distance matrix reporting the distance values.
- `units.by.cluster`: a list of the input dataset's units, grouped by cluster membership.
- `avr.silh.width.by.n.of.clusters`: average silhouette width by number of clusters.
- `partition.silh.data`: silhouette data for the selected partition.

References

Arroyo, J., Maté, C., & Roque, A. M-S. (2006). Hierarchical Clustering for Boxplot Variables. In Studies in Classification, Data Analysis, and Knowledge Organization (pp. 59–66). Springer Berlin Heidelberg.

Ichino, M., & Yaguchi, H. (1994). Generalized Minkowski Metrics for Mixed Feature-Type Data Analysis. IEEE Transactions on Systems, Man, and Cybernetics, 24(4), 698-708.

Rousseeuw, P J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis, Journal of Computational and Applied Mathematics 20, 53-65.

See Also

`silhouette`, `hclust`

Examples

```
## EXAMPLE 1

# Create a toy dataset in WIDE format
df <- data.frame(
a = rnorm(30, mean = 30, sd = 5),
b = rnorm(30, mean = 40, sd = 5),
c = rnorm(30, mean = 20, sd = 5),
d = rnorm(30, mean = 25, sd = 5),
e = rnorm(30, mean = 50, sd = 5),
f = rnorm(30, mean = 10, sd = 5),
g = rnorm(30, mean = 100, sd = 5),
h = rnorm(30, mean = 20, sd = 5),
i = rnorm(30, mean = 40, sd = 5),
l = rnorm(30, mean = 35, sd = 5),
m = rnorm(30, mean = 35, sd = 5),
n = rnorm(30, mean = 70, sd = 5),
o = rnorm(30, mean = 20, sd = 5),
p = rnorm(30, mean = 70, sd = 5),
q = rnorm(30, mean = 90, sd = 5)
)

# Run the function
result <- boxplotcluster(df)

# Same as above, but selecting a 4-cluster solution
result <- boxplotcluster(df, part=4)

# Same as above, but the rows are clustered
result <- boxplotcluster(df, calc.type="rows", part=4)



## EXAMPLE 2
# Create a toy dataset in WIDE format, representing archaeological stone
# flake length (cm) by raw material

df <- data.frame(
Basalt = c(7.0, 7.0, 7.7, 8.2, 10.3, 10.3, 10.3, 10.8, 11.0, 13.0, 13.9, 14.6, 1.0),
Chert = c(2.9, 4.8, 5.3, 5.8, 5.8, 6.2, 6.5, 7.7, 7.7, 7.9, 8.9, 9.6, 2.0),
Obsidian = c(2.2, 2.4, 3.1, 4.3, 5.0, 5.5, 5.8, 6.0, 6.2, 7.2, 7.4, 7.7, 2.0),
Quartzite = c(5.5, 5.5, 7.0, 7.4, 7.7, 7.9, 8.6, 8.9, 9.4, 9.6, 10.6, 10.8, 1.0),
Granite = c(4.0, 4.5, 6.0, 6.8, 7.0, 7.8, 8.1, 8.4, 9.0, 10.0, 10.5, 11.0, 1.0),
Sandstone = c(3.0, 3.2, 4.0, 4.5, 4.9, 5.2, 6.0, 6.5, 7.0, 7.5, 8.0, 8.5, 1.0),
Limestone = c(3.5, 4.0, 4.8, 5.2, 6.0, 6.5, 7.0, 7.5, 8.0, 8.5, 9.0, 9.5, 1.5),
```

```
Slate = c(2.0, 2.4, 3.0, 3.6, 4.0, 4.4, 5.0, 5.6, 6.0, 6.4, 7.0, 7.6, 1.2)
)

# Run the function to cluster the columns (default); cluster solution is
# selected by the iterative method (default)

result <- boxplotcluster(df)



## EXAMPLE 3
#  Create a toy dataset in LONG format

n_units <- 20
n_groups <- 10
measurements_per_group <- 4

long_data <- data.frame(
SubjectID = rep(paste0("Unit", 1:n_units), each = n_groups * measurements_per_group),
Grouping_var = rep(rep(paste0("M", 1:n_groups), each = measurements_per_group), n_units),
Value = runif(n_units * n_groups * measurements_per_group))

#  Run the analysis, specifying the target variable and the grouping variable,
# and selecting a 3-cluster solution
result <- boxplotcluster(long_data, target.var = "Value", group.var = "Grouping_var", part=3)
```

Index