Package 'corclass'

May 8, 2026

Type Package

Title Correlational Class Analysis

Version 0.2.1

Date 2023-09-01

Author Andrei Boutyline

Maintainer Andrei Boutyline <aboutyl@umich.edu>

Description Perform a correlational class analysis of the data, resulting in a partition of the data into separate modules.

License GPL-2

Depends igraph, methods

Suggests Cairo

NeedsCompilation no

Repository CRAN

Date/Publication 2023-09-06 19:02:32 UTC

Contents

corclass-package *Correlational Class Analysis package*

Description

This package implements the Correlational Class Analysis methodology described by Boutyline (under review). The correlational class analysis of a survey dataset produces a partition of the population into separate modules. This is done in four steps:

1. Create a matrix G of absolute row correlations. This is the network adjacency matrix.
2. Set statistically insignificant correlations to 0 to reduce noise.
3. Use igraph's leading.eigenvector.community to partition this network into modules.
4. Return an object describing the resulting class assignments (as well as the separate data frames describing the individual modules.)

CCA substantially improves the accuracy of the Relational Class Analysis (RCA) algorithm proposed by Goldberg (2011). See Boutyline (under review) for details.

Details

The main function is `cca`. `plot.cca` plots the modules produced by `cca`. Sample data can be accessed via data(cca.example).

Author(s)

Written and maintained by Andrei Boutyline, <andrei.boutyline@gmail.com>.

References

Boutyline, Andrei. 2017. "Improving the Measurement of Shared Cultural Schemas with Correlational Class Analysis: Theory and Method." Sociological Science 4:353-93. https://www.sociologicalscience.com/articles-v4-15-353/

See Also

This package makes heavy use of `igraph`.
The CCA algorithm is an improvement of RCA `https://cran.r-project.org/package=RCA`

Examples

```
data(cca.example)
res1 <- cca(cca.example)
plot(res1, 1)
```

cca *Main function for CCA.*

Description

Perform a correlational class analysis of the data, resulting in a partition of the data into separate modules. This consists of four steps:

1. Create a matrix G of absolute row correlations. This is the network adjacency matrix.
2. Set statistically insignificant correlations to 0 to reduce noise.
3. Use igraph's leading.eigenvector.community to partition this network into modules.
4. Return an object describing the resulting class assignments (as well as the separate data frames describing the individual modules.)

Usage

```
cca(dtf, filter.significance = TRUE, filter.value = 0.01,
    zero.action = c("drop", "ownclass"), verbose = TRUE)
```

Arguments

dtf	The data frame containing the variables of interest.
filter.significance	Significance filtering sets "insignificant" ties to 0 to decrease noise and increase stability. Simulation results show that this greatly increases accuracy in many settings. Set filter.significance = FALSE to disable this.
filter.value	Minimum significance cutoff. Absolute row correlations below this value will be set to 0.
zero.action	What to do with 0-variance rows before partitioning the graph. If zero.action is "drop", CCA drop rows with 0 variance from the analyses (default). If zero.action is "ownclass", the correlations between 0-variance rows and all other rows is set to 0, and the correlations between all pairs of 0-var rows are set to 1. This effectively creates a "zero class".
verbose	Whether to print details of what CCA is doing to the screen.

Value

membership	The class assignments produced by CCA.
cormat	The row correlation matrix that was partitioned by CCA. It has a "dtf" attribute which holds the dataframe. Note that, if 0-variance were dropped, they will be excluded from the dataframe as well as the correlation matrix. The "zeros" attribute which holds the indexes of the dropped rows.
modules	For convenience, the dataframe is separated into the modules found by the algorithm. A separate dataframe for each module i can be found in modules[[i]]$dtf. The matrix of column correlations are in modules[[i]]$cormat. modules[[i]]$degenerate indicates whether this module contains undefined. Note that these modules can be plotted via the S3 plot method. See example below.

Author(s)

Andrei Boutyline, <aboutyl@umich.edu>.

References

Boutyline, Andrei. 2017. "Improving the Measurement of Shared Cultural Schemas with Correlational Class Analysis: Theory and Method." Sociological Science 4:353-93. https://www.sociologicalscience.com/articles-v4-15-353/

See Also

plot.cca

Examples

```
 data(cca.example)
res1 <- cca(cca.example) # with igraph 0.7, this should find 3 classes of sizes 218 391 144.
 plot(res1, 1) # plot them
 plot(res1, 2)
 plot(res1, 3)

print (round(res1$modules[[1]]$cormat, 2)) # examine the correlation matrix for the 1st module
 print (summary(res1$modules[[1]]$dtf)) # look at its variable ranges
 plot(res1, 1, bw = TRUE) # Plot it again in a more journal-friendly format.

 # now let's try setting the filter value too high
 res2 <- cca(cca.example, filter.value = 0.001)
 # With igraph 0.7, the above now finds 17 classes
 #    of sizes 216 1 1 371 1 1 1 1 1 1 1 1 11 141 1 1 2
# The small isolate classes can either be dropped manually, or by increasing filter.value
```

`cca.example` *Sample data for Correlational Class Analysis.*

Description

A randomly generated sample dataset for correlational class analysis, created using the approach described in Boutyline (2017). rownames(cca.example) contain the true schematic class membership for each row. Every row belonging to one schematic class was created from noisy linear transformations of the same vector.

Usage

```
data(cca.example)
```

Format

The format is: num [1:754, 1:10] 4 7 4 7 4 4 7 3 9 8 ... - attr(*, "dimnames")=List of 2 ..$: chr [1:754] "1" "1" "1" "1" ... ..$: NULL

References

Boutyline, Andrei. 2017. "Improving the Measurement of Shared Cultural Schemas with Correlational Class Analysis: Theory and Method." Sociological Science 4:353-93. https://www.sociologicalscience.com/articles-v4-15-353/

Examples

```
data(cca.example)
res1 <- cca(cca.example)
```

`plot.cca` *Plotting function for CCA modules.*

Description

Plot a CCA-produced module as a network diagram. The network nodes are survey variables (columns), and the ties are their correlations. Red (or dashed) ties represent negative correlations. This is a convenience function wrapping igraph's graphing functionality. Writing to a file is done via the Cairo package.

Usage

```
## S3 method for class 'cca'
plot(x, module.index, cutoff = 0.05, LAYOUT = igraph::layout.kamada.kawai,
    drop.neg.ties.for.layout = TRUE, bw = FALSE, main = NULL, file = NULL, ...)
```

Arguments

`x`	The cca object returned by `cca`.
`module.index`	Index of module to plot, between 1 and length(x$modules).
`cutoff`	Minimum absolute column correlation to plot.
`LAYOUT`	If this is a function, it is assumed to be one of the layout routines from igraph (or something that returns data in the same format). Otherwise, it is assumed to be the layout returned by such a function.
`drop.neg.ties.for.layout`	Whether to drop negative ties for the purpose of layout. This may be necessary because some layout algorithms do not work if negative ties are present. Note that the negative ties are only dropped for the purposes of layout. They will still included in the actual plot.
`bw`	Whether to print in color for screen viewing (FALSE), or in b&w with dashed lines for negative ties for a journal manuscript (TRUE).
`main`	Caption at the top of the graph. If NULL, the module number is used as the caption.
`file`	If a filename is provided, the graph is saved as a pdf with that filename. Note that this requires the Cairo package.
`...`	Unused.

Value

If the LAYOUT paramter is a layout function, then the return value is the static layout generated by this function (this allows the same exact layout to be reproduced in the future–see example below). Otherwise, it is the same static layout that was passed to plot.cca.

Author(s)

Andrei Boutyline, <andrei.boutyline@gmail.com>

See Also

cca

Examples

```
 data(cca.example)
res1 <- cca(cca.example) # with igraph 0.7, this should find 3 classes of sizes 218 391 144.
 plot(res1, 1) # plot the first module
 plot(res1, 2) # plot the second module
 plot(res1, 3) # plot the third module

 plot(res1, 1, bw = TRUE) # check out first module in black and white
plot(res1, 1, LAYOUT = layout.fruchterman.reingold) # try a different layout algorithm

 # example of saving a fixed layout
 layout1 <- plot(res1, 1) # try out a layout ...
 layout1 <- plot(res1, 1) # ... try again
 layout1 <- plot(res1, 1) # ... until one looks good

# Now plot the result with the chosen layout. To save image to disk,
#  replace NULL below with file name (e.g., file = "module1.pdf")
plot(res1, 1, LAYOUT = layout1, file = NULL)
```

---

print.cca *Print description of CCA results.*

Description

Prints a concise description of CCA results, including module membership counts. Reports if any of the modules are degenerate.

Usage

```
## S3 method for class 'cca'
print(x, ...)
```

Arguments

x	The cca object returned by cca.
...	Unused.

Author(s)

Andrei Boutyline, <andrei.boutyline@gmail.com>

See Also

plot.cca, cca

Examples

```
data(cca.example)
res1 <- cca(cca.example)
print(res1)
```

Index