Package ‘cata’

May 23, 2022

Type    Package
Title   Analysis of Check-All-that-Apply (CATA) Data
Version 0.0.10.9
Date    2022-05-22
Author  J.C. Castura [aut, cre, ctb] (<https://orcid.org/0000-0002-1640-833X>)
Maintainer J.C. Castura <jcastura@compusense.com>
Description Functions for analyzing check-all-that-apply (CATA) data from consumer and sensory tests. Cochran’s Q test, McNemar’s test, and Penalty-Lift analysis provided, as described in for CATA data analysis by Meyners, Castura & Carr (2013) <doi:10.1016/j.foodqual.2013.06.010>. Cluster analysis can be performed using b-cluster analysis. The quality of cluster analysis solutions can be evaluated using various measures. The methods related to b-cluster analysis are described in a manuscript by Castura, Meyners, Varela & Naes (2022) <doi:10.1016/j.foodqual.2022.104564>.
Depends R (>= 4.1.0)
Imports stats, utils, graphics
License GPL (>= 2)
LazyData TRUE
Encoding UTF-8
RoxygenNote 7.1.2
NeedsCompilation no
Repository CRAN
Date/Publication 2022-05-23 15:30:24 UTC

R topics documented:

barray ................................................................. 2
bcluster .............................................................. 3
bcluster.h .......................................................... 4
bcluster.n ......................................................... 5
cochranQ .......................................................... 6
Consumer CATA data set: bread .................................. 7
Convert 3d array of CATA data to 4d array of CATA differences

Description
Converts a three-dimensional array (I assessors, J products, M attributes) to a four-dimensional array of product comparisons (I assessors, J(J-1)/2 product comparisons, two outcomes (of type b or c), M attributes)

Usage
barray(X, values = "bc", type.in = "binary", type.out = "binary")

Arguments
X three-dimensional array (I assessors, J products, M attributes) where values are 0 (not checked) or 1 (checked)
values "bc" (default) returns two outcomes: b and c; otherwise "abcd" returns four outcomes: a, b, c, d.
type.in type of data submitted; default (binary) may be set to ordinal or scale.
type.out currently only binary is implemented

Value
A four-dimensional array of product comparisons having I assessors, J(J-1)/2 product comparisons, outcomes (see values parameter), M attributes

References
**Examples**

```r
data(bread)

# Get the 4d array of CATA differences for the first 10 consumers
b <- barray(bread$cata[1:10,,])
```

---

**bcluster**

*Wrapper function for b-cluster analysis*

**Description**

By default, `bcluster` calls a function to perform b-cluster analysis by a non-hierarchical iterative ascent algorithm, then inspects results if there are multiple runs.

**Usage**

```r
bcluster(X, inspect = TRUE, inspect.plot = TRUE, algorithm = "n", measure = "b", G = NULL, M = NULL, max.iter = 500, tol = exp(-32), runs = 1, seed = 2021, verbose = FALSE)
```

**Arguments**

- **X**: three-way array with $I$ assessors, $J$ products, $M$ attributes where CATA data have values 0 (not checked) and 1 (checked)
- **inspect**: default (TRUE) calls the `inspect` function to evaluate all solutions (when runs>1)
- **inspect.plot**: default (TRUE) plots results from the `inspect` function
- **algorithm**: default is n for non-hierarchical; h for hierarchical
- **measure**: default is b for the b-measure; Q for Cochran’s Q test
- **G**: number of clusters (required for non-hierarchical algorithm)
- **M**: initial cluster memberships
- **max.iter**: maximum number of iteration allowed (default 500)
- **tol**: non-hierarchical algorithm stops if variance over 5 iterations is less than tol (default: exp(-32))
- **runs**: number of runs (defaults to 1)
- **seed**: for reproducibility (default is 2021)
- **verbose**: for reproducibility (default is FALSE)

**Value**

list with elements:

- **runs**: b-cluster analysis results from `bcluster.n` or `bcluster.h` (in a list if runs>1)
- **inspect**: result from `inspect` (the plot from this function is rendered if inspect.plot is TRUE)
References


Examples

```r
data(bread)
# b-cluster analysis on the first 14 consumers and the first 6 attributes
(b1 <- bcluster(bread$cata[1:14,,1:6], G=2))
# identical to:
# (b2 <- bcluster.n(bread$cata[1:10,,1:6], G=2))
```

bcluster.h

*b-cluster analysis by hierarchical agglomerative strategy*

Description

Perform b-clustering using the hierarchical agglomerative clustering strategy.

Usage

```r
bcluster.h(X, measure = "b", runs = 1, seed = 2021, verbose = FALSE)
```

Arguments

- **X**: three-way array; the I, J, M array has I assessors, J products, M attributes where CATA data have values 0 (not checked) and 1 (checked)
- **measure**: currently only b (the b-measure) is implemented
- **runs**: number of runs (defaults to 1; use a higher number of runs for a real application)
- **seed**: for reproducibility (default is 2021)
- **verbose**: maximum number of iterations

Value

An object of class hclust from hierarchical b-cluster analysis results (a list of such objects if runs>1), where each hclust object has the structure described in hclust as well as the item retainedB (a vector indicating the retained sensory differentiation at each iteration (merger)).

References

Examples

```r
data(bread)

# hierarchical b-cluster analysis on first 10 consumers and first 6 attributes
b <- bcluster.h(bread$cata[1:14,,1:6])

plot(as.dendrogram(b),
     main = "Hierarchical b-cluster analysis",
     sub = "10 bread consumers on 6 attributes")
```

**bcluster.n**

*b-cluster analysis by non-hierarchical iterative ascent clustering strategy*

**Description**

Non-hierarchical b-cluster analysis transfers assessors iteratively to reach a local maximum in sensory differentiation retained.

**Usage**

```r
bcluster.n(X, G, M = NULL, measure = "b", max.iter = 500, runs = 1, X.input = "data", tol = exp(-32), seed = 2021, verbose = FALSE)
```

**Arguments**

- **X**: CATA data organized in a three-way array (assessors, products, attributes)
- **G**: number of clusters (required for non-hierarchical algorithm)
- **M**: initial cluster memberships
- **measure**: b (default) for the b-measure is implemented
- **max.iter**: maximum number of iteration allowed (default 500)
- **runs**: number of runs (defaults to 1)
- **X.input**: either "data" (default) or "bc" if X is obtained from the function barray
- **tol**: algorithm stops if variance over 5 iterations is less than tol (default: exp(-32))
- **seed**: for reproducibility (default is 2021)
- **verbose**: maximum number of iterations

**Value**

An object of class bclust.n (or a list of such objects if runs>1), where each such object has the following components:

- **cluster**: vector of the final cluster memberships
- **totalB**: value of the total sensory differentiation in data set
• retainedB: value of sensory differentiation retained in b-cluster analysis solution
• progression: vector of sensory differentiation retained in each iteration
• iter: number of iterations completed
• finished: boolean indicates whether the algorithm converged before max.iter

References


Examples
data(bread)

# b-cluster analysis on the first 10 consumers and the first 6 attributes
(b <- bcluster.n(bread$cata[1:10, , 1:6], G=2))

---

**cochranQ**  
*Cochran’s Q test*

**Description**

Calculate Cochran’s Q test statistic. The null hypothesis that is assumed is that product proportions are all equal. The alternative hypothesis is that product proportions are not all equal.

**Usage**

```r
cochranQ(X, na.rm = TRUE, quiet = FALSE, digits = getOption("digits"))
```

**Arguments**

- **X**: matrix of I assessors (rows) and J products (columns) where values are 0 (not checked) or 1 (checked)
- **na.rm**: should NA values be removed?
- **quiet**: if FALSE (default) then it prints information related to the test; if TRUE it returns only the test statistic (Q)
- **digits**: significant digits (to display)

**Value**

Q test statistic
Consumer CATA data set: bread

References


See Also

mcnemarQ

Examples

data(bread)
# Cochran’s Q test on the first 40 consumers on the first attribute ("Fresh")
cochranQ(bread$cata[1:40,,1])

Description

Raw results from CATA and Liking evaluations of six bread products samples by 161 consumers.

Format

A list with 4 items:

- `$cata`: check-all-that-apply (CATA) data (array, 161 consumers x 6 breads x 31 sensory attributes)
- `$liking`: 9-point hedonic scale data (matrix, 161 consumers x 6 breads)
- `$ideal.cata`: check-all-that-apply (CATA) data for ideal bread (matrix, 161 consumers x 31 sensory attributes)
- `$liking`: 9-point hedonic scale data for ideal bread(vector, 161 consumers)

CATA data is coded 1 if the attribute is checked; otherwise it is coded 0

References


Examples

data(bread)
head(bread$cata)
evaluateClusterQuality

Evaluate Quality of Cluster Analysis Solution

Description

Evaluate the quality of cluster analysis solutions using measures related to within-cluster product discrimination, between-cluster non-redundancy, overall diversity (coverage), average RV, sensory differentiation retained, and within-cluster homogeneity.

Usage

```
evaluateClusterQuality(X, M, alpha = .05, M.order = NULL,
quiet = FALSE, digits = getOption("digits"), ...)  
```

Arguments

- **X**: three-way array; the I, J, M array has I assessors, J products, codeM attributes where CATA data have values 0 (not checked) and 1 (checked)
- **M**: cluster memberships
- **alpha**: significance level to be used for two-tailed tests
- **M.order**: can be used to change the cluster numbers (e.g. to label cluster 1 as cluster 2 and vice versa); defaults to NULL
- **quiet**: if FALSE (default) then it prints information quality measures; if TRUE then returns results without printing
- **digits**: significant digits (to display)
- **...**: other parameters for print.default (if quiet = TRUE).

Value

A list containing cluster analysis quality measures:

- **$solution**:
  - Pct.b = percentage of the total sensory differentiation retained in the solution
  - min(NR) = smallest observed between-cluster non-redundancy
  - Div_G = overall diversity (coverage)
  - H_G = overall homogeneity (weighted average of within-cluster homogeneity indices)
  - avRV = average RV coefficient for all between-cluster comparisons
- **$clusters**:
  - ng = number of cluster members
  - bg = sensory differentiation retained in cluster
  - xbarg = average citation rate in cluster
  - Hg = homogeneity index within cluster (see homogeneity)
getb

- \( D_g \) = within-cluster product discrimination

- $nonredundancy.clusterpairs$:
  - square data frame showing non-redundancy for each pair of clusters (low values indicate high redundancy)

- $rv.clusterpairs$:
  - square data frame with RV coefficient for each pair of clusters (high values indicate higher similarity in product configurations)

References


See Also

homogeneity

Examples

data(bread)
evaluateClusterQuality(bread$cata[1:14,,1:6], M = rep(1:2, each = 7))

\[
\text{getb} \quad \text{Calculate the b-measure}
\]

Description

Function to calculate the b-measure, which quantifies the sensory differentiation retained.

Usage

getb(X.b, X.c)

Arguments

- \( X.b \) three-way (I, \( J(J-1)/2 \), M) array with I assessors, \( J(J-1)/2 \) product comparisons, M CATA attributes, where values are counts of type b from the function \text{barray}\

- \( X.c \) array of same dimension as \( X.b \), where values are counts of type b from the function \text{barray}\

Value

b-measure
References


Examples

```r
data(bread)

bread.bc <- barray(bread$cata)
getb(bread.bc[1:10,,1], bread.bc[1:10,,2])
```

| homogeneity | Calculate within-cluster homogeneity |

Description

Within a group of \( N \) consumers, the Homogeneity index lies between \( 1/N \) (no homogeneity) to 1 (perfect homogeneity).

Usage

```r
homogeneity(X)
```

Arguments

- `X` three-way array; the \( I,J,M \) array has \( I \) assessors, \( J \) products, codeM attributes where CATA data have values 0 (not checked) and 1 (checked)

Value

- homogeneity index

References


Examples

```r
data(bread)

# homogeneity index for the first 7 consumers on the first 6 attributes
homogeneity(bread$cata[1:7,,1:6])
```
inspect

Inspect/summarize many b-cluster analysis runs

Description
Inspect many runs of b-cluster analysis. Calculate sensory differentiation retained and recurrence rate.

Usage
inspect(X, G = 2, bestB = NULL, bestM = NULL, inspect.plot = TRUE)

Arguments
X three-way array; the I, J, M array has I assessors, J products, codeM attributes where CATA data have values 0 (not checked) and 1 (checked)
G number of clusters (required for non-hierarchical algorithm)
bestB total sensory differentiation retained in the best solution. If not provided, then bestB is determined from best solution in the runs provided (in X).
bestM cluster memberships for best solution. If not provided, then the best solution is determined from the runs provided (in X).
inspect.plot default (TRUE) plots results from the inspect function

Value
A data frame with unique solutions in rows and the following columns:
• B : Sensory differentiation retained
• pctB : Percentage of the total sensory differentiation retained
• B.prop : Proportion of sensory differentiation retained compared to best solution
• raw.agree : raw agreement with best solution
• count : number of runs for which this solution was observed
• c.1, c.2, ... : remaining columns gives index of the cluster to which the consumers (columns) are allocated

References

Examples
data(bread)

res <- bcluster.n(bread$cata[1:10, , 1:8], G = 2, runs = 5)
inspect(res)
### mcnemarQ

**McNemar’s test**

**Description**

Pairwise tests are conducted using the two-tailed binomial test. These tests can be conducted after Cochran’s Q test.

**Usage**

```r
mcnemarQ(X, na.rm = TRUE, quiet = FALSE, digits = getOption("digits"))
```

**Arguments**

- `X` matrix of I assessors (rows) and J products (columns) where values are 0 (not checked) or 1 (checked)
- `na.rm` should NA values be removed?
- `quiet` if FALSE (default) then it prints information related to the test; if TRUE it returns only the test statistic (Q)
- `digits` significant digits (to display)

**Value**

Test results for all McNemar pairwise tests conducted via the binomial test

**References**


**See Also**

- `cochranQ`

**Examples**

```r
data(bread)

# McNemar's exact pairwise test for all product pairs
# on the first 40 consumers and the first attribute ("Fresh")
mcnemarQ(bread$cata[1:40,,1])
```
Penalty-Lift Analysis

**Description**

Penalty-Lift analysis for CATA variables, which is the difference between the average hedonic response when CATA attribute is checked vs. the average hedonic response when CATA attribute is not checked.

**Usage**

```
plift(X, Y)
```

**Arguments**

- **X**: either a matrix of CATA data with I consumers (rows) and J products (columns) or an array of CATA data with I consumers, J products, and M attributes.
- **Y**: matrix of hedonic data with I consumers (rows) and J products (columns).

**Value**

Penalty lift for the attribute if X is a matrix; otherwise, penalty-lift for each attribute if X is a 3d array.

**References**


**Examples**

```r
data(bread)

# penalty lift, based only on the first 20 consumers
plift(bread$cata[1:20,,1], bread$liking[1:20, ])

# for the first attribute ("Fresh")
plift(bread$cata[1:20,,1:3], bread$liking[1:20, ])
```

# for the first 3 attributes
Calculate RV Coefficient

Description
Calculate RV coefficient

Usage
\[ \text{rv.coef}(X, Y, \text{method} = 1) \]

Arguments
- \( X \): input matrix (same dimensions as \( Y \))
- \( Y \): input matrix (same dimensions as \( X \))
- \( \text{method} \): 1 (default) and 2 give identical RV coefficients

Value
RV coefficient

References

Examples
```r
# Generate some data
set.seed(123)
X <- matrix(rnorm(8), nrow = 4)
Y <- matrix(rnorm(8), nrow = 4)
# get the RV coefficient
rv.coef(X, Y)
```

Salton’s cosine measure

Description
Calculate Salton’s cosine measure

Usage
\[ \text{salton}(X, Y) \]
Arguments

X  input matrix (same dimensions as Y)
Y  input matrix (same dimensions as X)

Value

Salton’s cosine measure

References


Examples

```r
# Generate some data
set.seed(123)
X <- matrix(rnorm(8), nrow = 4)
Y <- matrix(rnorm(8), nrow = 4)

# get Salton’s cosine measure
salton(X, Y)
```

Description

Converts a three-dimensional array (I assessors, J products, M attributes) to a two-dimensional matrix with (I assessors, J products) rows and (M attributes) columns, optionally preceded by two columns of row headers.

Usage

```r
toMatrix(X, header.rows = TRUE)
```

Arguments

X  three-dimensional array (I assessors, J products, M attributes) where values are 0 (not checked) or 1 (checked)
header.rows  TRUE (default) includes row headers; set to FALSE to exclude these headers

Value

A matrix with I assessors * J products in rows and M attributes in columns (preceded by 2 columns) of headers if header.rows = TRUE
Examples

data(bread)

# convert CATA results from the first 10 consumers and the first 4 attributes
# to a tall matrix
toMatrix(bread$cata[1:10,,1:4])

Description

Converts a three-dimensional array (I assessors, J products, M attributes) to a two-dimensional matrix (J products, I assessors * M attributes)

Usage

toWideMatrix(X)

Arguments

X three-dimensional array (I assessors, J products, M attributes) where values are 0 (not checked) or 1 (checked)

Value

A matrix with J products in rows and I assessors * M attributes in columns

Examples

data(bread)

# convert CATA results from the first 10 consumers and the first 4 attributes
# to a wide matrix
toWideMatrix(bread$cata[1:10,,1:4])
Index

barray, 2, 5, 9
bcluster, 3
bcluster.h, 3, 4
bcluster.n, 3, 5
bread (Consumer CATA data set: bread), 7
cochranQ, 6, 12
Consumer CATA data set: bread, 7
evaluateClusterQuality, 8
getb, 9
hclust, 4
homogeneity, 8, 9, 10
inspect, 3, 11, 11
mcnemarQ, 7, 12
pLift, 13
print.default, 8
rv.coef, 14
salton, 14
toMatrix, 15
toWideMatrix, 16