Package ‘CompR’

July 1, 2015

Version 1.0
Date 2015-07-01
Title Paired Comparison Data Analysis
Author Michel Semenou
Maintainer Michel Semenou <michel.semenou@oniris-nantes.fr>
Depends R (>= 3.1), methods, utils, MASS, graphics, stats
Description Different tools for describing and analysing paired comparison data are presented. Main methods are estimation of products scores according Bradley Terry Luce model. A segmentation of the individual could be conducted on the basis of a mixture distribution approach. The number of classes can be tested by the use of Monte Carlo simulations. This package deals also with multi-criteria paired comparison data.
License GPL-2
NeedsCompilation no
Repository CRAN
Date/Publication 2015-07-01 16:06:23

R topics documented:

CompR-package .......................................................... 2
BradleyEstim-class ..................................................... 4
ClassDataPairComp .................................................... 6
ClassifPaired .......................................................... 6
Cocktail ................................................................. 7
Cocktail_Cum .......................................................... 8
C_piBTL ................................................................. 8
DataPairComp-class ................................................... 10
DataSimulH0 ........................................................... 11
EstimBradley .......................................................... 11
getCons ............................................................. 13
getCons-methods ..................................................... 14
getCrit ............................................................... 14
Different tools for describing and analysing paired comparison data are presented. Main methods are estimation of products scores according Bradley Terry Luce model. A segmentation of the individual could be conducted on the basis of a mixture distribution approach. The number of classes can be tested by the use of Monte Carlo simulations. This package deals also with multi-criteria paired comparison data.
Details
Function to estimate products configurations (Bradley’s scores) and weights of the classes is `EstimBradley()`.  
Function to perform a test concerning the number of classes is `ResSimulLvrRatio()`.  
Function to obtain a graphical representation of Bradley’s scores is `Piplot()`.  

**Author(s)**  
Michel Semenou  
Maintainer: <michel.semenou@oniris-nantes.fr>  

**See Also**  
`EstimBradley`, `ResSimulLvrRatio`, `Piplot`  

**Examples**  
```r  
data(Cocktail)  
show(Cocktail)  
ResCock1 <- EstimBradley(Cocktail, Constraint=0, Tcla=1, eps=0.001, eps1=0.001, TestPi=TRUE)  
show(ResCock1)  
Res_LvrRatio1 <- ResSimulLvrRatio(Cocktail, ResCock1, 0, 3, level=0.05, eps=0.001, eps1=0.001)  
getSimu(Res_LvrRatio1)  
getTest(Res_LvrRatio1)  
```  

---  

**BradleyEstim-class**  

A class for Bradley’s scores estimation results  

**Objects from the Class**  

Objects can be created by the function `EstimBradley()`.  

---  

**Description**  

A class for Bradley’s scores estimation results  

---  

**BradleyEstim-class**  

Class "BradleyEstim"  

---  

**Description**  

A class for Bradley’s scores estimation results  

**Objects from the Class**  

Objects can be created by the function `EstimBradley()`.  

---  

**Description**  

A class for Bradley’s scores estimation results  

**Objects from the Class**  

Objects can be created by the function `EstimBradley()`.  

---  

**Description**  

A class for Bradley’s scores estimation results  

**Objects from the Class**  

Objects can be created by the function `EstimBradley()`.  

---  

**Description**  

A class for Bradley’s scores estimation results  

**Objects from the Class**  

Objects can be created by the function `EstimBradley()`.  

---  

**Description**  

A class for Bradley’s scores estimation results  

**Objects from the Class**  

Objects can be created by the function `EstimBradley()`.  

---  

**Description**  

A class for Bradley’s scores estimation results  

**Objects from the Class**  

Objects can be created by the function `EstimBradley()`.  

---  

**Description**  

A class for Bradley’s scores estimation results  

**Objects from the Class**  

Objects can be created by the function `EstimBradley()`.  

---  

**Description**  

A class for Bradley’s scores estimation results  

**Objects from the Class**  

Objects can be created by the function `EstimBradley()`.  

---  

**Description**  

A class for Bradley’s scores estimation results  

**Objects from the Class**  

Objects can be created by the function `EstimBradley()`.  

---  

**Description**  

A class for Bradley’s scores estimation results  

**Objects from the Class**  

Objects can be created by the function `EstimBradley()`.  

---  

**Description**  

A class for Bradley’s scores estimation results  

**Objects from the Class**  

Objects can be created by the function `EstimBradley()`.  

---  

**Description**  

A class for Bradley’s scores estimation results  

**Objects from the Class**  

Objects can be created by the function `EstimBradley()`.  

---  

**Description**  

A class for Bradley’s scores estimation results  

**Objects from the Class**  

Objects can be created by the function `EstimBradley()`.  

---  

**Description**  

A class for Bradley’s scores estimation results  

**Objects from the Class**  

Objects can be created by the function `EstimBradley()`.  

---  

**Description**  

A class for Bradley’s scores estimation results  

**Objects from the Class**  

Objects can be created by the function `EstimBradley()`.  

---  

**Description**  

A class for Bradley’s scores estimation results  

**Objects from the Class**  

Objects can be created by the function `EstimBradley()`.  

---  

**Description**  

A class for Bradley’s scores estimation results  

**Objects from the Class**  

Objects can be created by the function `EstimBradley()`.  

---  

**Description**  

A class for Bradley’s scores estimation results  

**Objects from the Class**  

Objects can be created by the function `EstimBradley()`.  

---  

**Description**  

A class for Bradley’s scores estimation results  

**Objects from the Class**  

Objects can be created by the function `EstimBradley()`.
**Slots**

Lvr: Object of class "matrix" corresponding to the number of iterations of the EM algorithm,
LogLikelihoods at the previous step and the current step, and the differences between these 2
LogLikelihoods

Lvr: Object of class "numeric" final value of the LogLikelihood

Lambda: Object of class "matrix" weights of the different classes

Pi: Object of class "list" Bradley’s scores for each class and each criteria

Zh: Object of class "matrix" with the posterior probabilities for each individual to belong to the
different classes and the class with the higher probability

Ic: Object of class "matrix" value of the different Information criterion (AIC, BIC, CAIC)

Restestglob: Object of class "list" result of testing the whole Bradley’s scores equality for each
class and each criteria

Restestprod: Object of class "list" result of multiple comparison tests for Bradley’s scores in
each class and for each criteria

Varcov: Object of class "list" of covariance matrices of Bradley’s scores in each class and for
each criteria

**Methods**

getIc signature(object = "BradleyEstim")

getLambda signature(object = "BradleyEstim")

getLvr signature(object = "BradleyEstim")

getLvr signature(object = "BradleyEstim")

getPi signature(object = "BradleyEstim")

getRestestglob signature(object = "BradleyEstim")

getRestestprod signature(object = "BradleyEstim")

getVarcov signature(object = "BradleyEstim")

getZh signature(object = "BradleyEstim")

show signature(object = "BradleyEstim")

**Examples**

data(ResCocktail1)

show(ResCocktail1)
ClassDataPairComp

Create an object of class DataPairComp

Description

return an object of DataPairComp class

Usage

ClassDataPairComp(Mat, labelprod = NULL, labelcons = NULL, labelcrit = NULL)

Arguments

Mat: Paired comparison matrix with a number of rows equal to nsubject*nitems and nitems columns.
labelprod: names of the different items (default labelprod=NULL)
labelcons: names of the different subjects (default labelcons=NULL)
labelcrit: name of the criterium (default labelcrit=NULL)

Value

Object of class DataPairComp with the following elements:
Cons: corresponding to the label of consumers (default : Number of consumers)
Crit: name of the different criteria contained
Prod: names of the different products (default : number of the product)
Paircomp: list of number of criteria elements each corresponding to the results of paired comparisons performed by the consumers.

ClassifPaired

Classification of paired comparison data

Description

Returns the result of consumers classification

Usage

ClassifPaired(Data, Tcla)

Arguments

Data: Object of class DataPairComp
Tcla: Number of classes to use for classification
Details

The function performs a hierarchical cluster analysis on a set of dissimilarities based on pairwise comparison matrices, using the functions `hclust` and `cutree` of stats package.

Value

vector with group memberships resulting from the classification with Tcla clusters.

See Also

`hclust`, `cutree` of stats package

Description

Paired comparison of 7 beverages by 112 subjects according their preferences

Usage

data(Cocktail)

Format

A `DataPairComp` class object with the following elements:

Cons: corresponding to the label of consumers (default: Number of consumers)

Crit: name of the different criteria contained

Prod: names of the different products (default: number of the product)

Paircomp: list of number of criteria elements each corresponding to the results of paired comparisons performed by the consumers.

Examples

data(Cocktail)
show(Cocktail)
Cocktail_Cum  Beverages paired comparison

Description

Paired comparison of 7 beverages by 112 subjects according their preferences

Usage

data(Cocktail)

Format

A matrix resulting of the cumulative paired comparison results of 7 products by 112 consumers. The (i,j) element corresponds to the number of times product i was preferred to product j among all comparisons between these two products.

Examples

data(Cocktail_Cum)
Cocktail_Cum

C_piBTL  Estimation of Bradley’s scores

Description

Returns the Bradley’s scores of the different items and the value of the LogLikelihood

Usage

C_piBTL(Matpair, Constraint=0, eps1=1e-04, Pi=NULL, TestPi=FALSE, Zht=NULL)

Arguments

Matpair  Matrix of the cumulative sum of the results of paired comparisons or object of class DataPairComp
Constraint  Kind of constraint on Bradley’s scores. If Constraint=0, the sum of Bradley’s scores should be equal to 1. For other values for Constraint, the product of Bradley’s scores should be equal to 1. (default is Constraint=0)
eps1  value to take into account for the convergence criteria of the algorithm of Bradley’s scores estimation. (default is eps1=1e-04)
Pi

Initial values for Bradley’s scores. If Pi=NULL the initialisation is based on a mean score for each item obtained from the data Matpair. Else, initial values for Bradley’s scores are Pi given by the user. (default is Pi=NULL)

TestPi

Indicate if the user wants to perform a multiple comparison tests on the Bradley’s scores. (default TestPi=FALSE)

Zht

Indicate the individuals probabilities to belong to the different classes. Zht has not to be provided for external use of this function. It is used in the main function EstimBradley (default Zht=NULL)

Details

The algorithm is based on a maximum likelihood approach using Dykstra method.

Value

List of following components:

- Pi Bradley’s scores
- lnl value of the log-likelihood
- lvrHO value of the log-likelihood under the hypothesis of equal values for the Bradley’s scores
- lvrH1 value of the log-likelihood at the end of the Bradley’s scores estimation algorithm
- lRatio value of the likelihood ration statistic
- Pvalue Pvalue of the test
- H1 logical value, FALSE if Bradley’s scores should be considered as equal, TRUE otherwise
- VarcovPi Matrix of covariances of Bradley’s scores
- restestij Matrix of the following elements
  - products i and j compared
  - value of the test statistic
  - Pvalue of the test
  - decision at a 0.05 level

Examples

data(Cocktail_Cum)
res<-C_piBTL(Cocktail_Cum, Constraint=0, epsl=1e-04, Pi=NULL, TestPi=TRUE)
res
DataPairComp-class  Class "DataPairComp"

Description

A class for Paired comparison data

Objects from the Class

Objects can be created by calls of the form new("DataPairComp", ...), or by the function ImportData().

Slots

Cons: Object of class "character" label for the individuals
Crit: Object of class "character" label for the criterion
Prod: Object of class "character" label for the products
Paircomp: Object of class "list" corresponding to the individual results of paired comparisons for each criteria, when products i and j are presented to individual h, the (i,j) element resulting is coded by 1 if i is choosen against j and 0 otherwise

Methods

getCons signature(object = "DataPairComp")
getCrit signature(object = "DataPairComp")
getPaircomp signature(object = "DataPairComp")
getProd signature(object = "DataPairComp")
show signature(object = "DataPairComp")

See Also

ImportData

Examples

data(Cocktail)
show(Cocktail)
DataSimulH0  

Simulation of paired comparison data

Description

Returns paired comparison data according a given configuration

Usage

DataSimulH0(Data, ResH0)

Arguments

- **Data**: Object of class `datapaircomp`
- **ResH0**: Object of class `bradleyestim`.

Details

The paired comparison data are simulated according the products configuration, the weight of the different classes for the different criteria (stored in the object `ResH0` of class `BradleyEstim`) obtained on the basis of the results of `EstimBradley` function for the paired comparison data contained in the object `Data` of class `DataPairComp`.

Value

Object of class `DataPairComp` with the following components:

- **Cons**: corresponding to the label of consumers
- **Crit**: names of the different criteria
- **Prod**: names of the different products
- **Paircomp**: list of number of criteria elements each corresponding to the results of simulated paired comparisons performed by the consumers according their belonging to the different classes.

EstimBradley  

Estimation of Bradley's scores in the different classes of subjects

Description

Estimates Bradley’s scores according the desired number of classes.

Usage

EstimBradley(Data, Constraint=0, Tcla=1, eps=1e-04, eps1=1e-04, TestPi=TRUE)
Arguments

Data Object of class DataPairComp
Constraint Kind of constraint on Bradley's scores. If Constraint=0, the sum of Bradley's scores should be equal to 1. For other values for Constraint, the product of Bradley's scores should be equal to 1. (default constraint=0)
Tcla Number of classes, default=1, no segmentation.
eps value of the convergence criteria for the EM algorithm (default eps=1e-04).
eps1 value of the criteria convergence for Dykstra algorithm (default eps1=1e-04).
TestPi if TestPi=TRUE multiple comparison tests for Bradley's scores are performed. Else no multiple comparison test. (default is TestPi=TRUE)

Details

The estimation is based on maximum likelihood for mixture distributions with E.M. algorithm.

Value

Object of class BradleyEstim with the following components:

Lrwriter matrix describing the evolution of log likelihood at the different steps of the maximization procedure.
Lvr Final value of the log likelihood
Lambda numeric Final estimates of classes' weight
Pi list of Tcla elements containing Bradley's scores for the different criteria
Zh matrix of the belongings probabilities of the individuals to the different classes and the belonging class according to these probabilities
IC value of Information Criterion (AIC,BIC,CAIC)
Restestglob (given if TestPi=TRUE) list of five elements:
lvrH0 matrix of size (Tcla * number of criteria), giving the value of the log likelihood under the hypothesis of equality of Bradley's scores
lvrH1 matrix of size (Tcla * number of criteria), giving the value of the log likelihood under the hypothesis of non equality of Bradley's scores
lratio matrix of size (Tcla * number of criteria), giving the value of the log likelihood Ratio statistic
pvalue matrix of size (Tcla * number of criteria), giving the P value of the log likelihood Ratio test
h1 matrix of size (Tcla * number of criteria) giving the result of rejection of equality of Bradley's scores
Restestprod (given if TestPi=TRUE and if Bradley's scores are not equal) list of Tcla elements of type matrix of size (number of paired comparison * 7), each column corresponding to:
class identification,
criterion identification,
product identification i,
getCons

13

product identification j,
value for the statistic corresponding to H0: equality of the Bradley’s scores of
products i and j,
P value of this test,
Rejection or acceptation of H0 for a level of 5%.

Varcov

(given if TestPi=TRUE)
list of Tcla elements containing Bradley's scores covariance matrices for the
different criteria.

Examples

data(Cocktail)
show(Cocktail)
ResCock1<-EstimBradley(Cocktail,Constraint=0,Tcla=1,eps=0.001,eps1=0.001,TestPi=TRUE)
show(ResCock1)

getCons

Gets the individuals labels.

Description

Gets the individuals labels.

Usage

getCons(object)

Arguments

object An object of class DataPairComp

Value

vector of the individuals labels.

Examples

data(Cocktail)
Cocktail_Cons<-getCons(Cocktail)
**getCons-methods**

Methods for function `getCons`

**Methods**

```
signature(object = "DataPairComp")
```

---

**getCrit**

*Gets the criteria’s labels.*

---

**Description**

Gets the criteria’s labels.

**Usage**

```
getCrit(object)
```

**Arguments**

- `object` An object of class `DataPairComp`

**Value**

vector of the criteria’s labels.

**Examples**

```
data(Cocktail)
Cocktail_Crit<-getCrit(Cocktail)
```

---

**getCrit-methods**

Methods for function `getCrit`

**Methods**

```
signature(object = "DataPairComp")
```
**getIc**

*Gets the Information criteria’s labels.*

**Description**

Gets the Information criteria’s labels (AIC, BIC, CAIC).

**Usage**

`getIc(object)`

**Arguments**

- `object` An object of class BradleyEstim

**Value**

vector of Information criteria.

**Examples**

```r
data(Cocktail)
ResCock<-EstimBradley(Cocktail,Constraint=0,Tcla=1,eps=1e-04,eps1=1e-04,TestPi=TRUE)
ResCock_ic<-getIc(ResCock)
```

---

**getIc-methods**

*Methods for Function getIc*

**Description**

Methods for function `getIc`

**Methods**

`signature(object = "BradleyEstim")`
getLambda

Description

Get the weight of the different classes.

Usage

getlambda(object)

Arguments

object  An object of class BradleyEstim

Value

A vector of the weights of the different classes.

Examples

data(Cocktail)
ResCock<-EstimBradley(Cocktail,Constraint=0,Tcla=1,eps1=1e-04,eps2=1e-04,TestPi=TRUE)
ResCock_Lambda<-getLambda(ResCock)

getLambda-methods

Methods for Function getLambda

Description

Methods for function getLambda

Methods

signature(object = "BradleyEstim")
getLvr

Gets the final value of loglikelihood.

Description

Gets the final value of loglikelihood from the function `estimbradley()`.

Usage

`getLvr(object)`

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>An object of class BradleyEstim</td>
</tr>
</tbody>
</table>

Value

Numeric value of the loglikelihood.

Examples

```r
data(Cocktail)
ResCock<-estimbradley(Cocktail,Constraint=0,Tcla=1,eps=1e-04,eps1=1e-04,TestPi=TRUE)
ResCock_Lvr<-getLvr(ResCock)
```

getLvr-methods

Methods for function `getLvr`

Methods

```r
signature(object = "BradleyEstim")
```
getLvrter-methods

getLvrter

Gets the iteration done until convergence of the loglikelihood estimation of Bradley’s scores.

Description

Gets the iteration done until convergence from the function EstimBradley().

Usage

getLvrter(object)

Arguments

object An object of class BradleyEstim

Value

A matrix with numbers of iteration rows and 4 columns giving the iteration, the previous value of loglikelihood, the current value of the loglikelihood, and the difference between these loglikelihoods.

Examples

data(Cocktail)
ResCock<->EstimBradley(Cocktail,Constraint=0,Tcla=1,eps=1e-04,eps1=1e-04,TestPi=TRUE)
ResCock_Lvriter<->getLvrter(ResCock)

getLvrter-methods

Methods for Function getLvrter

Description

Methods for function getLvrter

Methods

signature(object = "BradleyEstim")
**Description**

Gets the individual paired comparisons.

**Usage**

```r
getPaircomp(object)
```

**Arguments**

- `object` An object of class `DataPairComp`

**Value**

A list of number of criteria elements each corresponding to the results of paired comparisons performed by the consumers.

**Examples**

```r
data(Cocktail)
Cocktail_Paircomp<-getPaircomp(Cocktail)
```

**Description**

Methods for function `getPaircomp`

**Methods**

```r
signature(object = "DataPairComp")
```
getPi-methods

---

getPi  

*Gets the Bradley's scores.*

---

**Description**

Gets the Bradley's scores from the function `estimBradley()`.

**Usage**

`getPi(object)`

**Arguments**

- **object**
  
  An object of class `BradleyEstim`

**Value**

A list of the Bradley’s scores for the different criteria.

**Examples**

```r
data(Cocktail)
ResCock <- estimBradley(Cocktail, Constraint=0, Tcla=1, eps=1e-04, eps1=1e-04, TestPi=TRUE)
ResCock_Pi <- getPi(ResCock)
```

---

getPi-methods  

*Methods for Function `getPi`*

---

**Description**

Methods for function `getPi`

**Methods**

`signature(object = "BradleyEstim")`
getProd

gets the products labels.

Description

gets the products labels.

Usage

getProd(object)

Arguments

object An object of class DataPairComp

Value

vector of the products labels.

Examples

data(Cocktail)
Cocktail_Prod<-getProd(Cocktail)

getProd-methods

Methods for Function getProd

Description

Methods for function getProd

Methods

signature(object = "DataPairComp")
getRestestglob-methods

getRestestglob

*Gets the result of the test of Bradley’s scores equality.*

**Description**

Gets the result of the test of Bradley’s scores equality from the function `estimBradley()`.

**Usage**

```r
getRestestglob(object)
```

**Arguments**

- `object` An object of class `BradleyEstim`

**Value**

list of five elements:

- `lvrH0` matrix of size \((Tcla \times \text{number of criteria})\), giving the value of the log likelihood under the hypothesis of equality of Bradley’s scores.
- `lvrH1` matrix of size \((Tcla \times \text{number of criteria})\), giving the value of the log likelihood under the hypothesis of non equality of Bradley’s scores.
- `lratio` matrix of size \((Tcla \times \text{number of criteria})\), giving the value of the log likelihood Ratio statistic.
- `pvalue` matrix of size \((Tcla \times \text{number of criteria})\), giving the P value of the log likelihood Ratio test.
- `h1` matrix of size \((Tcla \times \text{number of criteria})\) giving the result of rejection of equality of Bradley’s scores.

**Examples**

```r
data(Cocktail)
ResCock<-estimBradley(Cocktail,Constraint=0,Tcla=1,eps=1e-04,eps1=1e-04,TestPi=TRUE)
ResCock_Restestglob<-getRestestglob(ResCock)
```

---

getRestestglob-methods

*Methods for Function* `getRestestglob`

**Description**

Methods for function `getRestestglob`

**Methods**

signature(object = "BradleyEstim")
**getRestestprod**

*Gets the result of the Bradley’s scores multiple comparison tests.*

**Description**

Gets the result of the Bradley’s scores multiple comparison tests from the function `estimbradley()`.

**Usage**

```r
getRestestprod(object)
```

**Arguments**

- `object`: An object of class `BradleyEstim`

**Value**

A list of `Tcla` elements of type `matrix` of size (number of paired comparison * 7), each column corresponding to:
- class identification,
- criterion identification,
- product identification i,
- product identification j,
- value for the statistic corresponding to $H_0$: equality of the Bradley’s scores of products i and j,
- P value of this test,
- Rejection or acceptance of $H_0$ for a level of 5%.

**Examples**

```r
data(Cocktail)
ResCock<-estimbradley(Cocktail,Constraint=0,Tcla=1,eps=1e-04,eps1=1e-04,TestPi=TRUE)
ResCock_Restestprod<-getRestestprod(ResCock)
```

**getRestestprod-methods**

*Methods for Function getRestestprod*

**Description**

Methods for function `getRestestprod`

**Methods**

```r
signature(object = "BradleyEstim")
```
**getSimu**

*Gets the results of Likelihood Ratio Test.*

**Description**

Gets the results of Likelihood Ratio Test obtained by Monte-Carlo simulations.

**Usage**

```r
getSimu(object)
```

**Arguments**

- `object`  
  An object of class `LvrRatio`

**Value**

A matrix with the number of classes under H0, the values of Loglikelihood under H0 and H1 and the differences between these Loglikelihoods.

**getSimu-methods**

*Methods for Function getSimu*

**Description**

Methods for function `getSimu`

**Methods**

```r
signature(object = "LvrRatio")
```

**getTest**

*Gets the level and the quantile of Likelihood ratio test.*

**Description**

Gets the level and the quantile of Likelihood ratio test from the function `ResSimuLvrRatio()`

**Usage**

```r
getTest(object)
```

**Arguments**

- `object`  
  An object of class `LvrRatio`
getTest-methods

Value

Matrix with the level and the associated quantile after performing Likelihood Ratio test.

getTest-methods Methods for Function getTest

Description

Methods for function getTest

Methods

signature(object = "LvrRatio")

getVarcov Gets the Bradley's scores covariance matrices.

Description

Gets the Bradley's scores covariance matrices from the function EstimBradley().

Usage

getVarcov(object)

Arguments

object An object of class BradleyEstim

Value

list of Tcla elements containing Bradley's scores covariance matrices for the different criteria.

Examples

data(Cocktail)
ResCock<-EstimBradley(Cocktail,Constraint=0,Tcla=1,eps=1e-04,eps1=1e-04,TestPi=TRUE)
ResCock_Varcov<-getVarcov(ResCock)
Description

Methods for function getVarcov

Methods

signature(object = "BradleyEstim")

getZh

Gets the result of the function EstimBradley()

Description

Gets the posterior probabilities for each individual to belong to the different classes and the class with the higher probability.

Usage

getch(object)

Arguments

object An object of class BradleyEstim

Value

Object of class matrix with the posterior probabilities for each individual to belong to the different classes and the class with the higher probability.

Examples

data(Cocktail)
ResCock2<-EstimBradley(Cocktail,Constraint=0,Tcla=2,eps=1e-04,eps1=1e-04,TestPi=TRUE)
ResCock2_Zh<-getZh(ResCock2)
**getZh-methods**

**Methods for Function getZh**

**Description**

Methods for function getZh

**Methods**

signature(object = "BradleyEstim")

---

**ImportData**

**Import data file**

**Description**

Import the different paired comparison data files in cvs format and create an object of class DataPairComp

**Usage**

ImportData(name, labelprod=FALSE, labelconso=NULL, sep =";", dec=".")

**Arguments**

- **name**: part of name of the different data files (.csv files)
- **labelprod**: indicate the existence of labels of the different products in data files (default=FALSE) given in the header of each column of the data files.
- **labelconso**: vector of label of consumers given by the user (default=NULL)
- **sep**: the field separator character. Values on each line of the file are separated by this character.(default=";")
- **dec**: the character used in the file for decimal points.(default=".")

**Value**

Object of class DataPairComp with the following elements:

- **Cons**: corresponding to the label of consumers (default: Number of consumer)
- **Crit**: names of the different criteria contained in the name of the different data files
- **Prod**: names of the different products (default: number of the product)
- **Paircomp**: list of number of criteria elements each corresponding to the results of paired comparisons performed by the consumers.
LvrRatio-class  Class "LvrRatio"

Description

A class for Likelihood Ration Test results

Objects from the Class

Objects can be created by ResSimullvrRatio().

Slots

Simu: Object of class "matrix" with the number of classes under H0, Loglikelihoods under H0 and H1, difference between these Loglikelihoods.

Test: Object of class "matrix" with the level and the associated quantile after performing Likelihood Ratio test.

Methods

getSimu signature(object = "LvrRatio")

getTest signature(object = "LvrRatio")

Examples

showClass("LvrRatio")

Piplot  Graphical representation of the Bradley's scores

Description

Gives a graphical representation of the Bradley’s scores.

Usage

Piplot(Pi, SigmaPi = NULL, level=0.05, main = NULL, ylab = "Bradley's scores", xlab = "Item", labelprod = NULL)
Arguments

- **Pi**: vector of Bradley’s scores
- **SigmaPi**: vector of Bradley’s scores standard deviation given by the user. (default SigmaPi=NULL)
- **level**: level to use for the confidence intervals. (default level=0.05)
- **main**: Title of the plot. (default main=NULL)
- **ylab**: value for ylab. (default ylab= Bradley’s scores)
- **xlab**: value for xlab. (default xlab=Item)
- **labelprod**: label vector of the Item. (default labelprod=NULL)

Details

The representation is based on plot(x) function, with Item on x axis, and Bradley’s scores on y axis. If SigmaPi is provided by user, a 1-level (default 95%) confidence interval is drawn for each Item.

Value

A graphical representation of bradley’s scores.

Examples

```r
data(Cocktail_Cum)
res<-C_piBMI(Cocktail_Cum,Constraint=0,epsI=0.0001,Pi=NULL,TestPi=TRUE)
Res_Pi<-res$Pi
Res_Varcov<-res$VarcovPi
Res_Sigma<-sqrt(diag(Res_Varcov))
Piplot(Res_Pi, SigmaPi = Res_Sigma, level=0.01, main = NULL, ylab = "Bradley's scores", xlab = "Item", labelprod = NULL)
```

---

**ResCocktail1**

*Result of EstimBradley function for 1 class and data Cocktail*

Description

Result of EstimBradley function for 1 class and data Cocktail

Usage

data(ResCocktail1)

Format

A BradleyEstim class object with the following elements:
Examples

```r
data(ResCocktail1)
show(ResCocktail1)
```

---

**ResSimulLvrRatio**  
*Log Likelihood Ratio Test for Paired comparison data*

**Description**

Returns the result of Log Likelihood Ratio Test of the number of classes for Paired comparison data (T classes versus (T+1) classes)

**Usage**

```r
ResSimulLvrRatio(data, ResH0, Constraint, nsimul, level, eps=1e-04, eps1=1e-04)
```

**Arguments**

- **Data**: Object of class `DataPairComp`
- **ResH0**: Object of class `BradleyEstim` corresponding to the result of `BradleyEstim()` function for T classes (H0)
- **Constraint**: Kind of constraint on Bradley’s scores. If `Constraint=0`, the sum of Bradley’s scores should be equal to 1. For other values for `Constraint`, the product of Bradley’s scores should be equal to 1 (default `Constraint=0`).
- **nsimul**: number of Monte Carlo simulations
- **level**: level of the Log Likelihood Ratio test defined by the user (default `level=0.05`).
- **eps**: value of the convergence criteria for the EM algorithm (default `eps=1e-04`).
- **eps1**: value of the criteria convergence for Dykstra algorithm (default `eps1=1e-04`).

**Details**

The likelihood ratio test is based on a Monte Carlo procedure. A simulation of `nsimul` data set is done. We perform estimation of the different parameters for the number of classes defined in the object `ResH0` of class `BradleyEstim` (corresponding to the null hypothesis) and for one more class corresponding to the alternative hypothesis.

We obtain a set of Log Likelihoods under the null and alternative hypothesis on the basis of simulated data and so of the Log Likelihood Ratio Statistic.

We replace the observed value of this statistic for the true data set. And we conclude on the acceptance or not of the null hypothesis (no differences between T and T+1 classes).

**Value**

Object of class `LvrRatio` with the following components:

- **Simu**: Matrix with the number of classes under H0, Loglikelihoods under H0 and H1, difference between these Loglikelihoods.
- **Test**: Matrix with the level of the test and the associated quantile
show-methods

Examples

data(Cocktail)
ResCock1<-EstimBradley(Cocktail,Constraint=0,Tcla=1,eps=1e-04,eps1=1e-04,TestPi=TRUE)
Res_LvrRat1<--ResSimuLvrRatio(Cocktail,ResCock1,0,3,level=0.05,eps=0.001,eps1=0.001)
getSimu(Res_LvrRat1)
getTest(Res_LvrRat1)

Description

Methods for function show

Methods

signature(object = "BradleyEstim")
signature(object = "DataPairComp")
Index

*Topic **package**
   CompR-package, 2
   BradleyEstim-class, 4

C_pibtl, 8
CClassDataPairComp, 6
ClassifPaired, 6
Cocktail, 7
Cocktail_Cum, 8
CompR (CompR-package), 2
CompR-package, 2

DataPairComp-class, 10
DataSimuH0, 11

EstimBradley, 4, 11

getCons, 13
getCons, DataPairComp-method
   (DataPairComp-class), 10
getCons-methods, 14
getCrit, 14
getCrit, DataPairComp-method
   (DataPairComp-class), 10
getCrit-methods, 14
getIc, 15
getIc, BradleyEstim-method
   (BradleyEstim-class), 4
getIc-methods, 15
getLambda, 16
getLambda, BradleyEstim-method
   (BradleyEstim-class), 4
getLambda-methods, 16
getLvr, 17
getLvr, BradleyEstim-method
   (BradleyEstim-class), 4
getLvr-methods, 17
getLvrriter, 18
getLvrriter, BradleyEstim-method
   (BradleyEstim-class), 4
getLvrriter-methods, 18
getPaircomp, 19
getPaircomp, DataPairComp-method
   (DataPairComp-class), 10
getPaircomp-methods, 19
getPi, 20
getPi, BradleyEstim-method
   (BradleyEstim-class), 4
getPi-methods, 20
getProd, 21
getProd, DataPairComp-method
   (DataPairComp-class), 10
getProd-methods, 21
getRestestglob, 22
getRestestglob, BradleyEstim-method
   (BradleyEstim-class), 4
getRestestglob-methods, 22
getRestestprod, 23
getRestestprod, BradleyEstim-method
   (BradleyEstim-class), 4
getRestestprod-methods, 23
getSimu, 24
getSimu, LvrRatio-method
   (LvrRatio-class), 28
getSimu-methods, 24
getTest, 24
getTest, LvrRatio-method
   (LvrRatio-class), 28
getTest-methods, 25
getVarcov, 25
getVarcov, BradleyEstim-method
   (BradleyEstim-class), 4
getVarcov-methods, 26
getZh, 26
getZh, BradleyEstim-method
   (BradleyEstim-class), 4
getZh-methods, 27

ImportData, 27
INDEX

LvrRatio-class, 28

Piplot, 4, 28

ResCocktail1, 29
ResSimulLvrRatio, 4, 30

show, BradleyEstim-method
   (BradleyEstim-class), 4
show, DataPairComp-method
   (DataPairComp-class), 10
show-methods, 31