Package ‘gosset’

October 13, 2022

Type Package
Title Tools for Data Analysis in Experimental Agriculture
Version 0.5
URL https://agrdatasci.github.io/gosset/

BugReports https://github.com/agrdatasci/gosset/issues

Description Toolkit for a workflow to analyse experimental agriculture data, from data synthesis to model selection and visualisation. The package is named after W.S. Gosset aka ‘Student’, a pioneer of modern statistics in small sample experimental design and analysis.

License MIT + file LICENSE
Encoding UTF-8
LazyData true
Depends R (>= 3.5.0), methods
Imports BradleyTerry2, DescTools, ggplot2, ggparty, ggrepel, partykit, PlackettLuce, psychotools, qvcalc, stats, tidyr, utils, patchwork

Suggests climatrends, ClimMobTools, chirps, gnm, knitr, psychotree, nasapower, rmarkdown

Language en-GB

RoxygenNote 7.2.1
VignetteBuilder knitr

NeedsCompilation no

Author Kauê de Sousa [aut, cre] (<https://orcid.org/0000-0002-7571-7845>), Jacob van Etten [aut] (<https://orcid.org/0000-0001-7554-2558>), David Brown [aut] (<https://orcid.org/0000-0003-2859-1618>), Jonathan Steinke [aut] (<https://orcid.org/0000-0001-5742-702X>)

Maintainer Kauê de Sousa <desousa.kaue@gmail.com>

Repository CRAN

Date/Publication 2022-09-23 16:20:06 UTC
**R topics documented:**

- akaike_weights .................................................. 2
- breadwheat ....................................................... 3
- btpermute .......................................................... 5
- compare ............................................................. 6
- crossvalidation .................................................... 7
- gosset ............................................................... 10
- kendallT tau ....................................................... 10
- kendallW ............................................................ 12
- kenyachoice ....................................................... 13
- nicabe a n .......................................................... 14
- node_labels ......................................................... 15
- pseudoR2 ............................................................ 16
- pseudo_rank ........................................................ 17
- rank_binomial ...................................................... 18
- rank_numeric ....................................................... 19
- rank_paircomp ..................................................... 21
- rank_tricot .......................................................... 22
- regret ............................................................... 24
- reliability ........................................................... 26
- rowbind ............................................................. 28
- worth_map .......................................................... 29

**Index**

<table>
<thead>
<tr>
<th>akaike_weights</th>
<th>Akaike weights</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>31</td>
</tr>
</tbody>
</table>

**Description**

Akaike weights represent the relative likelihood of a model. It can be used in model averaging and selection.

**Usage**

akaike_weights(object)

**Arguments**

- object  
  a numerical vector with models goodness of fit coefficients

**Value**

A data frame containing the coefficients:

- delta  
  the delta overall change in the coefficients
- relative_logLik  
  the relative log-likelihood
- akaike_weights  
  the Akaike weights
Author(s)
Kauê de Sousa and Jacob van Etten

References

Examples

data("airquality")

# try three model approaches
mod1 <- glm(Temp ~ 1,
  data = airquality,
  family = poisson())

mod2 <- glm(Temp ~ Ozone,
  data = airquality,
  family = poisson())

mod3 <- glm(Temp ~ Ozone + Solar.R,
  data = airquality,
  family = poisson())

# models AICs together in a single vector
models <- c(mod1 = AIC(mod1),
  mod2 = AIC(mod2),
  mod3 = AIC(mod3))

# calculate akaike weights
aw <- akaike_weights(models)

# the higher the better
names(models[which.max(aw$akaike_weights)])

---

breadwheat  Preferred bread wheat varieties

Description
Data from decentralized on-farm trials of bread wheat (*Triticum aestivum* L.) varieties in Vaishali, India over the 2014’s Rabi season. Farmers were asked to test three varieties of bread wheat randomly assigned as incomplete blocks of three varieties (out of 16 varieties) and assess which one had the best and worst performance considering four traits, germination, grain quality, yield and overall performance.
Usage

breadwheat

Format

A data frame with 493 records and 19 variables:

variety_a  The name of variety A in the comparison.
variety_b  The name of variety B in the comparison.
variety_c  The name of variety C in the comparison.
district  The administrative region where the experiment was established.
village  The village within the administrative region where the experiment was established.
participant_name  The participant name (omitted for protection and privacy).
age  The participant age.
gender  The participant gender M = Male; F = Female.
planting_date  The date which the experiment started.
lon The longitude in which the experiment was established.
lat The latitude in which the experiment was established.
germination_best  The variety ranked as best for germination ("A", "B" or "C").
germination_worst  The variety ranked as worst for germination ("A", "B" or "C").
grainquality_best  The variety ranked as best for grain quality ("A", "B" or "C").
grainquality_worst  The variety ranked as worst for grain quality ("A", "B" or "C").
yield_best  The variety ranked as best for yield ("A", "B" or "C").
yield_worst  The variety ranked as worst for yield ("A", "B" or "C").
overall_best  The variety ranked as best for overall performance ("A", "B" or "C").
overall_worst  The variety ranked as worst for overall performance ("A", "B" or "C").

Source

The data were provided by Bioversity International https://www.bioversityinternational.org/, a CGIAR Research Centre, through the ClimMob platform https://climmob.net.
btpermute

Variable selection with Permuted Inclusion Criterion

Description

Method of forward variable selection based on deviance for Bradley-Terry models using pairwise ranking data. The selection procedure consists of two steps, first, permuting the variables from the original predictors with \texttt{n.iterations}, then performing a forward selection to retain the predictors with highest contribution to the model, see details.

Usage

\texttt{btpermute(}
\texttt{contests = NULL,}
\texttt{predictors = NULL,}
\texttt{n.iterations = 15,}
\texttt{seed = NULL,}
\texttt{...)}

Arguments

- \texttt{contests} \hspace{1cm} a data frame with pairwise binary contests with these variables 'id','player1','player2','win1','win2'; in that order. The id should be equivalent to the index of each row in \texttt{predictors}
- \texttt{predictors} \hspace{1cm} a data frame with player-specific variables with row indices that should match with the ids in \texttt{contests}. An id is not required, only the predictor variables, the ids are the index for each row
- \texttt{n.iterations} \hspace{1cm} integer, number of iterations to compute
- \texttt{seed} \hspace{1cm} integer, the seed for random number generation. If NULL (the default), \texttt{gosset} will set the seed randomly
- ... \hspace{1cm} additional arguments passed to \texttt{BradleyTerry2} methods

Details

The selection procedure consists of two steps. In the first step, \texttt{btpermute} adds to the set of original (candidate) predictors variables an additional set of 'fake', permuted variables. This set of permuted predictors is created by assigning to each ranking the variables from another, randomly selected ranking. The permuted variables are not expected to have any predictive power for pairwise rankings. In the second step, \texttt{btpermute} adds predictors to the Bradley-Terry model in a forward selection procedure. Each predictors (real and permuted) is added to the null model individually, and \texttt{btpermute} retains which variable reduces model deviance most strongly. The two-step process is replicated \texttt{n} times with argument \texttt{n.iterations}. At each iteration, a new random permutation is generated and all variables are tested. Replicability can be controlled using argument \texttt{seed}. Across the \texttt{n n.iterations}, the function identifies the predictor that appeared most often as the most deviance-reducing one. When this is a real variable, it is constantly added to the model and the forward selection procedure moves on – again creating new permutations, adding
real and fake variables individually, and examining model deviance. Variable selection stops when a permuted variable is found to be most frequently the most deviance-reducing predictors across n.iterations. In turn, variable selection continuous as long as any real variable has stronger explanatory power for pairwise rankings than the random variables.

Value

an object of class gosset_btpermute with the final BTm() model, selected variables, seeds (random numbers) used for permutations and deviances

Author(s)

Jonathan Steinke and Kauê de Sousa

References


See Also

rank_binomial, BTm

Other model selection functions: crossvalidation()

Examples

require("BradleyTerry2")
data("kenyachoice", package = "gosset")
mod <- btpermute(contests = kenyachoice$contests,
predictors = kenyachoice$predictors,
n.iterations = 10,
seed = 1)
mod

---

**compare**

*Compare agreement between two methods*

**Description**

Measures the agreement between two methods
Usage

```
crossvalidation
```

### Default S3 method:
```
crossvalidation(x, y, labels = NULL, ...)
```

### S3 method for class 'PlackettLuce'
```
crossvalidation(x, y, ...)
```

Arguments

- `x`: a numeric vector, or an object of class `PlackettLuce`
- `y`: a numeric vector, or an object of class `PlackettLuce`
- `...`: additional arguments passed to methods
- `labels`: optional, a vector with the same length `x` to plot values

Value

a ggplot with the agreement

References


Examples

```
set.seed(1)
x <- runif(10, -1, 2)
set.seed(2)
y <- runif(10, -1, 2)
compare(x, y)
```

---

crossvalidation    Cross-validation

Description

Methods for measuring the performance of a predictive model on sets of test data in Bradley-Terry model from **psychotree**, Generalized Linear and Generalized Nonlinear models from **gnm**, and Plackett-Luce model from **PlackettLuce**
Usage

crossvalidation(formula, data, k = 10, folds = NULL, seed = NULL, ...)

## S3 method for class 'bttree'
AIC(object, newdata = NULL, ...)

## S3 method for class 'bttree'
deviance(object, newdata = NULL, ...)

## S3 method for class 'pltree'
deviance(object, newdata = NULL, ...)

## S3 method for class 'gnm'
AIC(object, newdata = NULL, ...)

## S3 method for class 'gnm'
deviance(object, newdata = NULL, ...)

Arguments

- **formula**: an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted, of the form \( y \ x_1 + \ldots + x_n \)
- **data**: a data frame (or object coercible by as.data.frame to a data frame) containing the variables in the model
- **k**: an integer for the number of bins in the cross-validation
- **folds**: an optional vector or list of vectors specifying the \( k \)-folds in the cross-validation
- **seed**: integer, the seed for random number generation. If NULL (the default), gosset will set the seed randomly
- **...**: additional arguments passed the methods of the chosen model
- **object**: a model object
- **newdata**: a data.frame with test data

Value

An object of class gosset.cv with the cross-validation goodness-of-fit estimates, which are:

- **AIC**: Akaike Information Criterion
- **deviance**: Model deviance
- **logLik**: Log-Likelihood
- **MaxLik**: Maximum likelihood pseudo R-squared
- **CraggUhler**: Cragg and Uhler’s pseudo R-squared
- **McFadden**: McFadden pseudo R-squared
- **kendallTau**: the Kendall correlation coefficient
Author(s)
Kauê de Sousa, Jacob van Etten and David Brown

References

See Also
bttree, gnm, pltree
Other model selection functions: btpermute()

Examples

# Generalized Linear Models
if (require("gnm")) {
data("airquality")

cv <- crossvalidation(Temp ~ Wind + Solar.R,
   data = airquality,
   k = 3,
   seed = 999,
   family = poisson())
}

# Plackett-Luce Model
if(require("PlackettLuce")) {
# beans data from PlackettLuce
data("beans", package = "PlackettLuce")

G <- rank_tricot(data = beans,
   items = c(1:3),
   input = c(4:5),
   additional.rank = beans[c(6:8)],
   group = TRUE)

beans <- cbind(G, beans)

# take seasons as bins
k <- length(unique(beans$season))
folds <- as.integer(as.factor(beans$season))

cv <- crossvalidation(G ~ maxTN,
   data = beans,
   k = k,
   folds = folds,
gosset

Tools for Data Analysis in Experimental Agriculture

Description

Toolkit for a workflow to analyse experimental agriculture data, from data synthesis to model selection and visualisation. The package is named after W.S. Gosset aka ‘Student’, a pioneer of modern statistics in small sample experimental design and analysis.

Author(s)

Kauê de Sousa and Jacob van Etten and David Brown and Jonathan Steinke

See Also

Useful links:

- Development repository: https://github.com/AgrDataSci/gosset
- Static documentation: https://AgrDataSci.github.io/gosset/
- Report bugs: https://github.com/AgrDataSci/gosset/issues

kendallTau

Kendall rank correlation coefficient

Description

Compute Kendall rank correlation coefficient between two objects. Kendall is a coefficient used in statistics to measure the ordinal association between two measured quantities. A tau test is a non-parametric hypothesis test for statistical dependence based on the tau coefficient. The 'kendallTau' function applies the "kendall" method from 'stats::cor' with some previous treatment in the data, such as converting floating numbers into ranks (from the higher being the first and negative being the last) and the possibility to remove zeros from incomplete ranks.
Usage

kendallTau(x, y, null.rm = TRUE, ...)

## Default S3 method:
kendallTau(x, y, null.rm = TRUE, ...)

## S3 method for class 'matrix'
kendallTau(x, y, ...)

## S3 method for class 'rankings'
kendallTau(x, y, ...)

## S3 method for class 'grouped_rankings'
kendallTau(x, y, ...)

## S3 method for class 'paircomp'
kendallTau(x, y, ...)

Arguments

x a numeric vector, matrix or data frame
y a vector, matrix or data frame with compatible dimensions to x
null.rm logical, to remove zeros from x and y
... further arguments affecting the Kendall tau produced. See details

Value

The Kendall correlation coefficient and the Effective N, which is the equivalent N needed if all items were compared to all items. Can be used for significance testing.

Author(s)

Kauê de Sousa and Jacob van Etten

References


See Also

cor

Other goodness-of-fit functions: kendallW(), pseudoR2()

Examples

library("PlackettLuce")

R <- matrix(c(1, 2, 4, 3, 2, 4, 3, 1, 4, 3, 1, 2, 3, 4, 1, 3, 2), nrow = 4, byrow = TRUE)

kendallW(R, null.rm = TRUE)
kendallW

1, 4, 2, 3,
1, 2, 4, 3,
1, 2, 4, 3,
1, 3, 4, 2,
1, 4, 3, 2), nrow = 6, byrow = TRUE)
colnames(R) <- LETTERS[1:4]

G <- group(as.rankings(R), 1:6)

mod <- pltree(G ~ 1, data = G)
preds <- predict(mod)

k <- kendallTau(R, preds)

# also applies to a single observation in the matrix
k <- kendallTau(R[1,], preds[1,])

---

kendallW

Kendall’s W (coefficient of concordance)

Description

Compute Kendall’s W, also known as coefficient of concordance among observed rankings and those predicted by the Plackett-Luce model.

Usage

kendallW(x, ...)

## Default S3 method:
kendallW(x, y, ...)

## S3 method for class 'pltree'
kendallW(x, newdata = NULL, ...)

Arguments

x an object of class pltree

... additional arguments passed to methods

y an object of class pltree

newdata data for predictions
Details
It is as wrapper of the function DescTools::KendallW, adapted to compute the Kendall’s W on the observed and predicted rankings.
Kendall’s W values range between 0 (no agreement) to 1 (full agreement)

Value
Kendall’s W (coefficient of concordance)

Author(s)
David Brown, Kauê de Sousa and Jacob van Etten

See Also
Other goodness-of-fit functions: kendallTau(), pseudoR2()

---

kenyachoice Kenyan farmers’ preferences for agricultural and livelihood practices

Description
Data from a preference experiment in Makueni County, Kenya. Twenty-six smallholder farmers ordered 9 different livelihood improvement practices by their personal preference. Full rankings were broken down into multiple pairwise rankings (kenyachoice["contests"]). To each respondent, a set of nine socio-economic variables is available (kenyachoice["predictors"]).

Usage
kenyachoice

Format
A list with two dataframes. kenyachoice["contests"] contains the pairwise rankings from farmers choices. kenyachoice["predictors"] contains the socioeconomic data for each farmer. Codes for contests are described:

B Opening a business
D Dry planting
G Collective crop marketing
J Finding off-farm job
M Machine tillage
O Renting out traction animals
R Mulching
T Terracing
Z Zai pits
Common bean on-farm trial in Nicaragua

Description

Data from decentralized on-farm trials of common bean (*Phaseolus vulgaris* L.) varieties in Nicaragua over five seasons between 2015 and 2016. Following the tricot approach, farmers were asked to test three varieties of common bean randomly assigned as incomplete blocks of three varieties (out of 10 varieties) and assess which of those three had the best and worst performance in nine traits (Vigor, Architecture, Resistance to Pests, Resistance to Diseases, Tolerance to Drought, Yield, Marketability, Taste, and Overall Appreciation).

Usage

nicabean

Format

A list with two data frames, nicabean[“trial”] contains the trial data:

- **id** the plot id
- **item** the variety name
- **trait** the trait for the given variety and plot id
- **rank** the rank for the given variety and trait, with 1 being higher and 3 the lowest

nicabean[“bean_covar”] contains the covariates associated with the data:

- **id** the plot id
- **adm0** the country name where trials were set
- **longitude** the longitude of the trial plot
- **latitude** the latitude of the trial plot
- **trial** the trial name as registered on ClimMob
- **variety_a** the variety assigned as label A in the incomplete block
- **variety_b** the variety assigned as label B in the incomplete block
- **variety_c** the variety assigned as label C in the incomplete block
- **planting_date** the planting date
- **gender** the farmer gender
- **age** the farmer age

Source


van Etten, J., et. al. (2016). Experimental Agriculture, 55, 275-296. doi:10.1017/S0014479716000739

node_labels

Get node labels and rules used in a party tree

Description

Returns the covariates used to split a recursive partitioning tree and the rules that were applied to build the tree.

Usage

node_labels(x)
node_rules(x)
top_items(x, top = 5)

## S3 method for class 'pltree'
plot(x, log = TRUE, ref = NULL, ci.level = 0.95, ...)

Arguments

- **x**: an object of class party
- **top**: an integer for the number of items to return
- **log**: logical, if TRUE log-worth coefficients are displayed instead of worth
- **ref**: optional, character for the reference item when log = TRUE
- **ci.level**: an integer for the confidence interval levels
- **...**: additional arguments passed to methods

Value

a vector with the node labels, a data.frame with node rules, a ggplot

Author(s)

Kauê de Sousa

Examples

```
library("PlackettLuce")
data("beans", package = "PlackettLuce")
G <- rank_tricot(data = beans,
    items = c(1:3),
    input = c(4:5),
    group = TRUE,
    additional.rank = beans[c(6:8)])
```
pld <- cbind(G, beans[,c("maxTN", "season", "lon")])
tree <- pltree(G ~ maxTN + season + lon, data = pld)
node_labels(tree)
node_rules(tree)
top_items(tree)
plot(tree)
plot(tree, log = TRUE)

---

**pseudoR2**

---

**Pseudo R-squared**

**Description**

Regression coefficient to evaluate goodness-of-fit in a given model when ordinary least squares (OLS) are not available. The algorithm computes estimates from the maximum likelihood through an iterative process. These estimates are called 'pseudo R-squared' because they look like 'R-squared' in the sense that they are on a similar scale (from 0 to 1), with higher values indicating better model fit.

**Usage**

pseudoR2(object, ...)

## Default S3 method:
pseudoR2(object, ...)

## S3 method for class 'pltree'
pseudoR2(object, newdata = NULL, ...)

## S3 method for class 'bttree'
pseudoR2(object, ...)

**Arguments**

- **object**
  a model object of class glm, gnm, lm, pltree or bttree
- **...**
  additional arguments affecting the R-squared produced
- **newdata**
  a data.set with testing data
Value

A data frame containing the pseudo R-squared coefficients:

- logLik: log-likelihood
- logLikNull: Null log-likelihood
- MaxLik: maximum likelihood pseudo R-squared
- Cragg Uhler: Cragg and Uhler's pseudo R-squared
- McFadden: McFadden pseudo R-squared

Author(s)

Kauê de Sousa and Jacob van Etten

References


See Also

Other goodness-of-fit functions: `kendallTau()`, `kendallW()`

Examples

data("airquality")

mod <- glm(Temp ~ Wind + Solar.R,
          data = airquality,
          family = poisson())
pseudoR2(mod)
Arguments

object an object of class rankings, grouped_rankings or paircomp
drop.null logical, an optional argument to remove null contests
disaggregate logical, if TRUE binaries are disaggregated by individual contests

Value

a matrix or PlackettLuce rank

Examples

```r
library("PlackettLuce")
R <- matrix(c(1, 2, 0, 0,
               4, 1, 0, 3,
               2, 1, 0, 3,
               1, 2, 0, 0,
               2, 1, 0, 0,
               1, 0, 0, 2), nrow = 6, byrow = TRUE)
colnames(R) <- c("apple", "banana", "orange", "pear")
# summary(PlackettLuce(R))
R <- pseudo_rank(R)
summary(PlackettLuce(R))
```

---

**Description**

Binary comparisons from a ranking object. Ties are not taken into account, then they are added as NA's.

**Usage**

`rank_binomial(object, drop.null = FALSE, disaggregate = FALSE)`
**Value**

A data.frame with binary rank of pairwise contests:

- player1: a factor with n levels for the first player in the contests
- player2: a factor with n levels (same as player1) for the second player in the contests
- win1: number of times player1 wins against player2
- win2: number of times player2 wins against player1

**Author(s)**

Kauê de Sousa

**References**


**See Also**

Other rank functions: `rank_numeric()`, `rank_paircomp()`, `rank_tricot()`

**Examples**

```r
library("PlackettLuce")

R <- matrix(c(1, 2, 0, 0,
              4, 1, 2, 3,
              2, 4, 3, 1,
              1, 2, 3, 0,
              2, 1, 1, 0,
              1, 0, 3, 2), nrow = 6, byrow = TRUE)
colnames(R) <- c("apple", "banana", "orange", "pear")

R <- as.rankings(R)
rank_binomial(R)
rank_binomial(R, disaggregate = TRUE)
```

---

**Description**

Group and coerce numeric values into Plackett-Luce rankings.
rank_numeric

Usage

rank_numeric(
  data,
  items,
  input,
  id = NULL,
  group = FALSE,
  ascending = FALSE,
  ...
)

Arguments

data a data.frame with columns specified by items and input values
items a character or numerical vector for indexing the column(s) containing the item
       names in data
input a character or numerical vector for indexing the column(s) containing the values
       in data to be ranked
id an index of data indicating the ids for "long" data
group logical, if TRUE return an object of class "grouped_rankings"
ascending logical, only for floating point numbers, to compute rankings from lower to
       higher values
... additional arguments passed to methods

Value

a PlackettLuce "rankings" object, which is a matrix of dense rankings

Author(s)

Kauê de Sousa

See Also

rankings

Other rank functions: `rank_binomial()`, `rank_paircomp()`, `rank_tricot()`

Examples

# A matrix with 10 rankings of 5 items (A, B, C, D, E)
# with numeric values as "rank"
set.seed(123)
df <- cbind(id = rep(1:10, each = 5),
            items = rep(LETTERS[1:5], times = 10),
            input = runif(50, 1, 3))

# return an object of class 'rankings'
R <- rank_numeric(df,
Pairwise comparison rankings

Description

Pairwise comparisons from a ranking object. Ties are not taken into account, then they are added as NA's.

Usage

rank_paircomp(object)

Arguments

object an object of class "rankings" or "grouped_rankings" which is a matrix of dense rankings.

Value

an object of class "paircomp" which is a matrix of pairwise comparisons

Author(s)

Kauê de Sousa and Jacob van Etten

References

**See Also**

`rankings`, `paircomp`

Other rank functions: `rank_binomial()`, `rank_numeric()`, `rank_tricot()`

**Examples**

```r
colnames(R) <- c("apple", "banana", "orange", "pear")
R <- as.rankings(R)

PC <- rank_paircomp(R)
```

```r#
# using breadwheat data
data("breadwheat", package = "gosset")
# convert the tricot rankings from breadwheat data
# into a object of class 'rankings' from PlackettLuce
R <- rank_tricot(breadwheat,
                 items = c("variety_a", "variety_b", "variety_c"),
                 input = c("overall_best", "overall_worst"))
PC <- rank_paircomp(R)
```

**Description**

Create an object of class "rankings" from tricot data. Tricot stands for "triadic comparison of technologies". It is a methodology to carry out large agronomic field experiments allowing the comparison between many alternative technologies, in many different environments. Each participant evaluates a set of three randomised technologies from a larger set. A comparison with a local item can be added as additional rankings with the argument `additional.rank`.

**Usage**

```r
rank_tricot(data, items, input, group = FALSE, additional.rank = NULL, ...)
```
Arguments

data  a data.frame with columns specified by items and input values
items a character or numerical vector for indexing the column(s) containing the item
names in data
input a character or numerical vector for indexing the column(s) containing the values
in data to be ranked

Details

full.output: logical, to return a list with a "rankings", a "grouped_rankings" and the ordered items

Value

a PlackettLuce "rankings" object, which is a matrix of dense rankings

Author(s)

Kauê de Sousa and Jacob van Etten, with ideas from Heather Turner

References

van Etten J., et al. (2016). Experimental Agriculture, 55(S1), 275–296. doi:10.1017/S0014479716000739

See Also

rankings, breadwheat

Other rank functions: rank_binomial(), rank_numeric(), rank_paircomp()

Examples

# using breadwheat data
data("breadwheat", package = "gosset")

# convert the tricot rankings from breadwheat data
# into a object of class 'rankings' from PlackettLuce
R <- rank_tricot(data = breadwheat, 
items = c("variety_a","variety_b","variety_c"), 
input = c("overall_best","overall_worst"))

# beans data where each observer compares 3 varieties randomly distributed
# from a list of 11 and additionally compares these 3 varieties
# with their local variety
library("PlackettLuce")
data("beans", package = "PlackettLuce")

# first build rankings with only tricot items
# and return an object of class 'rankings'
R <- rank_tricot(data = beans,
                 items = c(1:3),
                 input = c(4:5))
head(R)

# pass the comparison with local item as an additional rankings, then
# each of the 3 varieties are compared separately with the local item
# and return an object of class grouped_rankings
G <- rank_tricot(data = beans,
                 items = c(1:3),
                 input = c(4:5),
                 group = TRUE,
                 additional.rank = beans[c(6:8)])
head(G)

---

**regret**

*Regret-based values for risk assessment*

**Description**

Regret is an important heuristic in the behavioural sciences. Minimizing worst regret (the loss under the worst possible outcome) is a criterion that takes a conservative approach to risk analysis in diversification strategies.

**Usage**

regret(object, ..., bootstrap = TRUE, normalize = TRUE)

## Default S3 method:
regret(object, ..., values, items, group, bootstrap = TRUE, normalize = TRUE)

## S3 method for class 'pltree'
regret(object, bootstrap = TRUE, normalize = TRUE, ...)

## S3 method for class 'list'
regret(object, bootstrap = TRUE, normalize = TRUE, ...)

------------------

---

**regret**

*Regret-based values for risk assessment*

**Description**

Regret is an important heuristic in the behavioural sciences. Minimizing worst regret (the loss under the worst possible outcome) is a criterion that takes a conservative approach to risk analysis in diversification strategies.

**Usage**

regret(object, ..., bootstrap = TRUE, normalize = TRUE)

## Default S3 method:
regret(object, ..., values, items, group, bootstrap = TRUE, normalize = TRUE)

## S3 method for class 'pltree'
regret(object, bootstrap = TRUE, normalize = TRUE, ...)

## S3 method for class 'list'
regret(object, bootstrap = TRUE, normalize = TRUE, ...)
Arguments

- object: a data.frame, an object of class pltree, or a list with PlackettLuce models
- ...: further arguments passed to methods
- bootstrap: logical, to run a bayes bootstrap on object
- normalize: logical, to normalize values to sum to 1
- values: an index in object with the values to compute regret
- items: an index in object for the different items
- group: an index in object for the different scenarios

Value

A data frame with regret estimates
- items: the item names
- worth: the worth parameters
- regret: the squared regret
- worst_regret: the worst regret

Author(s)

Jacob van Etten and Kauê de Sousa

References


Examples

```r
# Case 1 ####
library("PlackettLuce")
data("breadwheat", package = "gosset")

# convert the tricot rankings from breadwheat data
# into a object of class 'grouped_rankings'
G <- rank_tricot(breadwheat, 
                   items = c("variety_a","variety_b","variety_c"),
                   input = c("overall_best","overall_worst"),
                   group = TRUE)

# combine grouped rankings with temperature indices
mydata <- cbind(G, breadwheat[c("lon","lat")])

# fit a pltree model using geographic data
```
mod <- pltree(G ~ ., data = mydata)

regret(mod)

# Case 2 ####
# list of PlackettLuce models
R <- matrix(c(1, 2, 3, 0,
              4, 1, 2, 3,
              2, 1, 3, 4,
              1, 2, 3, 0,
              2, 1, 3, 0,
              1, 0, 3, 2), nrow = 6, byrow = TRUE)
colnames(R) <- c("apple", "banana", "orange", "pear")
mod1 <- PlackettLuce(R)

R2 <- matrix(c(1, 2, 0, 3,
               2, 1, 0, 3,
               2, 1, 0, 3,
               1, 2, 0, 3,
               2, 1, 0, 3,
               1, 3, 4, 2), nrow = 6, byrow = TRUE)
colnames(R2) <- c("apple", "banana", "orange", "pear")
mod2 <- PlackettLuce(R2)

mod <- list(mod1, mod2)
regret(mod, n1 = 500)

reliability

**Probability of outperforming a check**

**Description**

Measures the precision of estimated values, and the potential response to selection on those estimated values compared to a check

**Usage**

reliability(x, ...)

## Default S3 method:
reliability(x, y = NULL, ...)

## S3 method for class 'PlackettLuce'
reliability(x, ref, ...)

## S3 method for class 'pltree'
reliability(x, ref, ...)
Arguments

- **x** a numeric vector, or an object of class `PlackettLuce` or `pltree`
- ... additional arguments passed to methods
- **y** numeric, the reference value
- **ref** a character or integer for indexing the element containing reference values in `x`

Value

the reliability based on the worth parameters

Author(s)

Kauê de Sousa, David Brown, Jacob van Etten

References


Examples

# Case 1. vector example
x <- c(9.5, 12, 12.3, 17)
y <- 11.2
reliability(x, y)

# Case 2. PlackettLuce model
library("PlackettLuce")
R <- matrix(c(1, 2, 4, 3,
        4, 1, 2, 3,
        2, 3, 1, 4,
        4, 2, 3, 1,
        2, 1, 4, 3,
        1, 4, 3, 2), nrow = 6, byrow = TRUE)
colnames(R) <- c("apple", "banana", "orange", "pear")

mod <- PlackettLuce(R)

reliability(mod, ref = "orange")

# Case 3. PlackettLuce tree

data("beans", package = "PlackettLuce")
G <- rank_tricot(data = beans,
items = c(1:3),
input = c(4:5),
group = TRUE,
additional.rank = beans[c(6:8)])

pld <- cbind(G, beans[,c("maxTN", "season", "lon")])

tree <- pltree(G ~ maxTN + season + lon, data = pld)

reliability(tree, ref = "Local")

---

rowbind

Combine R objects by rows

Description

Combine R objects when number and names of columns do not match

Usage

rowbind(x, ...)

## Default S3 method:
rowbind(x, y, ...)

## S3 method for class 'list'
rowbind(x, ...)

Arguments

x a R object, typically a data.frame, matrix or list
...
additional arguments passed to methods
y a matrix, a data.frame (or any other object that can be coerced to data.frame)

Value

a data.frame with the combined data

Examples

# two data frames
rowbind(airquality, mtcars)

# a list of data frames
l <- list(airquality, mtcars)
rowbind(l)
Description

Produces plots to highlight worth coefficients of items in a party tree of a list of PlackettLuce models

Usage

worth_map(object, ...)

## Default S3 method:
worth_map(object, ...)

## S3 method for class 'list'
worth_map(object, labels, ...)

worth_bar(object, ...)

Arguments

object an object of class party or an object of class PlackettLuce or a list objects of class PlackettLuce

... additional arguments passed to methods

labels a vector with the name of models in object

Examples

library("psychotree")
library("PlackettLuce")
data("Topmodel2007", package = "psychotree")
R <- as.grouped_rankings(Topmodel2007$preference)

tm_tree <- pltree(R ~ ., data = Topmodel2007[, -1],
                  minsize = 5,
                  npseudo = 0)

worth_map(tm_tree)

##########################################
# Ranking of preference on four fruits
# based on traits taste, texture,
# price and storability
# taste
R1 <- matrix(c(1, 2, 3, 4,
               4, 1, 3, 2,
4, 1, 2, 3,
1, 2, 0, 3), nrow = 4, byrow = TRUE)
colnames(R1) <- c("apple", "banana", "orange", "pear")
mod1 <- PlackettLuce(R1)

# texture
R2 <- matrix(c(1, 4, 2, 3,
1, 4, 3, 2,
1, 4, 2, 3,
1, 4, 2, 3), nrow = 4, byrow = TRUE)
colnames(R2) <- c("apple", "banana", "orange", "pear")
mod2 <- PlackettLuce(R2)

# price
R3 <- matrix(c(2, 4, 3, 1,
4, 1, 2, 3,
3, 4, 2, 1,
4, 3, 1, 2), nrow = 4, byrow = TRUE)
colnames(R3) <- c("apple", "banana", "orange", "pear")
mod3 <- PlackettLuce(R3)

# storability
R4 <- matrix(c(1, 4, 3, 2,
3, 4, 1, 2,
1, 3, 2, 4,
2, 3, 4, 1), nrow = 4, byrow = TRUE)
colnames(R4) <- c("apple", "banana", "orange", "pear")
mod4 <- PlackettLuce(R4)

# models in a list
mods <- list(mod1, mod2, mod3, mod4)

# name for each model
labels <- c("Taste", "Texture", "Price", "Storability")

worth_map(mods, labels)

# plot only one model as bar
worth_bar(mod1)
Index

* dataset
  breadwheat, 3
  kenyachoice, 13
  nicabeans, 14
* goodness-of-fit functions
  kendallTau, 10
  kendallW, 12
  pseudoR2, 16
* model selection functions
  btpermute, 5
  crossvalidation, 7
* rank functions
  rank_binomial, 18
  rank_numeric, 19
  rank_paircomp, 21
  rank_tricot, 22
* utility functions
  rowbind, 28

AIC.bttree (crossvalidation), 7
AIC.gnm (crossvalidation), 7
akaike_weights, 2

breadwheat, 3, 23
BTm, 6
btpermute, 5, 9
btree, 9

compare, 6
cor, 11
crossvalidation, 6, 7

deviance.bttree (crossvalidation), 7
deviance.gnm (crossvalidation), 7
deviance.pltree (crossvalidation), 7

gnm, 9
gosset, 10
gosset-package (gosset), 10

kendallTau, 10, 13, 17
kendallW, 11, 12, 17
kenyachoice, 13

nicabeans, 14

node_labels, 15
node_rules (node_labels), 15

paircomp, 22
plot.pltree (node_labels), 15
pltree, 9
pseudo_rank, 17
pseudoR2, 11, 13, 16

rank_binomial, 6, 18, 20, 22, 23
rank_numeric, 19, 19, 22, 23
rank_paircomp, 19, 20, 21, 23
rank_tricot, 19, 20, 22, 22
rankings, 20, 22, 23
regret, 24
reliability, 26
rowbind, 28

top_items (node_labels), 15

worth_bar (worth_map), 29
worth_map, 29