Package 'rapidsplit'

June 13, 2024

Type Package

Title Fast split-half reliability algorithm

Version 0.0.1.0

Date 2021-11-24

Author Sercan Kahveci

Maintainer Sercan Kahveci <sercan.kahveci@plus.ac.at>

Description Fast and flexible split-half reliability algorithm.

License GPL (>= 2)

Imports Rcpp (>= 1.0.5), AATtools

LinkingTo Rcpp

RoxygenNote 7.3.1

Encoding UTF-8

Contents

`applyItersplits` *applyItersplits*

Description

generate splits for splithalf

Usage

```
applyItersplits(iters, splits, replace = FALSE)
```

Arguments

iters	number of iterations
splits	list of vectors of row numbers
replace	Sample without (default) or with replacement

---

bootstrapWeights *Bootstrap Weights*

Description

Create a matrix of bootstrap samples expressed as frequency weights

Usage

```
bootstrapWeights(size, times)
```

Arguments

size	Number of values to bootstrap
times	Number of bootstraps

Value

A matrix with bootstrap samples expressed as frequency weights. Each column represents a single bootstrap iteration and each row represents a case.

Examples

```
myweights<-bootstrapWeights(size=50, times=100)
rapidsplit::mediansByWeight(x=rnorm(50),weights=myweights)
```

---

corByColumns *Correlate each column of 1 matrix with the same column in another matrix*

Description

Correlate each column of 1 matrix with the same column in another matrix

Usage

```
corByColumns(x, y)

corByColumns_mask(x, y, mask)
```

Arguments

x, y	Matrices whose values to correlate by column
mask	Logical matrix marking which data points to include

Value

A numeric vector of correlations per column

---

cormean *Compute a minimally biased average of correlation values*

Description

This function computes a minimally biased average of correlation values. This is needed because simple averaging of correlations is negatively biased, and the often used z-transformation method of averaging correlations is positively biased. The algorithm was developed by Olkin & Pratt (1958).

Usage

```
cormean(
  r,
  n,
  weights = c("none", "n", "df"),
  type = c("OP5", "OP2", "OPK"),
  na.rm = F
)
```

Arguments

r	a vector containing correlation values
n	a single value or vector containing sample sizes
weights	Character. How should the correlations be weighted? none leads to no weighting, n weights by sample size, df weights by sample size minus one.
type	Character. Determines which averaging algorithm to use, with "OP5" being the most accurate.
na.rm	Logical. Should missing values be removed?

Value

An average correlation.

References

Olkin, I., & Pratt, J. (1958). Unbiased estimation of certain correlation coefficients. The Annals of Mathematical Statistics, 29. https://doi.org/10.1214/aoms/1177706717

Shieh, G. (2010). Estimation of the simple correlation coefficient. Behavior Research Methods, 42(4), 906-917. https://doi.org/10.3758/BRM.42.4.906

Examples

```
cormean(c(0,.3,.5),c(30,30,60))
```

`correlation-tools` *Correlation tools*

Description

Helper functions to compute important statistics from correlation coefficients.

Usage

```
r2z(r)

z2r(z)

r2t(r, n)

t2r(t, n)

r2p(r, n)

rconfint(r, n, alpha = 0.05)

compcorr(r1, r2, n1, n2)

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

Arguments

`r`, `r1`, `r2`	a correlation value
`z`	a Z-score
`n`, `n1`, `n2`	sample sizes
`t`	a t-score
`alpha`	the significance level to use
`x`	a `compcorr` object to print
`...`	Ignored

Functions

- `r2z()`: converts correlation coefficients to z-scores
- `z2r()`: converts z-scores to correlation coefficients
- `r2t()`: Converts correlation coefficients to t-scores
- `t2r()`: Converts t-scores to correlation coefficients
- `r2p()`: Computes the two-sided p-value for a given correlation
- `rconfint()`: Computes confidence intervals for a given correlation coefficient
- `compcorr()`: computes the significance of the difference between two correlation coefficients
- `print(compcorr)`: computes the significance of the difference between two correlation coefficients

See Also

cormean

Examples

```
z <- r2z(.5)
r <- z2r(z)
t<-r2t(r,30)
r2p(r,30)
print(rconfint(r,30))
print(compcorr(.5,.7,20,20))
```

---

excludeOutliersByMask *Exclude SD-based outliers*

Description

Update a mask matrix based on outlyingness

Usage

```
excludeOutliersByMask(x, mask, sdlim = 3)

ExcludeSDOutliers_nomask(x, sdlim = 3)
```

Arguments

x	Matrix in which to mark SD-based outliers by column (with FALSE)
mask	a logical matrix determining which data points to include and which not to
sdlim	Standard deviation limit to apply; values beyond are classified as outliers and masked

Value

An updated mask

---

rapidsplit *rapidsplit*

Description

A very fast algorithm for permutated split-half reliability

Usage

```
rapidsplit(
  ds,
  subjvar,
  diffvars = NULL,
  stratvars = NULL,
  rtvar,
  iters,
  agg = c("means", "medians"),
  standardize = F,
  include.scores = T
)

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

## S3 method for class 'rapidsplit'
plot(x, type = c("average", "minimum", "maximum", "random", "all"), ...)
```

Arguments

ds	Dataset, a data.frame
subjvar	Subject ID variable name, a character
diffvars	Variables that determine which conditions need to be subtracted from each other, a character
stratvars	Additional variables that the splits should be stratified by, if possible; a character
rtvar	Reaction time variable name, a character
iters	Number of split-halves to average, an integer
agg	The function by which to aggregate the RTs; can be "means" or "medians"
standardize	Whether to divide by scores by the subject's SD; a logical
include.scores	Include all individual split-half scores?
x	rapidsplit object to print or plot
...	Ignored
type	Character argument indicating what should be plotted. By default, this plots the random split whose correlation is closest to the average. However, this can also plot the random split with the "minimum" or "maximum" split-half correlation, or any "random" split. "all" splits can also be plotted together in one figure.

Value

A list containing * the averaged reliability * a vector with the reliability of each iteration

Examples

```
print("no example yet")

ds<-AATtools::aat_simulate()
frel<-rapidsplit(ds=ds,subjvar="subj",diffvars=c("is_pull","is_target"),
                 rtvar="rt",iters=1000,agg="mean")
print(frel)
plot(frel,type="all")
```

`SpearmanBrown` *Spearman-Brown correction Perform a Spearman-Brown correction on the provided correlation score.*

Description

Spearman-Brown correction Perform a Spearman-Brown correction on the provided correlation score.

Usage

```
SpearmanBrown(corr, ntests = 2)
```

Arguments

`corr`	To-be-corrected correlation coefficient
`ntests`	An integer indicating how many times larger the full test is, for which the corrected correlation coefficient is being computed.

Details

When `ntests=2`, the formula will compute what the correlation coefficient would be if the test were twice as long.

Examples

```
SpearmanBrown(.5)
```

`stratifiedItersplits` *stratifiedItersplits*

Description

generate stratified splits for a single participant

Usage

```
stratifiedItersplits(itercount, groupsizes)
```

Arguments

`itercount`	number of iterations
`groupsizes`	vector of number of RTs per group to stratify

Details

This first equally splits what can be equally split within groups. Then it randomly splits all the leftovers.

Value

A matrix with zeroes and ones

Index