Package ‘RankAggregator’

October 12, 2022

Title Aggregation of (Partial) Ordinal Rankings

Version 0.0.1

Description Easily compute an aggregate ranking (also called a median ranking or a consensus ranking) according to the axiomatic approach presented by Cook et al. (2007). This approach minimises the number of violations between all candidate consensus rankings and all input (partial) rankings, and draws on a branch and bound algorithm and a heuristic algorithm to drastically improve speed. The package also provides an option to bootstrap a consensus ranking based on resampling input rankings (with replacement). Input rankings can be either incomplete (partial) or complete. Reference: Cook, W.D., Golany, B., Penn, M. and Raviv, T. (2007) <doi:10.1016/j.cor.2005.05.030>.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Depends R (>= 2.10)

Suggests knitr

NeedsCompilation no

Author Jay Burns [aut, cre],
Adam Butler [aut]

Maintainer Jay Burns <jay.burns@sruc.ac.uk>

Repository CRAN

Date/Publication 2020-08-31 09:20:10 UTC

R topics documented:

consensusRanking 2
consensusRankingBoot 3
cook_example 4
evaluationMatrix 5

extendRanking 5
lowerBound 6
RankAggregator 7
upperBound 7

Index **9**

`consensusRanking` *Rank aggregation of partial rankings*

Description

This function is the core function for the `RankAggregator` package. This function uses a branch and bound algorithm, described by Cook et al. (2007), to return a best consensus (or median) ranking for a set of (partial) rankings.

Usage

```
consensusRanking(x)
```

Arguments

`x`	a `data.frame` containing columns titled `Reviewer`, `Item`, `Ranking`. On data structure, `Reviewer` and `Item` must be character, and `Ranking` must be numeric. Each row of `x` identifes the rank position that a single `Reviewer` awarded a single `Item`

Value

A `data.frame` is returned, with two columns: `Item` and `Rank`, where `Item` is a `Factor` containing all unique `Item`'s from the input `data.frame` `x`, and where `Rank` is the estimated (numeric) rank position based on the branch and bound rank aggregation procedure.

See Also

This function calls internal functions `evaluationMatrix`, `extendRanking`, `lowerBound`, and `upperBound`

Examples

```
consensusRanking(cook_example)
```

`consensusRankingBoot` *Rank aggregation of partial rankings with optonal bootstrapping*

Description

This funciton calls `RankAggregator`::`consensusRanking` to return a best consensus (or median) ranking for a set of (partial) rankings.

This function also provides an optional bootstrap resampling procedure to give user-defined confidence intervals and average rank positions with the consensus ranking.

Usage

```
consensusRankingBoot(
  x,
  bootstrap,
  nboot = 10000,
  conf.int = 0.95,
  prog.upd = TRUE
)
```

Arguments

`x`	a `data.frame` containing columns titled `Reviewer`, `Item`, `Ranking`. On data structure, `Reviewer` and `Item` must be character, and `Ranking` must be numeric. Each row of `x` identifes the rank position that a single `Reviewer` awarded a single `Item`
`bootstrap`	a logical value indicating whether to bootstrap the rank aggregation procedure.
`nboot`	a numeric value for bootstrap replicates. Default value is `10000`.
`conf.int`	a numeric value >0 and <1. Default value is `0.95`, which sets confidence interval at 95% level.
`prog.upd`	a logical value indicating whether the user wants progress updates on the bootstrap procedure.

Value

If `bootstrap` is `FALSE`, a `data.frame` is returned, with two columns: `Item` and `Rank.est`, where `Item` is a `Factor` containing all unique `Item`'s from the input `data.frame` `x`, and where `Rank.est` is the estimated (numeric) rank position based on the `consensusRanking()` rank aggregation procedure.#'

If `bootstrap` is `TRUE`, a `list` is returned, with two elements:

- `$summaryTable` is a `data.frame` with six columns: `Item` `Rank.est`, `Rank.cilo`, `Rank.cihi`, `Rank.median`, `Rank.mean`. Where `Item` and `Rank.est` are as described above, `Rank.cilo` and `Rank.cihi` are the estimates for the low and high confidence intervals, respectively. `Rank.median` and `Rank.mean` both describe the average rank positions.

- `$bootstrapData` is an array containing estimated (numeric) rank positions based on the `consensusRanking()` rank aggregation procedure with resampled data. `NA` denotes estimated rankings that were discarded due to not containing all `Items`.

References

Cook, W.D., Golany, B., Penn, M. and Raviv, T., 2007. Creating a consensus ranking of proposals from reviewers partial ordinal rankings. Computers & Operations Research, 34, pp.954-965.

Marshall, E.C., Sanderson, C., Spiegelhalter, D.J. and McKee, M., 1998. Reliability of league tables of in vitro fertilisation clinics: retrospective analysis of live birth ratesCommentary: How robust are rankings? The implications of confidence intervals. Bmj, 316, pp.1701-1705.

See Also

Calls the internal function `consensusRanking`, which calls the other internal functions `evaluationMatrix`, `consensusRanking`, `extendRanking`, `lowerBound`, `upperBound`

`cook_example` *Example data: partial rankings*

Description

A dataset containing 5 partial rankings of 6 items. This is the example used by Cook et al (2007).

Usage

```
cook_example
```

Format

A data frame of 20 rows and 3 columns

Item Character values giving one of 6 items

Reviewer Character values giving one of 5 reviewers

Ranking Numeric values giving a rank position

Source

Cook, W.D., Golany, B., Penn, M. and Raviv, T., 2007. Creating a consensus ranking of proposals from reviewers' partial ordinal rankings. Computers & Operations Research, 34, pp.954-965.

`evaluationMatrix` *Evaluation matrix*

Description

This function is called by `RankAggregator`::`consensusRanking`. For each pair of `Items`, whenever both `Items` are ranked by the same `Reviewer`, this function sums the occurances when each of the two `Items` is preferred to the other.

Usage

```
evaluationMatrix(x)
```

Arguments

`x`	a `data.frame` containing columns titled `Reviewer`, `Item`, `Ranking`. On data structure, `Reviewer` and `Item` must be character, and `Ranking` must be numeric. Each row of `x` identifes the rank position that a single `Reviewer` awarded a single `Item`

Value

An `m x n` pairwise `matrix` giving the number of times `Item[m]` is preferred to (i.e. receives a ranking value lower than) `Item[n]` across all `Reviewer Rankings`

Examples

```
evaluationMatrix(cook_example)
```

`extendRanking` *Fully extend a partial ranking*

Description

This function is called by `RankAggregator`::`consensusRanking`. The heuristic procedure orders unranked `Items` according the proportion of times an item was preferred in all pairwise comparisons with other unranked `Items`.

Usage

```
extendRanking(umat, node)
```

Arguments

`umat`	a `matrix`, which is either the output of `evaluationMatrix`, or a subset of the output of `evaluationMatrix`.
`node`	a `list` of elements, containing information about a node in the branch and bound search space. The relevant elements here are `$partial.ranking`, `$included`, and `$prl`. Where, `$partial.ranking` is a vector of rank positions for each `Item` in `umat` that is ranked so far; partial rankings may contain some - or all - `NA` values. `$included` is a logical vector denoting if an `Item` in `umat` is ranked in `$partial.ranking`. And `$prl` is a numeric value denoting how many of the `Item`s in `umat` are ranked in `$partial.ranking`.

Value

A vector of rank positions.

lowerBound *Lower bound value*

Description

This function is called by `RankAggregator`::`consensusRanking`. The lower bound is the absolute lowest value a complete candidate ranking could attain. Note, this value is not always achievable, so may differ from the value returned by `upperBound`.

```
For each pair of \code{Item}s, there are three possible calculations, depending
  on whether both \code{Item}s are in the \code{partial.ranking}, one is in
  and the other is out the \code{partial.ranking}, or both are not in
  the \code{partial.ranking}.
```

Usage

```
lowerBound(umat, partial.ranking)
```

Arguments

`umat`	a `matrix`, which is either the output of `evaluationMatrix`, or a subset of the output of `evaluationMatrix`.
`partial.ranking`	a vector of rank positions for each `Item` in `umat` that is ranked so far; partial rankings may contain some - or all - `NA` values.

Value

A numeric value for the lower bound of a `partial.ranking`

`RankAggregator` *RankAggregator*

Description

This package provides a set of functions to easily compute an aggregate ranking (also called a median ranking or a compromise ranking) according to the axiomatic approach presented by Cook et al. (2007). This approach minimises the number of violations between all candidate consensus rankings and all input (partial) rankings, and draws on a branch and bound algorithm, and a heuristic algorithm to drastically improve speed. Input rankings can be either incomplete (partial) or complete.

The package also provides an option to bootstrap resulting consensus ranking based on resampling input rankings (with replacement). This approach was inspired by Marshall et al. (1998).

Author(s)

Jay Burns <`jay.burns@sruc.ac.uk`>, Adam Butler <`adam.butler@bioss.ac.uk`>

References

Cook, W.D., Golany, B., Penn, M. and Raviv, T., 2007. Creating a consensus ranking of proposals from reviewers partial ordinal rankings. Computers & Operations Research, 34, pp.954-965.

Marshall, E.C., Sanderson, C., Spiegelhalter, D.J. and McKee, M., 1998. Reliability of league tables of in vitro fertilisation clinics: retrospective analysis of live birth ratesCommentary: How robust are rankings? The implications of confidence intervals. Bmj, 316, pp.1701-1705.

`upperBound` *Upper bound value*

Description

This function is called by `RankAggregator`::`consensusRanking`. The upper bound value is the value used by the branch and bound algorithm in determining whether or not to replace the current incumbent solution.

Usage

```
upperBound(ccr, umat)
```

Arguments

`ccr`	a vector of rank positions that is a candidate complete ranking
`umat`	a `matrix`, which is either the output of `evaluationMatrix`, or a subset of the output of `evaluationMatrix`.

Value

A numeric value for the upper bound of a candidate complete ranking

Index

∗ **datasets**
    cook_example, 4

consensusRanking, 2, *3–7*
consensusRankingBoot, 3
cook_example, 4

evaluationMatrix, *2*, *4*, 5, *6, 7*
extendRanking, *2*, *4*, 5

lowerBound, *2*, *4*, 6

RankAggregator, *2, 3*, *5–7*, 7
RankAggregator-package
        *(*RankAggregator*)*, 7

upperBound, *2*, *4*, *6*, 7