Package 'rafsi'

September 25, 2024

Type Package

Title Ranking of Alternatives with the RAFSI Method

Version 0.0.2

Description Ranking of Alternatives through Functional mapping of criterion sub-intervals into a Single Interval Method is designed to perform multi-criteria decision-making (MCDM), developed by Mališa Žižovic in 2020 (<doi:10.3390/math8061015>).
It calculates the final sorted rankings based on a decision matrix where rows represent alternatives and columns represent criteria. The method uses:
- A numeric vector of weights for each criterion (the sum of weights must be 1).
- A numeric vector of ideal values for each criterion.
- A numeric vector of anti-ideal values for each criterion.
- Numeric values representing the extent to which the ideal value is preferred over the anti-ideal value,
and the extent to which the anti-ideal value is considered worse.
The function standardizes the decision matrix, normalizes the data, applies weights, and returns the
final sorted rankings.

Language en-US

Depends R (>= 4.2.0)

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.1

Suggests knitr, rmarkdown, spelling, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Mateus Vanzetta [aut, cre],
Marcos Santos [ctb] (<https://orcid.org/0000-0003-1533-5535>)

Maintainer Mateus Vanzetta <mateusvanzetta@id.uff.br>

Repository CRAN

Date/Publication 2024-09-25 08:10:02 UTC

Contents

`rafsi_method` *Rank Reversal Problem Using a New Multi-Attribute Model - RAFSI Method for Multi-Criteria Decision Making*

Description

This function implements the (Ranking of Alternatives Through Functional Mapping of Criterion Sub-Intervals Into a Single Interval) RAFSI method, Rank Reversal Problem Using a New Multi-Attribute Model. More information about the method can be found at https://doi.org/10.3390/math8061015. More information about the implementation at https://github.com/mateusvanzetta/rafsi. used for multi-criteria decision-making problems. It calculates the standardized decision matrix, normalizes the data, applies weights, and returns the final sorted rankings.

Usage

```
rafsi_method(
  dataset,
  weights,
  criterion_type,
  ideal = numeric(),
  anti_ideal = numeric(),
  n_i,
  n_k
)
```

Arguments

`dataset`	A matrix of criterion values where rows represent alternatives and columns represent criteria.
`weights`	A numeric vector representing the weights of each criterion. The sum of the weights must be 1.
`criterion_type`	A character vector indicating the type of each criterion ('max' for maximization, 'min' for minimization).
`ideal`	A numeric vector representing the ideal values for each criterion.
`anti_ideal`	A numeric vector representing the anti-ideal values for each criterion.
`n_i`	A numeric value representing the ratio that shows to what extent the anti-ideal value is worse than the value.
`n_k`	A numeric value representing the ratio that shows to what extent the ideal value is preferred over the anti-ideal value.

Value

A list containing:

Standardized_matrix The matrix after applying the RAFSI transformation, which standardizes the data according to the ideal and anti-ideal values.

Normalized_matrix The matrix after normalizing the standardized data, adjusted according to the criteria weights.

Ranking A data frame showing the final ranking of the alternatives. The alternatives are sorted in descending order of preference.

#'

Examples

```
# Define the dataset
dataset <- matrix(c(
  180, 165, 160, 170, 185, 167,   # Criterion 1
  10.5, 9.2, 8.8, 9.5, 10, 8.9,   # Criterion 2
  15.5, 16.5, 14, 16, 14.5, 15.1, # Criterion 3
  160, 131, 125, 135, 143, 140,   # Criterion 4
  3.7, 5, 4.5, 3.4, 4.3, 4.1      # Criterion 5
), nrow = 6, ncol = 5)

# Set the names of alternatives
rownames(dataset) <- c("A1", "A2", "A3", "A4", "A5", "A6")

# Define the weights and criterion types
weights <- c(0.35, 0.25, 0.15, 0.15, 0.10)
criterion_type <- c('max', 'max', 'min', 'min', 'max')

# Specify ideal and anti-ideal values
ideal <- c(200, 12, 10, 100, 8)
anti_ideal <- c(120, 6, 20, 200, 2)

# Set n_i and n_k values
n_i <- 1
n_k <- 6

# Apply the RAFSI method
result <- rafsi_method(dataset, weights, criterion_type, ideal, anti_ideal, n_i, n_k)

# View the result
print(result)
```

Index