Package 'ahp'

January 6, 2017

Type Package

Title Analytic Hierarchy Process

Version 0.2.11

Date 2017-01-05

Author Christoph Glur

Maintainer Christoph Glur <christoph.glur@ipub.com>

VignetteBuilder knitr

Imports utils, data.tree, yaml, formattable, DiagrammeR

Suggests shiny, shinyAce, shinythemes, shinyjs, testthat, knitr, rmarkdown

Description Model and analyse complex decision making problems using the Analytic Hierarchy Process (AHP) by Thomas Saaty.

License GPL (>= 2)

URL http://github.com/gluc/ahp

BugReports http://github.com/gluc/ahp/issues

Depends R (>= 3.2.0)

RoxygenNote 5.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2017-01-06 14:44:18

R topics documented:

---

ahp *ahp AHP (Analytic Hierarchy Process) Modeling for R*

Description

AHP (Analytic Hierarchy Process) is a decision making framework developed by Thomas Saaty. This package lets you model and analyse complex decision making problems according to the AHP framework.

Details

The basic workflow with this package is: 1. specify your ahp problem in an ahp file. See `vignette("file-format", packag` for details 2. load ahp file, using `Load` 3. calculate model, using `Calculate` 4. visualize the ahp model using `Visualize` 5. output model analysis, either using `Analyze` or using `AnalyzeTable`

For more information, see the package vignette using `vignette("examples", package = "ahp")`. To learn the details about the ahp file format, type `vignette("file-format", package = "ahp")`

References

Saaty, T. L. (2001). Decision Making for Leaders: The Analytic Hierarchy Process for Decisions in a Complex World, New Edition 2001 (3 Revised). Pittsburgh, PA: RWS Publications, ISBN 978-0962031786.

Examples

```
library(ahp)
#list example files provided by the package
list.files(system.file("extdata", package="ahp"))
#load a specific example
ahpFile <- system.file("extdata", "car.ahp", package="ahp")
carAhp <- Load(ahpFile)
Calculate(carAhp)
Analyze(carAhp)
AnalyzeTable(carAhp)

#the vacation.ahp file provides an example with multiple decision makers
ahpFile <- system.file("extdata", "vacation.ahp", package="ahp")
vacationAhp <- Load(ahpFile)
Calculate(vacationAhp)
Visualize(vacationAhp)
Analyze(vacationAhp, decisionMaker = "Dad")
AnalyzeTable(vacationAhp, decisionMaker = "Mom")
```

```
AnalyzeTable(vacationAhp,
             decisionMaker = "Kid",
             variable = "priority",
             sort = "orig",
             pruneFun = function(node, dm) PruneByCutoff(node, dm, minWeight = 0.1))
```

`AhpMatrix` *Create the AHP preference matrix from a dataframe containing the pairwiswe preferences.*

Description

Create the AHP preference matrix from a dataframe containing the pairwiswe preferences.

Usage

```
AhpMatrix(preferenceCombinations)
```

Arguments

`preferenceCombinations`
: a data.frame containing category or alternative A in the first column, B in the second colum, and the preference in the third column.

Value

an AHP preference matrix

`Analyze` *Analyze your AHP model*

Description

Converts the calculated AHP tree into a `data.frame` or an HTML table, containing all the weight contributions or priorities to/of the overall decision. You can also sort and filter the output.

Displays the AHP analysis in form of an html table, with gradient colors and nicely formatted.

Usage

```
Analyze(ahpTree, decisionMaker = "Total", variable = c("weightContribution",
  "priority", "score"), sort = c("priority", "totalPriority", "orig"),
  pruneFun = function(node, decisionMaker) TRUE)

AnalyzeTable(ahpTree, decisionMaker = "Total",
  variable = c("weightContribution", "priority", "score"),
  sort = c("priority", "totalPriority", "orig"), pruneFun = function(node,
  decisionMaker) TRUE, weightColor = "honeydew3",
  consistencyColor = "wheat2", alternativeColor = "thistle4")

PruneByCutoff(node, decisionMaker, minWeight = 0)

PruneLevels(node, decisionMaker, levelsToPrune = 0)
```

Arguments

ahpTree	the calculated AHP data.tree
decisionMaker	the name of the decision maker. The default returns the joint decision.
variable	the variable to display, i.e. either weightContribution (the default), priority, or score
sort	sort either by priority according to the decision maker (the default), by totalPriority, or as originally specified (orig)
pruneFun	use this to filter the what rows are shown in the analysis pruneFun must be a function taking a Node as its first argument, and the decisionMaker as its second argument. The default (NULL) returns the full AHP tree
weightColor	The name of the color to be used to emphasize weights of categories. See color for a list of possible colors.
consistencyColor	The name of the color to be used to highlight bad consistency
alternativeColor	The name of the color used to highlight big contributors to alternative choices.
node	the Node
minWeight	prunes the nodes whose weightContribution is smaller than the minWeight
levelsToPrune	cuts the n hightest levels of the ahp tree

Value

Analyze returns a data.frame containing the contribution of each criteria

AnalyzeTable returns a formattable data.frame object which, in most environments, will be displayed as an HTML table

the Prune methods must return TRUE for nodes to be kept, FALSE for nodes to be pruned

Examples

```
ahpFile <- system.file("extdata", "car.ahp", package="ahp")
carAhp <- Load(ahpFile)
Calculate(carAhp)
Analyze(
   carAhp,
   pruneFun = function(x, decisionMaker) {
    PruneLevels(x, decisionMaker, 1) && PruneByCutoff(x, decisionMaker, minWeight = 0.05)
   }
)


ahpFile <- system.file("extdata", "vacation.ahp", package="ahp")
vacationAhp <- Load(ahpFile)
Calculate(vacationAhp)
AnalyzeTable(
   vacationAhp,
   decisionMaker = "Kid",
   variable = "score",
   sort = "totalPriority"
)
```

`Calculate` *Calculate the Ahp tree*

Description

Calculate the Ahp tree

Usage

```
Calculate(ahpTree, pairwiseFun = PrioritiesFromPairwiseMatrixEigenvalues,
  scoresFun = PrioritiesFromScoresDefault)
```

Arguments

`ahpTree`	a data.tree tree containing the Ahp model specification
`pairwiseFun`	lets you overwrite the function that is used to calculate the priorities from the pairwise preference matrix.
`scoresFun`	lets you overwrite the function that is used to calculate the priorites from scores.

GetPairwiseFromFunction *Create table from function*

Description

Create table from function

Usage

```
GetPairwiseFromFunction(node)
```

Arguments

node	The node

Load *Load an ahp model from file*

Description

ahp files are in YAML format, and they are self-contained, fully specified ahp problems. They contain two sections: **alternatives** and **goal**.

Usage

```
Load(ahpFile)

LoadString(ahpString)
```

Arguments

ahpFile	The full path to the file to be loaded, or a connection.
ahpString	A character string to be loaded.

Details

The **alternatives** section contains a list of alternatives, where each alternative may have a number of attributes.

The **goal** section is a tree of criteria, each criteria having a `name`, a `preferences` attribute, and possible child criteria or alternatives.

To look at a sample file, type, see examples below or type `vignette("examples", package = "ahp")`. To learn the details about the ahp file format, type `vignette("file-format", package = "ahp")`.

Value

A data.tree containing the model specification.

Examples

```
ahpFile <- system.file("extdata", "car.ahp", package="ahp")

#look at a sample file
cat(readChar(ahpFile, file.info(ahpFile)$size))

#load the file into R
carAhp <- Load(ahpFile)
```

PrioritiesFromPairwiseMatrixEigenvalues

Calculate the ahp priority weights from the AHP matrix.

Description

For a comparison of different methods, see for example **How to derive priorities in AHP: a comparative study**, by Alessio Ishizaka and Markus Lusti, as available here: http://eprints.port.ac.uk/9041/1/filetodownload,7063

Usage

```
PrioritiesFromPairwiseMatrixEigenvalues(mat, allowedConsistency = 1)

PrioritiesFromPairwiseMatrixMeanNormalization(mat)

PrioritiesFromPairwiseMatrixGeometricMean(mat)
```

Arguments

mat	The AHP preference matrix
allowedConsistency	if the AHP consistency ratio is larger than this value, AHP is not applied and equal weights are returned.

Value

the ahp preference weights

`PrioritiesFromScoresDefault` *Converts a vector of scores into priority weights.*

Description

While pure AHP limits itself to pairwise preferences, scoring alternatives on an arbitrary scale is often much less time consuming in practice. This method calculates the priority weight as `score / sum(scores)`

Usage

```
PrioritiesFromScoresDefault(scores)
```

Arguments

`scores`	a vector of scores

Value

a vector of priority weights

`RunGUI` *Shows a GUI (Graphical User Interface) that lets you specify AHP models and view the results.*

Description

Shows a GUI (Graphical User Interface) that lets you specify AHP models and view the results.

Usage

```
RunGUI(port = getOption("shiny.port"))
```

Arguments

`port`	The TCP port that the application should listen on. If the `port` is not specified, and the `shiny.port` option is set (with `options(shiny.port = XX)`), then that port will be used. Otherwise, use a random port.

`Visualize` *Draw a diagram of the AHP tree.*

Description

The function uses graphviz via DiagrammeR. For details on styling, refer to these.

Usage

```
Visualize(ahpTree, criteriaNodesStyle = list(style = "filled,rounded", shape =
  "box", color = "honeydew4", fillcolor = "honeydew", penwidth = 4, fontname =
  "helvetica"), alternativeNodesStyle = list(style = "filled,rounded", shape =
  "box", color = "thistle4", fillcolor = "thistle", penwidth = 4, fontname =
  "helvetica"), criteriaEdgesStyle = list(arrowhead = "vee", color = "grey35",
  penwidth = 2), alternativeEdgesStyle = list(dir = "none", color = "grey35",
  penwidth = 2))

GetGraph(ahpTree, criteriaNodesStyle = list(style = "filled,rounded", shape =
  "box", color = "honeydew4", fillcolor = "honeydew", penwidth = 4, fontname =
  "helvetica"), alternativeNodesStyle = list(style = "filled,rounded", shape =
  "box", color = "thistle4", fillcolor = "thistle", penwidth = 4, fontname =
  "helvetica"), criteriaEdgesStyle = list(arrowhead = "vee", color = "grey35",
  penwidth = 2), alternativeEdgesStyle = list(dir = "none", color = "grey35",
  penwidth = 2))
```

Arguments

`ahpTree`	The tree to be drawn
`criteriaNodesStyle`	a list of graphviz / dot language styling elements
`alternativeNodesStyle`	a list of graphviz / dot language styling elements
`criteriaEdgesStyle`	a list of graphviz / dot language styling elements
`alternativeEdgesStyle`	a list of graphviz / dot language styling elements

Index