Package: QAIG (via r-universe)

August 31, 2024

Type Package

Title Automatic Item Generator for Quantitative Multiple-Choice Items

Version 0.1.7

Date 2020-05-19

Maintainer Subhabrata Patra (Shubh) <shubh.b.patra@gmail.com>

Description A tool for automatic generation of sibling items from a parent item model defined by the user. It is an implementation of the process automatic item generation (AIG) focused on generating quantitative multiple-choice type of items (see Embretson, Kingston (2018) <doi:10.1111/jedm.12166>).

URL https://github.com/shubh-b/QAIG

BugReports https://github.com/shubh-b/QAIG/issues

Depends R (>= 3.1.0)

License GPL-3

Imports stringr, Formula, stats, utils

Encoding UTF-8

LazyData TRUE

RoxygenNote 6.1.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

Repository https://shubh-b.r-universe.dev

RemoteUrl https://github.com/shubh-b/qaig

RemoteRef HEAD

RemoteSha 82ccdbe7743846fae43db2af9c4193de78939930

Contents

`itemgen` *Automatic Item Generator for Quantitative Multiple-Choice Items*

Description

`itemgen` function generates group of sibling items from a parent item model defined by user.

Usage

```
itemgen(stem_text = stem_text, formulae = formulae, N = N, C,
  options_affix, ans_key, save.csv)
```

Arguments

`stem_text`	The stem of the parent item with specified number-variables and character-variables.
`formulae`	A raw text that contains necessary formulae for the options (response choices) along with necessary values or functions that help to calculate the numeric value for each option.
`N`	A list of numeric input vector(s) for the number variable(s) in the stem.
`C`	(Optional) A list of character input vector(s) for the character variable(s) in the stem if there is any.
`options_affix`	(Optional) A list that consists of vectors with prefixes and suffixes (if there is any) of the numeric values in the options along with any text that can be included as an option of the items.
`ans_key`	(Optional) A text that indicates the correct response if it is NOT specified within formulae by '~'.
`save.csv`	(Optional) A name text given by the user for the output .csv file, if user wants to save the newly generated sibling items in working directory as a data.frame.

Details

User has to develop a short schema for the parent item model that contains formation of stem along with formula for each of the response choices of the parent item as the input. Number-variables and character-variables must be specified in particular manner in the stem. Each formula must be written in new line as text and should be declared together as an object. `itemgen` function delivers the changes in the positions of the variables in stem and calculates the response choices automatically by taking members from the input vectors given by user in the schema. As a result, several permutations of changes in the variables lead to generation of new group of items. Please see vignette of 'QAIG' for more details.

Value

This function returns a data frame that contains stem, options, answer key etc. for all the generated sibling items within its rows to display in console and within its columns in the saved .csv file if the input for the argument 'save.csv' is given in `itemgen` function.

Note

The formula model for each option must be distinct. `itemgen` function does NOT permit same numeric value as two or more response choices and hence it will throw an error. If same numeric value needs to be produced as more than one response choices, those models can be made different by adding 0 or multiplying 1 with the terms in the model.

The model for the distractor options in formulae must be written using "?". Correct response option can be written using EITHER "~" OR "?". In OR case correct response must be indicated by the function argument "ans_key" to stop `itemgen` function throw an error. Please see section 2 and 3 in vignette of 'QAIG'.

Author(s)

Shubh Patra and Bao Sheng Loe

References

Mark J. Gierl, Hollis Lai (2011). The Role of Item Models in Automatic Item Generation. https://www.researchgate.net/publication/239794821_The_Role_of_Item_Models_in_Automatic_Item_Generation

Susan E. Embretson, Neal M. Kingston (2018). Automatic Item Generation: A More Efficient Process for Developing Mathematics Achievement Items? https://onlinelibrary.wiley.com/doi/epdf/10.1111/jedm.12166

Examples

```
stem_text <- "The sum value of all the odd [C1] between [N1] and [N2] is"
n1 <- c(20, 24, 28, 32)
n2 <- c(48, 52, 56)
c1 <- c("natural numbers", "integers")
N <- list(n1 = n1, n2 = n2)
C <- list(c1 = c1)
formulae <- "Option_A ? sum((n1+1) : (n2-1))/2\n
Option_B ~ (length(seq(n1+1, n2-1, by = 2)))*(n1+n2)/2\n
Option_C ? sum(n1 : n2)/2\n
Option_D ? (length(seq(n1, n2, by = 2)))*(n1+n2)/2\n"

options_affix <- list(Option_A = c("", ""), Option_B = c("", ""), Option_C = c("", ""),
Option_D = c("", ""), Difficulty = "MEDIUM")

# itemgen() function can be used as:
itemgen(stem_text = stem_text, formulae = formulae, N = N, C = C, options_affix = options_affix)
```

Index