Package ‘ShinyItemAnalysis’

May 4, 2020

Type Package

Title Test and Item Analysis via Shiny

Version 1.3.3

Date 2020-05-04

Author Patricia Martinkova [aut, cre],
Adela Hladka [aut],
Ondrej Leder [ctb],
Jakub Houdek [ctb],
Lubomir Stepanek [ctb],
Tomas Jurica [ctb],
Jana Vorlickova [ctb],
Jan Netik [ctb]

Maintainer Patricia Martinkova <`martinkova@cs.cas.cz`>

Depends R (>= 3.5.0)

Imports corrplot, cowplot, CTT, data.table, deltaPlotR, DT, difNLR (>= 1.3.2), difR (>= 5.0), ggdendro, ggplot2 (>= 2.2.1), gridExtra, knitr, latticeExtra, ltm, mirt (>= 1.24), moments, msm, nnet, plotly, psych, psychometric, reshape2, rmarkdown, shiny (>= 1.0.3), shinyBS, shinydashboard, shinyjs (>= 0.9), stringr, VGAM, xtable

Description Interactive shiny application for analysis of educational tests and their items.

License GPL-3

LazyData TRUE

RoxygenNote 7.1.0

BugReports `https://github.com/patriciamar/ShinyItemAnalysis/issues`

URL `http://www.ShinyItemAnalysis.org`,
`https://CRAN.R-project.org/package=ShinyItemAnalysis`

Encoding UTF-8

NeedsCompilation no

Repository CRAN

Date/Publication 2020-05-04 15:20:02 UTC

R topics documented:

`ShinyItemAnalysis-package`

ShinyItemAnalysis: Test and Item Analysis via Shiny

Description

The ShinyItemAnalysis package containts an interactive shiny application for analysis of educational tests and their items which can be accessed using function `startShinyItemAnalysis()`. The shiny application covers a broad range of psychometric methods and offers data examples, model equations, parameter estimates, interpretation of results, together with a selected R code, and is therefore suitable for teaching psychometric concepts with R. It also allows the users to upload and analyze their own data and to automatically generate analysis reports in PDF or HTML.

Besides, the package provides its own functions for test and item analysis within classical test theory framework (e.g., functions `gDiscrim()`, `ItemAnalysis()`, `DistractorAnalysis()`, or `DDplot()`), using various regression models (e.g., `plotCummulative()`, `plotAdjacent()`, `plotMultinomial()`, or `plotDIFLogistic()`), and under IRT framework (e.g., `ggWrightMap()`, or `plotDIFirt()`).

Package also contains several demonstration datasets including `HCI` dataset from the reference paper Martinkova and Drabinova (2018).

Details

Package: ShinyItemAnalysis
Type: Package
Version: 1.3.3
Date: 2020-05-04
Depends: R (>= 3.5.0)
Imports: corrplot, cowplot, CTT, data.table, deltaPlotR, DT, difNLR (>= 1.3.2), difR (>= 5.0), ggdendro, ggplot2 (>= 2.2.1), gridExtra, knitr, latticeExtra, ltm, mirt (>= 1.24), moments, msm, nnet, plotly, psych, psychometric, reshape2, rmarkdown, shiny (>= 1.0.3), shinyBS, shinydashboard, shinyjs (>= 0.9), stringr, VGAM, xtable License: GPL-3
BugReports: `https://github.com/patriciamar/ShinyItemAnalysis/issues`
Website: `http://www.ShinyItemAnalysis.org/`
Encoding: UTF-8

Functions

- `startShinyItemAnalysis`
- `DDplot`
- `DistractorAnalysis`
- `plotDistractorAnalysis`
- `gDiscrim`
- `ggWrightMap`
- `ItemAnalysis`
- `plotAdjacent`, `plotCumulative`, `plotMultinomial`
- `plotDIFirt`, `plotDIFLogistic`

Datasets

- `dataMedical`
- `dataMedicalgraded`
- `HCI`
- `LearningToLearn`

Author(s)

Patricia Martinkova
Institute of Computer Science of the Czech Academy of Sciences
`<martinkova@cs.cas.cz>`

Adela Hladka (nee Drabinova)
Institute of Computer Science of the Czech Academy of Sciences
Faculty of Mathematics and Physics, Charles University
`<hladka@cs.cas.cz>`

References

Martinkova, P., & Drabinova, A. (2018). ShinyItemAnalysis for Teaching Psychometrics and to Enforce Routine Analysis of Educational Tests. The R Journal, 10(2), 503-515. `https://doi.org/10.32614/RJ-2018-074`

See Also

Useful links:

- `http://www.ShinyItemAnalysis.org`
- `https://CRAN.R-project.org/package=ShinyItemAnalysis`
- Report bugs at `https://github.com/patriciamar/ShinyItemAnalysis/issues`

`dataMedical` *Dichotomous Dataset of Admission Test to Medical School*

Description

The `dataMedical` dataset consists of the responses of 2,392 subjects (750 males, 1,633 females and 9 subjects without gender specification) to admission test to a medical school. It contains 100 items. A correct answer is coded as `"1"` and incorrect answer as `"0"`. Missing answers were evaluated as incorrect, i.e. `"0"`.

Usage

```
data(dataMedical)
```

Format

A `dataMedical` is a `data.frame` consisting of 2,392 observations on the following 102 variables.

X the first 100 columns represent dichotomously scored items of the test.

gender variable describing gender; values `"0"` and `"1"` refer to males and females.

StudySuccess criterion variable; value `"1"` means that student studies standardly, `"0"` otherwise (e.g. leaving or interrupting studies).

Author(s)

Cestmir Stuka
First Faculty of Medicine, Charles University

Martin Vejrazka
First Faculty of Medicine, Charles University

Patricia Martinkova
Institute of Computer Science of the Czech Academy of Sciences
`<martinkova@cs.cas.cz>`

References

Stuka, C. Vejrazka, M., Martinkova, P. Komenda, M. & Stepanek, L. (2016). The Use of Test and Item Analysis for Improvement of Tests. Workshop held at conference MEFANET, 2016, Brno, Czech Republic.

See Also

`dataMedicaltest`, `dataMedicalkey`, `dataMedicalgraded`

`dataMedicalgraded` *Graded Dataset of Admission Test to Medical School*

Description

The `dataMedicalgraded` dataset consists of the responses of 2,392 subjects (750 males, 1,633 females and 9 subjects without gender specification) to multiple-choice admission test to a medical school. It contains 100 items. Each item is graded with 0 to 4 points. Maximum of 4 points were set if all correct answers and none of incorrect answers were selected.

Usage

```
data(dataMedicalgraded)
```

Format

A `dataMedicalgraded` is a `data.frame` consisting of 2,392 observations on the following 102 variables.

X the first 100 columns represent ordinal item scores of the test.

gender variable describing gender; values `"0"` and `"1"` refer to males and females.

StudySuccess criterion variable; value `"1"` means that student studies standardly, `"0"` otherwise (e.g. leaving or interrupting studies).

Author(s)

Cestmir Stuka
First Faculty of Medicine, Charles University

Martin Vejrazka
First Faculty of Medicine, Charles University

Patricia Martinkova
Institute of Computer Science of the Czech Academy of Sciences
`<martinkova@cs.cas.cz>`

References

Stuka, C. Vejrazka, M., Martinkova, P. Komenda, M. & Stepanek, L. (2016). The Use of Test and Item Analisis for Improvement of Tests. Workshop held at conference MEFANET, 2016, Brno, Czech Republic.

See Also

`dataMedical`, `dataMedicaltest`, `dataMedicalkey`

`dataMedicalkey` *Key of Correct Answers for* `dataMedicaltest` *Dataset*

Description

The `dataMedicalkey` is a vector of factors representing correct answers of `dataMedicaltest` dataset.

Usage

```
data(dataMedicalkey)
```

Format

A vector with 100 values representing correct answers to items of `dataMedicaltest` dataset. For more details see `dataMedicaltest`.

Author(s)

Cestmir Stuka
First Faculty of Medicine, Charles University

Martin Vejrazka
First Faculty of Medicine, Charles University

Patricia Martinkova
Institute of Computer Science of the Czech Academy of Sciences
`<martinkova@cs.cas.cz>`

References

Stuka, C. Vejrazka, M., Martinkova, P. Komenda, M. & Stepanek, L. (2016). The Use of Test and Item Analisis for Improvement of Tests. Workshop held at conference MEFANET, 2016, Brno, Czech Republic.

See Also

`dataMedical`, `dataMedicaltest`, `dataMedicalgraded`

`dataMedicaltest` *Dataset of Admission Test to Medical School*

Description

The `dataMedicaltest` dataset consists of the responses of 2,392 subjects (750 males, 1,633 females and 9 subjects without gender specification) to multiple-choice admission test to a medical school. It contains 100 items, possible answers were A, B, C, D, while any combination of these can be correct.

Usage

```
data(dataMedicaltest)
```

Format

A ‘dataMedicaltest‘ is a ‘data.frame‘ consisting of 2,392 observations on the following 102 variables.

X the first 100 columns represent items answers.

gender variable describing gender; values ‘"0"‘ and ‘"1"‘ refer to males and females.

StudySuccess criterion variable; value ‘"1"‘ means that student studies standardly, ‘"0"‘ otherwise (e.g. leaving or interrupting studies).

Author(s)

Cestmir Stuka
First Faculty of Medicine, Charles University

Martin Vejrazka
First Faculty of Medicine, Charles University

Patricia Martinkova
Institute of Computer Science of the Czech Academy of Sciences
martinkova@cs.cas.cz

References

Stuka, C. Vejrazka, M., Martinkova, P. Komenda, M. & Stepanek, L. (2016). The Use of Test and Item Analisis for Improvement of Tests. Workshop held at conference MEFANET, 2016, Brno, Czech Republic.

See Also

`dataMedical`, `dataMedicalkey`, `dataMedicalgraded`

`DDplot` *Graphical representation of difficulty and (generalized) discrimination in item analysis*

Description

Plots difficulty and (generalized) discrimination for items ordered by difficulty.

Usage

```
DDplot(data, item.names, k = 3, l = 1, u = 3,
discrim = "ULI", maxscore, minscore, bin = FALSE, cutscore, average.score = FALSE,
thr = 0.2)
```

Arguments

`data`	numeric: binary or ordinal data matrix or data frame. See **Details**.
`item.names`	character: the names of items.
`k`	numeric: number of groups to which may be data.frame x divided by the total score. Default value is 3. See **Details**.
`l`	numeric: lower group. Default value is 1. See **Details**.
`u`	numeric: upper group. Default value is 3. See **Details**.
`discrim`	character: type of discrimination index to be calculated. Deafult value is "ULI". See **Details**.
`maxscore`	vector or numeric: maximal scores of items. If numeric, the same maximal score is used for all items. If missing, vector of achieved maximal scores is calculated and used in calculations.
`minscore`	vector or numeric: minimal scores of items. If numeric, the same minimal score is used for all items. If missing, vector of achieved minimal scores is calculated and used in calculations.
`bin`	logical: should the ordinal data be binarized. Deafult value is FALSE. See **Details**.
`cutscore`	vector or numeric: cutscore used to binarize the data.set. If numeric, the same cutscore is used for all items. If missing, vector of maximal scores is used in calculations.
`average.score`	logical: should average score of the item disaplyed instead of difficulty. Default value is FALSE. See **Details**.
`thr`	numeric: value of discrimination threshold. See **Details**.

Details

The `data` is a matrix or data frame whose rows represents examinee answers (`1` correct, `0` incorrect, or ordinal item scores) and columns correspond to the items.

The `item.names` argument stands for names of items. If not specified, the names of dataset columns are used. Difficulty and discrimination indices are plotted for each item, items are ordered by their difficulty.

Discrimination is calculated using method specified in `discrim`. Default option "ULI" calculates difference in ratio of correct answers in upper and lower third of students. "RIT" index caluclates correlation between item score and test total score. "RIR" index caclulates correlation between item score and total score for the rest of the items. With option "none", only difficulty is displayed.

"ULI" index can be generalized using arguments `k`, `l` and `u`. Generalized ULI discrimination is then computed as follows: The function takes data on individuals, computes their total test score and then divides individuals into `k` groups. The lower and upper group are determined by `l` and `u` parameters, i.e. l-th and u-th group where the ordering is defined by increasing total score.

For ordinal data, difficulty is defined as relative score (achieved - minimal)/(maximal - minimal). Minimal score can be specified by `minscore`, maximal score can be specified by `maxscore`. Average score of items can be displayed with argument `average.score = T`. Note that for binary data difficulty estimate is the same as average score of the item.

Binarization of data is allowed in `bin`, for this purpose `cutscore` is used.

By rule of thums, discrimination of items should not be lower than 0.2. The value of threshold can be specified via `thr` argument. In case that `thr = NULL`, no horizontal line is displayed in the plot.

Note

Generalized discrimination is calculated by `gDiscrim` function.

Author(s)

Adela Hladka
Institute of Computer Science of the Czech Academy of Sciences
Faculty of Mathematics and Physics, Charles University
`<hladka@cs.cas.cz>`

Lubomir Stepanek
First Faculty of Medicine, Charles University

Jana Vorlickova
Institute of Computer Science of the Czech Academy of Sciences

Patricia Martinkova
Institute of Computer Science of the Czech Academy of Sciences
`<martinkova@cs.cas.cz>`

References

Martinkova, P., Stepanek, L., Drabinova, A., Houdek, J., Vejrazka, M., & Stuka, C. (2017). Semi-real-time analyses of item characteristics for medical school admission tests. In: Proceedings of the 2017 Federated Conference on Computer Science and Information Systems.

See Also

`gDiscrim`, `discrim`

Examples

```
# loading 100-item medical admission test data sets
data(dataMedical, dataMedicalgraded)
# binary data set
dataBin <- dataMedical[, 1:100]
# ordinal data set
dataOrd <- dataMedicalgraded[, 1:100]

# DDplot of binary data set
DDplot(dataBin)
## Not run:
#' # DDplot of binary data set without threshold
DDplot(dataBin, thr = NULL)
# compared to DDplot using ordinal data set and 'bin = TRUE'
DDplot(dataOrd, bin = TRUE)
# compared to binarized data set using bin = TRUE and cutscore equal to 3
DDplot(dataOrd, bin = TRUE, cutscore = 3)

# DDplot of binary data using generalized ULI
# discrimination based on 5 groups, comparing 4th and 5th
# threshold lowered to 0.1
DDplot(dataBin, k = 5, l = 4, u = 5, thr = 0.1)

# DDplot of ordinal data set using ULI
DDplot(dataOrd)
# DDplot of ordinal data set using generalized ULI
# discrimination based on 5 groups, comparing 4th and 5th
# threshold lowered to 0.1
DDplot(dataOrd, k = 5, l = 4, u = 5, thr = 0.1)
# DDplot of ordinal data set using RIT
DDplot(dataOrd, discrim = "RIT")
# DDplot of ordinal data set using RIR
DDplot(dataOrd, discrim = "RIR")
# DDplot of ordinal data set disaplaying only difficulty
DDplot(dataBin, discrim = "none")

# DDplot of ordinal data set disaplaying difficulty estimates
DDplot(dataOrd)
# DDplot of ordinal data set disaplaying average item scores
DDplot(dataOrd, average.score = TRUE)

## End(Not run)
```

`DistractorAnalysis` *Function for item distractor analysis*

Description

Performs distractor analysis for each item and optional number of groups.

Usage

```
DistractorAnalysis(data, key, p.table = FALSE, num.groups = 3, matching = NULL,
match.discrete = FALSE, cut.points)
```

Arguments

`data`	character: data matrix or data frame. See **Details**.
`key`	character: answer key for the items.
`p.table`	logical: should the function return the proportions. If `FALSE` (default) the counts are returned.
`num.groups`	numeric: number of groups to that should be respondents splitted.
`matching`	numeric: numeric vector. If not provided, total score is calculated and distractor analysis is performed based on it.
`match.discrete`	logical: is `matching` discrete? Default value is `FALSE`. See details.
`cut.points`	numeric: numeric vector specifying cut points of `matching`. See details.

Details

This function is adapted version of `distractor.analysis` function from `CTT` package.

The `data` is a matrix or data frame whose rows represents unscored item response from a multiple-choice test and columns correspond to the items.

The `key` must be a vector of the same length as `ncol(data)`.

In case, no `matching` is provided, the scores are calculated using the item data and key. The respondents are by default splitted into the `num.groups`-quantiles and the number (or proportion) of respondents in each quantile is reported with respect to their answers. In case that `matching` is discrete (`match.discrete = TRUE`), `matching` is splitted based on its unique levels. Other cut points can be specified via `cut.points` argument.

Author(s)

Adela Hladka
Institute of Computer Science of the Czech Academy of Sciences
Faculty of Mathematics and Physics, Charles University
<hladka@cs.cas.cz>

Patricia Martinkova
Institute of Computer Science of the Czech Academy of Sciences

<martinkova@cs.cas.cz>

See Also

`distractor.analysis`

Examples

```
# loading 100-item medical admission test data
data(dataMedicaltest, dataMedicalkey)
data <- dataMedicaltest[, 1:100]
dataBin <- dataMedical[, 1:100]
key <- unlist(dataMedicalkey)

# distractor analysis for dataMedicaltest data set
DistractorAnalysis(data, key)
## Not run:
# distractor analysis for dataMedicaltest data set with proportions
DistractorAnalysis(data, key, p.table = T)

# distractor analysis for dataMedicaltest data set for 6 groups
DistractorAnalysis(data, key, num.group = 6)

# distractor analysis for dataMedicaltest using specified matching
matching <- round(rowSums(databin), -1)
DistractorAnalysis(data, key, matching = matching)

# distractor analysis for dataMedicaltest using discrete matching
DistractorAnalysis(data, key, matching = matching, match.discrete = T)

# distractor analysis for dataMedicaltest using groups specified by cut.points
DistractorAnalysis(data, key, cut.points = seq(10, 100, 10))

## End(Not run)
```

`gDiscrim` *Generalized Item Discrimination*

Description

gDiscrim function computes various generalizations of discrimination index ULI. It enumerates the ablitity of item to distinguish between individuals from upper (U) vs. lower (L) ability groups, i.e. between respondents with high vs. low overall score on the test. Number of groups, as well as upper and lower groups can be specified by user. Maximal and minimal score in ordinal data sets can be specified by user.

Usage

```
gDiscrim(x, k = 3, l = 1, u = 3, maxscore, minscore)
```

Arguments

`x`	matrix or data.frame of items to be examined. Rows represent persons, columns reperesent items.
`k`	numeric: number of groups to which may be data.frame `x` divided by the total score. Default value is 3. See **Details**.
`l`	numeric: lower group. Default value is 1. See **Details**.
`u`	numeric: upper group. Default value is 3. See **Details**.
`maxscore`	numeric: maximal score in ordinal items. If missing, vector of obtained maximal scores is imputed. See **Details**.
`minscore`	numeric: minimal score in ordinal items. If missing, vector of obtained minimal scores is imputed. See **Details**.

Details

The function computes total test scores for all respondents and then divides the respondents into `k` groups. The lower and upper groups are determined by `l` and `u` parameters, i.e. l-th and u-th group where the ordering is defined by increasing total score.

In ordinal items, difficulty is calculated as difference of average score divided by range (maximal possible score `maxscore` minus minimal possible score `minscore` for given item).

Discrimination is calculated as difference in difficulty between upper and lower group.

Note

`gDiscrim` is used by `DDplot` function.

Author(s)

Adela Hladka
Institute of Computer Science of the Czech Academy of Sciences
Faculty of Mathematics and Physics, Charles University
`<hladka@cs.cas.cz>`

Lubomir Stepanek
First Faculty of Medicine, Charles University

Jana Vorlickova
Institute of Computer Science of the Czech Academy of Sciences

Patricia Martinkova
Institute of Computer Science of the Czech Academy of Sciences
`<martinkova@cs.cas.cz>`

References

Martinkova, P., Stepanek, L., Drabinova, A., Houdek, J., Vejrazka, M., & Stuka, C. (2017). Semi-real-time analyses of item characteristics for medical school admission tests. In: Proceedings of the 2017 Federated Conference on Computer Science and Information Systems. https://doi.org/10.15439/2017F380

See Also

`DDplot`

Examples

```
# loading 100-item medical admission test data sets
data(dataMedical, dataMedicalgraded)
# binary data set
dataBin <- dataMedical[, 1:100]
# ordinal data set
dataOrd <- dataMedicalgraded[, 1:100]

# ULI for first 5 items for binary data set
# compare to psychometric::discrim(x)
gDiscrim(dataBin)[1:5]
# generalized ULI using 5 groups, compare 4th and 5th for binary data set
gDiscrim(dataBin, k = 5, l = 4, u = 5)[1:5]

# ULI for first 5 items for ordinal data set
gDiscrim(dataOrd)[1:5]
# generalized ULI using 5 groups, compare 4th and 5th for binary data set
gDiscrim(dataOrd, k = 5, l = 4, u = 5)[1:5]
# maximum (4) and minimum (0) score are same for all items
gDiscrim(dataOrd, k = 5, l = 4, u = 5, maxscore = 4, minscore = 0)[1:5]
```

`ggWrightMap` *Wright Map using ggplot*

Description

This function allows to generate Wright Map (also called item-person map) using `ggplot` function from package `ggplot2` and `plot_grid` function from `cowplot`. Wright Map is used to display histogram of factor scores and the item difficulty parameters estimated by the Rasch IRT model.

Usage

```
ggWrightMap(theta, b, binwidth = 0.5, color = "blue", size = 15, item.names)
```

Arguments

`theta`	numeric: vector of ability estimates.
`b`	numeric: vector of difficulty estimates.
`binwidth`	numeric: the width of the bins of histogram.
`color`	character: color of histogram.
`size`	text size in pts.
`item.names`	names of items to be displayed.

Author(s)

Adela Hladka
Institute of Computer Science of the Czech Academy of Sciences
Faculty of Mathematics and Physics, Charles University
<hladka@cs.cas.cz>

Patricia Martinkova
Institute of Computer Science of the Czech Academy of Sciences
<martinkova@cs.cas.cz>

References

Wright, B. D., & Stone, M. H. (1979). Best test design.

See Also

`wrightMap`

Examples

```
library(mirt)

# loading 100-item medical admission test data sets
data(dataMedical)
# binary data set
dataBin <- dataMedical[, 1:100]

# fit Rasch model with mirt package
fit <- mirt(dataBin, model = 1, itemtype = "Rasch")
# factor scores
theta <- as.vector(fscores(fit))
# difficulty estimates
b <- coef(fit, simplify = TRUE)$items[, "d"]

ggWrightMap(theta, b)

item.names <- paste("Item", 1:20)
ggWrightMap(theta, b, item.names = item.names)
```

`HCI` *Homeostasis Concept Inventory Dichotomous Dataset*

Description

(`HCI`) dataset consists of the dichotomously scored responses of 651 students (405 males, 246 females) to Homeostasis Concept Inventory (HCI) multiple-choice test. It containts 20 items, vector of gender membership and identificator whether students plan to major.

Usage

```
data(HCI)
```

Format

`HCI` is a `data.frame` consisting of 651 observations on the 22 variables.

Item1-Item20	dichotomously scored items of the HCI test.
gender	gender membership vector, `"0"` males, `"1"` females.
major	identificator whether students planning to major in the life sciences.

Author(s)

Jenny L. McFarland
Biology Department, Edmonds Community College

References

McFarland, J. L., Price, R. M., Wenderoth, M. P., Martinkova, P., Cliff, W., Michael, J., ... & Wright, A. (2017). Development and validation of the homeostasis concept inventory. CBE-Life Sciences Education, 16(2), ar35.

See Also

`HCItest`, `HCIkey`

`HCIkey` *Key of Correct Answers for Homeostasis Concept Inventory Dataset*

Description

The `HCIkey` is a vector of factors representing correct answers of `HCItest` dataset.

Usage

```
data(HCIkey)
```

Format

A nominal vector with 20 values representing correct answers to items of `HCItest` dataset. For more details see `HCItest`.

Author(s)

Jenny L. McFarland
Biology Department, Edmonds Community College

References

McFarland, J. L., Price, R. M., Wenderoth, M. P., Martinkova, P., Cliff, W., Michael, J., ... & Wright, A. (2017). Development and validation of the homeostasis concept inventory. CBE-Life Sciences Education, 16(2), ar35.

See Also

`HCI`, `HCItest`

`HCItest` *Homeostasis Concept Inventory Dataset*

Description

(`HCItest`) dataset consists of the responses of 651 students (405 males, 246 females) to Homeostasis Concept Inventory (HCI) multiple-choice test. It containts 20 items, vector of gender membership and identificator whether students plan to major.

Usage

```
data(HCItest)
```

Format

`HCItest` is a `data.frame` consisting of 651 observations on the 22 variables.

Item1-Item20 multiple-choice items of the HCI test.

gender gender membership vector, `"0"` males, `"1"` females.

major identificator whether students planning to major in the life sciences.

Author(s)

Jenny L. McFarland
Biology Department, Edmonds Community College

References

McFarland, J. L., Price, R. M., Wenderoth, M. P., Martinkova, P., Cliff, W., Michael, J., ... & Wright, A. (2017). Development and validation of the homeostasis concept inventory. CBE-Life Sciences Education, 16(2), ar35.

See Also

`HCI`, `HCIkey`

`ItemAnalysis` *Item Analysis*

Description

`ItemAnalysis` function computes various traditional item analysis indices including difficulty, discrimination and item validity. For ordinal items the difficulty and discrimination indices take into account minimal item score as well as range.

Usage

```
ItemAnalysis(data, y = NULL, k = 3, l = 1, u = 3,
maxscore, minscore, cutscore, add.bin = FALSE)
```

Arguments

`data`	matrix or data.frame of items to be examined. Rows represent respondents, columns reperesent items.
`y`	vector of criterion values.
`k`	numeric: number of groups to which may be data.frame x divided by the total score. Default value is 3. See **Details**.
`l`	numeric: lower group. Default value is 1. See **Details**.
`u`	numeric: upper group. Default value is 3. See **Details**.

`maxscore`	numeric or vector: maximal score in ordinal items. If missing, vector of obtained maximal scores is imputed. See **Details**.
`minscore`	numeric or vector: minimal score in ordinal items. If missing, vector of obtained minimal scores is imputed. See **Details**.
`cutscore`	numeric or vector: cut score used for binarization of ordinal data. If missing, vector of maximal scores is imputed. See **Details**.
`add.bin`	logical: If TRUE, indices are printed also for binarized data. See **Details**.

Details

For ordinal items the difficulty and discrimination indices take into account minimal item score as well as range.

For calculation of discimination ULI index, it is possible to specify the number of groups `k`, and which two groups `l` and `u` are to be compared.

In ordinal items, difficulty is calculated as difference of average score divided by range (maximal possible score `maxscore` minus minimal possible score `minscore`).

If `add.bin` is set to `TRUE`, item analysis of binarized data is included in the output table. In such a case, `cutscore` is used for binarization. When binarizing the data, values greater or equal to cut-score are set to `1`, other values are set to `0`.

Value

`ItemAnalysis` function computes various traditional item analysis indices. Output is a `data.frame` with following columns:

`Difficulty`	average score of the item divided by its range
`Average score`	average score of the item
`SD`	standard deviation of the item score
`SD bin`	standard deviation of the item score for binarized data
`Correct answers`	proportion of correct answers
`Min score`	minimal score specified in `minscore`; if not provided, observed minimal score
`Max score`	maximal score specified in `maxscore`; if not provided, observed maximal score
`Obtained min`	observed minimal score
`Obtained max`	observed maximal score
`Cut score`	cut-score specified in `cutscore`
`ULI`	generalized ULI
`ULI default`	dscrimination with ULI
`RIT`	correlation between item score and overall test score
`RIR`	correlation between item score and overall test score
`Item criterion`	correlation of item score with criterion
`Item reliability`	item reliability index calculated as `cor(item,test)*sqrt(((N-1)/N)*var(item))`, see Allen & Yen (1979), Ch.6.4

`Item reliability woi`	item reliability index (scored without item)
`Item validity`	item validity index calculated as `cor(item,y)*sqrt(((N-1)/N)*var(item))`, see Allen & Yen (1979), Ch.6.4
`Item criterion`	correlation between item and criterion `y`
`Alpha drop`	Cronbach's alpha without given item

With `add.bin == TRUE`, indices based on binarized data set are also provided and marked with `bin`.

Author(s)

Patricia Martinkova
Institute of Computer Science of the Czech Academy of Sciences
`<martinkova@cs.cas.cz>`

Jana Vorlickova
Institute of Computer Science of the Czech Academy of Sciences

Adela Hladka
Institute of Computer Science of the Czech Academy of Sciences
Faculty of Mathematics and Physics, Charles University
`<hladka@cs.cas.cz>`

References

Martinkova, P., Stepanek, L., Drabinova, A., Houdek, J., Vejrazka, M., & Stuka, C. (2017). Semi-real-time analyses of item characteristics for medical school admission tests. In: Proceedings of the 2017 Federated Conference on Computer Science and Information Systems. https://doi.org/10.15439/2017F380

Allen, M. J. & Yen, W. M. (1979). Introduction to measurement theory. Monterey, CA: Brooks/Cole.

See Also

`DDplot`, `gDiscrim`

Examples

```
## Not run:
# loading 100-item medical admission test data sets
data(dataMedical, dataMedicalgraded)
# binary data set
dataBin <- dataMedical[, 1:100]
# ordinal data set
dataOrd <- dataMedicalgraded[, 1:100]
# study success is the same for both data sets
StudySuccess <- dataMedical[, 102]

# item analysis for binary data
head(ItemAnalysis(dataBin))
```

```
# item analysis for binary data using also study success
head(ItemAnalysis(dataBin, y = StudySuccess))

# item analysis for binary data
head(ItemAnalysis(dataOrd))
# item analysis for binary data using also study success
head(ItemAnalysis(dataOrd, y = StudySuccess))
# including also item analysis for binarized data
head(ItemAnalysis(dataOrd,
  y = StudySuccess, k = 5, l = 4, u = 5,
  maxscore = 4, minscore = 0, cutscore = 4, add.bin = TRUE
))

## End(Not run)
```

`LearningToLearn` *Dichotomous Data Set of Learning to Learn Test*

Description

`LearningToLearn` is a real longitudinal dataset used in Martinkova et al (2020) study, demonstrating differential item functioning in change (DIF-C) on Learning to Learn (LtL) test. Among other variables, it primarily contains binary-coded responses of 782 subjects to (mostly) multiple-choice test consisting of 41 items within 7 subscales (see **Format** for details). Each respondent was tested twice in total – the first time in Grade 6 and the second time in Grade 9. Most importantly, school track (variable `track_01` or `track`) is available, with 391 students attending basic school (BS) and 391 pursuing selective academic school (AS). This dataset was created using propensity score matching algorithm to achieve similar characteristics in both tracks (see **References** for details). To further simplify the work with `LtL` dataset, we provide computed total scores as well as 7 subscores, both for Grade 6 and Grade 9. The dataset also includes *change* variables for each item (see **Format** for details) for more detailed DIF-C analysis using multinomial regression model.

Usage

```
data(LearningToLearn)
```

Format

A `LearningToLearn` data frame consists of 782 observations on the following 141 variables:

track_01 dichotomously scored school track, where `"1"` denotes the selective academic school one.

track school track, where `"AS"` represents the selective academic school track, and `"BS"` stands for basic school track.

score_6 & score_9 total test score value obtained by summing all 41 items of `LtL`, the number denotes the Grade which the respondent was taking at the time of testing.

score_6_subtest1–score_6_subtest7 scores of respective cognitive subtest (1–7) of `LtL` in Grade 6.

score_9_subtest1–score_9_subtest7 scores of respective cognitive subtest (1–7) of `LtL` in Grade 9.

Item1A_6–Item7F_6 dichotomously coded 41 individual items obtained at Grade 6, `"1"` represents the correct answer to the particular item.

Item1A_9–Item7F_9 dichotomously coded 41 individual items obtained at Grade 9, `"1"` represents the correct answer to the particular item.

Item1A_changes–Item7F_changes change patterns with those possible values:

- a student responded correctly in neither Grade 6 nor in Grade 9 (did not improve, `"00"`)
- a student responded correctly in Grade 6 but not in Grade 9 (deteriorated, `"10"`)
- a student did not respond correctly in Grade 6 but responded correctly in Grade 9 (improved, `"01"`), and
- a student responded correctly in both grades (did not deteriorate, `"11"`)

Author(s)

Patricia Martinkova
Faculty of Education, Charles University
Institute of Computer Science of the Czech Academy of Sciences
`<martinkova@cs.cas.cz>`

Adela Hladka
Institute of Computer Science of the Czech Academy of Sciences
Faculty of Mathematics and Physics, Charles University
`<hladka@cs.cas.cz>`

Eva Potuznikova
Faculty of Education, Charles University

References

Martinkova, P., Hladka, A., & Potuznikova, E. (2020). Is academic tracking related to gains in learning competence? Using propensity score matching and differential item change functioning analysis for better understanding of tracking implications. *Learning and Instruction*, *66*, 101286. https://doi.org/10.1016/j.learninstruc.2019.101286

`plotAdjacent` *Function for plotting category probabilities of adjacent logistic regression model*

Description

Function for plotting category probabilities function estimated by `vglm()` from `VGAM` package

Usage

```
plotAdjacent(x, matching.name = "matching")
```

Arguments

`x`	object of class `vglm`
`matching.name`	character: name of matching criterion used for estimation in `x`.

Author(s)

Tomas Jurica

Adela Hladka
Institute of Computer Science of the Czech Academy of Sciences
Faculty of Mathematics and Physics, Charles University
`<hladka@cs.cas.cz>`

Patricia Martinkova
Institute of Computer Science of the Czech Academy of Sciences
`<martinkova@cs.cas.cz>`

See Also

`vglm`

Examples

```
# loading packages
library(VGAM)

# loading data
data <- dataMedicalgraded[, 1:100]

# total score calculation
score <- apply(data, 1, sum)
data[, 1] <- ordered(factor(data[, 1], levels = 0:max(data[, 1])))

# cummulative logistic model for item 1
fit <- vglm(data[, 1] ~ score, family = acat(reverse = FALSE, parallel = TRUE))
# coefficients for item 1
coefs <- coef(fit)

plotAdjacent(fit, matching.name = "Total score")
```

`plotCumulative` *Function for plotting cumulative and category probabilities of cumulative logistic regression model*

Description

Function for plotting cumulative and category probabilities function estimated by `vglm()` from `VGAM` package

Usage

```
plotCumulative(x, type = "cumulative", matching.name = "matching")
```

Arguments

`x`	object of class `vglm`
`type`	character: type of plot to be displayed. Options are `"cumulative"` (default) for cumulative probabilities and `"category"` for category probabilities.
`matching.name`	character: name of matching criterion used for estimation in `x`.

Author(s)

Tomas Jurica

Adela Hladka
Institute of Computer Science of the Czech Academy of Sciences
Faculty of Mathematics and Physics, Charles University
<hladka@cs.cas.cz>

Patricia Martinkova
Institute of Computer Science of the Czech Academy of Sciences
<martinkova@cs.cas.cz>

See Also

`vglm`

Examples

```
# loading packages
library(VGAM)

# loading data
data <- dataMedicalgraded[, 1:100]

# total score calculation
```

```
score <- apply(data, 1, sum)
data[, 1] <- ordered(factor(data[, 1], levels = 0:max(data[, 1])))

# cummulative logistic model for item 1
fit <- vglm(data[, 1] ~ score, family = cumulative(reverse = TRUE, parallel = TRUE))
# coefficients for item 1
coefs <- coef(fit)

plotCumulative(fit, type = "cumulative", matching.name = "Total score")
plotCumulative(fit, type = "category", matching.name = "Total score")
```

`plotDIFirt` *Function for characteristic curve of DIF IRT model*

Description

Plots characteristic curve of IRT model.

Usage

```
plotDIFirt(parameters, test = "Lord", item = "all", item.name, same.scale = F)
```

Arguments

`parameters`	numeric: data matrix or data frame. See **Details**.
`test`	character: type of statistic to be shown. See **Details**.
`item`	either character ("all"), or numeric vector, or single number corresponding to column indicators. See **Details**.
`item.name`	character: the name of item.
`same.scale`	logical: are the item `parameters` on the same scale? (default is "FALSE"). See **Details**.

Details

This function plots characteristic curve of DIF IRT model.

The `parameters` matrix has a number of rows equal to twice the number of items in the data set. The first J rows refer to the item parameter estimates in the reference group, while the last J ones correspond to the same items in the focal group. The number of columns depends on the selected IRT model: 2 for the 1PL model, 5 for the 2PL model, 6 for the constrained 3PL model and 9 for the unconstrained 3PL model. The columns of irtParam have to follow the same structure as the output of `itemParEst`, `difLord` or `difRaju` command from `difR` package.

Two possible type of `test` statistics can be visualized - `"Lord"` gives only characteristic curves, `"Raju"` also highlights area between these curves.

For default option `"all"`, all characteristic curves are plotted.

Author(s)

Adela Hladka
Institute of Computer Science of the Czech Academy of Sciences
Faculty of Mathematics and Physics, Charles University
`<hladka@cs.cas.cz>`

Patricia Martinkova
Institute of Computer Science of the Czech Academy of Sciences
`<martinkova@cs.cas.cz>`

See Also

`itemParEst`, `difLord`, `difRaju`

Examples

```
# loading libraries
library(difR)
library(ltm)

# loading data based on GMAT2
data(GMAT2, package = "difNLR")

# Estimation of 2PL IRT model and Lord's statistic
# by difR package
fitLord <- difLord(GMAT2, group = 21, focal.name = 1, model = "2PL")
# plot of item 1 and Lord's statistic
plotDIFirt(fitLord$itemParInit, item = 1)

# Estimation of 2PL IRT model and Raju's statistic
# by difR package
fitRaju <- difRaju(GMAT2, group = 21, focal.name = 1, model = "2PL")
# plot of item 1 and Lord's statistic
plotDIFirt(fitRaju$itemParInit, test = "Raju", item = 1)
```

`plotDIFLogistic` *Function for characteristic curve of 2PL logistic DIF model*

Description

Plots characteristic curve of 2PL logistic DIF model

Usage

```
plotDIFLogistic(x, item = 1, item.name, group.names = c("Reference", "Focal"),
Data, group, match, draw.empirical = TRUE)
```

Arguments

`x`	an object of `"Logistic"` class. See **Details**.
`item`	numeric: number of item to be plotted
`item.name`	character: the name of item to be used as title of plot.
`group.names`	character: names of reference and focal group.
`Data`	numeric: the data matrix. See **Details**.
`group`	numeric: the vector of group membership. See **Details**.
`match`	character or numeric: specifies matching criterion. Can be either `"score"`, or numeric vector of the same length as number of observations in `Data`. See **Details**.
`draw.empirical`	logical: whether empirical probabilities should be calculated and plotted. Default value is `TRUE`.

Details

This function plots characteristic curves of 2PL logistic DIF model fitted by `difLogistic()` function from difR package using ggplot2.

`Data` and `group` are used to calculate empirical probabilities for reference and focal group. `match` should be the same as in `x$match`. In case that matching variable is used instead of total score or standardized score, `match` needs to be a numeric vector of the same the same length as number of observations in `Data`.

Author(s)

Adela Hladka
Institute of Computer Science of the Czech Academy of Sciences
Faculty of Mathematics and Physics, Charles University
`<hladka@cs.cas.cz>`

Patricia Martinkova
Institute of Computer Science of the Czech Academy of Sciences
`<martinkova@cs.cas.cz>`

See Also

`difLogistic`, `ggplot`

Examples

```
# loading libraries
library(difR)

# loading data based on GMAT
data(GMAT, package = "difNLR")
Data <- GMAT[, 1:20]
group <- GMAT[, 21]
```

```
# DIF detection using difLogistic() function
x <- difLogistic(Data, group, focal.name = 1)
# Characteristic curve by logistic regression model
plotDIFLogistic(x, item = 1, Data = Data, group = group)

# Using name of column as item identifier
plotDIFLogistic(x, item = "Item1", Data = Data, group = group)

# Renaming reference and focal group
plotDIFLogistic(x, item = 1, group.names = c("Group 1", "Group 2"), Data = Data, group = group)

# Not plotting empirical probabilities
plotDIFLogistic(x, item = 1, draw.empirical = FALSE)
```

`plotDistractorAnalysis` *Function for graphical representation of item distractor analysis*

Description

Plots graphical representation of item distractor analysis with proportions and optional number of groups.

Usage

```
plotDistractorAnalysis(data, key, num.groups = 3, item = 1, item.name,
multiple.answers = TRUE, matching = NULL, match.discrete = FALSE, cut.points)
```

Arguments

`data`	character: data matrix or data frame. See **Details**.
`key`	character: answer key for the items.
`num.groups`	numeric: number of groups to that should be respondents splitted.
`item`	numeric: the number of item to be plotted.
`item.name`	character: the name of item.
`multiple.answers`	logical: should be all combinations plotted (default) or should be answers splitted into distractors. See **Details**.
`matching`	numeric: numeric vector. If not provided, total score is calculated and distractor analysis is performed based on it.
`match.discrete`	logical: is `matching` discrete? Default value is `FALSE`. See details.
`cut.points`	numeric: numeric vector specifying cut points of `matching`. See details.

Details

This function is graphical representation of `DistractorAnalysis` function. In case, no `matching` is provided, the scores are calculated using the item data and key. The respondents are by default splitted into the `num.groups`-quantiles and the proportions of respondents in each quantile are displayed with respect to their answers. In case that `matching` is discrete (`match.discrete = TRUE`), `matching` is splitted based on its unique levels. Other cut points can be specified via `cut.points` argument.

The `data` is a matrix or data frame whose rows represents unscored item response from a multiple-choice test and columns correspond to the items.

The `key` must be a vector of the same length as `ncol(data)`. In case it is not provided, `matching` need to be specified.

If `multiple.answers = TRUE` (default) all reported combinations of answers are plotted. If `multiple.answers = FALSE` all combinations are splitted into distractors and only these are then plotted with correct combination.

Author(s)

Adela Hladka
Institute of Computer Science of the Czech Academy of Sciences
Faculty of Mathematics and Physics, Charles University
`<hladka@cs.cas.cz>`

Patricia Martinkova
Institute of Computer Science of the Czech Academy of Sciences
`<martinkova@cs.cas.cz>`

See Also

`DistractorAnalysis`, `distractor.analysis`

Examples

```
# loading 100-item medical admission test data
data(dataMedical, dataMedicaltest, dataMedicalkey)
data <- dataMedicaltest[, 1:100]
dataBin <- dataMedical[, 1:100]
key <- unlist(dataMedicalkey)

# distractor plot for items 48, 57 and 32 displaying distractors only
plotDistractorAnalysis(data, key, item = 48, multiple.answers = FALSE)
# correct answer B does not function well
plotDistractorAnalysis(data, key, item = 57, multiple.answers = FALSE)
# all options function well, thus the whole item discriminates well
plotDistractorAnalysis(data, key, item = 32, multiple.answers = FALSE)
# functions well, thus the whole item discriminates well
## Not run:
# distractor plot for items 48, 57 and 32 displaying all combinations
```

```
plotDistractorAnalysis(data, key, item = 48)
plotDistractorAnalysis(data, key, item = 57)
plotDistractorAnalysis(data, key, item = 32)

# distractor plot for item 57 with all combinations and 6 groups
plotDistractorAnalysis(data, key, item = 57, num.group = 6)

# distractor plot for item 57 using specified matching and key option
matching <- round(rowSums(dataBin), -1)
plotDistractorAnalysis(data, key, item = 57, matching = matching)
# distractor plot for item 57 using specified matching without key option
plotDistractorAnalysis(data, item = 57, matching = matching)

# distractor plot for item 57 using discrete matching
plotDistractorAnalysis(data, key, item = 57, matching = matching, match.discrete = T)

# distractor plot for item 57 using groups specified by cut.points
plotDistractorAnalysis(data, key, item = 57, cut.points = seq(10, 100, 10))

## End(Not run)
```

`plotMultinomial` *Function for plotting category probabilities of multinomial log-linear regression model*

Description

Plots category probabilities functions estimated by `multinom()` from `nnet` package.

Usage

```
plotMultinomial(x, matching, matching.name = "matching")
```

Arguments

`x`	object of class `multinom`
`matching`	numeric: vector of matching criterion used for estimation in `x`.
`matching.name`	character: name of matching criterion used for estimation in `x`.

Author(s)

Adela Hladka
Institute of Computer Science of the Czech Academy of Sciences
Faculty of Mathematics and Physics, Charles University
`<hladka@cs.cas.cz>`

Tomas Jurica

Patricia Martinkova
Institute of Computer Science of the Czech Academy of Sciences
`<martinkova@cs.cas.cz>`

See Also

`multinom`

Examples

```
# loading data
data(GMAT, GMATtest, GMATkey, package = "difNLR")

matching <- scale(apply(GMAT[, 1:20], 1, sum)) # Z-score
data <- GMATtest[, 1:20]
key <- GMATkey

# multinomial model for item 1
fit <- nnet::multinom(relevel(data[, 1], ref = paste(key[1])) ~ matching)

# plotting category probabilities
plotMultinomial(fit, matching, matching.name = "Z-score")
```

`startShinyItemAnalysis` *This function will start ShinyItemAnalysis application.*

Description

An interactive shiny application for running test and item analysis.

Usage

```
startShinyItemAnalysis()
```

Author(s)

Patricia Martinkova
Institute of Computer Science of the Czech Academy of Sciences
`<martinkova@cs.cas.cz>`

Adela Hladka
Institute of Computer Science of the Czech Academy of Sciences
Faculty of Mathematics and Physics, Charles University
`<hladka@cs.cas.cz>`

Examples

```
## Not run:
rm(list = ls())
startShinyItemAnalysis()

## End(Not run)
```

`theme_app` *Complete theme for* `ShinyItemAnalysis` *graphics*

Description

This complete theme is based on `theme_bw` and it was modified for purposes of `ShinyItemAnalysis`.

Usage

```
theme_app(base_size = 15, base_family = "")
```

Arguments

`base_size`	base font size
`base_family`	base font family

See Also

`ggtheme`

Examples

```
library(ggplot2)
data(GMAT, package = "difNLR")
data <- GMAT[, 1:20]
# total score calculation
df <- data.frame(score = apply(data, 1, sum))
# histogram
g <- ggplot(df, aes(score)) +
  geom_histogram(binwidth = 1) +
  xlab("Total score") +
  ylab("Number of respondents")

g
g + theme_app()
```

Index