Package 'fsia'

July 22, 2025

Type Package

Title Import and Analysis of OMR Data from FormScanner

Version 1.1.1

Date 2017-06-23

Author Michela Battauz

Maintainer Michela Battauz <michela.battauz@uniud.it>

Description Import data of tests and questionnaires from FormScanner. FormScanner is an open source software that converts scanned images to data using optical mark recognition (OMR) and it can be downloaded from <http://sourceforge.net/projects/formscanner/>. The spreadsheet file created by FormScanner is imported in a convenient format to perform the analyses provided by the package. These analyses include the conversion of multiple responses to binary (correct/incorrect) data, the computation of the number of corrected responses for each subject or item, scoring using weights,the computation and the graphical representation of the frequencies of the responses to each item and the report of the responses of a few subjects.

License GPL-3

NeedsCompilation no

Repository CRAN

Date/Publication 2017-06-23 12:55:29 UTC

Contents

`fsia-package` *Import and Analysis of OMR Data from FormScanner*

Description

Import data of tests and questionnaires from FormScanner. FormScanner is an open source software that converts scanned images to data using optical mark recognition (OMR) and it can be downloaded from <http://sourceforge.net/projects/formscanner/>. The spreadsheet file created by FormScanner is imported in a convenient format to perform the analyses provided by the package. These analyses include the conversion of multiple responses to binary (correct/incorrect) data, the computation of the number of corrected responses for each subject or item, scoring using weights, the computation and the graphical representation of the frequencies of the responses to each item and the report of the responses of a few subjects.

Details

Package:	fsia
Type:	Package
Version:	1.1.1
Date:	2017-06-23
License:	GPL-3

Data of questionnaires and tests are often collected on paper forms. FormScanner is an open source software that converts scanned images to data using optical mark recognition (OMR). Function `read.formscanner` of the fsia package can be used to import data from FormScanner in R. The correct response (key) can be specified using function `addkey`. It is also possible to specify weights for each response using function `addweights`. If items have a key, data can be converted to binary variables using function `resp2binary`. In this case, the number of corrected responses for each person and for each item can also be computed by using functions `person.stat` and `item.stat`. These functions can also be used to compute a score using the weights previously specified. Function `freq` calculates the absolute or percentage frequencies of the responses to each item. The frequencies can be printed on screen or plotted on a graph. In both cases, the true responses (if any) are highlighted. The responses given by one or a few subjects can be displayed on a graph by using function `report`. The key is shown on the right and wrong responses can be immediately identified by the red colour.

The package includes two data sets for illustrative purposes. Data sets `test` and `questionnaire` contain the result of importing csv files with function `read.formscanner`. Data set `key` contains the correct responses of the items of the test data sets. Data set `weights` contains the weights of each correct response, while data set `weights_multiple` contains the weights of each response.

Author(s)

Michela Battauz

Maintainer: Michela Battauz <michela.battauz@uniud.it>

References

Borsetta, A. (2017). FormScanner [Computer Software], URL http://sourceforge.net/projects/formscanner/.

Examples

```
# IMPORT DATA FROM FORMSCANNER
# find the directory with package fsia
dir_pkg <- find.package("fsia")
# the example files are in the directory examples
# create the path
questionnaire_path <- file.path(dir_pkg, "examples", "scan_results_questionnaire.csv")
test_path <- file.path(dir_pkg, "examples", "scan_results_test.csv")
# import file "scan_results_questionnaire.csv"
questionnaire_imp<-read.formscanner(questionnaire_path, dummy = "Q5.sources")
questionnaire_imp
# questionnaire_imp is equal to the data questionnaire
# import file "scan_results_test.csv"
test_imp <- read.formscanner(test_path, conc = paste("id", 1:6, sep = ""), id = "id1")
test_imp
# test_imp is equal to the data test

# ADD THE KEY
# create the path for file "key.csv"
key_path <- file.path(dir_pkg, "examples", "key.csv")
# add the key
testk <- addkey(test_imp, keyfile = key_path)
testk$key

# ADD WEIGHTS
# create the path for file "weights.csv"
weights_path <- file.path(dir_pkg, "examples", "weights.csv")
# specify the weights for each correct response
testw <- addweights(testk, weightsfile = weights_path)
testw$weights
# create the path for file "weights_multiple.csv"
weights_mult_path <- file.path(dir_pkg, "examples", "weights_multiple.csv")
# specify the weights for each response
testwm <- addweights(test_imp, weightsfile = weights_mult_path)
testwm$weights

# CONVERT DATA TO BINARY VARIABLES
resp01 <- resp2binary(obj = testk, col = 2:41)
resp01[, 2:5]

# ASSIGN WEIGHTS TO RESPONSES
resps <- resp2scores(obj = testw, col =2:41)
```

```
resps[, 2:5]

# ASSIGN WEIGHTS TO RESPONSES (MULTIPLE WEIGHTS)
resps <- resp2scores(obj = testwm, col =2:41)
resps[, 2:5]

# PERSON STATISTICS (selected only 4 items)
pst <- person.stat(obj = testk, col = 2:5)
pst
pst <- person.stat(obj = testw, col = 2:5, weights = TRUE)
pst
pst <- person.stat(obj = testwm, col = 2:5, weights = TRUE)
pst

# ITEM STATISTICS
ist <- item.stat(obj = testk, col = 2:41)
head(ist)
ist <- item.stat(obj = testw, col = 2:41, weights = TRUE)
head(ist)
ist <- item.stat(obj = testwm, col = 2:41, weights = TRUE)
head(ist)

# FREQUENCIES OF THE RESPONSES
fr <- freq(obj = testk, col = c("Question03", "Question04"))
fr
par(mfrow = c(1, 2))
plot(fr, ask = FALSE)

# RESPONSES OF TWO SUBJECTS
par(mfrow = c(1, 2))
report(obj = testk, col = 2:11, whichid = c("102344", "245784"))
report(obj = testw, col = 2:11, whichid = c("102344", "245784"), weights = TRUE)
par(mfrow = c(1, 1))
report(obj = testwm, col = 2:11, whichid = c("102344", "245784"), weights = TRUE)
```

---

addkey *Add a Key to a Data Frame*

Description

Adds the correct responses (key) to a data frame.

Usage

```
addkey(obj, keyline = NULL, keyfile = NULL, keydata = NULL)
```

Arguments

obj	An object containing the data imported by function read.formscanner.

keyline	the number of line of the data containing the correct responses to the items. This line is deleted from the data and used as key.
keyfile	the name of the file with the correct responses to the items. Column names should match the names of the items.
keydata	an R data frame containing the correct responses to the items. Column names should match the names of the items.

Details

keyfile and keydata can contain more items than obj. In this case, items not contained in obj are dropped in the output.

Value

A list with data, key, and eventually weights.

Author(s)

Michela Battauz

See Also

addweights

Examples

```
data(test)
data(key)
testk <- addkey(test, keydata = key)
```

---

addweights *Add a Weights to a Data Frame*

Description

Adds the weights associated with the responses to the items to a data frame.

Usage

```
addweights(obj, weightsfile = NULL, weightsdata = NULL)
```

Arguments

obj	An object containing the data imported by function read.formscanner.
weightsfile	the name of the file with the weights. Column names should match the names of the items.
weightsdata	an R data frame containing the weights. Column names should match the names of the items.

Details

If `weightsfile` and `weightsdata` have only one row, they should contain the weights that are assigned to the correct responses. These are defined by `addkey`. To specify a different weight to every response of each item, `weightsfile` and `weightsdata` should have one row for each response. In this case there should be a field named `response`.

`weightsfile` and `weightsdata` can contain more items than `obj`. In this case, items not contained in `obj` are dropped in the output.

Value

A list with `data`, eventually `key`, and `weights`.

Author(s)

Michela Battauz

See Also

`addkey`

Examples

```
data(test)
data(key)
data(weights)
data(weights_multiple)
testk <- addkey(test, keydata = key)
testw <- addweights(testk, weightsdata = weights)
testwm <- addweights(test, weightsdata = weights_multiple)
```

---

freq *Absolute and Percentage Frequencies of the Responses to the Items.*

Description

Calculates and plots the absolute or percentage frequencies of the responses to each item.

Usage

```
freq(obj, columns, perc = FALSE)
## S3 method for class 'frlist'
plot(x, display = TRUE, ask = TRUE, ...)
```

Arguments

obj	An object containing the data imported by function read.formscanner and eventually the key added by function addkey.
columns	A vector containing which columns to use. Columns can be specified by name or number.
perc	logical; if TRUE percentage frequencies are calculated.
x	An object of class frlist returned by function freq.
display	logical; if TRUE the frequencies are displayed on the plot.
ask	logical; if TRUE the user is asked for input, before a new figure is drawn.
...	further arguments passed to or from other methods.

Value

Function freq returns an abject of class frlist containing a list with components

item	the name of the item.
tab	an object of class table containing the frequencies.
key	the key of the item.

Author(s)

Michela Battauz

See Also

read.formscanner

Examples

```
data(test)
data(key)
testk <- addkey(test, keydata = key)

fr <- freq(obj = testk, col = c("Question03", "Question04"))
fr
par(mfrow=c(1,2))
plot(fr, ask = FALSE)
fr <- freq(obj = testk, col = 2:11, perc = TRUE)
fr
par(mfrow = c(2,5))
plot(fr, ask = FALSE)
```

item.stat *Item Statistics*

Description

This function computes some statistics for each item.

Usage

```
item.stat(obj, columns, weights = FALSE)
```

Arguments

obj	An object containing the data imported by function read.formscanner, the key added by function addkey and/or weights added by function addweights.
columns	A vector containing which columns to use. Columns can be specified by name or number.
weights	Logical. If TRUE weights are used to compute the score.

Value

A data frame with the following variables.

item	item label.
score	total score for each item. If weights is FALSE score is equal to the number of corrected responses.
max	maximum score for each item.
perc	ratio between score and max.

Author(s)

Michela Battauz

See Also

person.stat

Examples

```
data(test)
data(key)
data(weights)
data(weights_multiple)

testk <- addkey(test, keydata = key)
testw <- addweights(testk, weightsdata = weights)
testwm <- addweights(test, weightsdata = weights_multiple)
```

```
# number of correct responses for each item
ist <- item.stat(obj = testk, col = 2:41)
head(ist)
# sum of weights of correct responses for each item
ist <- item.stat(obj = testw, col = 2:41, weights = TRUE)
head(ist)
# sum of weights of every response for each item
ist <- item.stat(obj = testwm, col = 2:41, weights = TRUE)
head(ist)
```

---

key *Key of Items*

Description

This data set contains the correct responses of the items in the test data set.

Usage

```
data("key")
```

Format

A data frame with variables `Question01` - `Question40`.

Author(s)

Michela Battauz

See Also

addkey

Examples

```
data(key)
key
```

person.stat *Person Statistics*

Description

This function computes some statistics for each person.

Usage

```
person.stat(obj, columns, weights = FALSE)
```

Arguments

obj	An object containing the data imported by function read.formscanner, the key added by function addkey and/or weights added by function addweights.
columns	A vector containing which columns to use. Columns can be specified by name or number.
weights	Logical. If TRUE weights are used to compute the score.

Value

A data frame with the following variables.

id	person label.
score	total score for each person. If weights is FALSE score is equal to the number of corrected responses.
max	maximum score for each person.
perc	ratio between score and max.

Author(s)

Michela Battauz

See Also

person.stat

Examples

```
data(test)
data(key)
data(weights)
data(weights_multiple)

testk <- addkey(test, keydata = key)
testw <- addweights(testk, weightsdata = weights)
testwm <- addweights(test, weightsdata = weights_multiple)
```

```
# number of correct responses for each person (only 4 items)
pst <- person.stat(obj = testk, col = 2:5)
pst
# sum of weights of correct responses for each person
pst <- person.stat(obj = testw, col = 2:5, weights = TRUE)
pst
# sum of weights of every response for each person
pst <- person.stat(obj = testwm, col = 2:5, weights = TRUE)
pst
```

`questionnaire` *Questionnaire Responses*

Description

This data set contains responses to a questionnaire.

Usage

```
data("questionnaire")
```

Format

A data frame with 5 observations on the following 14 variables.

`File.name`	file name.
`Q1.gender`	gender.
`Q2.age`	age.
`Q3.restaurants`	how often eat at restaurants.
`Q4.movies`	watching movies is fun.
`Q4.music`	I like listening to music.
`Q4.reading`	reading is an indispensable part of life.
`Q5.sources`	sources of information.
`Q6.interviewer`	interviewer.
`Q5.sources.internet`	dummy variable for response internet.
`Q5.sources.magazines`	dummy variable for response magazines.
`Q5.sources.newspapers`	dummy variable for response newspapers.
`Q5.sources.radio`	dummy variable for response radio.
`Q5.sources.TV`	dummy variable for response TV.

Details

This data set is obtained by importing file "scan_results_questionnaire.csv" with function `read.formscanner`.

Author(s)

Michela Battauz

See Also

read.formscanner

Examples

```
data(questionnaire)
questionnaire
```

---

read.formscanner *Import Data From FormScanner*

Description

This function imports data from the FormScanner software.

Usage

```
read.formscanner(file, col.names, conc = NULL, id = NULL, dummy = NULL)
```

Arguments

file	the name of the csv file to be imported.
col.names	the names of the columns. If NULL the names of the columns are those given in the csv file.
conc	a vector containing which columns to concatenate. Columns can be specified by name or number.
id	name of the column that uniquely identifies the row.
dummy	a vector containing the columns to convert to dummy variables. Columns can be specified by name or number.

Details

If some columns are concatenated using argument conc, the name of the newly created variable is the name of the first column.

test and questionnaire are instances of the output of function read.formscanner.

When id is obtained as the concatenation of different columns using argument conc, id should be set equal to the first column concatenated.

Value

A data frame.

Author(s)

Michela Battauz

References

Borsetta, A. (2016). FormScanner, [Computer Software], URL http://sourceforge.net/projects/formscanner/.

See Also

addkey, addweights

Examples

```
# find the directory with package fsia
dir_pkg <- find.package("fsia")
# the example files are in the directory examples
# create the path
questionnaire_path <- file.path(dir_pkg, "examples", "scan_results_questionnaire.csv")
test_path <- file.path(dir_pkg, "examples", "scan_results_test.csv")
# import file "scan_results_questionnaire.csv"
questionnaire_imp<-read.formscanner(questionnaire_path, dummy = "Q5.sources")
questionnaire_imp
# questionnaire_imp is equal to the data questionnaire
# import file "scan_results_test.csv"
test_imp <- read.formscanner(test_path, conc = paste("id", 1:6, sep = ""), id = "id1")
test_imp
# test_imp is equal to the data test
```

---

report *Report the Responses*

Description

This function produces a graphic with the responses given by one or a few subjects and shows the correct ones.

Usage

```
report(obj, columns, whichid, grid = TRUE, main = "", las = 0, itemlab = NULL,
  weights = FALSE)
```

Arguments

obj	An object containing the data imported by function read.formscanner, the key added by function addkey and/or weights added by function addweights.
columns	A vector containing which columns to use. Columns can be specified by name or number.
whichid	A vector containing the values of variable id that are shown on the graph.

grid	logical; if TRUE horizontal lines are drown on the graph.
main	an overall title for the plot.
las	numeric in 0,1,2,3; the style of axis labels (see par).
itemlab	labels of the items.
weights	logical. If TRUE the weights are displayed.

Details

Correct responses are colored green, wrong responses are colored red.

Author(s)

Michela Battauz

See Also

read.formscanner

Examples

```
data(test)
data(key)
data(weights)
data(weights_multiple)

testk <- addkey(test, keydata = key)
testw <- addweights(testk, weightsdata = weights)
testwm <- addweights(test, weightsdata = weights_multiple)

par(mfrow = c(1, 2))
report(obj = testk, col = 2:11, whichid = c("102344", "245784"))
report(obj = testw, col = 2:11, whichid = c("102344", "245784"), weights = TRUE)
par(mfrow = c(1, 1))
report(obj = testwm, col = 2:11, whichid = c("102344", "245784"), weights = TRUE)
```

---

resp2binary *Convert Responses to Binary Data*

Description

Converts data to binary (correct/incorrect) responses, according to the key.

Usage

```
resp2binary(obj, columns)
```

Arguments

obj	An object containing the data imported by function read.formscanner and the key added by function addkey.
columns	A vector containing which columns to use. Columns can be specified by name or number.

Value

The data frame data contained in obj with columns replaced by binary data.

Author(s)

Michela Battauz

See Also

read.formscanner

Examples

```
data(test)
data(key)

testk <- addkey(test, keydata = key)

resp01 <- resp2binary(obj = testk, col = 2:41)
resp01
```

---

resp2scores *Convert Responses to Scores*

Description

Assigns a weight to the responses.

Usage

```
resp2scores(obj, columns)
```

Arguments

obj	An object containing the data imported by function read.formscanner and weights added by function addweights.
columns	A vector containing which columns to use. Columns can be specified by name or number.

Value

The data frame `data` contained in `obj` with `columns` replaced by scored responses.

Author(s)

Michela Battauz

See Also

`read.formscanner`

Examples

```
data(test)
data(key)
data(weights)
data(weights_multiple)

testk <- addkey(test, keydata = key)
testw <- addweights(testk, weightsdata = weights)
testwm <- addweights(test, weightsdata = weights_multiple)

# ASSIGN WEIGHTS TO RESPONSES
resps <- resp2scores(obj = testw, col =2:41)
resps[, 2:5]

# ASSIGN WEIGHTS TO RESPONSES (MULTIPLE WEIGHTS)
resps <- resp2scores(obj = testwm, col =2:41)
resps[, 2:5]
```

---

test *Test Responses*

Description

This data set contains multiple choice responses to 40 items.

Usage

```
data("test")
```

Format

A data frame with 5 observations on the following 44 variables.

`File.name`	file name.
`Question01-Question40`	responses given to the items.
`i.course`	course,

i.university	university.
id	identification number.

Details

This data set is obtained by importing file "scan_results_test.csv" with function read.formscanner.

Author(s)

Michela Battauz

See Also

key, read.formscanner, weights, weights_multiple

Examples

```
data(test)
test
```

---

weights *Weights of Items*

Description

This data set contains the weights of the correct responses of the items in the test data set.

Usage

```
data("weights")
```

Format

A data frame with variables Question01 - Question40.

Author(s)

Michela Battauz

See Also

addweights

Examples

```
data(weights)
weights
```

`weights_multiple` *Weights of each Response of the Items*

Description

This data set contains the weights of each responses of the items in the test data set.

Usage

```
data("weights_multiple")
```

Format

A data frame with variables

`response`	responses A, B, C, D.
`Question01-Question40`	weights for each response to each item.

Author(s)

Michela Battauz

See Also

`addweights`

Examples

```
data(weights_multiple)
weights_multiple
```

Index