Package 'LOGAN'

January 20, 2025

Title Log File Analysis in International Large-Scale Assessments

Version 1.0.1

Date 2022-10-24

Description Enables users to handle the dataset cleaning for conducting specific analyses with the log files from two international educational assessments: the Programme for International Student Assessment (PISA, <https://www.oecd.org/pisa/>) and the Programme for the International Assessment of Adult Competencies (PIAAC, <https://www.oecd.org/skills/piaac/>). An illustration of the analyses can be found on the LOGAN Shiny app (<https://loganpackage.shinyapps.io/shiny/>) on your browser.

BugReports https://github.com/derecost/LOGAN/issues

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 3.5)

Imports pander (>= 0.6.1), psych (>= 1.7.8), foreign (>= 0.8-69), dplyr, ggplot2, rlang, stringr, magrittr, modules, methods

Suggests knitr, rmarkdown

RoxygenNote 7.2.1

NeedsCompilation no

Author Denise Reis Costa [aut], Waldir Leoncio [aut, cre]

Maintainer Waldir Leoncio <w.l.netto@medisin.uio.no>

Repository CRAN

Date/Publication 2022-10-25 08:47:56 UTC

Contents

`BoxplotStrategybyPerformance`

Plot: Boxplot PV1Math by Strategy var

Description

Plot: Boxplot PV1Math by Strategy var

Usage

```
BoxplotStrategybyPerformance(
  data,
  strategy.var,
  performance.test,
  ylab.text,
  xlab.text
)
```

Arguments

data	data
strategy.var	strategy.var
performance.test	performance.test
ylab.text	ylab.text
xlab.text	xlab.text

`CleanActions` *Clean events*

Description

This function allows you to clean events in the 'event.type' variable

Usage

```
CleanActions(data, event.type, clear.events)
```

Arguments

data	A `matrix` or `data.frame` where the 'event.type' variable is
event.type	a vector with concatenate events. See `ConcatActions` function.
clear.events	a vector where all the events to be cleaned are listed. Each element of this vector needs to be of a `"event"=""` type.

Value

This function returns a `data.frame` with the "new.event.type" variable that cleaned events from the "event.type" variable.

Examples

```
# Data preparation
df <- cp025q01
df$id <- paste(df[, 1], df[, 2], df[, 3], sep = "-")
df <- m0$TrimVar(df, c("event", "event_type", "diag_state"))
df <- m0$ConcatActions(df, c(rlang::quo(event), rlang::quo(event_type)))

# Function demonstration
df.clean <- m0$CleanActions(df, event_type, c("ACER_EVENT_" = ""))
table(df$event.type)
table(df.clean$new.event.type) # cleaned version
```

ConcatActions *Concatenate events*

Description

This function allows you to concatenate event actions from diferent variables in a unique vector.

Usage

```
ConcatActions(data, concat.events)
```

Arguments

data	A matrix or data.frame where the concatenated events are
concat.events	a vector where all the events are listed. Each element of this vector needs to be of a quo() type.

Details

The output dataset will be identical to the input dataset, except for the addition of one column in the end, called "event.type". Each row of event.type contains the values of concat.events of all the rows.

Value

This function returns a data.frame with the concatenated events in the 'event.type' variable.

Examples

```
# Data preparation
df <- cp025q01
df$id <- paste(df[, 1], df[, 2], df[, 3], sep = "-")
df <- m0$TrimVar(df, c("event", "event_type", "diag_state"))

# Function demonstration
df.conc <- m0$ConcatActions(df, c(rlang::quo(event), rlang::quo(event_type)))
names(df)
names(df.conc) # notice the extra variable in the end
table(df.conc$event.type)
```

cp025q01 *Log file for PISA 2012, CP025, Q01 (selected countries)*

Description

Log file for PISA 2012, CP025, Q01 (selected countries)

cp025q01.treated *Treated log file and microdata for PISA 2012, CP025, Q01 (selected countries)*

Description

Treated log file and microdata for PISA 2012, CP025, Q01 (selected countries)

DataActionsbyID *Wide format dataset with the sequence of actions by ID*

Description

This is a function that translates a long to wide format dataset.

Usage

```
DataActionsbyID(data, id.var, event.var, name.var.action)
```

Arguments

data	A matrix or data.frame where the 'event.type' variable is
id.var	a vector with the individuals identification. It is a quo() type.
event.var	a vector with the cleaned concatenate events. See CleanActions function.
name.var.action	A character string that will name the new variable of events

Value

This function returns a data.frame with the only one entry by individual identification and a new 'action.var' variable.

Examples

```
# Data preparation
df <- cp025q01
df$id <- paste(df[, 1], df[, 2], df[, 3], sep = "-")
df <- m0$TrimVar(df, c("event", "event_type", "diag_state"))
df <- m0$ConcatActions(df, c(rlang::quo(event), rlang::quo(event_type)))
df <- m0$CleanActions(df, event.type, c("ACER_EVENT_" = ""))

# Function demonstration
m0$DataActionsbyID(df, id, new.event.type, "actions")
```

DataArcSinebyPerformance *Data: Percentage in arcsine values x PISA scores by Country*

Description

This is a function that calculates the percentage in arcsine and plots it against the PISA scores

Usage

```
DataArcSinebyPerformance(data, strategy.var, performance.test, country.id)
```

Arguments

`data`	A `matrix` or `data.frame` where the 'strategy.var' and performance variables are
`strategy.var`	A string with the name of the strategy variable. It is "quo()" type.
`performance.test`	A string with the name of the test performance variable. It is "quo()" type.
`country.id`	A string with the name of the countries variable. It is "quo()" type.

Value

This function returns a data frame and a plot

DescriptiveStrategy *Report: Descriptive statistics by strategy*

Description

This is a function that reports a descriptive analysis of the strategy and students performance

Usage

```
DescriptiveStrategy(
  data,
  strategy.var,
  performance.item,
  performance.test,
  PartialCredit = FALSE
)
```

Arguments

data	A matrix or data.frame where the 'strategy.var' and performance variables are
strategy.var	A character string with the name of the strategy variable
performance.item	A character string with the name of the item performance variable
performance.test	A character string with the name of the test performance variable
PartialCredit	Logical. It can be used when the item is partial credit score.

Value

This function returns a report with a descriptive analysis of the strategy and students performance

Examples

```
m2$DescriptiveStrategy(cp025q01.treated, "votat", "CP025Q01", "PV1CPRO")
```

FreqActionsSummary *Frequency of specifics events in a variable of Actions - Summary*

Description

This is a function that locates specific events (using the actions.search argument) and create new variables associate with this strategy.

Usage

```
FreqActionsSummary(data, freqact.var, var)
```

Arguments

data	A matrix or data.frame where the 'action.var' variable is
freqact.var	freqact.var
var	var

Value

This function returns a data.frame with the frequency of each specific events from the actions.search argument and "Freq.Actions.Search" summary.

`ImportSPSS` *Read SPSS process data*

Description

This is a simple function that, by default, reads an SPSS data file and save it as a data frame. It is essentially a wrapper for foreign::read.spss with arguments common to log file datasets.

Usage

```
ImportSPSS(filename)
```

Arguments

`filename`	character string: the name of the file or URL to read.

Value

This function returns a data frame.

`LOGAN` *LOGAN: Log File Analysis in International Large-scale Assessments*

Description

This package enables users to handle the dataset cleaning for conducting specific analyses with the log files from two international educational assessments: the Programme for International Student Assessment (PISA, <http://www.oecd.org/pisa/>) and the Programme for the International Assessment of Adult Competencies (PIAAC, <http://www.oecd.org/skills/piaac/>). An illustration of the analyses can be found on the LOGAN Shiny app (<https://loganpackage.shinyapps.io/shiny/>) on your browser.

LOGAN functions

The LOGAN functions The LOGAN functions are organized in modules, so to call a function you must prefix it with, e.g., 'm0$', where "m0" is the module to which a certain function pertains.

What follows is a list of Functions organized per module:

Module 0:

- CleanActions
- ConcatActions
- DataActionsbyID
- ImportSPSS
- RangeNumberActionsbyVar

- TrimVar

Module 1:

- NumericTimeVar
- PlotTimeonTaskbyVar
- SummaryTOTbyVar
- TOTVar
- VarTimebyID

Module 2:

- DescriptiveStrategy
- PlotStrategybyCatPerformance
- VarActionSearch

Author(s)

- Denise Reis Costa [aut, cre],
- Waldir Leoncio Netto [aut]

`m0` *Module 0: Data preparation*

Description

Module 0: Data preparation

Usage

```
m0
```

Format

An object of class `module` (inherits from `list`) of length 6.

Details

This module contains the following functions, which should be called by issuing "m0$<function_name>()": CleanActions, ConcatActions, DataActionsbyID, ImportSPSS, RangeNumberActionsbyVar, TrimVar

m1 *Module 1: Time*

Description

Module 1: Time

Usage

```
m1
```

Format

An object of class `module` (inherits from `list`) of length 5.

Details

This module contains the following functions, which should be called by issuing "m1$<function_name>()": NumericTimeVar, PlotTimeonTaskbyVar, SummaryTOTbyVar, TOTVar, VarTimebyID

m2 *Module 2: Actions (cognitive related)*

Description

Module 2: Actions (cognitive related)

Usage

```
m2
```

Format

An object of class `module` (inherits from `list`) of length 3.

Details

This module contains the following functions, which should be called by issuing "m2$<function_name>()": DescriptiveStrategy, PlotStrategybyCatPerformance, VarActionSearch.

NumericTimeVar *Time var as a numeric vector*

Description

This is a function that transforms a factor var time in numeric.

Usage

```
NumericTimeVar(data, vector.time)
```

Arguments

data	A matrix or data.frame
vector.time	variable containing the time

Value

This function returns a data.frame with the number of students and number de actions (min-max) aggregated by a specific variable.

Examples

```
vector.time <- c("CP025Q01.END", "CP025Q01.START")
m1$NumericTimeVar(cp025q01.treated, vector.time)
```

pisa *Microdata for PISA 2012 (selected countries)*

Description

Microdata for PISA 2012 (selected countries)

`PlotStrategybyCatPerformance` *Check response time by var*

Description

This is a function that reports the number of students and number of actions (min-max) aggregated by a specific variable.

Usage

```
PlotStrategybyCatPerformance(data, strategy.var, categ.var, namexlab, nameylab)
```

Arguments

`data`	A `matrix` or `data.frame`
`strategy.var`	strategy variable
`categ.var`	categorizing variable
`namexlab`	name of the variable in the x-axis
`nameylab`	name of the variable in the y-axis

Value

This function returns a `data.frame` with the number of students and number de actions (min-max) aggregated by a specific variable.

Examples

```
# Data preparation
df <- cp025q01.treated
df$categ <- cut(df$PV1CPRO, c(0, 423, 488, 553, 900))
df.dataplot <- df[, c("top", "categ")]
df.dataplot[, 1] <- as.factor(df.dataplot[, 1])
df.dataplot[, 2] <- as.factor(df.dataplot[, 2])

# Function demonstration
m2$PlotStrategybyCatPerformance(
  df.dataplot, top, categ,
  "Proficiency levels", "Percentage"
)
```

`PlotTimeonTaskbyVar` *Check response time by var*

Description

This is a function that reports the number of students and number de actions (min-max) aggregated by a specific variable.

Usage

```
PlotTimeonTaskbyVar(
  data,
  tot.var,
  performance.item,
  namexlab,
  nameylab = "Density"
)
```

Arguments

`data`	A `matrix` or `data.frame`
`tot.var`	a vector with the total time. It is a `quo()` type.
`performance.item`	name of the item variable
`namexlab`	name of the plot's x-axis
`nameylab`	name of the plot's y-axis. Defaults to "Density"

Value

This function returns a `data.frame` with the number of students and number de actions (min-max) aggregated by a specific variable.

Examples

```
m1$PlotTimeonTaskbyVar(cp025q01.treated, "CP025Q01.TOT", "CP025Q01",
  namexlab = "Time on task (minutes)"
)
```

RangeNumberActionsbyVar

Check number of students and actions by var

Description

This is a function that reports the number of students and number de actions (min-max) aggregated by a specific variable.

Usage

```
RangeNumberActionsbyVar(data, id.var, var.group, save.table = TRUE)
```

Arguments

data	A matrix or data.frame
id.var	a vector with the individuals identification. It is a quo() type.
var.group	a vector with the group variable. It is a quo() type.
save.table	if TRUE, will save the table generated as an object of class data.frame. Otherwise, will print the table in pandoc format, but the object will not be saved (even if the user assigns it to an object)

Value

This function returns a data.frame with the number of students and number de actions (min-max) aggregated by a specific variable.

Examples

```
m0$RangeNumberActionsbyVar(cp025q01.treated, NewID, CNT, save.table = FALSE)
```

RangeTimeonTaskbyVar *Check response time by var*

Description

This is a function that reports the number of students and number de actions (min-max) aggregated by a specific variable.

Usage

```
RangeTimeonTaskbyVar(data, tot.var, var.group)
```

Arguments

data	A matrix or data.frame
tot.var	a vector with the total time. It is a quo() type.
var.group	a vector with the group variable. It is a quo() type.

Value

This function returns a data.frame with the number of students and number de actions (min-max) aggregated by a specific variable.

ScatterPlotbyPerformance

Plot: Percentage in arcsine values x PISA scores by Country

Description

This is a function that calculates the percentage in arcsine and plots it against the PISA scores

Usage

```
ScatterPlotbyPerformance(
  data,
  strategy.summary,
  performance.mean,
  country.id,
  ylab.text,
  xlab.text,
  ylim.vector,
  xlim.vector
)
```

Arguments

data	A matrix or data.frame where the 'strategy.var' and performance variables are
strategy.summary	strategy.summary
performance.mean	performance.mean
country.id	A string with the name of the countries variable. It is "quo()" type.
ylab.text	A character string giving the text of the y-axis in the plot
xlab.text	A character string giving the text of the x-axis in the plot
ylim.vector	A numeric vector with the limits of the y-axis in the plot
xlim.vector	A numeric vector with the limits of the x-axis in the plot

Value

This function returns a data frame and a plot

SummaryTOTbyVar *Summary of time on task by var*

Description

This is a function that reports the number of students and a summary of time on task aggregated by a specific variable.

Usage

```
SummaryTOTbyVar(data, tot.var, performance.item, na.rm = FALSE)
```

Arguments

data	A matrix or data.frame
tot.var	a vector with the time on task.
performance.item	a vector with the group variable. It is a quo() type.
na.rm	remove missing data in 'performance.item'? Default is 'FALSE'

Value

This function returns a data.frame with the number of students and number de actions (min-max) aggregated by a specific variable.

Examples

```
m1$SummaryTOTbyVar(cp025q01.treated, "CP025Q01.TOT", "CP025Q01", TRUE)
```

TOTVar *Time on task variable*

Description

This is a function that reports the number of students and a summary of time on task aggregated by a specific variable.

Usage

```
TOTVar(data, starttime.vec, endtime.vec, divBy = NA, tot.var)
```

Arguments

data	A matrix or data.frame
starttime.vec	a vector with the individuals' identifications. It is a quo() type.
endtime.vec	a vector with the group variable. It is a quo() type.
divBy	a vector with the group variable. It is a quo() type.
tot.var	string containing the name of the output variable

Value

This function returns a data.frame with the number of students and number de actions (min-max) aggregated by a specific variable.

Examples

```
m1$TOTVar(cp025q01.treated, "CP025Q01.START", "CP025Q01.END",
  divBy = 60,
  tot.var = "CP025Q01.TOT"
)
```

TrimVar *Trim variables*

Description

TrimVar() is a function that allows you to remove whitespace inside the strings of a vector.

Usage

```
TrimVar(data, trim.vector)
```

Arguments

data	dataset
trim.vector	vector of variables on the dataset to be trimmed

Value

This function returns a vector removing trailing and leading spaces inside the original vector.

Examples

```
head(m0$TrimVar(cp025q01, "event"))
```

VarActionPosition *Identify the position of specific events in a variable of Actions*

Description

This is a function that locates specific events (using the `actions.search` argument) and create new variables associate with this strategy.

Usage

```
VarActionPosition(data, action.var, actions.search)
```

Arguments

`data`	A `matrix` or `data.frame` where the 'action.var' variable is
`action.var`	a vector with actions. See `DataActionsbyID` function.
`actions.search`	A character vector with the actions to be searched.

Value

This function returns a `data.frame` with the frequency of each specific events from the `actions.search` argument and "Freq.Actions.Search" summary.

VarActionSearch *Frequency of specifics events in a variable of Actions*

Description

This is a function that locates specific events (using the `actions.search` argument) and create new variables associate with this strategy.

Usage

```
VarActionSearch(data, action.var, actions.search)
```

Arguments

`data`	A `matrix` or `data.frame` where the 'action.var' variable is
`action.var`	a vector with actions. See `DataActionsbyID` function.
`actions.search`	A character vector with the actions to be searched.

Value

This function returns a `data.frame` with the frequency of each specific events from the `actions.search` argument and "Freq.Actions.Search" summary.

Examples

```
# Counting the instances of top_setting == 1
df <- m2$VarActionSearch(cp025q01.treated, "CP025Q01.ACTIONS", "1_apply")
table(df$freq.1_apply) # checking results
```

`VarTimebyID` *Extracting the start or end time*

Description

Extracting the start or end time

Usage

```
VarTimebyID(data, id.var, time.var, event.var, name.var.time, new.name)
```

Arguments

`data`	data frame
`id.var`	vector of unique identification
`time.var`	vector with the time variable
`event.var`	vector with the events
`name.var.time`	name of the time string to filter (ex.: "START_ITEM" or "END_ITEM")
`new.name`	name of the output variable

Value

a data frame with ‘time‘ replaced with ‘new.name‘. The variable ‘event.var‘ is dropped.

Examples

```
# Data preparation
df <- cp025q01
df$id <- paste(df[, 1], df[, 2], df[, 3], sep = "-")
df <- m0$TrimVar(df, c("event", "event_type", "diag_state"))
df <- m0$ConcatActions(df, c(rlang::quo(event), rlang::quo(event_type)))
df <- m0$CleanActions(df, event.type, c("ACER_EVENT_" = ""))

# Function demonstration
m1$VarTimebyID(df, id, time, new.event.type, "START_ITEM", "start")
```

Index