Package 'datacheck'

August 29, 2016

Type Package

Title Tools for Checking Data Consistency

Version 1.2.2

Date 2015-04-16

Maintainer ``Simon, Reinhard (CIP)" <R.SIMON@CGIAR.ORG>

Author ``Simon, Reinhard (CIP)" <R.SIMON@CGIAR.ORG>, Jose Francisco Loff <jfloff at phistat.com>

Copyright International Potato Center (CIP)

License MIT + file LICENSE

URL https://github.com/c5sire/datacheck

BugReports https://github.com/c5sire/datacheck/issues

Description Functions to check variables against a
set of data quality rules. A rule file can be accompanied by look-up tables. In
addition, there are some convenience functions that may
serve as an example for defining clearer 'data rules'. An
HTML based user interface facilitates initial exploration of the
functionality.

LazyData yes

Encoding UTF-8

Imports stringr, Hmisc, shiny

Depends R (>= 3.1.0), grDevices

Repository CRAN

NeedsCompilation no

Date/Publication 2015-04-18 08:56:10

R topics documented:

`datacheck-package` *Check a table against a set of constraints or rules defined in R.*

Description

The rules can be written in standard R syntax. A rule must contain the names of 'columns' or variables present in the table and use R operators or simple functions. If not, the rule will simply be ignored. Each line must 'test' one rule and return a vector of boolean values as many as the table has rows. Rules must not contain an assignment. The set of rules is simply defined as a set of R statements and can be mixed with empty lines and comments. Comments after a rule will be used for summarizing rule check results in a table and should therefore be short - usually short names. This allows to visually organize rules in a file and also document them. One may put more extensive comments just before the rule and add a short name or comment on the same line after it. This also allows to use standard R editors for development of the rules.

Details

A simple score is calculated based on the number of rules a datapoint (= table cell) complies with. Like in a school test only the number of correct answers (or rule compliances) are counted. Summaries of scores by row (record) and column (variable) are added to a score data frame.

The table itself must be a simple dataframe or .csv file.

The package includes a simple graphical user interface as a web page. This can be started with `run_datacheck()`. This interface shows summaries of the checks by rule and by record. The score table can be 'downloaded'. The user interface is meant as an easy way to get to know the package. All results can be also created using the command line interface of R.

The main function and the principal example can be found under `datadict_profile`.

Several helper functions like `is_proper_name` or `is_only_lowers` are for convenience and illustration on how to express rules more clearly or succinct.

`as.rules` *Converts a vector of lines into a rules data frame*

Description

The rules must be one per line and should evaluate to a vector of TRUE or FALSE.

Usage

```
as.rules(lines = "")
```

Arguments

`lines`	R statements with conditions

Details

A rule must only refer to one 'column' name. Rule statements may not have assignment operators (= or <-). Rules may be separated by empty or commented lines. A comment after a rule is used to document the specific rule in the summary table.

Value

The results as a datadict object or NA for 'empty' rule lines

Author(s)

Reinhard Simon

See Also

Other datadict: `as_rules`; `datadict.profile`; `datadict_profile`; `has.ruleErrors`; `has_rule_errors`; `is.datadict.profile`; `is_datadict_profile`; `prep4rep`; `read.rules`; `read_rules`

as_rules *Converts a vector of lines into a rules data frame*

Description

The rules must be one per line and should evaluate to a vector of TRUE or FALSE.

Usage

```
as_rules(lines = "")
```

Arguments

lines	R statements with conditions

Details

A rule must only refer to one 'column' name. Rule statements may not have assignment operators (= or <-). Rules may be separated by empty or commented lines. A comment after a rule is used to document the specific rule in the summary table.

Value

The results as a datadict object or NA for 'empty' rule lines

Author(s)

Reinhard Simon

See Also

Other datadict: as.rules; datadict.profile; datadict_profile; has.ruleErrors; has_rule_errors; is.datadict.profile; is_datadict_profile; prep4rep; read.rules; read_rules

datadict.profile *Create a data quality profile (main function)*

Description

Tests a database against a set of rules (one per line) in a 'data dictionary file'. Rules will be summarized in the returned object: the variable/column, the rule, any comment after the rule, the execution success, the total number of rule violations if any, the record id for any non-compliant records. Rules that can't be executed for any reason will be marked as 'failed'.

Usage

```
datadict.profile(atable, adictionary)
```

Arguments

atable	a data.frame
adictionary	a list of rules in rule format

Details

The rule file must be a simple list of one rule per line. Functions can be used but since they are applied on a 'vector' (the column) they should be used within a sapply statement (see example rule file). Rules may be separated by empty lines or lines with comment character #. Comments after a rule within the same line will be used for display in the summary table and should be short. A rule must only test one variable and one aspect at a time.

Value

a data.profile object or NA

Author(s)

Reinhard Simon

See Also

Other datadict: as.rules; as_rules; datadict_profile; has.ruleErrors; has_rule_errors; is.datadict.profile; is_datadict_profile; prep4rep; read.rules; read_rules

Examples

```
library(stringr)
# Get example data files
atable <- system.file("examples/db.csv", package = "datacheck")
arule <- system.file("examples/rules1.R", package = "datacheck")
aloctn <- system.file("examples/location.csv", package = "datacheck")  # for use in is.oneOf

ctable <- basename(atable)
crule <- basename(arule)
cloctn <- basename(aloctn)

cwd <- tempdir()
owd <- getwd()
setwd(cwd)

file.copy(atable, ctable)
file.copy(arule, crule)
file.copy(aloctn, cloctn)

at <- read.csv(ctable, stringsAsFactors = FALSE)
ad <- read_rules(crule)

db <- datadict_profile(at, ad)

is_datadict_profile(db) == TRUE
```

```
db

setwd(owd)
```

`datadict_profile` *Create a data quality profile (main function)*

Description

Tests a database against a set of rules (one per line) in a 'data dictionary file'. Rules will be summarized in the returned object: the variable/column, the rule, any comment after the rule, the execution success, the total number of rule violations if any, the record id for any non-compliant records. Rules that can't be executed for any reason will be marked as 'failed'.

Usage

```
datadict_profile(atable, adictionary)
```

Arguments

`atable`	a data.frame
`adictionary`	a list of rules in rule format

Details

The rule file must be a simple list of one rule per line. Functions can be used but since they are applied on a 'vector' (the column) they should be used within a sapply statement (see example rule file). Rules may be separated by empty lines or lines with comment character #. Comments after a rule within the same line will be used for display in the summary table and should be short. A rule must only test one variable and one aspect at a time.

Value

a data.profile object or NA

Author(s)

Reinhard Simon

See Also

Other datadict: `as.rules`; `as_rules`; `datadict.profile`; `has.ruleErrors`; `has_rule_errors`; `is.datadict.profile`; `is_datadict_profile`; `prep4rep`; `read.rules`; `read_rules`

Examples

```
library(stringr)
# Get example data files
atable <- system.file("examples/db.csv", package = "datacheck")
arule <- system.file("examples/rules1.R", package = "datacheck")
aloctn <- system.file("examples/location.csv", package = "datacheck")  # for use in is.oneOf

ctable <- basename(atable)
crule <- basename(arule)
cloctn <- basename(aloctn)

cwd <- tempdir()
owd <- getwd()
setwd(cwd)

file.copy(atable, ctable)
file.copy(arule, crule)
file.copy(aloctn, cloctn)

at <- read.csv(ctable, stringsAsFactors = FALSE)
ad <- read_rules(crule)

db <- datadict_profile(at, ad)

is_datadict_profile(db) == TRUE

db

setwd(owd)
```

---

has.punct *Tests for presence of most common punctuation characters*

Description

These include: .?'!_#$^&*()+=<>,;' - and [tab] and [at]

Usage

```
has.punct(s)
```

Arguments

s	a character string

Value

boolean TRUE if detects anything

Author(s)

Reinhard Simon

See Also

Other rule_checks: has_punct; is.oneOf; is.onlyLowers; is.properName; is.withinRange; is_one_of; is_only_lowers; is_proper_name; is_within_range

Examples

```
has_punct(".") == TRUE
```

---

has.ruleErrors *Quick check if a rule profile on a table has any errors.*

Description

Quick check if a rule profile on a table has any errors.

Usage

```
has.ruleErrors(profile.rules)
```

Arguments

profile.rules	a data.profile object

Value

boolean

Author(s)

Reinhard Simon

See Also

Other datadict: as.rules; as_rules; datadict.profile; datadict_profile; has_rule_errors; is.datadict.profile; is_datadict_profile; prep4rep; read.rules; read_rules

Examples

```
# Get example data file with some errors in it
atbler <- system.file("examples/db-err.csv", package = "datacheck")
arule <- system.file("examples/rules1.R", package = "datacheck")

at <- read.csv(atbler, stringsAsFactors = FALSE)
ad <- read_rules(arule)

db_e <- datadict_profile(at, ad)

has_rule_errors(db_e) == TRUE
```

---

has_punct *Tests for presence of most common punctuation characters*

Description

These include: .?'!_#$^&*()+=<>,;' - and [tab] and [at]

Usage

```
has_punct(s)
```

Arguments

s	a character string

Value

boolean TRUE if detects anything

Author(s)

Reinhard Simon

See Also

Other rule_checks: has.punct; is.oneOf; is.onlyLowers; is.properName; is.withinRange; is_one_of; is_only_lowers; is_proper_name; is_within_range

Examples

```
has_punct(".") == TRUE
```

has_rule_errors *Quick check if a rule profile on a table has any errors.*

Description

Quick check if a rule profile on a table has any errors.

Usage

```
has_rule_errors(profile_rules)
```

Arguments

profile_rules	a data.profile object

Value

boolean

Author(s)

Reinhard Simon

See Also

Other datadict: as.rules; as_rules; datadict.profile; datadict_profile; has.ruleErrors; is.datadict.profile; is_datadict_profile; prep4rep; read.rules; read_rules

Examples

```
# Get example data file with some errors in it
atbler <- system.file("examples/db-err.csv", package = "datacheck")
arule <- system.file("examples/rules1.R", package = "datacheck")

at <- read.csv(atbler, stringsAsFactors = FALSE)
ad <- read_rules(arule)

db_e <- datadict_profile(at, ad)

has_rule_errors(db_e) == TRUE
```

heatmap.quality *Draws a heatmap based on data quality scores*

Description

Knows to extract the quality matrix from the profile object and pass it on to the heatmap function. Plots a heatmap.

Usage

```
heatmap.quality(profile, recLab = NULL, recMax = 100, scoreMax = NULL,
  cols = NULL, ...)
```

Arguments

profile	a datadict.profile object
recLab	variable that should be used for labeling the records
recMax	maximum first n records for display
scoreMax	maxim quality score to filter out
cols	color scheme
...	as in heatmap function

Details

Currently this function is limited a table siZé of 300 records.

Author(s)

Reinhard Simon

See Also

Other visuals: heatmap_quality; ruleCoverage; rule_coverage; scoreSum; score_sum

heatmap_quality *Draws a heatmap based on data quality scores*

Description

Knows to extract the quality matrix from the profile object and pass it on to the heatmap function. Plots a heatmap.

Usage

```
heatmap_quality(profile, recLab = NULL, recMax = 100, scoreMax = NULL,
  cols = NULL, ...)
```

Arguments

profile	a datadict.profile object
recLab	variable that should be used for labeling the records
recMax	maximum first n records for display
scoreMax	maxim quality score to filter out
cols	color scheme
...	as in heatmap function

Details

Currently this function is limited a table siZé of 300 records.

Author(s)

Reinhard Simon

See Also

Other visuals: heatmap.quality; ruleCoverage; rule_coverage; scoreSum; score_sum

---

is.datadict.profile *is.datadict.profile*

Description

Is this a datadict.profile object

Usage

```
is.datadict.profile(x)
```

Arguments

x	The object to be tested.

Value

boolean

Author(s)

Reinhard Simon

See Also

Other datadict: as.rules; as_rules; datadict.profile; datadict_profile; has.ruleErrors; has_rule_errors; is_datadict_profile; prep4rep; read.rules; read_rules

Examples

```
library(stringr)
# Get example data files
atable <- system.file("examples/db.csv", package = "datacheck")
arule <- system.file("examples/rules1.R", package = "datacheck")
aloctn <- system.file("examples/location.csv", package = "datacheck")  # for use in is.oneOf

ctable <- basename(atable)
crule <- basename(arule)
cloctn <- basename(aloctn)

cwd <- tempdir()
owd <- getwd()
setwd(cwd)

file.copy(atable, ctable)
file.copy(arule, crule)
file.copy(aloctn, cloctn)

at <- read.csv(ctable, stringsAsFactors = FALSE)
ad <- read_rules(crule)

db <- datadict_profile(at, ad)

is_datadict_profile(db) == TRUE

db

setwd(owd)
```

---

is.oneOf *Tests if a string or 'factor level' is one of a pre-defined set*

Description

The aset parameter may point to a file with level names. This is useful if there are many levels like in database of world countries. The file path may be an absolute one or relative to the current working directory.

Usage

```
is.oneOf(x, aset)
```

Arguments

x	a factor level as character string
aset	a vector of character strings or a path to a custom file (full pathname where necessary)

Details

The supporting table must have two columns named 'VALUES' and 'LABELS'. The lookup file must be in comma separated format and using the '.csv' extension. It must also be encoded using UTF-8 character set for being able to use foreign characters across operating systems. This is often an issue when using Excel to develop the file.

The x parameter may have just one level or multiple levels separated by ';'. Likewise the aset parameter may have just one level or multiple levels separated by ';'. In any case the x parameter must be a subset of aset (or the lookup file): see the example section.

Value

boolean TRUE if detects anything

Author(s)

Reinhard Simon, Jose Francisco Loff

See Also

Other rule_checks: has.punct; has_punct; is.onlyLowers; is.properName; is.withinRange; is_one_of; is_only_lowers; is_proper_name; is_within_range

Examples

```
# Case 1: define the reference set or lookup set within the function. Useful for small or binary
# sets like m(ale)/f(emale)
is_one_of("m", "m") == TRUE

is_one_of("m", c("f", "m")) == TRUE

is_one_of("y", c("f", "m")) == FALSE

is_one_of(c("b", "c", "d"), c("a", "b", "c", "d", "e")) == TRUE


# Case 2: use an external lookup table. The external lookup table must have at least one column
# called exactly 'VALUES'. May have also another one 'LABELS'. Useful for long lookup tables like
# list of countries.

# some preparation work for using a temporary directory
owd <- getwd()
td <- tempdir()
setwd(td)
```

```
VALUES <- LETTERS[1:10]
LABELS <- VALUES
db <- cbind(VALUES, LABELS)
db <- as.data.frame(db, stringsAsFactors = FALSE)
names(db) <- c("VALUES", "LABELS")
write.csv(db, "sample.csv", row.names = FALSE)

is_one_of("A", "sample.csv") == TRUE
is_one_of("Z", "sample.csv") == FALSE

# switching back to your working directory
setwd(owd)
```

`is.onlyLowers` *Tests if a string has only lower case letters*

Description

Tests if a string has only lower case letters

Usage

```
is.onlyLowers(s)
```

Arguments

s	a character string

Value

boolean TRUE if detects anything

Author(s)

Reinhard Simon

See Also

Other rule_checks: `has.punct`; `has_punct`; `is.oneOf`; `is.properName`; `is.withinRange`; `is_one_of`; `is_only_lowers`; `is_proper_name`; `is_within_range`

Examples

```
is_only_lowers("example") == TRUE

is_only_lowers("Example") == FALSE
```

`is.properName` *Tests if string is like a proper name with inital letter in upper case*

Description

Tests if string is like a proper name with inital letter in upper case

Usage

```
is.properName(aname)
```

Arguments

`aname`	a character string

Value

boolean TRUE if ok

Author(s)

Reinhard Simon, Jose Francisco Loff

See Also

Other rule_checks: `has.punct`; `has_punct`; `is.oneOf`; `is.onlyLowers`; `is.withinRange`; `is_one_of`; `is_only_lowers`; `is_proper_name`; `is_within_range`

Examples

```
# Valid proper names

is_proper_name("John") == TRUE

is_proper_name("john") == FALSE

is_proper_name(123) == FALSE
```

`is.withinRange` *Tests if a numeric value is between a minimal and maximum value. Serves as convenience function.*

Description

Tests if a numeric value is between a minimal and maximum value. Serves as convenience function.

Usage

```
is.withinRange(val, min, max)
```

Arguments

`val`	The value to be checked
`min`	The minimal value (inclusive)
`max`	The maximum value (inclusive)

Value

boolean TRUE if detects anything

Author(s)

Reinhard Simon

See Also

Other rule_checks: `has.punct`; `has_punct`; `is.oneOf`; `is.onlyLowers`; `is.properName`; `is_one_of`; `is_only_lowers`; `is_proper_name`; `is_within_range`

Examples

```
is_within_range(1, 0, 2) == TRUE

is_within_range(-1, 0, 2) == FALSE
```

`is_datadict_profile` *is.datadict.profile*

Description

Is this a datadict.profile object

Usage

```
is_datadict_profile(x)
```

Arguments

x	The object to be tested.

Value

boolean

Author(s)

Reinhard Simon

See Also

Other datadict: as.rules; as_rules; datadict.profile; datadict_profile; has.ruleErrors; has_rule_errors; is.datadict.profile; prep4rep; read.rules; read_rules

Examples

```
library(stringr)
# Get example data files
atable <- system.file("examples/db.csv", package = "datacheck")
arule <- system.file("examples/rules1.R", package = "datacheck")
aloctn <- system.file("examples/location.csv", package = "datacheck")  # for use in is.oneOf

ctable <- basename(atable)
crule <- basename(arule)
cloctn <- basename(aloctn)

cwd <- tempdir()
owd <- getwd()
setwd(cwd)

file.copy(atable, ctable)
file.copy(arule, crule)
file.copy(aloctn, cloctn)

at <- read.csv(ctable, stringsAsFactors = FALSE)
```

```
ad <- read_rules(crule)

db <- datadict_profile(at, ad)

is_datadict_profile(db) == TRUE

db

setwd(owd)
```

`is_one_of` *Tests if a string or 'factor level' is one of a pre-defined set*

Description

The aset parameter may point to a file with level names. This is useful if there are many levels like in database of world countries. The file path may be an absolute one or relative to the current working directory.

Usage

```
is_one_of(x, aset)
```

Arguments

`x`	a factor level as character string
`aset`	a vector of character strings or a path to a custom file (full pathname where necessary)

Details

The supporting table must have two columns named 'VALUES' and 'LABELS'. The lookup file must be in comma separated format and using the '.csv' extension. It must also be encoded using UTF-8 character set for being able to use foreign characters across operating systems. This is often an issue when using Excel to develop the file.

The x parameter may have just one level or multiple levels separated by ';'. Likewise the aset parameter may have just one level or multiple levels separated by ';'. In any case the x parameter must be a subset of aset (or the lookup file): see the example section.

Value

boolean TRUE if detects anything

Author(s)

Reinhard Simon, Jose Francisco Loff

See Also

Other rule_checks: has.punct; has_punct; is.oneOf; is.onlyLowers; is.properName; is.withinRange; is_only_lowers; is_proper_name; is_within_range

Examples

```
# Case 1: define the reference set or lookup set within the function. Useful for small or binary
# sets like m(ale)/f(emale)
is_one_of("m", "m") == TRUE

is_one_of("m", c("f", "m")) == TRUE

is_one_of("y", c("f", "m")) == FALSE

is_one_of(c("b", "c", "d"), c("a", "b", "c", "d", "e")) == TRUE


# Case 2: use an external lookup table. The external lookup table must have at least one column
# called exactly 'VALUES'. May have also another one 'LABELS'. Useful for long lookup tables like
# list of countries.

# some preparation work for using a temporary directory
owd <- getwd()
td <- tempdir()
setwd(td)


VALUES <- LETTERS[1:10]
LABELS <- VALUES
db <- cbind(VALUES, LABELS)
db <- as.data.frame(db, stringsAsFactors = FALSE)
names(db) <- c("VALUES", "LABELS")
write.csv(db, "sample.csv", row.names = FALSE)

is_one_of("A", "sample.csv") == TRUE
is_one_of("Z", "sample.csv") == FALSE

# switching back to your working directory
setwd(owd)
```

---

is_only_lowers *Tests if a string has only lower case letters*

Description

Tests if a string has only lower case letters

Usage

```
is_only_lowers(s)
```

Arguments

s	a character string

Value

boolean TRUE if detects anything

Author(s)

Reinhard Simon

See Also

Other rule_checks: has.punct; has_punct; is.oneOf; is.onlyLowers; is.properName; is.withinRange; is_one_of; is_proper_name; is_within_range

Examples

```
is_only_lowers("example") == TRUE

is_only_lowers("Example") == FALSE
```

---

is_proper_name *Tests if string is like a proper name with inital letter in upper case*

Description

Tests if string is like a proper name with inital letter in upper case

Usage

```
is_proper_name(aname)
```

Arguments

aname	a character string

Value

boolean TRUE if ok

Author(s)

Reinhard Simon, Jose Francisco Loff

See Also

Other rule_checks: has.punct; has_punct; is.oneOf; is.onlyLowers; is.properName; is.withinRange; is_one_of; is_only_lowers; is_within_range

Examples

```
# Valid proper names

is_proper_name("John") == TRUE

is_proper_name("john") == FALSE

is_proper_name(123) == FALSE
```

---

is_within_range *Tests if a numeric value is between a minimal and maximum value. Serves as convenience function.*

Description

Tests if a numeric value is between a minimal and maximum value. Serves as convenience function.

Usage

```
is_within_range(val, min, max)
```

Arguments

val	The value to be checked
min	The minimal value (inclusive)
max	The maximum value (inclusive)

Value

boolean TRUE if detects anything

Author(s)

Reinhard Simon

See Also

Other rule_checks: has.punct; has_punct; is.oneOf; is.onlyLowers; is.properName; is.withinRange; is_one_of; is_only_lowers; is_proper_name

Examples

```
is_within_range(1, 0, 2) == TRUE

is_within_range(-1, 0, 2) == FALSE
```

---

pkg.version *Get the current version of a package*

Description

Uses the citation() function.

Usage

```
pkg.version(pkg)
```

Arguments

pkg	the package name

Value

a string with the package number

Author(s)

Reinhard Simon

See Also

Other helper: pkg_version; shortSummary; short_summary

---

pkg_version *Get the current version of a package*

Description

Uses the citation() function.

Usage

```
pkg_version(pkg)
```

Arguments

pkg	the package name

Value

a string with the package number

Author(s)

Reinhard Simon

See Also

Other helper: pkg.version; shortSummary; short_summary

`prep4rep` *Prepares a summary table for display in a 'printed' report.*

Description

Currently reduces the number of displayed record ids to 5 and adds a referral.

Usage

```
prep4rep(rule.checks, txt = "... more")
```

Arguments

`rule.checks`	table in a data.profile object
`txt`	text to be added after the first 5 record ids

Value

the modified rule.checks table

Author(s)

Reinhard Simon

See Also

Other datadict: as.rules; as_rules; datadict.profile; datadict_profile; has.ruleErrors; has_rule_errors; is.datadict.profile; is_datadict_profile; read.rules; read_rules

read.rules *Reads a file containing rules in data dictionary format.*

Description

The rules must be one per line and should evaluate to a vector of TRUE or FALSE.

Usage

```
read.rules(file = "")
```

Arguments

file	R file with conditions

Details

A rule must only refer to one 'column' name. Rule statements may not have assignment operators (= or <-). Rules may be separated by empty or commented lines. A comment after a rule is used to document the specific rule in the summary table.

Value

The results as a datadict object or NA for 'empty' rules file

Author(s)

Reinhard Simon

See Also

Other datadict: as.rules; as_rules; datadict.profile; datadict_profile; has.ruleErrors; has_rule_errors; is.datadict.profile; is_datadict_profile; prep4rep; read_rules

read_rules *Reads a file containing rules in data dictionary format.*

Description

The rules must be one per line and should evaluate to a vector of TRUE or FALSE.

Usage

```
read_rules(file = "")
```

Arguments

file	R file with conditions

Details

A rule must only refer to one 'column' name. Rule statements may not have assignment operators (= or <-). Rules may be separated by empty or commented lines. A comment after a rule is used to document the specific rule in the summary table.

Value

The results as a datadict object or NA for 'empty' rules file

Author(s)

Reinhard Simon

See Also

Other datadict: as.rules; as_rules; datadict.profile; datadict_profile; has.ruleErrors; has_rule_errors; is.datadict.profile; is_datadict_profile; prep4rep; read.rules

---

ruleCoverage *Dotchart of rules per variable*

Description

Summarizes rule coverage. There should be at least 3x coverage.

Usage

```
ruleCoverage(profile, rLowest = 1, rLow = 2, rOk = 3, rMax = 10)
```

Arguments

profile	a datadict profile object
rLowest	lowest acceptable number of rules per variable
rLow	an intermediate low number of rules per variable
rOk	the 'ok' number of rules per variable
rMax	= the maximum number of rules per variable

Author(s)

Reinhard Simon

See Also

Other visuals: heatmap.quality; heatmap_quality; rule_coverage; scoreSum; score_sum

rule_coverage *Dotchart of rules per variable*

Description

Summarizes rule coverage. There should be at least 3x coverage.

Usage

```
rule_coverage(profile, rLowest = 1, rLow = 2, rOk = 3, rMax = 10)
```

Arguments

profile	a datadict profile object
rLowest	lowest acceptable number of rules per variable
rLow	an intermediate low number of rules per variable
rOk	the 'ok' number of rules per variable
rMax	= the maximum number of rules per variable

Author(s)

Reinhard Simon

See Also

Other visuals: heatmap.quality; heatmap_quality; ruleCoverage; scoreSum; score_sum

runDatacheck *Presents the packages graphical user interface*

Description

Runs a web server to show the user interface.

Usage

```
runDatacheck(port = 1971L)
```

Arguments

port	the port where to listen; 1971 by default.

Author(s)

Reinhard Simon

See Also

Other interface: run_datacheck

`run_datacheck` *Presents the packages graphical user interface*

Description

Runs a web server to show the user interface.

Usage

```
run_datacheck(port = 1971L)
```

Arguments

`port`	the port where to listen; 1971 by default.

Author(s)

Reinhard Simon

See Also

Other interface: runDatacheck

`scoreSum` *Line chart of cumulative sum of rule scores.*

Description

A record receives one point per rule which evaluates TRUE. The total number of points is the 'quality score' per record.

Usage

```
scoreSum(profile)
```

Arguments

`profile`	a datadict profile object

Author(s)

Reinhard Simon

See Also

Other visuals: heatmap.quality; heatmap_quality; ruleCoverage; rule_coverage; score_sum

score_sum *Line chart of cumulative sum of rule scores.*

Description

A record receives one point per rule which evaluates TRUE. The total number of points is the 'quality score' per record.

Usage

```
score_sum(profile)
```

Arguments

profile	a datadict profile object

Author(s)

Reinhard Simon

See Also

Other visuals: heatmap.quality; heatmap_quality; ruleCoverage; rule_coverage; scoreSum

shortSummary *Produces a tabular summary of descriptive statistics using the 'Hmisc::describe' function from the Hmisc package.*

Description

Returns a dataframe of descriptive statistics

Usage

```
shortSummary(atable)
```

Arguments

atable	a data frame

Author(s)

Reinhard Simon

See Also

Other helper: pkg.version; pkg_version; short_summary

short_summary *Produces a tabular summary of descriptive statistics using the 'Hmisc::describe' function from the Hmisc package.*

Description

Returns a dataframe of descriptive statistics

Usage

```
short_summary(atable)
```

Arguments

atable	a data frame

Author(s)

Reinhard Simon

See Also

Other helper: pkg.version; pkg_version; shortSummary

Index