Package 'qualitycontrol'

November 28, 2022

Title Unified Framework for Data Quality Control

Version 0.1.0

Maintainer Luis Garcez <luisgarcez1@gmail.com>

Description An easy framework to set a quality control workflow on a dataset.
Includes a various range of functions that allow to establish an adaptable
data quality control.

Imports dplyr, stringr, janitor, openxlsx, readxl

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

URL https://github.com/luisgarcez11/qualitycontrol

BugReports https://github.com/luisgarcez11/qualitycontrol/issues

Suggests knitr, rmarkdown, testthat

Depends R (>= 2.10)

VignetteBuilder knitr

NeedsCompilation no

Author Luis Garcez [aut, cre, cph] (<https://orcid.org/0000-0002-8637-7946>)

Repository CRAN

Date/Publication 2022-11-28 09:30:02 UTC

R topics documented:

---

als_data   *Amyotrophic lateral sclerosis Example dataset*

Description

An Amyotrophic lateral sclerosis related example dataset.

Usage

```
als_data
```

Format

A list

- subjidSubject ID
- p1ALSFRS-R 1
- p2ALSFRS-R 2
- p3ALSFRS-R 3
- p4ALSFRS-R 4
- p5ALSFRS-R 5
- p6ALSFRS-R 6
- p7ALSFRS-R 7
- p8ALSFRS-R 8
- p9ALSFRS-R 9
- x1rALSFRS-R R1
- x2rALSFRS-R R2
- x3rALSFRS-R R3
- age_at_baselineAge at baseline
- age_at_onsetAge at onsite
- onsetRegion of onset
- baseline_dateBaseline date3
- death_dateDeath date

`als_data_qc_mapping`   *An example dataset containing a Quality Control mapping*

Description

An example dataset containing a Quality Control mapping

Usage

```
als_data_qc_mapping
```

Format

A list of 3 `tibbles`.

- missingTable with all the 'missing' tests.
- inconsistenciesTable with all the 'inconsistencies' tests.
- rangeTable with all the 'out of range' tests.

`qc_data`   *QC dataset using a specific variable mapping*

Description

QC dataset using a specific variable mapping

Usage

```
qc_data(data, qc_mapping, output_file = NULL)
```

Arguments

`data`	A data frame, data frame extension (e.g. a `tibble`) to be quality controlled.
`qc_mapping`	A list of data frame or data frame extension (e.g. a `tibble`) specifying the tests. Each data frame row represents a test to the `data`.
`output_file`	(optional) File path ended in `.xlsx` or `.xls`. If is not null, findings table to be written to this path.

Value

A data frame containing all the findings.

Examples

```
qc_data(als_data, als_data_qc_mapping)
```

---

read_qc_mapping   *Read Quality Control mapping file*

Description

read_qc_mapping reads an .xlsx file that contains the QC mapping.

Usage

```
read_qc_mapping(path)
```

Arguments

path   excel file path to be read. Each tab should contain 3 tabs with the names missing, inconsistencies and range. Each tab will correspond to one QC mapping table.

QC mapping excel file should contain 3 tabs:

- missing: columns should be named as "qc_type", "variable" and 'type".
- inconsistencies: columns should be named as "qc_type", "variable1", "type1", "relation", "variable2" and "type2".
- range: columns should be named as "qc_type", "variable", "type", "lower_value", "upper_value" and "categories".

The columns specified above should contain specific values:

- qc_type: "missing", "duplicated", "inconsistent_values" and "range"
- variable, variable1, variable2: variable name that is included in data.
- type, type1, type2: "numeric", text", "categorical", "date"
- relation: expected relation between variable1 and variable2 which can be "greater_than", "greater_than_or_equal", "lower_than", "lower_than_or_equal" or "equal".
- lower_value, upper_value: expected numeric values representing ranges
- categories: expected variable categories

Value

A list containing all the QC mapping tables

---

test_duplicated   *Test if variable values are duplicated*

Description

Test if variable values are duplicated

Usage

```
test_duplicated(data, variable)
```

Arguments

data	data to be tested.
variable	The variable to be tested.

Value

A data frame containing all the findings regarding the applied test.

Examples

```
test_duplicated(als_data, 'subjid')
```

---

test_inconsistencies   *Test the inconsistencies between variables on a dataset*

Description

Test the inconsistencies between variables on a dataset

Usage

```
test_inconsistencies(data, variable1, variable2, relation)
```

Arguments

data	data to be tested.
variable1	The variable to be tested.
variable2	The variable to be tested.
relation	String such as 'greater_than', 'greater_than_or_equal' 'lower_than_or_equal' and 'lower_than'.

Value

A data frame containing all the findings regarding the applied test.

Examples

```
test_inconsistencies(als_data, 'baseline_date', 'death_date', relation = 'lower_than')
test_inconsistencies(als_data, 'age_at_baseline', 'age_at_onset', relation = 'greater_than')
```

---

test_missing   *Test the variable missingness on a dataset*

Description

Test the variable missingness on a dataset

Usage

```
test_missing(data, variable)
```

Arguments

data	data to be tested.
variable	The variable to be tested.

Value

A data frame containing all the findings regarding the applied test.

Examples

```
test_missing(als_data, 'p8')
test_missing(als_data, 'p1')
```

---

test_range   *Test the range of a variable on a dataset*

Description

Test the range of a variable on a dataset

Usage

```
test_range(
  data,
  variable,
  type,
  categories = NULL,
  lower_value = NULL,
  upper_value = NULL
)
```

Arguments

`data`	data to be tested.
`variable`	The variable to be tested.
`type`	String such as 'categorical', 'date' or 'numeric'
`categories`	Only to be filled if type is 'categorical'. String of categories.
`lower_value`	Only to be filled if type is 'numeric' or 'date'. Can be numeric or string.
`upper_value`	Only to be filled if type is 'numeric' or 'date'. Can be numeric or string.

Value

A data frame containing all the findings regarding the applied test.

Examples

```
test_range(als_data, 'onset', c('bulbar','respiratory', 'spinal'), type = 'categorical')
test_range(als_data, 'age_at_baseline', lower_value = 20, upper_value = 100,
type = 'numeric')
test_range(als_data, 'age_at_onset', lower_value = 20, upper_value = 100,
type = 'numeric')
test_range(als_data, 'baseline_date', lower_value = '2000-01-01', upper_value = '2022-01-01',
type = 'date')
test_range(als_data, 'death_date', lower_value = '2000-01-01', upper_value = '2022-01-01',
 type = 'date')
```

Index