Package 'lab2clean'

September 9, 2024

Title Automation and Standardization of Cleaning Clinical Lab Data

Version 1.0.0

Description Navigating the shift of clinical laboratory data from primary everyday clinical use to secondary research purposes presents a significant challenge. Given the substantial time and expertise required for lab data pre-processing and cleaning and the lack of all-in-one tools tailored for this need, we developed our algorithm 'lab2clean' as an open-source R-package. 'lab2clean' package is set to automate and standardize the intricate process of cleaning clinical laboratory results. With a keen focus on improving the data quality of laboratory result values, our goal is to equip researchers with a straightforward, plug-and-play tool, making it smoother for them to unlock the true potential of clinical laboratory data in clinical research and clinical machine learning (ML) model development. Version 1.0 of the algorithm is described in detail in 'Zayed et al. (2024)' <doi:10.1186/s12911-024-02652-7>.

License GPL (>= 3)

Encoding UTF-8

Imports data.table, stats, utils

Suggests knitr, rmarkdown, printr

VignetteBuilder knitr

RoxygenNote 7.3.1

NeedsCompilation no

Author Ahmed Zayed [aut, cre] (<https://orcid.org/0000-0001-7797-1655>),
Arne Janssens [aut, ctb],
Pavlos Mamouris [ctb]

Maintainer Ahmed Zayed <ahmed.zayed@kuleuven.be>

Depends R (>= 3.5.0)

Repository CRAN

Date/Publication 2024-09-09 09:00:02 UTC

Contents

`clean_lab_result` *Clean and Standardize Laboratory Result Values*

Description

This function is designed to clean and standardize laboratory result values. It creates two new columns "clean_result" and "scale_type" without altering the original result values. The function is part of a comprehensive R package designed for cleaning laboratory datasets.

Usage

```
clean_lab_result(
  lab_data,
  raw_result,
  locale = "NO",
  report = TRUE,
  n_records = NA
)
```

Arguments

`lab_data`	A data frame containing laboratory data.
`raw_result`	The column in `lab_data` that contains raw result values to be cleaned.
`locale`	A string representing the locale for the laboratory data. Defaults to "NO".
`report`	A report is written in the console. Defaults to "TRUE".
`n_records`	In case you are loading a grouped list of distinct results, then you can assign the n_records to the column that contains the frequency of each distinct result. Defaults to NA

Details

The function undergoes the following methodology:

1. Clear Typos: Removes typographical errors and extraneous characters.
2. Handle Extra Variables: Identifies and separates extra variables from result values.
3. Detect and Assign Scale Types: Identifies and assigns the scale type using regular expressions.
4. Number Formatting: Standardizes number formats based on predefined rules and locale.
5. Mining Text Results: Identifies common words and patterns in text results.

Internal Datasets: The function uses an internal dataset; `common_words_languages.csv` which contains common words in various languages used for pattern identification in text result values.

Value

A modified `lab_data` data frame with additional columns:

- `clean_result`: Cleaned and standardized result values.
- `scale_type`: The scale type of result values (Quantitative, Ordinal, Nominal).
- `cleaning_comments`: Comments about the cleaning process for each record.

Note

This function is part of a larger data cleaning pipeline and should be evaluated in that context. The package framework includes functions for cleaning result values and validating quantitative results for each test identifier.

Performance of the function can be affected by the size of `lab_data`. Considerations for data size or pre-processing may be needed.

Author(s)

Ahmed Zayed ahmed.zayed@kuleuven.be

See Also

Function 2 for result validation,

`common_words` *Data for the common words*

Description

A dataset containing data for common words.

Usage

```
data(common_words)
```

Format

A data frame with 19 rows and 9 variables.

Details

- Language: Contains 19 different languages.
- Positive: Displays the word "Positive" in 19 different languages.
- Negative: Displays the word "Negative" in 19 different languages.
- Not_detected: Displays the phrase "Not detected" in 19 different languages.
- High: Displays the word "High" in 19 different languages.

- Low: Displays the word "Low" in 19 different languages.
- Normal: Displays the word "Normal" in 19 different languages.
- Sample: Displays the word "Sample" in 19 different languages.
- Specimen: Displays the word "Specimen" in 19 different languages.

`Function_1_dummy` *Dummy Data for demonstrating function 1*

Description

A dataset containing dummy data for demonstrating function 1 ("clean_lab_result").

Usage

```
data(Function_1_dummy)
```

Format

A data frame with 87 rows and 2 variables.

Details

- raw_result: The raw result.
- frequency: The frequency of the raw result.

`Function_2_dummy` *Dummy Data for demonstrating function 2*

Description

A dataset containing dummy data for demonstrating function 2 ("validate_lab_result").

Usage

```
data(Function_2_dummy)
```

Format

A data frame with 86,863 rows and 5 variables.

Details

- patient_id: Indicates the identifier of the tested patient.
- lab_datetime1: Indicates the date or datetime of the laboratory test.
- loinc_code: Indicates the LOINC code of the laboratory test.
- result_value: Indicates the quantitative result values for validation.
- result_unit: Indicates the result units in a UCUM-valid format.

logic_rules *Data for the logic rules*

Description

A dataset containing data for the logic rules.

Usage

```
data(logic_rules)
```

Format

A data frame with 18 rows and 4 variables.

Details

- rule_id: The rule ID.
- rule_index: The rule index.
- rule_part: The rule part.
- rule_part_type: The rule part type.

reportable_interval *Data for the reportable interval*

Description

A dataset containing data for the reportable interval.

Usage

```
data(reportable_interval)
```

Format

A data frame with 493 rows and 4 variables.

Details

- interval_loinc_code: The interval of the LOINC code.
- UCUM_unit: The UCUM unit.
- low_reportable_limit: The lower reportable limit.
- high_reportable_limit: The higher reportable limit.

validate_lab_result *Validate Quantitative Laboratory Result Values*

Description

This function is designed to validate quantitative laboratory result values. It modifies the provided lab_data dataframe in-place, adding one new column.

Usage

```
validate_lab_result(
  lab_data,
  result_value,
  result_unit,
  loinc_code,
  patient_id,
  lab_datetime,
  report = TRUE
)
```

Arguments

lab_data	A data frame containing laboratory data.
result_value	The column in lab_data with quantitative result values for validation.
result_unit	The column in lab_data with result units in a UCUM-valid format.
loinc_code	The column in lab_data indicating the LOINC code of the laboratory test.
patient_id	The column in lab_data indicating the identifier of the tested patient.
lab_datetime	The column in lab_data with the date or datetime of the laboratory test.
report	A report is written in the console. Defaults to "TRUE".

Details

The function employs the following validation methodology:

1. Reportable limits check: Identifies implausible values outside reportable limits.
2. Logic rules check: Identifies values that contradict some predefined logic rules.
3. Delta limits check: Flags values with excessive change from prior results for the same test and patient.

Internal Datasets: The function uses two internal datasets included with the package:

1. reportable_interval: Contains information on reportable intervals.
2. logic_rules: Contains logic rules for validation.

Value

A modified `lab_data` data frame with additional columns:

- `flag`: specifies the flag detected in the result records that violated one or more of the validation checks

Note

This function is a component of a broader laboratory data cleaning pipeline and should be evaluated accordingly. The package's framework includes functions for cleaning result values, validating quantitative results, standardizing unit formats, performing unit conversion, and assisting in LOINC code mapping.

Concerning performance, the function's speed might be influenced by the size of `lab_data`. Consider:

- Limiting the number of records processed.
- Optimize the function for larger datasets.
- Implement pre-processing steps to divide the dataset chronologically.

Author(s)

Ahmed Zayed ahmed.zayed@kuleuvne.be, Arne Janssens arne.janssens@kuleuven.be

See Also

Function 1 for result value cleaning,

Index