Package ‘vvconverter’

March 19, 2025

Title Apply Transformations to Data

Version 0.7.0

Description Provides a set of functions for data transformations.
Transformations are performed on character and numeric data. As the scope of the package is within Student Analytics, there are functions focused around the academic year.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.1

URL `https://vusaverse.github.io/vvconverter/`,
`https://github.com/vusaverse/vvconverter`

BugReports `https://github.com/vusaverse/vvconverter/issues`

Imports checkmate, dplyr, lubridate, magrittr, polyglotr, stringr

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Tomer Iwan [aut, cre, cph]

Maintainer Tomer Iwan <`t.iwan@vu.nl`>

Repository CRAN

Date/Publication 2025-03-19 15:00:02 UTC

Contents

`academic_year` *Academic year*

Description

In this function, a date is translated to the academic year in which it falls. This is based on a start of the academic year on the 1st of September.

Usage

```
academic_year(x, start_1_oct = FALSE)
```

Arguments

`x`	A date, or vector with multiple dates. POSIXct is also accepted.
`start_1_oct`	Does the academic year start on the 1st of October? default FALSE: based on September 1st

Value

The academic year in which the specified date falls

See Also

Other vector calculations: `clean_multiple_underscores()`, `interval_round()`, `month_name()`, `sum_0_1()`, `transform_01_to_ft()`

Examples

```
academic_year(lubridate::today())
```

`clean_multiple_underscores` *clean multiple underscores*

Description

Replaces multiple underscores into a single underscore in a vector or string.

Usage

```
clean_multiple_underscores(x)
```

Arguments

`x`	The vector or string to be cleaned.

Value

cleaned vector or string.

See Also

Other vector calculations: `academic_year()`, `interval_round()`, `month_name()`, `sum_0_1()`, `transform_01_to_ft()`

Examples

```
clean_multiple_underscores("hello___world")
```

`destring` *Convert character vector to numeric, ignoring irrelevant characters.*

Description

Convert character vector to numeric, ignoring irrelevant characters.

Usage

```
destring(x, keep = "0-9.-")
```

Arguments

`x`	A vector to be operated on
`keep`	Characters to keep in, in bracket regular expression form. Typically includes 0-9 as well as the decimal separator (. in the US and , in Europe).

Value

vector of type numeric

Examples

```
destring("24k")
destring("5,5")
```

`group_summary` *Group Summary*

Description

Calculate the means (or other function) per group to analyze how each segment behaves. It scales each variable mean into the 0 to 1 range to easily profile the groups according to its mean. It also calculates the mean regardless of the grouping. This function is also useful when you want to profile cluster results in terms of its means. It automatically adds a row representing the summary of the column regardless of the group_var categories, which is useful to compare each segment with the whole population. It will exclude all factor/character variables.

Usage

```
group_summary(data, group_var, group_func = mean)
```

Arguments

`data`	Input data source.
`group_var`	Variable to make the group by.
`group_func`	Function to be used in the group by. Default is mean.

Value

Grouped data frame.

`group_summary_rank` *Group Summary Rank*

Description

Similar to 'group_summary' function, this one computes the rank of each value in order to quickly know what is the value in each segment that has the highest value (rank=1). 1 represents the highest number. It will exclude all factor/character variables.

Usage

```
group_summary_rank(data, group_var, group_func = mean)
```

Arguments

`data`	Input data source.
`group_var`	Variable to make the group by.
`group_func`	Function to be used in the group by. Default is mean.

Value

Grouped data frame, showing the rank instead of the absolute values.

`interval_round` *Interval round*

Description

Function to round numeric values in a vector to values from an interval sequence.

Usage

```
interval_round(x, interval)
```

Arguments

`x`	The numeric vector to adjust
`interval`	The interval sequence

Value

The vector corrected for the given interval

See Also

Other vector calculations: `academic_year()`, `clean_multiple_underscores()`, `month_name()`, `sum_0_1()`, `transform_01_to_ft()`

Examples

```
interval_round(c(5, 4, 2, 6), interval = seq(1:4))
```

`ltrim` *LTrim*

Description

Trim leading whitespace from sting.

Usage

```
ltrim(x)
```

Arguments

`x`	A text string.

Value

Cleaned string.

Examples

```
trim(" hello")
```

`median_top_10` *Median top 10 percentage*

Description

Calculate the median of the top ten percentage of the values.

Usage

```
median_top_10(x, na.rm = FALSE)
```

Arguments

`x`	A numerical vector
`na.rm`	Default TRUE: Remove NAs, before calculations.

Value

A numerical value

Examples

```
median_top_10(mtcars$cyl)
```

`mode` *Mode (most common value)*

Description

Determine the most common value in a vector. If two values have the same frequency, the first occurring value is used.

Usage

```
mode(x, na.rm = FALSE)
```

Arguments

`x`	a vector
`na.rm`	If TRUE: Remove nas before the calculation is done

Value

the most common value in the vector x

Examples

```
mode(c(0, 3, 5, 7, 5, 3, 2))
```

`month_name` *Month Name*

Description

Transform month from numeric to equivalent in specified language.

Usage

```
month_name(month_numeric, lang = "nl")
```

Arguments

`month_numeric`	Numeric in range 1 - 12.
`lang`	The language of the month names. Default is "nl" (Dutch).

Value

Character string representation of month in specified language.

See Also

Other vector calculations: `academic_year()`, `clean_multiple_underscores()`, `interval_round()`, `sum_0_1()`, `transform_01_to_ft()`

`rtrim` *RTrim*

Description

Trim trailing whitespaces from string.

Usage

```
rtrim(x)
```

Arguments

`x`	A text string.

Value

Cleaned string.

Examples

```
trim("hello ")
```

`str_replace_all_in_file` *Replace all occurences of a pattern in a file*

Description

Replace all occurences of a pattern in a file

Usage

```
str_replace_all_in_file(
  file,
  pattern,
  replacement = "[...]",
  only_comments = TRUE,
  collapse = FALSE
)
```

Arguments

`file`	character, path of file to be modified
`pattern`	character, pattern to be replaced
`replacement`	character, replacement text
`only_comments`	logical, should the replacement only be done in comments
`collapse`	logical, should the lines be collapsed into a single line before replacement

Value

NULL, the file is modified in place

`sum_0_1` *Sum 0 1*

Description

This function is the same as sum(), with one exception: If the outcome value is higher than 1, it will always return 1.

Usage

```
sum_0_1(x)
```

Arguments

`x`	a vector with numeric values

Value

0 or 1. Depending on whether the sum is greater than 0 or not.

See Also

Other vector calculations: `academic_year()`, `clean_multiple_underscores()`, `interval_round()`, `month_name()`, `transform_01_to_ft()`

`test_01` *Test 01*

Description

This function tests whether the vector is actually a boolean, but is encoded as a 0/1 variable. The function checks for numeric vectors whether the only occurring values are 0, 1, or NA. At character and factor vectors checks whether the only occurring values are "0", "1", or NA to be. If there is a 0/1 variable, TRUE is returned, in all others cases FALSE.

Usage

```
test_01(x)
```

Arguments

`x`	The vector to test

Value

A TRUE/FALSE value on the test

See Also

Other booleans: `transform_01_to_ft()`

Examples

```
vector <- c(0, 1, 0, 1, 1, 1, 0)
test_01(vector)
```

`test_yes_no` *Test Yes/No Responses*

Description

This function tests if a vector of responses are yes or no.

Usage

```
test_yes_no(responses)
```

Arguments

`responses`	A vector of responses.

Value

A logical vector indicating if each response is yes or no.

`transform_01_to_ft` *Transform 01 to FT*

Description

If the vector is a 0/1 vector, it is converted to a logical one TRUE/FALSE vector. This transformation is performed only if the vector contains only values 0, 1, or NA. If this is not the case returns the original variable. This transformation can be done on numeric, string, and factor vectors.

Usage

```
transform_01_to_ft(x)
```

Arguments

`x`	the vector to be tested and transformed.

Value

The transformed vector if a transformation is possible. If no transformation is possible, the original vector returned.

See Also

Other vector calculations: `academic_year()`, `clean_multiple_underscores()`, `interval_round()`, `month_name()`, `sum_0_1()`

Other booleans: `test_01()`

Examples

```
vector <- c(0, 1, 0, 1, 1, 1, 0)
transform_01_to_ft(vector)
```

transform_logical_yes_no *Transform Logical to Yes/No and Vice Versa*

Description

This function transforms a logical vector to a vector of yes/no strings or vice versa.

Usage

```
transform_logical_yes_no(x, lang = "nl")
```

Arguments

x	A logical or character vector.
lang	The language of the yes/no strings. Default is "nl" (Dutch).

Value

A vector of yes/no strings or a logical vector.

translate_yes_no *Translate Yes/No Responses*

Description

This function translates yes/no responses from a given language to English.

Usage

```
translate_yes_no(responses, source_language = "nl")
```

Arguments

responses	A vector of responses.
source_language	The language of the responses. Default is "nl" (Dutch).

Value

A vector of translated responses.

`trim` *Trim*

Description

Trim both leading and trailing whitespaces from string.

Usage

```
trim(x)
```

Arguments

`x`	A text string.

Value

Cleaned string.

Examples

```
trim(" hello ")
```

Index