Package ‘dataMojo’

February 22, 2023

Title Reshape Data Table

Version 1.0.0

Description A grammar of data manipulation with 'data.table', providing a consistent a series of utility functions that help you solve the most common data manipulation challenges.

Suggests knitr, rmarkdown, testthat, dplyr

VignetteBuilder knitr

License MIT + file LICENSE

Imports data.table

Encoding UTF-8

LazyData true

RoxygenNote 7.2.2

NeedsCompilation no

Author Jiena McLellan [aut, cre] (<https://orcid.org/0000-0002-5578-088X>),
Michael Condouris [ctb] (<https://orcid.org/0000-0002-8862-4250>),
Brittney Zykan [ctb],
Sai Im [ctb]

Maintainer Jiena McLellan <jienagu90@gmail.com>

Depends R (>= 3.5.0)

Repository CRAN

Date/Publication 2023-02-22 15:00:02 UTC

R topics documented:

`col_cal_percent` *create a new column which is the percentage of other columns*

Description

create a new column which is the percentage of other columns

Usage

```
col_cal_percent(df, new_col_name, numerator_cols, denominator_cols)
```

Arguments

`df`	input data frame
`new_col_name`	new column name
`numerator_cols`	numerator columns
`denominator_cols`	denominator columns

Value

data frame with a new percentage column

Examples

```
test_df <- data.frame(
  hc1 = c(2, 0, 1, 5, 6, 7, 10),
  hc2 = c(1, 0, 10, 12, 4, 1, 9 ),
  total = c(10, 2, 0, 39, 23, 27, 30)
)
dataMojo::col_cal_percent(test_df,
  new_col_name = "hc_percentage",
  numerator_cols = c("hc1", "hc2"),
  denominator_cols = "total"
)
```

`dt_dates` *Anonymized sample data*

Description

Anonymized sample data

Usage

```
data(dt_dates)
```

Format

a data table with dates

Author(s)

Jiena Gu McLellan, 2020-05-26

Examples

```
data(dt_dates)
```

`dt_groups` *Anonymized sample data*

Description

Anonymized sample data

Usage

```
data(dt_groups)
```

Format

a data table with groups

Author(s)

Jiena Gu McLellan, 2020-05-26

Examples

```
data(dt_groups)
```

`dt_group_by` *group by columns and return a summarized table*

Description

group by columns and return a summarized table

Usage

```
dt_group_by(dt, group_by_cols, summarize_at, operation)
```

Arguments

`dt`	input data.table
`group_by_cols`	group by columns
`summarize_at`	column summarize at
`operation`	calculation operation, value should be one of following: sum, mean, median, max, min

Value

a summarized table

Examples

```
data("dt_groups")
dataMojo::dt_group_by(dt_groups,
                      group_by_cols = c("group1", "group2"),
                      summarize_at = "A1",
                      operation = "mean")
```

`dt_long` *Anonymized sample data*

Description

Anonymized sample data

Usage

```
data(dt_long)
```

Format

a data table in long format

Author(s)

Jiena Gu McLellan, 2020-05-26

Examples

```
data(dt_long)
```

`dt_missing` *Anonymized sample data*

Description

Anonymized sample data

Usage

```
data(dt_missing)
```

Format

a data table with missing values

Author(s)

Jiena Gu McLellan, 2020-05-26

Examples

```
data(dt_missing)
```

`dt_values` *Anonymized sample data*

Description

Anonymized sample data

Usage

```
data(dt_values)
```

Format

a data table with values

Author(s)

Jiena Gu McLellan, 2020-05-26

Examples

```
data(dt_values)
```

`fill_NA_with` *Fill missing values*

Description

Fill missing values

Usage

```
fill_NA_with(dt, fill_cols, fill_value)
```

Arguments

`dt`	input data table
`fill_cols`	filter by this columns
`fill_value`	fill NA with this value

Value

data table which NAs are filled

Examples

```
data("dt_missing")
fill_NA_with(dt_missing, fill_cols = c("Full_name"), fill_value = "pending")
```

filter_all *Filter all rows that meeting requirements*

Description

Filter all rows that meeting requirements

Usage

```
filter_all(dt, operator, cutoff_value)
```

Arguments

`dt`	input data.table
`operator`	operator should be one of l, g. l means less than, g means greater than.
`cutoff_value`	threshold value

Value

filtered data table

Examples

```
data("dt_values")
dataMojo::filter_all(dt_values, operator = "l", .2)
```

filter_all_at *Filter all rows that meet requirements with selected columns*

Description

Filter all rows that meet requirements with selected columns

Usage

```
filter_all_at(dt, operator, cutoff_value, selected_cols)
```

Arguments

`dt`	input data table
`operator`	operator should be one of l, or g. l means less than, g means greater than
`cutoff_value`	cutoff value
`selected_cols`	selected columns from input data table

Value

filtered data table

Examples

```
data("dt_values")
dataMojo::filter_all_at(dt_values, operator = "l", .1, c("A1", "A2"))
```

`filter_any` *Filter any rows that meeting requirements*

Description

Filter any rows that meeting requirements

Usage

```
filter_any(dt, operator, cutoff_value)
```

Arguments

`dt`	input data.table
`operator`	operator should be one of l, g. l means less than, g means greater than.
`cutoff_value`	threshold value

Value

fitlered data table

Examples

```
data("dt_values")
dataMojo::filter_any(dt_values, operator = "l", .1)
```

filter_any_at *Filter any rows that meet requirements with selected columns*

Description

Filter any rows that meet requirements with selected columns

Usage

```
filter_any_at(dt, operator, cutoff_value, selected_cols)
```

Arguments

`dt`	input data table
`operator`	operator should be one of l, or g. l means less than, g means greater than
`cutoff_value`	cutoff value
`selected_cols`	selected columns from input data table

Value

filtered data table

Examples

```
data("dt_values")
dataMojo::filter_all_at(dt_values, operator = "l", .1, c("A1", "A2"))
```

get_row_group_by *Fetch one row from each grouped by group*

Description

Fetch one row from each grouped by group

Usage

```
get_row_group_by(dt, group_by_cols, fetch_row)
```

Arguments

`dt`	input data table
`group_by_cols`	group by columns
`fetch_row`	first means to fetch first row and last means to fetch last row

Value

grouped by data table

Examples

```
data("dt_groups")
dataMojo::get_row_group_by(dt_groups,
                           group_by_cols = c("group1", "group2"),
                           fetch_row = "first")
```

---

pivot_percent_at *Create an aggregated data table with all proportion of one selected column*

Description

Create an aggregated data table with all proportion of one selected column

Usage

```
pivot_percent_at(dt, question_col, aggregated_by_cols)
```

Arguments

`dt`	data table
`question_col`	column selected as questions
`aggregated_by_cols`	grouped by columns

Value

aggregated data table

Examples

```
test_dt <- data.table::data.table(
Question = c(rep("Good", 3), rep("OK", 3), rep("Bad", 3)),
Gender = c(rep("F", 4), rep("M", 5))
)
dataMojo::pivot_percent_at(test_dt,
  question_col = "Question", aggregated_by_cols = "Gender")
```

`pivot_percent_at_multi` *Create an aggragated data table with all proportion of multiple selected column*

Description

Create an aggragated data table with all proportion of multiple selected column

Usage

```
pivot_percent_at_multi(dt, question_col, aggregated_by_cols)
```

Arguments

`dt`	data table
`question_col`	columns selected as questions
`aggregated_by_cols`	grouped by columns

Value

an aggragated data table

Examples

```
test_dt <- data.table::data.table(
  Question1 = c(rep("Good", 3), rep("OK", 3), rep("Bad", 3)),
  Question2 = c(rep("Good", 2), rep("OK", 2), rep("Bad", 5)),
  Gender = c(rep("F", 4), rep("M", 5))
)
dataMojo::pivot_percent_at_multi(test_dt,
  question_col = c("Question1","Question2") , aggregated_by_cols = "Gender")
```

`reshape_longer` *Reshape data frame to a longer format*

Description

Reshape data frame to a longer format

Usage

```
reshape_longer(dt, keep_cols, label_cols, value_cols)
```

Arguments

`dt`	input data
`keep_cols`	columns to be kept
`label_cols`	column name that contains the melted columns
`value_cols`	column name that contains the value of melted columns

Value

data table in a longer format

Examples

```
data("dt_dates")
reshape_longer(dt_dates,
                        keep_cols = "Full_name",
                        label_cols = c("Date_Type"),
                        value_cols = "Exact_date")
```

`reshape_wider` *Reshape data frame to a wider format*

Description

Reshape data frame to a wider format

Usage

```
reshape_wider(dt, keep_cols, col_label, col_value)
```

Arguments

`dt`	input data table
`keep_cols`	columns to be kept
`col_label`	columns that each unique values will be reshaped as a column name
`col_value`	columns that fill the reshaped columns

Value

reshaped widen data table

Examples

```
data("dt_long")
dataMojo::reshape_wider(dt_long,
  keep_cols = c("Full_name"),
  col_label = c("Date_Type"),
  col_value = "Exact_date")
```

`row_expand_dates` *Expand row given start and end dates*

Description

Expand row given start and end dates

Usage

```
row_expand_dates(dt, start_date_col, end_date_col, new_name)
```

Arguments

`dt`	input data table
`start_date_col`	start date column
`end_date_col`	end date column
`new_name`	new generated column name

Value

expanded data table

Examples

```
dt_dates_simple <- data.table::data.table(
  Start_Date = as.Date(c("2020-02-03", "2020-03-01") ),
  End_Date = as.Date(c("2020-02-05", "2020-03-02") ),
  group = c("A", "B")
)
row_expand_dates(dt_dates_simple, "Start_Date", "End_Date", "Date")[]
```

`row_expand_pattern` *Expand row based on pattern*

Description

Expand row based on pattern

Usage

```
row_expand_pattern(dt, col_name, split_by_pattern, new_name)
```

Arguments

`dt`	input data table
`col_name`	column to be expanded
`split_by_pattern`	split based on pattern
`new_name`	new generated column name

Value

expanded data table

Examples

```
data("starwars_simple")
row_expand_pattern(starwars_simple, "films", ", ", "film")[]
```

`row_percent_convert` *Convert count to percentage*

Description

Convert count to percentage

Usage

```
row_percent_convert(data, cols_rowsum)
```

Arguments

`data`	data frame
`cols_rowsum`	columns need to be converted to percentage

Value

data frame with calculated row percentage

Examples

```
test_df <- data.frame(
  Group = c("A", "B", "C"),
  Female = c(2,3,5),
  Male = c(10,11, 13)
)
dataMojo::row_percent_convert(test_df, cols_rowsum = c("Female", "Male"))
```

`select_cols` *Select columns*

Description

Select columns

Usage

```
select_cols(dt, cols)
```

Arguments

`dt`	input data table
`cols`	select columns

Value

data table with selected columns

Examples

```
data("dt_dates")
select_cols(dt_dates, c("Start_Date", "Full_name"))
```

`starwars_simple` *starwars data*

Description

starwars data

Usage

```
data(starwars_simple)
```

Format

a data table as example

Author(s)

Jiena Gu McLellan, 2020-05-26

Examples

```
data(starwars_simple)
```

`str_split_col` *Split one column to multiple columns based on patterns*

Description

Split one column to multiple columns based on patterns

Usage

```
str_split_col(dt, by_col, by_pattern, match_to_names = NULL)
```

Arguments

`dt`	input data table
`by_col`	by this column
`by_pattern`	split by this patter
`match_to_names`	created new columns names

Value

data table with new columns

Examples

```
data("dt_dates")
str_split_col(dt_dates,
              by_col = "Full_name",
              by_pattern = ", ",
              match_to_names = c("First Name", "Last Name"))
```

Index