Package ‘tidier’

September 11, 2023

Title Enhanced 'mutate'

Version 0.2.0

Description Provides 'Apache Spark' style window aggregation for R dataframes and remote 'dbplyr' tables via 'mutate' in 'dplyr' flavour.

Imports dplyr (>= 1.1.0), tidyr (>= 1.3.0), checkmate (>= 2.1.0), rlang (>= 1.0.6), slider (>= 0.2.2), magrittr (>= 1.5), furrr (>= 0.3.0), dbplyr (>= 2.3.1),

Suggests lubridate, stringr, testthat, RSQLite, tibble,

URL https://github.com/talegari/tidier

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.2.1

NeedsCompilation no

Author Srikanth Komala Sheshachala [aut, cre]

Maintainer Srikanth Komala Sheshachala <sri.teach@gmail.com>

Repository CRAN

Date/Publication 2023-09-11 18:10:02 UTC

R topics documented:

mutate ........................................... 2
mutate_ ......................................... 4
remove_common_nested_columns ................ 7

Index 8
mutate  

Drop-in replacement for mutate

Description

Provides supercharged version of mutate with group_by, order_by and aggregation over arbitrary window frame around a row for dataframes and lazy (remote) tbls of class tbl_lazy.

Usage

mutate(x, ..., .by, .order_by, .frame, .index, .complete = FALSE)

Arguments

x (data.frame or tbl_lazy)
...
expressions to be passed to mutate
.by (expression, optional: Yes) Columns to group by
.order_by (expression, optional: Yes) Columns to order by
.frame (vector, optional: Yes) Vector of length 2 indicating the number of rows to consider before and after the current row. When argument .index is provided (typically a column of type date or datetime), before and after can be interval objects. See examples. When input is tbl_lazy, only number of rows as vector of length 2 is supported.
.index (expression, optional: Yes, default: NULL) index column. This is supported when input is a dataframe only.
.complete (flag, default: FALSE) This will be passed to slider::slide / slider::slide_vec. Should the function be evaluated on complete windows only? If FALSE or NULL, the default, then partial computations will be allowed. This is supported when input is a dataframe only.

Details

A window function returns a value for every input row of a dataframe or lazy_tbl based on a group of rows (frame) in the neighborhood of the input row. This function implements computation over groups (partition_by in SQL) in a predefined order (order_by in SQL) across a neighborhood of rows (frame) defined by a (up, down) where

- up/down are number of rows before and after the corresponding row
- up/down are interval objects (ex: c(days(2), days(1))). Interval objects are currently supported for dataframe only. (not tbl_lazy)

This implementation is inspired by spark’s window API.

Implementation Details:

For dataframe input:

- Iteration per row over the window is implemented using the versatile slider.
• Application of a window aggregation can be optionally run in parallel over multiple groups (see argument .by) by setting a future parallel backend. This is implemented using furrr package.

• function subsumes regular usecases of mutate

For tbl_lazy input:

• Uses dbplyr::window_order and dbplyr::window_frame to translate to partition_by and window frame specification.

Value
data.frame or tbl_lazy

See Also
mutate_

Examples

library("magrittr")
# example 1 (simple case with dataframe)
# Using iris dataset,
# compute cumulative mean of column `Sepal.Length`
# ordered by `Petal.Width` and `Sepal.Width` columns
# grouped by `Petal.Length` column
iris %>%
  mutate(sl_mean = mean(Sepal.Length),
    .order_by = c(Petal.Width, Sepal.Width),
    .by = Petal.Length,
    .frame = c(Inf, 0),
  ) %>%
dplyr::slice_min(n = 3, Petal.Width, by = Species)

# example 2 (detailed case with dataframe)
# Using a sample airquality dataset,
# compute mean temp over last seven days in the same month for every row
set.seed(101)
airquality %>%
  # create date column
dplyr::mutate(date_col = lubridate::make_date(1973, Month, Day)) %>%
  # create gaps by removing some days
dplyr::slice_sample(prop = 0.8) %>%
dplyr::arrange(date_col) %>%
  # compute mean temperature over last seven days in the same month
tidier::mutate(avg_temp_over_last_week = mean(Temp, na.rm = TRUE),
    .order_by = Day,
    .by = Month,
    .frame = c(lubridate::days(7), # 7 days before current row
      lubridate::days(-1) # do not include current row
  )
mutate_

Drop-in replacement for mutate

Description

Provides supercharged version of mutate with group_by, order_by and aggregation over arbitrary window frame around a row for dataframes and lazy (remote) tbls of class tbl_lazy.

Usage

mutate_(
  x,
  ...,
  .by,
  .order_by,
  .frame,
  .index,
  .desc = FALSE,
  .complete = FALSE
)

Arguments

x  (data.frame or tbl_lazy)
...
.by  (character vector, optional: Yes) Columns to group by
.order_by  (string, optional: Yes) Columns to order by
mutate_

.frame (vector, optional: Yes) Vector of length 2 indicating the number of rows to consider before and after the current row. When argument .index is provided (typically a column of type date or datetime), before and after can be interval objects. See examples. When input is tbl_lazy, only number of rows as vector of length 2 is supported.

.index (string, optional: Yes, default: NULL) index column. This is supported when input is a dataframe only.

.desc (flag, default: FALSE) Whether to order in descending order

.complete (flag, default: FALSE) This will be passed to slider::slide / slider::slide_vec. Should the function be evaluated on complete windows only? If FALSE or NULL, the default, then partial computations will be allowed. This is supported when input is a dataframe only.

Details
A window function returns a value for every input row of a dataframe or lazy_tbl based on a group of rows (frame) in the neighborhood of the input row. This function implements computation over groups (partition_by in SQL) in a predefined order (order_by in SQL) across a neighborhood of rows (frame) defined by a (up, down) where

• up/down are number of rows before and after the corresponding row
• up/down are interval objects (ex: c(days(2), days(1))). Interval objects are currently supported for dataframe only. (not tbl_lazy)

This implementation is inspired by spark’s window API.

Implementation Details:
For dataframe input:

• Iteration per row over the window is implemented using the versatile slider.
• Application of a window aggregation can be optionally run in parallel over multiple groups (see argument .by) by setting a future parallel backend. This is implemented using furrr package.
• function subsumes regular usecases of mutate

For tbl_lazy input:

• Uses dbplyr::window_order and dbplyr::window_frame to translate to partition_by and window frame specification.

Value
data.frame or tbl_lazy

See Also
mutate
Examples

library("magrittr")
# example 1 (simple case with dataframe)
# Using iris dataset,
# compute cumulative mean of column `Sepal.Length`
# ordered by `Petal.Width` and `Sepal.Width` columns
# grouped by `Petal.Length` column

iris %>%
tidier::mutate_(sl_mean = mean(Sepal.Length),
    .order_by = c("Petal.Width", "Sepal.Width"),
    .by = "Petal.Length",
    .frame = c(Inf, 0),
) %>%
dplyr::slice_min(n = 3, Petal.Width, by = Species)

# example 2 (detailed case with dataframe)
# Using a sample airquality dataset,
# compute mean temp over last seven days in the same month for every row

set.seed(101)
airquality %>%
# create date column
dplyr::mutate(date_col = lubridate::make_date(1973, Month, Day)) %>%
# create gaps by removing some days
dplyr::slice_sample(prop = 0.8) %>%
dplyr::arrange(date_col) %>%
# compute mean temperature over last seven days in the same month
tidier::mutate_(avg_temp_over_last_week = mean(Temp, na.rm = TRUE),
    .order_by = "Day",
    .by = "Month",
    .frame = c(lubridate::days(7), # 7 days before current row
              lubridate::days(-1) # do not include current row
              ),
    .index = "date_col"
)

# example 3
airquality %>%
# create date column as character
dplyr::mutate(date_col =
    as.character(lubridate::make_date(1973, Month, Day))
) %>%
tibble::as_tibble() %>%
# as `tbl_lazy`
dplyr::memdb_frame() %>%
mutate_(avg_temp = mean(Temp),
    .by = "Month",
    .order_by = "date_col",
    .frame = c(3, 3)
) %>%
dplyr::collect() %>%
dplyr::select(Ozone, Solar.R, Wind, Temp, Month, Day, date_col, avg_temp)
remove_common_nested_columns

Remove non-list columns when same are present in a list column

Description
Remove non-list columns when same are present in a list column

Usage
remove_common_nested_columns(df, list_column)

Arguments
- df: input dataframe
- list_column: Name or expr of the column which is a list of named lists

Value
dataframe
Index

mutate, 2, 2, 3–5
mutate., 4

remove_common_nested_columns, 7