Package ‘quanteda.tidy’

December 17, 2025

Title 'tidyverse' Extensions for 'quanteda'

Version 0.4

Description

Enables 'tidyverse' operations on 'quanteda' corpus objects by extending 'dplyr' verbs to work directly with corpus objects and their document-level variables ('docvars'). Implements row operations for 'subsetting' and reordering documents; column operations for managing document variables; grouped operations; and two-table verbs for merging external data. For more on 'quanteda' see 'Benoit et al.' (2018) <doi:10.21105/joss.00774>. For 'dplyr' see 'Wickham et al.' (2023) <doi:10.32614/CRAN.package.dplyr>.

Depends R (>= 3.5.0), quanteda (>= 3.0.0)

Imports dplyr, rlang, tibble, tidyselect

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.3

Suggests covr, knitr, rmarkdown, spelling, testthat

VignetteBuilder knitr

Language en-GB

NeedsCompilation no

Author Kenneth Benoit [aut, cre, cph]

Maintainer Kenneth Benoit <kbenoit@smu.edu.sg>

Repository CRAN

Date/Publication 2025-12-17 10:10:08 UTC

Contents

`quanteda.tidy-package` *quanteda.tidy: Tidyverse Extensions for quanteda*

Description

Extends 'dplyr' verbs to work directly on 'quanteda' corpus objects, enabling users to manipulate document-level variables ("docvars") using familiar 'tidyverse' syntax. Implements row operations for subsetting and reordering documents; column operations for managing document variables; grouped operations via `add_count()` and `add_tally()`; and two-table verbs (such as `left_join()`) for merging external data.

Author(s)

Maintainer: Kenneth Benoit <`kbenoit@smu.edu.sg`> [copyright holder]

References

Benoit, K., Watanabe, K., Wang, H., Nulty, P., Obeng, A., Müller, S., & Matsuo, A. (2018). "quanteda: An R package for the quantitative analysis of textual data." *Journal of Open Source Software*, 3(30), 774. doi:10.21105/joss.00774

Wickham, H., François, R., Henry, L., Müller, K., & Vaughan, D. (2023). *dplyr: A Grammar of Data Manipulation*. R package version 1.1.4. doi:10.32614/CRAN.package.dplyr

`add_count.corpus` *Add count of observations to corpus*

Description

`add_count()` and `add_tally()` are wrappers around `dplyr::add_count()` and `dplyr::add_tally()` that add a new document variable with the number of observations. `add_count()` is a shortcut for `group_by()` + `add_tally()`.

Usage

```
## S3 method for class 'corpus'
add_count(x, ..., wt = NULL, sort = FALSE, name = NULL, .drop = NULL)

## S3 method for class 'corpus'
add_tally(x, ..., wt = NULL, sort = FALSE, name = NULL)
```

Arguments

`x` a **quanteda** corpus object

`...` for `add_count()`, document variables to group by; for `add_tally()`, additional arguments passed to the method

`wt` frequency weights. Can be `NULL` or a variable:

- If `NULL` (the default), counts the number of rows in each group
- If a variable, computes `sum(wt)` for each group

`sort` if `TRUE`, will sort output in descending order of `n`

`name` the name of the new column in the output. If omitted, it will default to `n`. If there's already a column called `n`, it will error, and require you to specify the name.

`.drop` not used for corpus objects; included for compatibility with the generic

Value

a corpus with an additional document variable containing counts

Examples

```
# Count documents by President and add as a variable
data_corpus_inaugural %>%
  add_count(President) %>%
  summary(n = 10)

# Add total count to each document
data_corpus_inaugural %>%
  head() %>%
  add_tally() %>%
  summary()

# Count by multiple variables
data_corpus_inaugural %>%
  add_count(Party, President) %>%
  summary(n = 10)

# Use custom name
data_corpus_inaugural %>%
  add_count(Party, name = ”party_count”) %>%
  summary(n = 10)
```

```
# Add tally to show total count
data_corpus_inaugural %>%
  slice(1:6) %>%
  add_tally() %>%
  summary()
```

add_tally *Add count of observations to corpus*

Description

add_tally is a generic function for adding a count column. The default method calls dplyr::add_tally().

Usage

```
add_tally(x, ...)
```

Arguments

x	an object
...	additional arguments passed to methods

Value

A corpus with an additional document variable containing counts.

arrange.corpus *Arrange the document order of a corpus by variables*

Description

Order the documents in a corpus by variables, including document variables.

Usage

```
## S3 method for class 'corpus'
arrange(.data, ...)
```

Arguments

.data	a corpus object whose documents will be sorted
...	comma-separated list of unquoted document variables, or expressions involving document variables. Use desc to sort a variable in descending order.

Value

A corpus with documents reordered according to the specified variables.

Examples

```
arrange(data_corpus_inaugural[1:5], President)
arrange(data_corpus_inaugural[1:5], c(3, 2, 1, 5, 4))
arrange(data_corpus_inaugural[1:5], desc(President))
```

`distinct.corpus` *Subset documents distinct/unique by document variables*

Description

Select only documents that are unique/distinct with respect to values of their document variables.

Usage

```
## S3 method for class 'corpus'
distinct(.data, ..., .keep_all = FALSE)
```

Arguments

`.data` a corpus object with document variables

`...` comma-separated list of unquoted document variables, or expressions involving document variables

`.keep_all` If `TRUE`, keep all variables in `.data`. If a combination of `...` is not distinct, this keeps the first row of values.

Value

A corpus containing only documents with unique combinations of the specified document variables.

Examples

```
distinct(data_corpus_inaugural[1:5], President) %>%
  summary()
distinct(data_corpus_inaugural[1:5], President, .keep_all = TRUE) %>%
  summary()
```

filter.corpus *Return documents with matching conditions*

Description

Use filter() to select documents where conditions evaluated on document variables are true. Documents where the condition evaluates to NA are dropped. A tidy replacement for corpus_subset().

Usage

```
## S3 method for class 'corpus'
filter(.data, ..., .preserve = FALSE)
```

Arguments

.data a **quanteda** object whose documents will be filtered

... Logical predicates defined in terms of the document variables in .data, or a condition supplied externally whose length matches the number of ndoc(.data)‘. See filter.

.preserve Relevant when the .data input is grouped. If .preserve = FALSE (the default), the grouping structure is recalculated based on the resulting data, otherwise the grouping is kept as is.

Value

A corpus containing only documents that satisfy the specified conditions.

Examples

```
data_corpus_inaugural %>%
    filter(Year < 1810) %>%
    summary()
```

left_join.corpus *Join corpus with a data frame*

Description

left_join() adds columns from y to the corpus x, matching documents based on document variables. This is a mutating join that keeps all documents from x and adds matching values from y. If a document in x has no match in y, the new columns will contain NA.

Usage

```
## S3 method for class 'corpus'
left_join(
  x,
  y,
  by = NULL,
  copy = FALSE,
  suffix = c(".x", ".y"),
  ...,
  keep = NULL
)
```

Arguments

`x` a **quanteda** corpus object

`y` a data frame or tibble to join

`by` a join specification. See `dplyr::left_join()` for details. Defaults to natural join using all variables with common names. Can use "docname" to join on document names (see Details).

`copy` if `y` is not a data frame or tibble, should it be copied?

`suffix` if there are non-joined duplicate variables in `x` and `y`, these suffixes will be added to disambiguate

`...` other arguments passed to `dplyr::left_join()`

`keep` should the join keys from both `x` and `y` be preserved?

Value

a corpus with document variables from both `x` and `y`

Special handling of "docname"

This function provides special handling for joining on document names:

- If `by = "docname"` (or "docname" appears in the `by` vector), the function will use `docnames(x)` as the joining column from the corpus, even if "docname" is not a document variable.
- If using `join_by(docname == other_col)`, the function will match `docnames(x)` to `other_col` in `y`.
- If "docname" exists as an actual document variable in `x`, that variable will be used instead of `docnames(x)`.

Examples

```
# Create example corpus and data
corp <- data_corpus_inaugural[1:5]

# Create data to join with document names
doc_data <- data.frame(
  docname = c("1789-Washington", "1793-Washington", "1797-Adams"),
  century = c(18, 18, 18),
  speech_number = c(1, 2, 1)
)

# Join using docname - matches docnames(corp) to doc_data$docname
left_join(corp, doc_data, by = "docname") %>%
  summary()

# Join using different column names with named vector
doc_data2 <- data.frame(
  doc_id = c("1789-Washington", "1793-Washington"),
  rating = c(5, 4)
)
left_join(corp, doc_data2, by = c("docname" = "doc_id")) %>%
  summary()

# Regular join on existing docvars
year_info <- data.frame(
  Year = c(1789, 1793, 1797, 1801, 1805),
  decade = c("1780s", "1790s", "1790s", "1800s", "1800s")
)
left_join(corp, year_info, by = "Year") %>%
  summary()
```

mutate.corpus *Create or transform document variables*

Description

mutate() adds new document variables and preserves existing ones; transmute() adds new document variables and drops existing ones. Both functions preserve the number of rows of the input. New variables overwrite existing variables of the same name.

Usage

```
## S3 method for class 'corpus'
mutate(.data, ...)

## S3 method for class 'corpus'
transmute(.data, ...)
```

Arguments

.data	a **quanteda** object whose document variables will be created or transformed
...	name-value pairs of expressions for document variable modification or assignment; see mutate.

Value

A corpus with new or modified document variables.

Examples

```
data_corpus_inaugural %>%
  mutate(fullname = paste(FirstName, President, sep = ", ")) %>%
  summary(n = 5)

data_corpus_inaugural %>%
  transmute(fullname = paste(FirstName, President, sep = ", ")) %>%
  summary(n = 5)
```

`pull.corpus` *Pull out a single document variable*

Description

Works like `$` for **quanteda** objects with document variables, or like `docvars(x, "varname")`.

Usage

```
## S3 method for class 'corpus'
pull(.data, var = -1, name = NULL, ...)

## S3 method for class 'tokens'
pull(.data, var = -1, name = NULL, ...)

## S3 method for class 'dfm'
pull(.data, var = -1, name = NULL, ...)
```

Arguments

`.data` a **quanteda** object with document variables

`var` A variable specified as:

- a literal variable name
- a positive integer, giving the position counting from the left
- a negative integer, giving the position counting from the right.

The default returns the last column (on the assumption that's the column you've created most recently).

This argument is taken by expression and supports quasiquotation (you can unquote column names and column locations).

`name` An optional parameter that specifies the column to be used as names for a named vector. Specified in a similar manner as `var`.

`...` For use by methods.

Value

A vector containing the values of the specified document variable.

Examples

```
tail(data_corpus_inaugural) %>% pull(President)
tail(data_corpus_inaugural) %>% pull(-1)
tail(data_corpus_inaugural) %>% pull(1)

toks <- data_corpus_inaugural %>%
  tail() %>%
  tokens()
pull(toks, President)

dfmat <- data_corpus_inaugural %>%
  tail() %>%
  tokens() %>%
  dfm()
pull(dfmat, President)
```

`relocate.corpus` *Change column order of document variables*

Description

Use `relocate()` to change the column positions of document variables, using the same syntax as select() to make it easy to move blocks of columns at once.

Usage

```
## S3 method for class 'corpus'
relocate(.data, ...)
```

Arguments

`.data` A data frame, data frame extension (e.g. a tibble), or a lazy data frame (e.g. from dbplyr or dtplyr). See *Methods*, below, for more details.

`...` <`tidy-select`> Columns to move.

Value

A corpus with document variables reordered.

Examples

```
data_corpus_inaugural %>%
  relocate(President, Party) %>%
  summary(n = 5)

data_corpus_inaugural %>%
  relocate(FirstName, President, .before = Year) %>%
  summary(n = 5)
```

rename.corpus *Rename document variables*

Description

`rename()` changes the names of individual document variables using `new_name` = `old_name` syntax; `rename_with()` renames columns using a function.

Usage

```
## S3 method for class 'corpus'
rename(.data, ...)

## S3 method for class 'corpus'
rename_with(.data, .fn, .cols = everything(), ...)
```

Arguments

`.data` a **quanteda** object with document variables

`...` For `rename()`: <`tidy-select`> Use `new_name` = `old_name` to rename selected variables.

For `rename_with()`: additional arguments passed onto `.fn`.

`.fn` A function used to transform the selected `.cols`. Should return a character vector the same length as the input.

`.cols` <`tidy-select`> Columns to rename; defaults to all columns.

Value

A corpus with renamed document variables.

Examples

```
data_corpus_inaugural %>%
  rename(LastName = President) %>%
  summary(n = 5)
data_corpus_inaugural %>%
  rename_with(toupper) %>%
  summary(n = 5)
```

```
data_corpus_inaugural %>%
  rename_with(toupper, starts_with("P")) %>%
  summary(n = 5)
```

select.corpus *Subset docvars using their names and types*

Description

Select (and optionally rename) document variables in a data frame, using a concise mini-language that makes it easy to refer to variables based on their name (e.g. a:f selects all columns from a on the left to f on the right). You can also use predicate functions like is.numeric to select variables based on their properties.

Usage

```
## S3 method for class 'corpus'
select(.data, ...)
```

Arguments

.data	a **quanteda** object with document variables
...	<tidy-select> One or more unquoted expressions separated by commas. Variable names can be used as if they were positions in the data frame, so expressions like x:y can be used to select a range of variables.

Details

For an overview of selection features, see dplyr::select().

Value

A corpus with the specified subset of document variables.

Examples

```
data_corpus_inaugural %>%
  select(Party, Year) %>%
  summary(n = 5)
```

`slice.corpus` *Subset documents using their positions*

Description

`slice()` lets you index documents by their (integer) locations. It allows you to select, remove, and duplicate documents. It is accompanied by a number of helpers for common use cases:

- `slice_head()` and `slice_tail()` select the first or last documents.
- `slice_sample()` randomly selects documents.
- `slice_min()` and `slice_max()` select documents with highest or lowest values of a document variable.

Usage

```
## S3 method for class 'corpus'
slice(.data, ..., .preserve = FALSE)

## S3 method for class 'corpus'
slice_head(.data, ..., n, prop)

## S3 method for class 'corpus'
slice_tail(.data, ..., n, prop)

## S3 method for class 'corpus'
slice_sample(.data, ..., n, prop, weight_by = NULL, replace = FALSE)

## S3 method for class 'corpus'
slice_min(.data, ..., n, prop, with_ties = TRUE)

## S3 method for class 'corpus'
slice_max(.data, ..., n, prop, with_ties = TRUE)
```

Arguments

`.data` a **quanteda** corpus object

`...` additional arguments passed to methods

`.preserve` Relevant when the `.data` input is grouped. If `.preserve = FALSE` (the default), the grouping structure is recalculated based on the resulting data, otherwise the grouping is kept as is.

`n, prop` Provide either `n`, the number of documents, or `prop`, the proportion of documents to select. If neither are supplied, `n = 1` will be used.

If `n` is greater than the number of rows in the group (or `prop > 1`), the result will be silently truncated to the group size. If the `proportion` of a group size is not an integer, it is rounded down.

`weight_by` <`data-masking`> Sampling weights. This must evaluate to a vector of non-negative numbers the same length as the input. Weights are automatically standardised to sum to 1.

`replace` Should sampling be performed with (`TRUE`) or without (`FALSE`, the default) replacement.

`with_ties` Should ties be kept together? The default, `TRUE`, may return more rows than you request. Use `FALSE` to ignore ties, and return the first `n` rows.

Value

An object of the same type as `.data`. The output has the following properties:

- Each document may appear 0, 1, or many times in the output. (If duplicated, then document names will be modified to remain unique.)
- Document variables are not modified.

Examples

```
slice(data_corpus_inaugural, 2:5)
slice(data_corpus_inaugural, 55:n())
slice_head(data_corpus_inaugural, n = 2)
slice_tail(data_corpus_inaugural, n = 3)
slice_tail(data_corpus_inaugural, prop = .05)

set.seed(42)
slice_sample(data_corpus_inaugural, n = 3)
slice_sample(data_corpus_inaugural, prop = .10, replace = TRUE)

data_corpus_inaugural <- data_corpus_inaugural %>%
    mutate(ntoks = ntoken(data_corpus_inaugural))
# shortest three texts
slice_min(data_corpus_inaugural, ntoks, n = 3)
# longest three texts
slice_max(data_corpus_inaugural, ntoks, n = 3)
```

Index