Package 'skimr'

February 25, 2019

Title Compact and Flexible Summaries of Data

Version 1.0.5

Description A simple to use summary function that can be used with pipes and displays nicely in the console. The default summary statistics may be modified by the user as can the default formatting. Support for data frames and vectors is included, and users can implement their own skim methods for specific object types as described in a vignette. Default summaries include support for inline spark graphs. Instructions for managing these on specific operating systems are given in the ``Using skimr'' vignette and the README.

Depends R (>= 3.1.2)

Imports cli, dplyr (>= 0.7), magrittr, pander, purrr, rlang, stats, stringr (>= 1.1), knitr (>= 1.2), tibble (>= 0.6), tidyr (>= 0.7), tidyselect (>= 0.2.4)

Suggests extrafont, rmarkdown, testthat (>= 2.0.0), withr, covr

License GPL-3

Encoding UTF-8

LazyData true

URL https://github.com/ropenscilabs/skimr

BugReports https://github.com/ropenscilabs/skimr/issues

VignetteBuilder knitr

RoxygenNote 6.1.1

Collate 'dplyr.r' 'skimr-package.R' 'formats.R' 'stats.R' 'functions.R' 'kable.R' 'options.R' 'pander.R' 'skim.R' 'skim_print.R' 'skim_v.R' 'summary.R' 'utils.R' 'wide.R'

NeedsCompilation no

Author Amelia McNamara [aut],
Eduardo Arino de la Rubia [aut],
Hao Zhu [aut],
Julia Lowndes [ctb],
Shannon Ellis [aut],

Elin Waring [cre],
Michael Quinn [aut],
Hope McLeod [ctb],
Hadley Wickham [ctb],
Kirill Müller [ctb],
RStudio, Inc. [cph] (Spark functions),
Connor Kirkpatrick [ctb],
Scott Brenstuhl [ctb],
Patrick Schratz [ctb],
lbusett [ctb],
Mikko Korpela [ctb],
Jennifer Thompson [ctb],
Harris McGehee [ctb],
Patrick Kennedy [ctb]

Maintainer Elin Waring <elin.waring@gmail.com>

Repository CRAN

Date/Publication 2019-02-25 05:50:02 UTC

R topics documented:

`skimr-package` *Skim a data frame*

Description

This package provides an alternative to the default summary functions within R. The package's API is tidy, functions take data frames, return data frames and can work as part of a pipeline. The returned `skimr` object is subsettable and offers a human readable output.

Details

skimr is opinionated, providing a strong set of summary statistics that are generated for a variety of different data types. It is also provides an API for customization. Users can change both the functions dispatched and the way the results are formatted.

`arrange.skim_df` *Use dplyr verb arrange on skim_df objects.*

Description

Use dplyr verb arrange on skim_df objects.

Usage

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

Arguments

`.data`	A tbl. All main verbs are S3 generics and provide methods for `tbl_df()`, `dtplyr::tbl_dt()` and `dbplyr::tbl_dbi()`.
`...`	Comma separated list of unquoted variable names, or expressions involving variable names. Use `desc()` to sort a variable in descending order.

Value

skim_df object coerced to a data frame.

See Also

`dplyr::arrange()`

`filter.skim_df` *Use dplyr verb filter on skim_df objects.*

Description

Use dplyr verb filter on skim_df objects.

Usage

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

Arguments

.data	A tbl. All main verbs are S3 generics and provide methods for tbl_df(), dtplyr::tbl_dt() and dbplyr::tbl_dbi().
...	Logical predicates defined in terms of the variables in .data. Multiple conditions are combined with &. Only rows where the condition evaluates to TRUE are kept. The arguments in ... are automatically quoted and evaluated in the context of the data frame. They support unquoting and splicing. See vignette("programming") for an introduction to these concepts.
.preserve	when FALSE (the default), the grouping structure is recalculated based on the resulting data, otherwise it is kept as is.

Value

skim_df object coerced to a data frame.

See Also

dplyr::filter()

---

fix_windows_histograms

Fix unicode histograms on Windows

Description

This functions changes your session's locale to address issues with printing histograms on Windows.

Usage

```
fix_windows_histograms()
```

Details

There are known issues with printing the spark-histogram characters when printing a data frame, appearing like this: "<U+2582><U+2585><U+2587>". This longstanding problem originates in the low-level code for printing dataframes.

kable *Create tables in LaTeX, HTML, Markdown and reStructuredText*

Description

This is a very simple table generator. It is simple by design. It is not intended to replace any other R packages for making tables.

Usage

```
kable(x, format = NULL, digits = getOption("digits"), row.names = NA,
  col.names = NA, align, caption = NULL, format.args = list(),
  escape = TRUE, ...)

## S3 method for class 'skim_df'
kable(x, format = NULL, digits = getOption("digits"),
  row.names = NA, col.names = NA, align = NULL, caption = NULL,
  format.args = list(), escape = TRUE, ...)

## S3 method for class 'summary_skim_df'
kable(x, ...)
```

Arguments

x	An R object, typically a matrix or data frame.
format	A character string. Possible values are latex, html, markdown, pandoc, and rst; this will be automatically determined if the function is called within **knitr**; it can also be set in the global option knitr.table.format. If format is a function, it must return a character string.
digits	Maximum number of digits for numeric columns, passed to round(). This can also be a vector of length ncol(x), to set the number of digits for individual columns.
row.names	Logical: whether to include row names. By default, row names are included if rownames(x) is neither NULL nor identical to 1:nrow(x).
col.names	A character vector of column names to be used in the table.
align	Column alignment: a character vector consisting of 'l' (left), 'c' (center) and/or 'r' (right). By default or if align = NULL, numeric columns are right-aligned, and other columns are left-aligned. If length(align) == 1L, the string will be expanded to a vector of individual letters, e.g. 'clc' becomes c('c', 'l', 'c'), unless the output format is LaTeX.
caption	The table caption.
format.args	A list of arguments to be passed to format() to format table values, e.g. list(big.mark = ',').
escape	Boolean; whether to escape special characters when producing HTML or LaTeX tables.
...	Other arguments (see Examples).

Value

The original skim_df object.

Methods (by class)

- skim_df: Produce kable output of a skimmed data frame
- summary_skim_df: Kable method for a summary_skim_df object

See Also

knitr::kable()

---

mutate.skim_df *Use dplyr verb mutate on skim_df objects.*

Description

Use dplyr verb mutate on skim_df objects.

Usage

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

Arguments

.data	A tbl. All main verbs are S3 generics and provide methods for tbl_df(), dtplyr::tbl_dt() and dbplyr::tbl_dbi().
...	Name-value pairs of expressions, each with length 1 or the same length as the number of rows in the group (if using group_by()) or in the entire input (if not using groups). The name of each argument will be the name of a new variable, and the value will be its corresponding value. Use a NULL value in mutate to drop a variable. New variables overwrite existing variables of the same name. The arguments in ... are automatically quoted and evaluated in the context of the data frame. They support unquoting and splicing. See vignette("programming") for an introduction to these concepts.

Value

skim_df object coerced to a data frame.

See Also

dplyr::mutate()

pander *Render data frames as markdown*

Description

Render data frames as markdown

Usage

```
pander(x, caption = attr(x, "caption"), ...)

pander.skim_df(x, caption = attr(x, "caption"), ...)

pander.summary_skim_df(x, ...)
```

Arguments

x	Object to be processed
caption	A character scalar specifying the table's caption.
...	Further arguments passed to or from other methods.

Functions

- pander: Produce pander output
- pander.skim_df: Produce pander output of a skimmed data frame
- pander.summary_skim_df: Pander method for a summary_skim_df object

See Also

pander::pander()

print *Print* skim *objects*

Description

Print skim objects

Usage

```
## S3 method for class 'skim_df'
print(x, ...)

## S3 method for class 'skim_vector'
print(x, ...)

## S3 method for class 'summary_skim_df'
print(x, ...)
```

Arguments

x	Either a skim_df, skim_vector or skim_summary object.
...	Further arguments passed to or from other methods.

Methods (by class)

- skim_df: Prints a skimmed data frame (skim_df from skim())
- skim_vector: Manages print for skim_vector objects.
- summary_skim_df: Print method for a summary_skim_df object.

---

select.skim_df *Use dplyr verb select on skim_df objects.*

Description

Use dplyr verb select on skim_df objects.

Usage

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

Arguments

.data	A tbl. All main verbs are S3 generics and provide methods for tbl_df(), dtplyr::tbl_dt() and dbplyr::tbl_dbi().
...	One or more unquoted expressions separated by commas. You can treat variable names like they are positions, so you can use expressions like x:y to select ranges of variables. Positive values select variables; negative values drop variables. If the first expression is negative, select() will automatically start with all variables. Use named arguments, e.g. new_name = old_name, to rename selected variables.

The arguments in ... are automatically quoted and evaluated in a context where column names represent column positions. They also support unquoting and splicing. See `vignette("programming")` for an introduction to these concepts.

See select helpers for more details and examples about tidyselect helpers such as `starts_with()`, `everything()`, ...

Value

skim_df object coerced to a data frame.

See Also

`dplyr::select()`

`skim` *Skim a data frame, getting useful summary statistics*

Description

`skim()` is an alternative to `summary()`, quickly providing a broad overview of a data frame. It handles data of all types, dispatching a different set of summary functions based on the types of columns in the data frame.

Usage

```
skim(.data, ...)

skim_tee(.data, ...)
```

Arguments

`.data`	A tibble, or an object that can be coerced into a tibble.
`...`	Additional options, normally used to list individual unquoted column names.

Details

Each call produces a `skim_df`, which is a fundamentally a tibble with a special print method. Instead of showing the result in a long format, `skim` prints a wide version of your data with formatting applied to each column. Printing does not change the structure of the `skim_df`, which remains a long tibble.

If you just want to see the printed output, call `skim_tee()` instead. This function returns the original data frame.

If you want to work with a data frame that resembles the printed output, call `skim_to_wide()` or for a named list of data frames by type `skim_to_list()`. Note that all of the columns in the data frames produced by these functions are character. The intent is that you will be processing the **printed** result further, not the original data.

skim() is designed to operate in pipes and to generally play nicely with other tidyverse functions. This means that you can use tidyselect helpers within skim to select or drop specific columns for summary. You can also further work with a skim_df using dplyr functions in a pipeline.

Value

A skim_df object, which can be treated like a tibble in most instances.

Customizing skim

skim() is an intentionally simple function, with minimal arguments like summary(). Nonetheless, this package provides two broad approaches to how you can customize skim()'s behavior. You can customize the functions that are called to produce summary statistics with skim_with(). You can customize how the output is displayed with skim_format().

Unicode rendering

If the rendered examples show unencoded values such as <U+2587> you will need to change your locale to allow proper rendering. Please review the *Using Skimr* vignette for more information (vignette("Using_skimr", package = "skimr")).

Examples

```
skim(iris)

# Use tidyselect
skim(iris, Species)
skim(iris, starts_with("Sepal"))

# Skim also works groupwise
dplyr::group_by(iris, Species) %>% skim()

# Skim pipelines; now we work with the tall format
skim(iris) %>% as.data.frame()
skim(iris) %>% dplyr::filter(type == "factor")

# Which column as the greatest mean value?
skim(iris) %>%
  dplyr::filter(stat == "mean") %>%
  dplyr::arrange(dplyr::desc(value))

# Use skim_tee to view the skim results and
# continue using the original data.
chickwts %>% skim_tee() %>% dplyr::filter(feed == "sunflower")
```

`skim_format` *Change the formatting options for printed skim objects*

Description

Formats are dispatched according to the type of value returned by the "skimmer," i.e. summary function. One special formatting "type" exists for the names of the returned vector. The names are used to assign the levels for statistics that have more than one value. Counts and quantiles are common cases.

Usage

```
skim_format(..., .list = list(), append = TRUE, drop_new = FALSE)

skim_format_defaults()

show_formats(which = NULL)
```

Arguments

`...`	Named arguments that contain named lists specifying formats to apply.
`.list`	Instead of individual named entries, you can provided a list instead. If most `...` arguments and `.list` is provided, values in `.list` take precedence.
`append`	Whether the provided options should be in addition to the defaults already in `skim`. Default is `TRUE`.
`drop_new`	Whether types outside of the defaults should be discarded.
`which`	A character vector. One or more of the classes whose formatting options you wish to display.

Details

When a vector is named, the name and the value are combined into a single formatted value. To deal with excessively long names for factor levels, only the first three characters of the name are returned by default. This can be changed by setting a new value for `max_char` within the `.levels` type.

Skim uses `format()` to convert the numeric values returned by the summary functions into displayed values. The default options are a subset of options available in that function.

Value

When setting formatting options, `invisible(NULL)`. When looking up values, a list of option-value pairs.

Functions

- skim_format_defaults: Use the default formatting options within skim
- show_formats: Show formatting options currently used, by data type. For each data type, options are returned as a list of option-value pairs.

Examples

```
# Format numbers to have more digits
skim_format(numeric = list(digits = 3))

# Show the values for the formats
show_formats()

# Show 4-character names in factor levels
skim_format(.levels = list(max_char = 4))

# Set multiple formats
skim_format(numeric = list(digits = 3), .levels = list(max_char = 4))

# Set multiple formats with a .list argument
my_formats <- list(numeric = list(digits = 3), .levels = list(max_char = 4))
skim_format(.list = my_formats)

# Alternatively, use rlang unquoting semantics
skim_format(!!!my_formats)

# Reset to the defaults
skim_format_defaults()
```

---

skim_with *Set or add the summary functions for a particular type of data*

Description

While skim is designed around having an opinionated set of defaults, you can use these functions to change the summary statistics that it returns. To do that, provide type you wish to change as an argument to this function, along with a list of named functions that you want to use instead of the defaults.

Usage

```
skim_with(..., .list = list(), append = TRUE, drop_new = FALSE)

skim_with_defaults()

show_skimmers(which = NULL)

get_skimmers(which = NULL)
```

Arguments

...	A list of functions, with an argument name that matches a particular data type.
.list	Instead of individual named entries, you can provided a list instead. If most ... arguments and .list is provided, values in .list take precedence.
append	Whether the provided options should be in addition to the defaults already in skim. Default is TRUE.
drop_new	Whether types outside of the defaults should be discarded.
which	A character vector. One or more of the classes whose summary functions you wish to display.

Details

This function is not pure. It sets values in within the package environment. This is an intentional design choice, with effects similar to setting options in base R. By setting options here for your entire session, you can continue to summarize using skim on its own.

Value

When setting values, invisible(NULL). When looking up values a list. The names of the list match the classes that have assigned summary functions. With show_skimmers(), each entry in the list is a character vector of function names. With get_skimmers(), each entry in the list is itself a list of named functions.

Functions

- skim_with_defaults: Use the default functions within skim
- show_skimmers: Access the names of the summary functions for a class.
- get_skimmers: Pull a list of summary functions for a class.

Examples

```
# Use new functions for numeric functions
skim_with(numeric = list(median = median, mad = mad), append = FALSE)
skim(faithful)

# If you want to remove a particular skimmer, set it to NULL
# This removes the inline histogram
skim_with(numeric = list(hist = NULL))
skim(faithful)

# This works with multiple skimmers. Just match names to overwrite
skim_with(numeric = list(iqr = IQR, p25 = NULL, p75 = NULL))
skim(faithful)

# Alternatively, set `append = FALSE` to replace the skimmers of a type.
skim_with(numeric = list(mean = mean, sd = sd), append = FALSE)

# Skimmers are unary functions. Partially apply arguments during assigment.
```

```
# For example, you might want to remove NA values.
skim_with(numeric = list(iqr = purrr::partial(IQR, na.rm = TRUE)))

# Or use an rlang-style formula constructor for the function
skim_with(numeric = list(med = ~median(., na.rm = TRUE)))

# Set multiple types of skimmers simultaneously
skim_with(numeric = list(mean = mean), character = list(len = length))

# Or pass the same as a list
my_skimmers <- list(numeric = list(mean = mean),
                    character = list(len = length))
skim_with(.list = my_skimmers)

# Alternatively, use rlang unquoting semantics
skim_with(!!!my_skimmers)

# Go back to defaults
skim_with_defaults()
skim(faithful)

# What are the names of the numeric skimmers?
show_skimmers("numeric")

# I want to create a set of skimmers for the hms class, using the date
# skimmers currently available.
funs <- get_skimmers()
skim_with(hms = funs$date)
```

`slice.skim_df` *Use dplyr verb slice on skim_df objects.*

Description

Use dplyr verb slice on skim_df objects.

Usage

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

Arguments

`.data`	A tbl. All main verbs are S3 generics and provide methods for `tbl_df()`, `dtplyr::tbl_dt()` and `dbplyr::tbl_dbi()`.
`...`	Logical predicates defined in terms of the variables in `.data`. Multiple conditions are combined with `&`. Only rows where the condition evaluates to `TRUE` are kept.

	The arguments in ... are automatically quoted and evaluated in the context of the data frame. They support unquoting and splicing. See `vignette("programming")` for an introduction to these concepts.
`.preserve`	when `FALSE` (the default), the grouping structure is recalculated based on the resulting data, otherwise it is kept as is.

Value

skim_df object coerced to a data frame.

See Also

`dplyr::slice()`

`stats` *Summary statistic functions*

Description

Skimr provides extensions to a variety of functions with R's stats package to simplify creating summaries of data. All functions are vectorized and take a single argument. Other parameters for these functions are set in the `skim_format()` function.

Usage

```
n_missing(x)

n_complete(x)

sorted_count(x)

inline_hist(x)

n_empty(x)

min_char(x)

max_char(x)

n_unique(x)

ts_start(x)

ts_end(x)

inline_linegraph(x)
```

```
list_lengths_min(x)

list_lengths_median(x)

list_lengths_max(x)

list_min_length(x)

list_max_length(x)
```

Arguments

x	A vector

Functions

- `n_missing`: Calculate the sum of `NA` and `NULL` (i.e. missing) values.
- `n_complete`: Calculate complete values; complete values are not missing.
- `sorted_count`: Create a contingency table and arrange its levels in descending order. In case of ties, the ordering of results is alphabetical and depends upon the locale. `NA` is treated as a ordinary value for sorting.
- `inline_hist`: Generate inline histogram for numeric variables. The character length of the histogram is controlled by the formatting options for character vectors.
- `n_empty`: Calculate the number of blank values in a character vector. A "blank" is equal to "".
- `min_char`: Calculate the minimum number of characters within a character vector.
- `max_char`: Calculate the maximum number of characters within a character vector.
- `n_unique`: Calculate the number of unique elements but remove `NA`.
- `ts_start`: Get the start for a time series without the frequency.
- `ts_end`: Get the finish for a time series without the frequency.
- `inline_linegraph`: Generate inline line graph for time series variables. The character length of the line graph is controlled by the formatting options for character vectors. Based on the function in the pillar package.
- `list_lengths_min`: Get the length of the shortest list in a vector of lists.
- `list_lengths_median`: Get the median length of the lists.
- `list_lengths_max`: Get the maximum length of the lists.
- `list_min_length`: Get the length of the shortest list in a vector of lists.
- `list_max_length`: Get the length of the longest list in a vector of lists.

See Also

`skim_format()` and `purrr::partial()` for setting arguments of a skimmer function.

summary.skim_df *Summary function for skim_df*

Description

This is a method of the generic function `summary()`.

Usage

```
## S3 method for class 'skim_df'
summary(object, ...)
```

Arguments

`object`	a skim dataframe.
`...`	Additional arguments affecting the summary produced. Not used.

Value

A summary of the skim data frame.

Examples

```
## Not run:
 a <- skim(mtcars)
 summary(a)

## End(Not run)
```

wide *Working with* `skimr`*'s printed output*

Description

These functions provide two approaches for handling the wide format produced when you print `skim_df`. `skim_to_wide()` returns a wide data frame with one row per variable and `NA` for statistics not calculated for a given type. `skim_to_list()` creates a list of wide tibbles, one for each type of vector within your data frame.

Usage

```
skim_to_wide(x, ...)

skim_to_list(x, ...)
```

Arguments

x	A data frame.
...	Further arguments passed to or from other methods.

Details

Note that in both cases, all columns are character vectors. This gives you additional control of the printed output, but not the original data.

Value

A wide data frame or a list of wide data frames.

Examples

```
# Treat the printed output as a wide data frame
skim_to_wide(iris)
iris %>% skim_to_wide()
iris %>%
  skim_to_wide() %>%
  dplyr::filter(type == "factor") %>%
  dplyr::select(top_counts)

# Treat the printed output as a list of data frames
skim_to_list(iris)
iris %>% skim_to_list()

# Save the result
sl <- iris %>% skim_to_list()
sl[["numeric"]]
kable(sl$numeric)

# Or grouped, this uses the magrittr exposition pipe
# see ?magrittr::`%$%`
library(magrittr)
iris %>%
  dplyr::group_by(Species) %>%
  skim_to_list() %$%
  kable(numeric)
```

Index