Package ‘statquotes’

October 10, 2023

Title Quotes on Statistics, Data Visualization and Science
Version 0.3.2
Date 2023-10-09
Language en-US
Description Generates a random quotation from a database of quotes on topics in statistics, data visualization and science. Other functions allow searching the quotes database by key term tags, or authors or creating a word cloud. The output is designed to be suitable for use at the console, in Rmarkdown and LaTeX.

Depends R (>= 3.5.0)
License GPL (>= 2)
Encoding UTF-8
LazyData true
Maintainer Michael Friendly <friendly@yorku.ca>

BugReports https://github.com/friendly/statquotes/issues
URL https://github.com/friendly/statquotes/
Imports stringr, tidytext, wordcloud
RoxygenNote 7.2.3
Suggests knitr, rmarkdown, dplyr, forcats, ggplot2
VignetteBuilder knitr
NeedsCompilation no

Author Michael Friendly [aut, cre] (<https://orcid.org/0000-0002-3237-0941>),
Kevin Wright [aut] (<https://orcid.org/0000-0002-0617-8673>),
Phil Chalmers [aut],
Matthew Sigal [ctb]

Repository CRAN
Date/Publication 2023-10-10 17:00:06 UTC
\section*{R topics documented:}

\begin{verbatim}
as.latex ......................................................... 2
as.markdown ....................................................... 3
find_duplicate_quotes ............................................. 4
quotes .............................................................. 5
quote_cloud ......................................................... 5
quote_tags .......................................................... 6
read_quotes_raw .................................................... 7
search_quotes ....................................................... 7
statquote ........................................................... 9
\end{verbatim}

\section*{Index}

\begin{verbatim}
  as.latex ......................................................... 11
\end{verbatim}

---

\textit{as.latex} \textit{Coerces statquote objects to strings suitable for \LaTeX}

\section*{Description}

This function coerces statquote objects to strings suitable for rendering in \LaTeX. Quotes and (potential \LaTeX) sources are placed within suitable "epigraph" output format via the \texttt{sprintf} function.

\section*{Usage}

\begin{verbatim}
as.latex(quotes, form = "\\epigraph{%s}{%s}\n\n", cite = TRUE)
\end{verbatim}

\section*{Arguments}

\begin{verbatim}
quotes \hspace{1cm} an object of class \texttt{statquote} returned from functions such as \texttt{search_quotes} or \texttt{statquote}
form \hspace{1cm} structure of the \LaTeX output for the text (first argument) and source (second argument) passed to \texttt{sprintf}
cite \hspace{1cm} logical; should the \texttt{cite} field be included in the source output?
\end{verbatim}

\section*{Value}

character vector of formatted \LaTeX quotes

\section*{Author(s)}

Phil Chalmers

\section*{See Also}

\begin{verbatim}
  as.data.frame.statquote, as.markdown
\end{verbatim}
as.markdown

Examples

```r
ll <- search_quotes("Tukey")
as.latex(ll)
```

Description

This function coerces statquote objects to strings suitable for rendering in markdown. Quotes and sources are placed within output formatted via the `sprintf` function.

This function formats a statquote object to the tagged key:value format used for maintaining the statquotes database. The key names are:

- `quo`: This is a quotation.
- `src`: Person or persons who said or wrote the quote.
- `cit`: Citation for the original quote.
- `url`: URL where the quote can be found (such as journal articles).
- `tag`: Comma-separated tags to categorize the quote.
- `tex`: TeX-formatted citation

Usage

```r
as.markdown(quotes, form = "> *%s* -- %s

", cite = TRUE)

as.tagged(quotes, qid = TRUE)
```

Arguments

- `quotes`: an object of class `statquote` returned from functions such as `search_quotes` or `statquote`
- `form`: structure of the markdown output for the text (first argument) and source (second argument) passed to `sprintf`
- `cite`: logical; should the `cite` field be included in the source output?
- `qid`: logical. Should the quote id number ‘qid’ be included in the output?

Value

- character vector of formatted markdown quotes
- A character vector of lines
find_duplicate_quotes

Check for duplicate quotes

Description

Returns a list with aggressively fuzzy matched quotations, along with their relevant citation information.

Usage

find_duplicate_quotes()

Author(s)

Phil Chalmers

Examples

# As the number of quotes has grown, this has become very slow.
# dups <- find_duplicate_quotes()
 quotes

Quotes on statistics, data visualization and science

Description
A data frame with quotations. The variables are:

Usage
data(quotes)

Format
A data frame

Details
- qid quote ID, a numeric vector
- text text of the quote
- source person(s) who said the quote.
- citation citation of the quote
- url URL of the quote
- tags tags used for searching
- tex TeX-style citation

quote_cloud
Generate a word cloud based upon quote database

Description
This function takes a search pattern (or regular expression) and generates a word cloud based upon that filter.

Usage
quote_cloud(search = ".*", max.words = 80, colors, ...)

Arguments
- search Character string (or regular expression) used to search the database. Default is to search all quotes.
- max.words Integer; The maximum number of words to be plotted.
- colors A character vector of colors to be used to designate word frequency. The default is 5 levels, from light to dark green.
- ... additional arguments passed to search_quotes and wordcloud
quote_tags

Value
None. A wordcloud is plotted.

See Also
statquote, quote_tags, quotes, search_quotes, wordcloud

Examples
quote_cloud()
quote_cloud(search = "graph")
quote_cloud(max.words = 10)

quote_tags List the tags of the quotes database

Description
This function finds the unique tags of items in the quotes database and returns them as vector or a one-way table giving their frequencies.

Usage
quote_tags(table = FALSE)

Arguments
table Logical. If table=TRUE, return a one-way frequency table of quotes for each tag; otherwise return the sorted vector of unique tags.

Value
Returns either a vector of tags in the quotes database or a one-way frequency table of the number of quotes for each tag.

Examples
quote_tags()
quote_tags(table=TRUE)

library(ggplot2)
qt <- quote_tags(table=TRUE)
qtdf <- as.data.frame(qt)
# bar plot of frequencies
ggplot2::ggplot(data=qtdf, aes(x=Freq, y=tags)) + geom_bar(stat = "identity")

# Sort tags by frequency
read_quotes_raw

```r
dplyr::mutate(tags = forcats::fct_reorder(tags, Freq)) |>
ggplot2::ggplot(aes(x=Freq, y=tags)) +
geom_bar(stat = "identity")
```

**read_quotes_raw**  
Parse quotes from the file quotes_raw.txt.

**Description**

There should be no reason for a person to call this function. This function parses 'data-raw/quotes_raw.txt'. The resulting dataframe is then saved to 'data/quotes.rda'. Although it would be possible to use this function to parse the quotes when loading the package, that would make it much slower to load the package.

**Usage**

```r
read_quotes_raw(file = file.path(getwd(), "data-raw/quotes_raw.txt"))
```

**Arguments**

- **file**  
The file of raw quotes.

**Value**

Dataframe with quotes

---

**search_quotes**  
Search the quote database for a string or regex pattern

**Description**

This function takes a search pattern (or regular expression) and returns all quotes that match the pattern.

A convenient wrapper for search quotes that by default returns all quotes

**Usage**

```r
search_quotes(
  search,
  ignore_case = TRUE,
  fuzzy = FALSE,
  fields = c("text", "source", "tags", "cite"),
  ...
)
```
search_quotes

search_text(search, fuzzy = FALSE, ...)

get_quotes(search = ".*", ...)

Arguments

search A character string or regex pattern to search the database.
ignore_case If TRUE, matching is done without regard to case.
fuzzy If TRUE, use `agrep` to allow approximate matches to the search string.
fields A character vector of the particular fields to search. The default is `c("text","source","tags")`. You can use the shortcut `fields="all"` to search all fields (including citation, url).
... additional arguments passed to `agrep` to fine-tune fuzzy search parameters.

Value

A data frame (also with class 'statquote') object containing all quotes that match the search parameters.

See Also

`agrep`, `statquote`.

Examples

search_quotes("^D") # regex to find all quotes that start with "D"
search_quotes("Tukey") # all quotes with "Tukey"
search_quotes("Turkey", fuzzy = TRUE) # fuzzy match

# to a data.frame
out <- search_quotes("bad data", fuzzy = TRUE)
as.data.frame(out)

search_text("omnibus")
qdb <- get_quotes()
nrow(qdb)
names(qdb)
**statquote**

*Display a randomly chosen statistical quote.*

**Description**

Display a randomly chosen statistical quote.

**Usage**

```r
c
statquote(ind = NULL, pattern = NULL, tag = NULL, source = NULL, topic = NULL)
```

```r
## S3 method for class 'statquote'
print(x, cite = TRUE, width = NULL, ...)
```

```r
## S3 method for class 'statquote'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

**Arguments**

- **ind**: Integer or character. If 'ind' is missing, a random quote is chosen from all quotations. If 'ind' is specified and is an integer, return the ind\(^{th}\) quote. If 'ind' is specified and is character, use it as the 'pattern'.
- **pattern**: Character string. Quotes are subset to those which match the pattern in the quote text.
- **tag**: Character string. Quotes are subset to those matching the specified tag.
- **source**: Character string. Quotes are subset to those matching the specified source (person).
- **topic**: Deprecated. Use 'tag' instead. Only kept for backward compatibility.
- **x**: object of class 'statquote'
- **cite**: logical; should the cite field be printed?
- **width**: Optional print width parameter
- **...**: Other optional arguments, unused here
- **row.names**: see `as.data.frame`
- **optional**: see `as.data.frame`

**Value**

A character vector containing one quote. It is of class statquote for which an S3 print method will be invoked, and for which other methods are available.

**See Also**

- `quote_tags`, `search_quotes`, `quotes`, Inspired by: `fortune`
- `as.latex`, `as.markdown`
Examples

```r
set.seed(1234)
statquote()
statquote(10)
statquote("boggled")
statquote(pattern="boggled")
statquote(source="Yates")
statquote(tag="anova")
print.data.frame(statquote(302))  # All information
```
Index

* datasets
  quotes, 5

agrep, 8
as.data.frame, 9
as.data.frame.statquote, 2, 4
as.data.frame.statquote(statquote), 9
as.latex, 2, 4, 9
as.markdown, 2, 3, 4, 9
as.tagged(as.markdown), 3

find_duplicate_quotes, 4
fortune, 9

get_quotes(search_quotes), 7

print.statquote(statquote), 9

quote_cloud, 5
quote_tags, 6, 6, 9
quotes, 5, 6, 9

read_quotes_raw, 7

search_quotes, 2, 3, 5, 6, 7, 9
search_text(search_quotes), 7
sprintf, 2, 3
statquote, 2, 3, 6, 8, 9

wordcloud, 5, 6