Package 'dhmeasures'

March 23, 2026

Title Digital History Measures

Version 1.0

Maintainer Steph Buongiorno <steph.buon@gmail.com>

Description Provides statistical functions to aid in the analysis of contemporary and historical corpora. These transparent functions may be useful to anyone, and were designed with the social sciences and humanities in mind. JSD (Jensen-Shannon Divergence) is a measure of the distance between two probability distributions. The JSD and Original JSD functions expand on existing functions, by calculating the JSD for distributions of words in text groups for all pairwise groups provided (Drost (2018) <doi:10.21105/joss.00765>). The Log Likelihood function is inspired by the work of digital historian Jo Guldi (Guldi (2022) <https://github.com/joguldi/digital-history>). Also includes helper functions that can count word frequency in each text grouping, and remove stop words.

URL https://github.com/stephbuon/dhmeasures

BugReports https://github.com/stephbuon/dhmeasures/issues

License MIT + file LICENSE

Encoding UTF-8

Imports Rcpp (>= 1.0.9), dplyr, tidytext

LinkingTo Rcpp

RoxygenNote 7.3.3

LazyData true

Depends R (>= 3.5.0)

NeedsCompilation yes

Author Ryan Schaefer [aut] (ORCID: <https://orcid.org/0000-0001-7694-3994>), Steph Buongiorno [aut, cre] (ORCID: <https://orcid.org/0000-0002-6965-0787>)

Repository CRAN

Date/Publication 2026-03-23 09:50:02 UTC

Contents

`count_tokens` *Count Tokens*

Description

Converts a data frame with columns for text and grouping variables into a data frame with each word and the count of each word in each group.

Usage

```
count_tokens(data, group = NA, text = "text")
```

Arguments

`data`	Data frame containing the raw data
`group`	The name of the column(s) containing the grouping variable. If not defined, the text will not be grouped. Can be given as either a string or a vector of strings.
`text`	The name of the column containing the text that needs to be tokenized

Value

Data frame containing columns for the word, the group(s) and the count labeled as 'word', the group name, and 'n'

Examples

```
test = data.frame (
  myText = c(
    "Hello! This is the first sentence I am using to test this function!",
    "This is the second sentence!"
  ),
  myGroup = c(
    "group1",
    "group2"
  )
)
```

```
count_tokens(test, text = "myText", group = "myGroup")
```

`dhmeasures` *Digital History Measures*

Description

Provides statistical and other helper functions to aid in the analysis of historical corpuses.

Details

The current statistical functions include Log Likelihood (log_likelihood), JSD (jsd) and Partial JSD (partial_jsd). The current helper functions include tokenize_counts.

Author(s)

Steph Buongiorno and Ryan Schaefer

`hansard_1870_example` *The 19th-century British Parliamentary debates for the decade 1820*

Description

Hansard corpus data for the decade 1820. This data has been pre-formatted to contain words counts by speaker. Stopwords have been removed from the data. To access the raw Hansard data, install the package hansardr. The variables are as follows:

Usage

```
hansard_1870_example
```

Format

A data frame with 482923 rows and 3 variables:

speaker The name of the speaker originally recorded in the transciptions of the debates.

word A word spoken by the given speaker.

n The number of times the given word was spoken by the given speaker

Source

../data/hansard_1870_example.RData

References

Buongiorno, Steph; Kalescky, Robert; Godat, Eric; Cerpa, Omar Alexander; Guldi, Jo (2021)

Examples

```
data(hansard_1870_example)
```

`jsd` *JSD*

Description

Calculates the JSD score for each word between group pairings. To use this function, the user must provide a data frame with a column for words, a column for the text group, and a column for the count of the word in that group. The default column names are "word", "group", and "n", but these can be changed using the parameters word, group, and n. The default settings will calculate the JSD for all words between the first two groups in the data set. However, the user can provide a list of words using the word_list parameter and/or a list of groups using the group_list parameter. If more than two groups are given, the function will provide the JSD scores all all pairs of groups.

Usage

```
jsd(
  text,
  group_list = as.character(c()),
  word_list = as.character(c()),
  group = "group",
  word = "word",
  n = "n"
)
```

Arguments

`text`	Data frame containing data
`group_list`	Vector containing all groups to find pairwise JSD scores for
`word_list`	Vector containing all words to find JSD scores for
`group`	Name of data frame column containing text group
`word`	Name of data frame column containing words
`n`	Name of data frame column containing word count in text group

Value

Data frame containing a column containing unique words and columns for JSD scores for each group pair

Examples

```
# Load example Hansard 1870 dataset
data(hansard_1870_example)
head(hansard_1870_example)

# Calculate JSD for given words and groups
output = jsd(
  hansard_1870_example,
  group = "speaker",
  group_list = c("MR. GLADSTONE", "MR. DISRAELI"),
  word_list = c("trade", "press", "industry")
)
head(output)
```

---

log_likelihood *Log Likelihood*

Description

Calculates word distinctiveness using the log likelihood algorithm. You input a data frame with columns for the word, the text group, and the number of times that word appears in that group. The column names are set to "word", "group", and "n" by default but they can be changed using the parameters word, group, and n. If any of these columns are not found, the function will not work. The output will be a new data frame with a column called "word" containing all unique words and subsequent columns for all unique groups with the name of that group. The data frame will contain the log likelihood scores for each word in each group. The larger a log likelihood score is, the more distinctive that word is to that group.

Usage

```
log_likelihood(
  text,
  group_list = as.character(c()),
  word_list = as.character(c()),
  group = "group",
  word = "word",
  n = "n"
)
```

Arguments

text	Data frame containing data
group_list	Vector containing all groups to find log likelihood scores for
word_list	Vector containing all words to find log likelihood scores for
group	Name of data frame column containing text group
word	Name of data frame column containing words
n	Name of data frame column containing word count in text group

Value

Data frame containing a column containing unique words and columns for log likelihood scores for each group

Examples

```
# Load example Hansard 1820 dataset
data(hansard_1870_example)
head(hansard_1870_example)

# Compute log likelihood
output = log_likelihood(
  hansard_1870_example,
  group = "speaker",
  group_list = c("MR. GLADSTONE", "MR. DISRAELI"),
  word_list = c("trade", "press", "industry")
)
head(output)
```

---

original_jsd *Original JSD*

Description

Calculates the JSD score between text groups. To use this function, the user must provide a data frame with a column for words, a column for the text group, and a column for the count of the word in that group. The default column names are "word", "group", and "n", but these can be changed using the parameters word, group, and n. The default settings will calculate the JSD for all words between the first two groups in the data set. However, the user can provide a list of words using the word_list parameter and/or a list of groups using the group_list parameter. If more than two groups are given, the function will provide the JSD scores all all pairs of groups.

Usage

```
original_jsd(
  text,
  group_list = as.character(c()),
  word_list = as.character(c()),
  group = "group",
  word = "word",
  n = "n"
)
```

Arguments

text	Data frame containing data
group_list	Vector containing all groups to find pairwise JSD scores for

word_list	Vector containing all words that should be used to calculate JSD
group	Name of data frame column containing text group
word	Name of data frame column containing words
n	Name of data frame column containing word count in text group

Value

Data frame containing a column containing unique words and columns for JSD scores for each group pair

Examples

```
# Load example Hansard 1870 dataset
data(hansard_1870_example)
head(hansard_1870_example)

# Calculate original JSD for given words and groups
output = original_jsd(
  hansard_1870_example,
  group = "speaker",
  group_list = c("MR. GLADSTONE", "MR. DISRAELI"),
  word_list = c("trade", "press", "industry")
)
head(output)
```

---

remove_stop_words *Remove Stop Words*

Description

Remove stop words and fully numeric words from a column in a dataframe that contains words. It is assumed that the text has been tokenized (see dhmeasures::tokenize_counts) prior to using this function.

Usage

```
remove_stop_words(
  data,
  words = "word",
  stop_words = dhmeasures::stop_word,
  remove_numbers = TRUE
)
```

Arguments

data	Data frame containing your data
words	The name of the column where stop words should be searched for
stop_words	Vector of stop words. Uses dhmeasures::stop_word as the default
remove_numbers	Set to true (default) to remove all numeric values from the words column

Value

Data frame with all prior data, but without rows containing stop words

Examples

```
test = data.frame (
  myText = c(
    "Hello! This is the first sentence I am using to test this function!",
    "This is the second sentence!"
  ),
  myGroup = c(
    "group1",
    "group2"
  )
)

test2 = count_tokens(test, text = "myText", group = "myGroup")
test2

remove_stop_words(test2)
```

---

stop_word *Stop Words List*

Description

A list of common words (known as stop words) that can be used to remove insignificant words from your data.

Usage

```
stop_word
```

Format

A character vector with 478 values

Source

../data/stop_word.RData

References

Buongiorno, Steph (2021)

Examples

```
data(stop_word)
```

Index