Package ‘textstem’

April 9, 2018

Title Tools for Stemming and Lemmatizing Text

Version 0.1.4

Maintainer Tyler Rinker <tyler.rinker@gmail.com>

Description Tools that stem and lemmatize text. Stemming is a process that removes endings such as affixes. Lemmatization is the process of grouping inflected forms together as a single base form.

Depends R (>= 3.3.0), koRpus.lang.en

Imports dplyr, hunspell, koRpus, lexicon (>= 0.4.1), quanteda (>= 0.99.12), SnowballC, stats, stringi, textclean, textshape, utils

Suggests testthat

Date 2018-04-09

License GPL-2

LazyData TRUE

RoxygenNote 6.0.1

URL http://github.com/trinker/textstem

BugReports http://github.com/trinker/textstem/issues

NeedsCompilation no

Author Tyler Rinker [aut, cre]

Repository CRAN

Date/Publication 2018-04-09 15:03:04 UTC

R topics documented:

`lemmatize_strings` *Lemmatize a Vector of Strings*

Description

Lemmatize a vector of strings.

Usage

```
lemmatize_strings(x, dictionary = lexicon::hash_lemmas, ...)
```

Arguments

`x`	A vector of strings.
`dictionary`	A dictionary of base terms and lemmas to use for replacement. The first column should be the full word form in lower case while the second column is the corresponding replacement lemma. The default makes the dictionary from the text using `make_lemma_dictionary`. For larger texts a dictionary may take some time to compute. It may be more useful to generate the dictionary prior to running the function and explicitly pass the dictionary in.
`...`	Other arguments passed to `split_token`.

Value

Returns a vector of lemmatized strings.

Note

The lemmatizer splits the string apart into tokens for speed optimization. After the lemmatizing occurs the strings are pasted back together. The strings are not guaranteed to retain exact spacing of the original.

See Also

`lemmatize_words`

Examples

```
x <- c(
    'the dirtier dog has eaten the pies',
    'that shameful pooch is tricky and sneaky',
    "He opened and then reopened the food bag",
    'There are skies of blue and red roses too!',
    NA,
    "The doggies, well they aren't joyfully running.",
    "The daddies are coming over...",
    "This is 34.546 above"
)

## Default lexicon::hash_lemmas dictionary
lemmatize_strings(x)

## Hunspell dictionary
lemma_dictionary <- make_lemma_dictionary(x, engine = 'hunspell')
lemmatize_strings(x, dictionary = lemma_dictionary)

## Bigger data set
library(dplyr)
presidential_debates_2012$dialogue %>%
    lemmatize_strings() %>%
    head()

## Not run:
## Treetagger dictionary
lemma_dictionary2 <- make_lemma_dictionary(x, engine = 'treetagger')
lemmatize_strings(x, lemma_dictionary2)

lemma_dictionary3 <- presidential_debates_2012$dialogue %>%
    make_lemma_dictionary(engine = 'treetagger')

presidential_debates_2012$dialogue %>%
     lemmatize_strings(lemma_dictionary3) %>%
     head()

## End(Not run)
```

lemmatize_words *Lemmatize a Vector of Words*

Description

Lemmatize a vector of words.

Usage

```
lemmatize_words(x, dictionary = lexicon::hash_lemmas, ...)
```

Arguments

x	A vector of words.
dictionary	A dictionary of base terms and lemmas to use for replacement. The first column should be the full word form in lower case while the second column is the corresponding replacement lemma. The default uses hash_lemmas. This may come from make_lemma_dictionary as well, giving a more targeted, smaller dictionary. make_lemma_dictionary has choices in engines to use for the lemmatization.
...	ignored.

Value

Returns a vector of lemmatized words.

See Also

lemmatize_strings

Examples

```
x <- c("the", NA, 'doggies', ',', 'well', 'they', "aren\'t", 'Joyfully', 'running', '.')
lemmatize_words(x)
```

make_lemma_dictionary *Generate a Lemma Dictionary*

Description

Given a set of text strings, the function generates a dictionary of lemmas corresponding to words that are not in base form.

Usage

```
make_lemma_dictionary(..., engine = "hunspell", path = NULL,
  lang = switch(engine, hunspell = {     "en_US" }, treetagger = {     "en" },
  lexicon = {     NULL }, stop("engine not found")))
```

Arguments

`engine`	One of: "hunspell", "treetragger" or "lexicon". The lexicon and hunspell choices use the **lexicon** and **hunspell** packages, which may be faster than TreeTagger, have the tooling available without installing external tools but are likely less accurate. TreeTagger is likely more accurate but requires installing the TreeTagger program (`http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger`.
`path`	Path to the TreeTagger program if `engine = "treetagger"`. If `NULL` `textstem` will attempt to locate the location of TreeTagger.
`lang`	A character string naming the language to be used in **koRpus** (treetagger) or **hunspell**. The default language is `'en'` for **koRpus** (treetagger) and `'en_US'` for **hunspell**. See `?koRpus::treetag` or `?hunspell::dictionary` for details. Note that for `koRpus::treetag` `lang` is passed to both `lang` and `prest` in the `TT.options` argument.
`...`	A vector of texts to generate lemmas for.

Value

Returns a two column `data.frame` with tokens and corresponding lemmas.

Examples

```
x <- c('the dirtier dog has eaten the pies',
    'that shameful pooch is tricky and sneaky',
    "He opened and then reopened the food bag",
    'There are skies of blue and red roses too!'
)
make_lemma_dictionary(x)
## Not run:
make_lemma_dictionary(x, engine = 'treetagger')

## End(Not run)
```

presidential_debates_2012 *2012 U.S. Presidential Debates*

Description

A dataset containing a cleaned version of all three presidential debates for the 2012 election.

Usage

```
data(presidential_debates_2012)
```

Format

A data frame with 2912 rows and 4 variables

Details

- person. The speaker
- tot. Turn of talk
- dialogue. The words spoken
- time. Variable indicating which of the three debates the dialogue is from

sam_i_am *Sam I Am Text*

Description

A dataset containing a character vector of the text from Seuss's 'Sam I Am'.

Usage

```
data(sam_i_am)
```

Format

A character vector with 169 elements

References

Seuss, Dr. (1960). Green Eggs and Ham.

stem_strings *Stem a Vector of Strings*

Description

Stem a vector of strings.

Usage

```
stem_strings(x, language = "porter", ...)
```

Arguments

`x`	A vector of strings.
`language`	The name of a recognized language (see `wordStem`).
`...`	Other arguments passed to `split_token`.

Value

Returns a vector of stemmed strings.

Note

The stemmer requires splitting the string apart into tokens. After the stemming occurs the strings are pasted back together. The strings are not guaranteed to retain exact spacing of the original.

See Also

`stem_words`

Examples

```
x <- c(
    'the dirtier dog has eaten the pies',
    'that shameful pooch is tricky and sneaky',
    "He opened and then reopened the food bag",
    'There are skies of blue and red roses too!',
    NA,
    "The doggies, well they aren't joyfully running.",
    "The daddies are coming over...",
    "This is 34.546 above"
)
stem_strings(x)
```

stem_words *Stem a Vector of Words*

Description

Stem a vector of words.

Usage

```
stem_words(x, language = "porter", ...)
```

Arguments

x	A vector of words.
language	The name of a recognized language (see wordStem).
...	ignored.

Value

Returns a vector of stemmed words.

See Also

stem_strings

Examples

```
x <- c("the", 'doggies', ',', 'well', 'they', "aren\'t", 'Joyfully', 'running', '.')
stem_words(x)
```

textstem *Tools for Stemming and Lemmatizing Text*

Description

Tools that stem and lemmatize text. Stemming is a process that removes endings such as suffixes. Lemmatization is the process of grouping inflected forms together as a single base form.

Index