Package ‘tardis’

November 18, 2022

Type Package
Title Text Analysis with Rules and Dictionaries for Inferring Sentiment
Version 0.1.4
Description Measure text's sentiment with dictionaries and simple rules covering negations and modifiers. User-supplied dictionaries are supported, including Unicode emojis and multi-word tokens, so this package can also be used to study constructs beyond sentiment.
License MIT + file LICENSE
Encoding UTF-8
LazyData true
RoxygenNote 7.2.1
Imports dplyr, magrittr, purrr, rlang, stringi, stringr, tidyr
Depends R (>= 2.10)
URL https://github.com/chris31415926535/tardis
BugReports https://github.com/chris31415926535/tardis/issues
Suggests covr, knitr, rmarkdown, testthat (>= 3.0.0)
Config/testthat/edition 3
LinkingTo cpp11
SystemRequirements C++11
VignetteBuilder knitr
NeedsCompilation yes
Author Christopher Belanger [aut, cre, cph]
  (<https://orcid.org/0000-0003-2070-5721>)
Maintainer Christopher Belanger <christopher.a.belanger@gmail.com>
Repository CRAN
Date/Publication 2022-11-18 20:10:02 UTC
**dict_modifiers**

**Description**

A `tbl_df` with two columns: `token` and `score`, identifying the tokens that increase or decrease other words’ sentiments, and the percentage by which they do so.

**Usage**

```r
dict_modifiers
```

**Format**

An object of class `spec_tbl_df` (inherits from `tbl_df`, `tbl`, `data.frame`) with 87 rows and 2 columns.

**Details**

Derived originally from the VADER dictionary, but modified.

**Source**

https://CRAN.R-project.org/package=vader

---

**dict_negations**

**Description**

A `tbl_df` with one column: `token`.

**Usage**

```r
dict_negations
```
dict_tardis_sentiment

Format
An object of class spec_tbl_df (inherits from tbl_df, tbl, data.frame) with 38 rows and 1 columns.

Details
Can include apostrophes or not, but they’re removed in processing so there’s no need to include both words with and without apostrophes.
Derived originally from the VADER dictionary, but modified.

Source
https://CRAN.R-project.org/package=vader

dict_tardis_sentiment  Sentiment dictionary for TARDIS package.

Description
Combines VADER and emoji dictionaries.

Usage
dict_tardis_sentiment

Format
An object of class spec_tbl_df (inherits from tbl_df, tbl, data.frame) with 7653 rows and 2 columns.

tardis  Text Analysis with Rules and Dictionaries for Inferring Sentiment (TARDIS)

Description
This function uses dictionaries (either the included defaults or user-supplied) custom dictionaries) and simple rules to measure the sentiment of supplied text. "Sentiment" means roughly the emotion expressed in the text, where emotions are collapsed into positive (e.g. happy) or negative (e.g. sad, angry).
Usage

tardis(
  input_text = c("I am happy.", "I am VERY happy!!", ":)", "Not sad.", "Bad.",
  "Not bad.", "A happy sentence! And a sad one. In the same text."),
  text_column = NA,
  dict_sentiments = NA,
  dict_modifiers = NA,
  dict_negations = NA,
  sigmoid_factor = 15,
  negation_factor = 0.75,
  allcaps_factor = 1.25,
  punctuation_factor = 1.15,
  use_punctuation = TRUE,
  summary_function = c("mean", "median", "max", "min", "sum"),
  simple_count = FALSE,
  verbose = FALSE
)

Arguments

input_text         Text to analyze, either a character vector or a data.frame with a column of text.
text_column        If using data.frame input, the name of the column of text to analyze.
dict_sentiments    Optional sentiment dictionary, defaults to internal tardis dictionary. A data.frame
                    with two columns: word and value.
dict_modifiers     Optional modifiers dictionary, or "none" to disable modifiers. Defaults to internal
                    tardis dictionary. A data.frame with two columns: word and value.
dict_negations     Optional negation dictionary, or "none" to disable negations. Defaults to internal
                    tardis dictionary. A data.frame with one column: word.
sigmoid_factor     Numeric, default 15. Factor for scaling sentence scores to -1/+1 using a sigmoid function. Set to NA to disable the sigmoid function and just return sums of scores, adjusted by any applicable negators, modifiers, or punctuation/caps effects.
negation_factor    Numeric, default 0.75. Multiplier for damping effects of sentiment-bearing terms after negations. Stacks multiplicatively. Should probably be less than 1.
allcaps_factor     Numeric, default 1.25. Multiplier for scaling effects of sentiment-bearing terms in ALL CAPS. Should probably be more than 1, to increase effects.
punctuation_factor Numeric, default 1.15. Multiplier for scaling effects of punctuation. A single question mark has no effect, but one or more exclamation marks does, and question marks have effects in the presence of exclamation marks, up to three punctuation marks total.
use_punctuation    Boolean, default TRUE. Should we consider sentence-level punctuation?
summary_function
For multi-sentence texts, how should we summarise sentence scores into a text score? Default "mean", also accepts "median", "max", "min", and "sum".

simple_count
Boolean, default FALSE. Convenience parameter that overrides many other parameters to enable simple counts of dictionary words: no modifiers, negations, capitalization, or punctuation effects are considered and no sigmoid function is applied.

verbose
For debugging–should it print lots of messages to the console?

Details
Roughly, each word’s sentiment is a property of its dictionary-given sentiment, whether it’s written in all-caps or not, and the three preceding words. A preceding negation (e.g. "not") will reverse and reduce the sentiment–turning a positive into a slightly less extreme negative, or vice-versa–and a preceding modifier can either increase/decrease the sentiment (e.g. "very" will increase it, "somewhat" will decrease it).

Sentences are scored based on their words and the presence of exclamation or question marks.

If a supplied text string has more than one sentence, this function will also return the mean, standard deviation, and range of sentiments expressed in its sentences. The rationale is that it doesn’t make sense to apply sentence-level analysis to paragraphs, especially for online communications where people can use quick swings in sentiment to express irony.

Input can be supplied in a data.frame or character vector.

Value
A tbl_df with one row for each input text and three new columns: sentiment_mean: the average sentiment for each sentence in each text. sentiment_sd: the standard deviation of sentence sentiments for each text. sentiment_range: the range of sentence sentiments for each text.

Usage
tardis_multidict(input_text, text_column = NA, dictionaries, ...)

Description
This convenience function takes a text and a set of dictionaries, and calls tardis::tardis() once for each dictionary. Other parameters are also passed along to tardis().
Arguments

- **input_text**: A text to be analyzed, either a tbl_df or a character vector.
- **text_column**: If tbl_df input, a character with the name of the input column containing the text to be analyzed.
- **dictionaries**: A single tbl_df with columns `dictionary`, `token`, and (optionally, for weighted dictionaries) `score`.
- **...**: Other parameters passed on to `tardis::tardis()`.

Details

Dictionaries must be in a single tbl_df with at least two columns: `token`, containing the tokens belonging to each dictionary; and `dictionary`, which contains a unique identifier mapping each token to a dictionary. Weights, if present, must be in a column named `score`.

Tokens can be mapped to multiple dictionaries, but each row maps one token to one dictionary.

Value

A tbl_df with new columns for each dictionary.

Examples

```r
## Not run:
library(magrittr)
# Get NRC emotions dataset from textdata package
nrc_emotion <- textdata::lexicon_nrc() %>%
  dplyr::rename(token = word, dictionary = sentiment) %>%
  dplyr::mutate(score = 1)

# set up some input text
text <- dplyr::tibble(body = c("I am so angry!", "I am angry.",
  "I'm not angry.", "Your mother and I aren't angry, we're just disappointed."))

emotions <- tardis_multidict(input_text = text, text_column = "body",
  dictionaries = nrc_emotion) %>%
  dplyr::select(body, score_anger, score_sadness)

emotions

## End(Not run)
```
Index

* datasets
  dict_modifiers, 2
  dict_negations, 2
  dict_tardis_sentiment, 3

dict_modifiers, 2
dict_negations, 2
dict_tardis_sentiment, 3

tardis, 3
tardis_multidict, 5