Package ‘vader’

June 9, 2020

Title Valence Aware Dictionary and sEntiment Reasoner (VADER)

Version 0.1.1

Description A lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media, and works well on texts from other domains. Hutto & Gilbert (2014) <https://www.aaai.org/ocs/index.php/ICWSM/ICWSM14/paper/view/8109/8122>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.0.2

Imports tm

Depends R (>= 2.10)

Suggests spelling

Language en-US

NeedsCompilation no

Author Katherine Roehrick [aut, cre]

Maintainer Katherine Roehrick <`kr.gitcode@gmail.com`>

Repository CRAN

Date/Publication 2020-06-09 05:30:02 UTC

R topics documented:

get_vader *Get a named vector of vader results for a single text document*

Description

Use get_vader() to calculate the valence of a single text document.

Usage

```
get_vader(text, incl_nt = T, neu_set = T)
```

Arguments

`text`	to be analyzed; for get_vader(), the text should be a character string
`incl_nt`	defaults to T, indicates whether you wish to incl n't contractions (e.g., can't) in negation analysis
`neu_set`	defaults to T, indicates whether you wish to count neutral words in calculations

Value

A named vector containing the valence score for each word; an overall, compound valence score for the text; the weighted percentage of positive, negative, and neutral words in the text; and the frequency of the word "but".

References

For the original Python Code, please see:

- https://github.com/cjhutto/vaderSentiment
- https://github.com/cjhutto/vaderSentiment/blob/master/vaderSentiment/vaderSentiment.py

For the original R Code, please see:

- https://github.com/nrguimaraes/sentimentSetsR/blob/master/R/ruleBasedSentimentFunctions.R

Modifications to the above scripts include, but are not limited to:

- ALL CAPS fx: updated to account for non-alpha words; i.e. "I'M 100 PERCENT SURE" would previously have been counted as mixed case due to the use of numbers
- IDIOMS fx: added capacity to check for idioms that do not contain any words found in the Vader Lexicon
- WORDS+EMOT: strip punctuation while preserving ALL emoticons found in dictionary
- Option to turn on/off neutral count

N.B.

In the examples below, "yesn't" is an internet neologism meaning "no", "maybe yes, maybe no", "didn't", etc.

See Also

`vader_df` to get vader results for multiple text documents

Examples

```
get_vader("I yesn't like it")
get_vader("I yesn't like it", incl_nt = FALSE)
get_vader("I yesn't like it", neu_set = FALSE)
```

vader_df *Get a dataframe of vader results for multiple text documents*

Description

Use vader_df() to calculate the valence of multiple texts contained within a vector or column in a dataframe.

Usage

```
vader_df(text, incl_nt = T, neu_set = T)
```

Arguments

`text`	to be analyzed; for vader_df(), the text should be a single vector (e.g. 1 column)
`incl_nt`	defaults to T, indicates whether you wish to incl n't contractions (e.g., can't) in negation analysis
`neu_set`	defaults to T, indicates whether you wish to count neutral words in calculations

Value

A dataframe containing the valence score for each word; an overall, compound valence score for the text; the weighted percentage of positive, negative, and neutral words in the text; and the frequency of the word "but".

N.B.

In the examples below, "yesn't" is an internet neologism meaning "no", "maybe yes, maybe no", "didn't", etc.

See Also

`get_vader` to get vader results for a single text document

Examples

```
vader_df(c("I'm happy", "I'm yesn't happy"))
vader_df(c("I'm happy", "I'm yesn't happy"), incl_nt = FALSE)
vader_df(c("I'm happy", "I'm yesn't happy"), neu_set = FALSE)
```

Index