Package 'syuzhet'

August 29, 2016

Type Package

Title Extracts Sentiment and Sentiment-Derived Plot Arcs from Text

Version 1.0.0

Date 2016-04-22

Maintainer Matthew Jockers <mjockers@gmail.com>

Description Extracts sentiment and sentiment-derived plot arcs
from text using three sentiment dictionaries conveniently
packaged for consumption by R users. Implemented dictionaries include
``syuzhet'' (default) developed in the Nebraska Literary Lab
``afinn'' developed by Finn {\{}AA}rup Nielsen, ``bing'' developed by Minqing Hu
and Bing Liu, and ``nrc'' developed by Mohammad, Saif M. and Turney, Peter D.
Applicable references are available in README.md and in the documentation
for the ``get_sentiment'' function. The package also provides a hack for
implementing Stanford's coreNLP sentiment parser. The package provides
several methods for plot arc normalization.

URL https://github.com/mjockers/syuzhet

License GPL-3

Imports openNLP, NLP, zoo, dtt, stats, graphics

LazyData true

Suggests knitr, pander, testthat (>= 0.9.1)

NeedsCompilation no

VignetteBuilder knitr

RoxygenNote 5.0.1

Author Matthew Jockers [aut, cre]

Repository CRAN

Date/Publication 2016-04-28 00:17:58

R topics documented:

`get_dct_transform` *Discrete Cosine Transformation with Reverse Transform.*

Description

Converts input values into a standardized set of filtered and reverse transformed values for easy plotting and/or comparison.

Usage

```
get_dct_transform(raw_values, low_pass_size = 5, x_reverse_len = 100,
  scale_vals = FALSE, scale_range = FALSE)
```

Arguments

`raw_values`	the raw sentiment values calculated for each sentence
`low_pass_size`	The number of components to retain in the low pass filtering. Default = 5
`x_reverse_len`	the number of values to return via decimation. Default = 100
`scale_vals`	Logical determines whether or not to normalize the values using the scale function Default = FALSE. If TRUE, values will be scaled by subtracting the means and scaled by dividing by their standard deviations. See ?scale
`scale_range`	Logical determines whether or not to scale the values from -1 to +1. Default = FALSE. If set to TRUE, the lowest value in the vector will be set to -1 and the highest values set to +1 and all the values scaled accordingly in between.

Value

The transformed values

Examples

```
s_v <- get_sentences("I begin this story with a neutral statement.
Now I add a statement about how much I despise cats.
I am allergic to them. I hate them. Basically this is a very silly test. But I do love dogs!")
raw_values <- get_sentiment(s_v, method = "syuzhet")
dct_vals <- get_dct_transform(raw_values)
plot(dct_vals, type="l", ylim=c(-0.1,.1))
```

---

get_nrc_sentiment *Get Emotions and Valence from NRC Dictionary*

Description

Calls the NRC sentiment dictionary to calculate the presence of eight different emotions and their corresponding valence in a text file.

Usage

```
get_nrc_sentiment(char_v)
```

Arguments

char_v	A character vector

Value

A data frame where each row represents a sentence from the original file. The columns include one for each emotion type as well as a positive or negative valence. The ten columns are as follows: "anger", "anticipation", "disgust", "fear", "joy", "sadness", "surprise", "trust", "negative", "positive."

References

Saif Mohammad and Peter Turney. "Emotions Evoked by Common Words and Phrases: Using Mechanical Turk to Create an Emotion Lexicon." In Proceedings of the NAACL-HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text, June 2010, LA, California. See: http://saifmohammad.com/WebPages/lexicons.html

get_nrc_values *Summarize NRC Values*

Description

Access the NRC dictionary to compute emotion types and valence for a set of words in the input vector.

Usage

```
get_nrc_values(word_vector)
```

Arguments

word_vector	A character vector.

Value

A vector of values for the emotions and valence detected in the input vector.

get_percentage_values *Chunk a Text and Get Means*

Description

Chunks text into 100 Percentage based segments and calculates means.

Usage

```
get_percentage_values(raw_values, bins = 100)
```

Arguments

raw_values	Raw sentiment values
bins	The number of bins to split the input vector. Default is 100 bins.

Value

A vector of mean values from each chunk

get_sentences *Sentence Tokenization*

Description

Parses a string into a vector of sentences.

Usage

```
get_sentences(text_of_file, strip_quotes = TRUE)
```

Arguments

text_of_file	A Text String
strip_quotes	Logical. Default of TRUE results in removal of quote marks from text input prior to sentence parsing.

Value

A Character Vector of Sentences

get_sentiment *Get Sentiment Values for a String*

Description

Iterates over a vector of strings and returns sentiment values based on user supplied method. The default method, "syuzhet" is a custom sentiment dictionary developed in the Nebraska Literary Lab. The default dictionary should be better tuned to fiction as the terms were extracted from a collection of 165,000 human coded sentences taken from a small corpus of contemporary novels.

Usage

```
get_sentiment(char_v, method = "syuzhet", path_to_tagger = NULL)
```

Arguments

char_v	A vector of strings for evaluation.
method	A string indicating which sentiment method to use. Options include "syuzhet", "bing", "afinn", "nrc" and "stanford." See references for more detail on methods.
path_to_tagger	local path to location of Stanford CoreNLP package

Value

Return value is a numeric vector of sentiment values, one value for each input sentence.

References

Bing Liu, Minqing Hu and Junsheng Cheng. "Opinion Observer: Analyzing and Comparing Opinions on the Web." Proceedings of the 14th International World Wide Web conference (WWW-2005), May 10-14, 2005, Chiba, Japan.

Minqing Hu and Bing Liu. "Mining and Summarizing Customer Reviews." Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2004), Aug 22-25, 2004, Seattle, Washington, USA. See: http://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html#lexicon

Saif Mohammad and Peter Turney. "Emotions Evoked by Common Words and Phrases: Using Mechanical Turk to Create an Emotion Lexicon." In Proceedings of the NAACL-HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text, June 2010, LA, California. See: http://saifmohammad.com/WebPages/lexicons.html

Finn Arup Nielsen. "A new ANEW: Evaluation of a word list for sentiment analysis in microblogs", Proceedings of the ESWC2011 Workshop on 'Making Sense of Microposts':Big things come in small packages 718 in CEUR Workshop Proceedings : 93-98. 2011 May. http://arxiv.org/abs/1103.2903. See: http://www2.imm.dtu.dk/pubdb/views/publication_details.php?id=6010

Manning, Christopher D., Surdeanu, Mihai, Bauer, John, Finkel, Jenny, Bethard, Steven J., and McClosky, David. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pp. 55-60. See: http://nlp.stanford.edu/software/corenlp.shtml

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher Manning, Andrew Ng and Christopher Potts. "Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank Conference on Empirical Methods in Natural Language Processing" (EMNLP 2013). See: http://nlp.stanford.edu/sentiment/

`get_sent_values` *Assigns Sentiment Values*

Description

Assigns sentiment values to words based on preloaded dictionary. The default is the syuzhet dictionary.

Usage

```
get_sent_values(char_v, method = "syuzhet")
```

Arguments

`char_v`	A string
`method`	A string indicating which sentiment dictionary to use

Value

A single numerical value (positive or negative) based on the assessed sentiment in the string

get_stanford_sentiment *Get Sentiment from the Stanford Tagger*

Description

Call the Stanford Sentiment tagger with a vector of strings. The Stanford tagger automatically detects sentence boundaries and treats each sentence as a distinct instance to measure. As a result, the vector that gets returned will not be the same length as the input vector.

Usage

```
get_stanford_sentiment(text_vector, path_to_stanford_tagger)
```

Arguments

text_vector	A vector of strings
path_to_stanford_tagger	a local file path indicating where the coreNLP package is installed.

get_text_as_string *Load Text from a File*

Description

Loads a file as a single text sting.

Usage

```
get_text_as_string(path_to_file)
```

Arguments

path_to_file	file path

Value

A character vector of length 1 containing the text of the file in the path_to_file argument.

get_tokens *Word Tokenization*

Description

Parses a string into a vector of word tokens.

Usage

```
get_tokens(text_of_file, pattern = "\\W")
```

Arguments

text_of_file	A Text String
pattern	A regular expression for token breaking

Value

A Character Vector of Words

get_transformed_values *Fourier Transform and Reverse Transform Values*

Description

Converts input values into a standardized set of filtered and reverse transformed values for easy plotting and/or comparison.

Usage

```
get_transformed_values(raw_values, low_pass_size = 2, x_reverse_len = 100,
  padding_factor = 2, scale_vals = FALSE, scale_range = FALSE)
```

Arguments

raw_values	the raw sentiment values calculated for each sentence
low_pass_size	The number of components to retain in the low pass filtering. Default = 3
x_reverse_len	the number of values to return. Default = 100
padding_factor	the amount of zero values to pad raw_values with, as a factor of the size of raw_values. Default = 2.
scale_vals	Logical determines whether or not to normalize the values using the scale function Default = FALSE. If TRUE, values will be scaled by subtracting the means and scaled by dividing by their standard deviations. See ?scale
scale_range	Logical determines whether or not to scale the values from -1 to +1. Default = FALSE. If set to TRUE, the lowest value in the vector will be set to -1 and the highest values set to +1 and all the values scaled accordingly in between.

Value

The transformed values

Examples

```
s_v <- get_sentences("I begin this story with a neutral statement.
Now I add a statement about how much I despise cats.
I am allergic to them.
Basically this is a very silly test.")
raw_values <- get_sentiment(s_v, method = "bing")
get_transformed_values(raw_values)
```

---

rescale *Vector Value Rescaling*

Description

Rescale Transformed values from -1 to 1

Usage

```
rescale(x)
```

Arguments

x	A vector of values

---

rescale_x_2 *Bi-Directional x and y axis Rescaling*

Description

Rescales input values to two scales (0 to 1 and -1 to 1) on the y-axis and also creates a scaled vector of x axis values from 0 to 1. This function is useful for plotting and plot comparison.

Usage

```
rescale_x_2(v)
```

Arguments

v	A vector of values

Value

A list of three vectors (x, y, z). x is a vector of values from 0 to 1 equal in length to the input vector v. y is a scaled (from 0 to 1) vector of the input values equal in lenght to the input vector v. z is a scaled (from -1 to +1) vector of the input values equal in length to the input vector v.

`simple_plot` *Plots simple and rolling shapes overlayed*

Description

A Simple function for comparing three smoothers

Usage

```
simple_plot(raw_values, title = "Syuzhet Plot", legend_pos = "top")
```

Arguments

`raw_values`	the raw sentiment values calculated for each sentence
`title`	for image
`legend_pos`	positon for legend

Index