Package ‘stylest’

October 14, 2022

Version 0.2.0
Title Estimating Speaker Style Distinctiveness
Depends R (>= 2.10)
Imports corpus, Matrix, stats
Suggests knitr, rmarkdown, testthat, kableExtra
Description
Estimates distinctiveness in speakers’ (authors’) style. Fits models that can be used for predicting speakers of new texts. Methods developed in Huang et al (2020) <doi:10.1017/pan.2019.49>.
License GPL-3
URL https://github.com/leslie-huang/stylest
BugReports https://github.com/leslie-huang/stylest/issues
LazyData true
Encoding UTF-8
VignetteBuilder knitr, rmarkdown
RoxygenNote 7.0.2
NeedsCompilation no
Author Leslie Huang [aut, cph, cre],
Patrick O. Perry [aut, cph],
Arthur Spirling [aut, cph]
Maintainer Leslie Huang <lesliehuang@nyu.edu>
Repository CRAN
Date/Publication 2021-03-04 17:10:02 UTC

R topics documented:

fit_term_usage .............................................. 2
novels_excerpts ........................................... 3
print.stylest_model ..................................... 3
stylest .................................................. 4
fit_term_usage

Computes speakers' term usage rates

Description
Computes speakers' term usage rates

Usage

fit_term_usage(
  x, speaker, terms, smooth, term_weights, fill_method, fill_weight, weight_varname
)

Arguments

x Text vector. May be a corpus_frame object
speaker Vector of speaker labels. Should be the same length as x
terms Vocabulary for document term matrix
smooth Numeric value used smooth term frequencies
term_weights Dataframe of distances (or any weights) per word in the vocab. This dataframe should have one column $word and a second column $weight_var containing the weight for the word
fill_method if "value" (default), fill_weight is used to fill any terms with NA weight. If "mean", the mean term_weight should be used as the fill value
fill_weight numeric value to fill in as weight for any term which does not have a weight specified in term_weights
weight_varname Name of the column in term_weights containing the weights
Value

named list of: terms, vector of num tokens uttered by each speaker, smoothing value, term weights (NULL if no weights), terms whose weights were imputed (NULL if no term_weights=NULL), fill_weight used to fill missing weights (NULL if no term_weights=NULL), and (smoothed) term usage rate matrix

Description

A dataset of text from English novels by Jane Austen, George Eliot, and Elizabeth Gaskell.

Usage

novels_excerpts

Format

A dataframe with 21 rows and 3 variables:

title Title
author Author
text Excerpt of text in complete sentences from the first 1,000 chars of the novel.

Source

Novel excerpts obtained from Project Gutenberg full texts in the public domain in the USA. http://gutenberg.org

print.stylest_model

Custom print method for stylest_model

Description

Custom print method for stylest_model

Usage

## S3 method for class 'stylest_model'
print(x, ...)

Arguments

x 'stylest_model' object
... Additional arguments
Value

Prints summary information about the ‘stylest_model’ object

Examples

```r
data(novels_excerpts)
speaker_mod <- stylest_fit(novels_excerpts$text, novels_excerpts$author)
print(speaker_mod)
```

---

**stylest**

*stylest: A package for estimating textual distinctiveness*

---

**stylest_fit**

*Fit speaker_model to a corpus*

---

**Description**

stylest provides a set of functions for fitting a model of speaker distinctiveness, including tools for selecting the optimal vocabulary for the model and predicting the most likely speaker (author) of a new text.

**Usage**

```r
stylest_fit(
  x,
  speaker,
  terms = NULL,
  filter = NULL,
  smooth = 0.5,
  term_weights = NULL,
  fill_method = "value",
  fill_weight = 0,
  weight_varname = "mean_distance"
)
```
**Arguments**

- **x**  
  Text vector. May be a corpus_frame object
- **speaker**  
  Vector of speaker labels. Should be the same length as x
- **terms**  
  If not NULL, terms to be used in the model. If NULL, use all terms
- **filter**  
  If not NULL, a text filter to specify the tokenization. See corpus for more information about specifying filter
- **smooth**  
  Numeric value used smooth term frequencies instead of the default of 0.5
- **term_weights**  
  Dataframe of distances (or any weights) per word in the vocab. This dataframe should have one column $word and a second column $weight_var containing the weight for the word. See the vignette for details.
- **fill_method**  
  if "value" (default), fill_weight is used to fill any terms with NA weight. If "mean", the mean term_weight should be used as the fill value
- **fill_weight**  
  numeric value to fill in as weight for any term which does not have a weight specified in term_weights, default=0.0 (drops any words without weights)
- **weight_varname**  
  Name of the column in term_weights containing the weights, default="mean_distance"

**Details**

The user may specify only one of terms or cutoff. If neither is specified, all terms will be used.

**Value**

A S3 stylest_model object containing:
- **speakers** Vector of unique speakers,
- **filter** text_filter used,
- **terms** terms used in fitting the model,
- **ntoken** Vector of number of tokens per speaker,
- **smooth** Smoothing value,
- **weights** If not NULL, a named matrix of weights for each term in the vocab,
- **rate** Matrix of speaker rates for each term in vocabulary

**Examples**

```r
data(novels_excerpts)
speaker_mod <- stylest_fit(novels_excerpts$text, novels_excerpts$author)
```

**Description**

Computes the mean log odds of the most likely speaker of each text over pairs of the speaker of a text and every other speaker in the stylest_model.

**Usage**

```r
stylest_odds(model, text, speaker, prior = NULL)
```
**Arguments**

- `model` : stylest_model object
- `text` : Text vector. May be a corpus_frame object
- `speaker` : Vector of speaker labels. Should be the same length as `x`
- `prior` : Prior probability of speakers. Uses equal prior if `NULL`

**Value**

A S3 stylest_odds object containing: a stylest_model object; vector of mean log odds that each actual speaker (compared with other speakers in the corpus) spoke their corresponding texts in the corpus; vector of SEs of the log odds

**Examples**

```r
data(novels_excerpts)
speaker_mod <- stylest_fit(novels_excerpts$text, novels_excerpts$author)
stylest_odds(speaker_mod, novels_excerpts$text, novels_excerpts$author)
```

---

**Description**

Use a fitted stylest_model to predict the most likely speaker of a text. This function may be used on in-sample or out-of-sample texts.

**Usage**

```r
stylest_predict(model, text, prior = NULL)
```

**Arguments**

- `model` : stylest_model object
- `text` : Text vector. May be a corpus_frame object
- `prior` : Prior probability, defaults to `NULL`

**Value**

`stylest_predict` object containing: `model` the fitted stylest_model object used in prediction, `predicted` the predicted speaker, `log_probs` matrix of log probabilities, `log_prior` matrix of log prior probabilities

**Examples**

```r
data(novels_excerpts)
speaker_mod <- stylest_fit(novels_excerpts$text, novels_excerpts$author)
stylest_predict(speaker_mod, "This is an example text, who wrote it?")
```
stylest_select_vocab  
Select vocabulary using cross-validated out-of-sample prediction

Description

Selects optimal vocabulary quantile(s) for model fitting using performance on predicting out-of-sample texts.

Usage

```r
stylest_select_vocab(
  x,  
speaker,  
filter = NULL,  
smooth = 0.5,  
nfold = 5,  
cutoff_pcts = c(50, 60, 70, 80, 90, 99),  
cutoffs_term_weights = NULL,  
fill_method = "value",  
fill_weight = 1,  
weight_varname = "mean_distance"
)
```

Arguments

- **x**: Corpus as text vector. May be a `corpus_frame` object
- **speaker**: Vector of speaker labels. Should be the same length as `x`
- **filter**: if not `NULL`, a `corpus text_filter`
- **smooth**: value for smoothing. Defaults to 0.5
- **nfold**: Number of folds for cross-validation. Defaults to 5
- **cutoff_pcts**: Vector of cutoff percentages to test. Defaults to `c(50, 60, 70, 80, 90, 99)`
- **cutoffs_term_weights**: Named list of dataframes of term weights, where the names correspond to the `cutoff_pcts`. Each dataframe should have one column `word` and a second column `weight_varname` containing the weight for the word. See the vignette for details.
- **fill_method**: if "value" (default), `fill_weight` is used to fill any terms with NA weight. If "mean", the mean term_weight should be used as the fill value
- **fill_weight**: numeric value to fill in as weight for any term which does not have a weight specified in `term_weights`. default=1.0
- **weight_varname**: Name of the column in each term_weights dataframe containing the weights, default="mean_distance"
Value

List of: best cutoff percent with the best speaker classification rate; cutoff percentages that were tested; matrix of the mean percentage of incorrectly identified speakers for each cutoff percent and fold; and the number of folds for cross-validation.

Examples

```r
## Not run:
data(novels_excerpts)
stylest_select_vocab(novels_excerpts$text, novels_excerpts$author, cutoff_pcts = c(50, 90))
## End(Not run)
```

stylest_terms

Use vocab cutoff to select terms for fitting the model

Description

The same text, speaker, and filter should be used in this model as in `fit_speaker` to select the terms for the latter function.

Usage

```r
stylest_terms(x, speaker, vocab_cutoff, filter = NULL)
```

Arguments

- `x`: Corpus as text vector. May be a `corpus_frame` object.
- `speaker`: Vector of speaker labels. Should be the same length as `x`.
- `vocab_cutoff`: Quantile cutoff for the vocabulary in (0, 100]
- `filter`: If not `NULL`, a corpus filter.

Value

List of terms

Examples

```r
data(novels_excerpts)
stylest_terms(novels_excerpts$text, novels_excerpts$author, vocab_cutoff = 50)
```
**stylest_term_influence**

*Compute the influence of terms*

**Description**

Compute the influence of terms

**Usage**

```r
stylest_term_influence(model, text, speaker)
```

**Arguments**

- `model`: `stylest_model` object
- `text`: Text vector. May be a `corpus_frame` object
- `speaker`: Vector of speaker labels. Should be the same length as `x`

**Value**

`data.frame` with columns representing terms, their mean influence, and their maximum influence

**Examples**

```r
data(novels_excerpts)
speaker_mod <- stylest_fit(novels_excerpts$text, novels_excerpts$author)
stylest_term_influence(speaker_mod, novels_excerpts$text, novels_excerpts$author)
```
Index

* datasets
  novels_excerpts, 3

fit_term_usage, 2

novels_excerpts, 3

print.stylest_model, 3

stylest, 4
stylest_fit, 4
stylest_odds, 5
stylest_predict, 6
stylest_select_vocab, 7
stylest_term_influence, 9
stylest_terms, 8