Package ‘topicdoc’

October 18, 2019

Type Package
Title Topic-Specific Diagnostics for LDA and CTM Topic Models
Version 0.1.0
License MIT + file LICENSE
URL https://github.com/doug-friedman/topicdoc
BugReports https://github.com/doug-friedman/topicdoc/issues
Depends R (>= 3.5.0)
Imports slam, topicmodels
Suggests knitr, rmarkdown, testthat (>= 2.1.0)
VignetteBuilder knitr
Encoding UTF-8
LazyData true
RoxygenNote 6.1.1
NeedsCompilation no
Author Doug Friedman [aut, cre]
Maintainer Doug Friedman <doug.nhp@gmail.com>
Repository CRAN
Date/Publication 2019-10-18 12:40:02 UTC

R topics documented:

  coherence ..................................................  2
  contain_equal_docs .......................................  2
  dist_from_corpus ........................................  3
contain_equal_docs

Description
Helper function to check that a topic model and a dtm contain the same number of documents

Usage
contain_equal_docs(topic_model, dtm_data)

doc_prominence ......................................................... 4
mean_token_length ..................................................... 5
n_topics ................................................................. 5
tf_df_dist ................................................................. 6
tf_df_dist_diff ......................................................... 7
topic_coherence ......................................................... 7
topic_diagnostics ....................................................... 8
topic_exclusivity ....................................................... 9
topic_size ............................................................... 10

Index

coherece  Helper function for calculating coherence for a single topic’s worth of terms

Description
Helper function for calculating coherence for a single topic’s worth of terms

Usage
coherence(dtm_data, top_terms, smoothing_beta)

Arguments

dtm_data a document-term matrix of token counts coercible to simple_triplet_matrix
top_terms a character vector of the top terms for a given topic
smoothing_beta a numeric indicating the value to use to smooth the document frequencies in order avoid log zero issues, the default is 1

Value
a numeric indicating coherence for the topic
dist_from_corpus

Arguments

- `topic_model` a fitted topic model object from one of the following: `tm-class`
- `dtm_data` a document-term matrix of token counts coercible to `simple_triplet_matrix`

Value

a logical indicating whether or not the two object contain the same number of documents

Description

The Hellinger distance between the token probabilities or betas for each topic and the overall probability for the word in the corpus is calculated.

Usage

```r
dist_from_corpus(topic_model, dtm_data)
```

Arguments

- `topic_model` a fitted topic model object from one of the following: `tm-class`
- `dtm_data` a document-term matrix of token counts coercible to `simple_triplet_matrix`

Value

A vector of distances with length equal to the number of topics in the fitted model

References


Examples

```r
# Using the example from the LDA function
library(topicmodels)
data("AssociatedPress", package = "topicmodels")
lda <- LDA(AssociatedPress[1:20,], control = list(alpha = 0.1), k = 2)
dist_from_corpus(lda, AssociatedPress[1:20,])
```
**Calculate the document prominence of each topic in a topic model**

**Description**

Calculate the document prominence of each topic in a topic model based on either the number of documents with an estimated gamma probability above a threshold or the number of documents where a topic has the highest estimated gamma probability.

**Usage**

```r
doc_prominence(topic_model, method = c("gamma_threshold", "largest_gamma"), gamma_threshold = 0.2)
```

**Arguments**

- `topic_model`: a fitted topic model object from one of the following: `tm-class`
- `method`: a string indicating which method to use - "gamma_threshold" or "largest_gamma", the default is "gamma_threshold"
- `gamma_threshold`: a number between 0 and 1 indicating the gamma threshold to be used when using the gamma threshold method, the default is 0.2

**Value**

A vector of document prominences with length equal to the number of topics in the fitted model.

**References**


**Examples**

```r
# Using the example from the LDA function
library(topicmodels)
data("AssociatedPress", package = "topicmodels")
lda <- LDA(AssociatedPress[1:20,], control = list(alpha = 0.1), k = 2)
doc_prominence(lda)
```
**mean_token_length**

*Calculate the average token length for each topic in a topic model*

**Description**

Using the the N highest probability tokens for each topic, calculate the average token length for each topic.

**Usage**

```r
mean_token_length(topic_model, top_n_tokens = 10)
```

**Arguments**

- `topic_model`: a fitted topic model object from one of the following: `tm-class`
- `top_n_tokens`: an integer indicating the number of top words to consider, the default is 10

**Value**

A vector of average token lengths with length equal to the number of topics in the fitted model.

**References**


**Examples**

```r
# Using the example from the LDA function
library(topicmodels)
data("AssociatedPress", package = "topicmodels")
lda <- LDA(AssociatedPress[1:20,], control = list(alpha = 0.1), k = 2)
mean_token_length(lda)
```

---

**n_topics**

*Helper function to determine the number of topics in a topic model*

**Description**

Helper function to determine the number of topics in a topic model.

**Usage**

```r
n_topics(topic_model)
```
tf_df_dist

Arguments

- **topic_model**: a fitted topic model object from one of the following: `tm-class`

Value

- an integer indicating the number of topics in the topic model

Description

Using the N highest probability tokens for each topic, calculate the Hellinger distance between the token frequencies and the document frequencies

Usage

```
tf_df_dist(topic_model, dtm_data, top_n_tokens = 10)
```

Arguments

- **topic_model**: a fitted topic model object from one of the following: `tm-class`
- **dtm_data**: a document-term matrix of token counts coercible to `simple_triplet_matrix`
- **top_n_tokens**: an integer indicating the number of top words to consider, the default is 10

Value

- A vector of distances with length equal to the number of topics in the fitted model

References


Examples

```
# Using the example from the LDA function
library(topicmodels)
data("AssociatedPress", package = "topicmodels")
lda <- LDA(AssociatedPress[1:20,], control = list(alpha = 0.1), k = 2)
tf_df_dist(lda, AssociatedPress[1:20,])
```
tf_df_dist_diff

Helper function to calculate the Hellinger distance between the token frequencies and document frequencies for a specific topic’s top N tokens

Description
Helper function to calculate the Hellinger distance between the token frequencies and document frequencies for a specific topic’s top N tokens

Usage
tf_df_dist_diff(dtm_data, top_terms)

Arguments
dtm_data a document-term matrix of token counts coercible to simple_triplet_matrix
top_terms - a character vector of the top N tokens

Value
a single value representing the Hellinger distance

topic_coherence
Calculate the topic coherence for each topic in a topic model

Description
Using the the N highest probability tokens for each topic, calculate the topic coherence for each topic

Usage
topic_coherence(topic_model, dtm_data, top_n_tokens = 10,
smoothing_beta = 1)

Arguments
topic_model a fitted topic model object from one of the following: tm-class
dtm_data a document-term matrix of token counts coercible to simple_triplet_matrix
top_n_tokens an integer indicating the number of top words to consider, the default is 10
smoothing_beta a numeric indicating the value to use to smooth the document frequencies in order avoid log zero issues, the default is 1
Value

A vector of topic coherence scores with length equal to the number of topics in the fitted model

References


See Also

semanticCoherence

Examples

```r
# Using the example from the LDA function
library(topicmodels)
data("AssociatedPress", package = "topicmodels")
lda <- LDA(AssociatedPress[1:20,], control = list(alpha = 0.1), k = 2)
topic_coherence(lda, AssociatedPress[1:20,])
```

```r
topic_diagnostics <- function(topic_model, dtm_data, top_n_tokens = 10, method = c("gamma_threshold", "largest_gamma"),
gamma_threshold = 0.2)
```

Description

Generate a dataframe containing the diagnostics for each topic in a topic model

Usage

```r
topic_diagnostics(topic_model, dtm_data, top_n_tokens = 10,
method = c("gamma_threshold", "largest_gamma"),
gamma_threshold = 0.2)
```

Arguments

- `topic_model`: a fitted topic model object from one of the following: `tm-class`
- `dtm_data`: a document-term matrix of token counts coercible to `slam_triplet_matrix` where each row is a document, each column is a token, and each entry is the frequency of the token in a given document
- `top_n_tokens`: an integer indicating the number of top words to consider for mean token length
- `method`: a string indicating which method to use - "gamma_threshold" or "largest_gamma"
- `gamma_threshold`: a number between 0 and 1 indicating the gamma threshold to be used when using the gamma threshold method, the default is 0.2
**topic_exclusivity**

**Value**

A dataframe where each row is a topic and each column contains the associated diagnostic values

**References**


**Examples**

```r
# Using the example from the LDA function
library(topicmodels)
data("AssociatedPress", package = "topicmodels")
lda <- LDA(AssociatedPress[1:20,], control = list(alpha = 0.1), k = 2)
topic_diagnostics(lda, AssociatedPress[1:20,])
```

<table>
<thead>
<tr>
<th>topic_exclusivity</th>
<th>Calculate the exclusivity of each topic in a topic model</th>
</tr>
</thead>
</table>

**Description**

Using the the N highest probability tokens for each topic, calculate the exclusivity for each topic

**Usage**

```r
topic_exclusivity(topic_model, top_n_tokens = 10, excl_weight = 0.5)
```

**Arguments**

- `topic_model`: a fitted topic model object from one of the following: `tm-class`
- `top_n_tokens`: an integer indicating the number of top words to consider, the default is 10
- `excl_weight`: a numeric between 0 and 1 indicating the weight to place on exclusivity versus frequency in the calculation, 0.5 is the default

**Value**

A vector of exclusivity values with length equal to the number of topics in the fitted model

**References**

See Also

exclusivity

Examples

# Using the example from the LDA function
library(topicmodels)
data("AssociatedPress", package = "topicmodels")
lda <- LDA(AssociatedPress[1:20,], control = list(alpha = 0.1), k = 2)
topic_exclusivity(lda)

---

<table>
<thead>
<tr>
<th>topic_size</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Calculate the size of each topic in a topic model</strong></td>
</tr>
</tbody>
</table>

Description

Calculate the size of each topic in a topic model based on the number of fractional tokens found in each topic.

Usage

```r
topic_size(topic_model)
```

Arguments

- `topic_model` 
  a fitted topic model object from one of the following: **tm-class**

Value

A vector of topic sizes with length equal to the number of topics in the fitted model

References


Examples

# Using the example from the LDA function
library(topicmodels)
data("AssociatedPress", package = "topicmodels")
lda <- LDA(AssociatedPress[1:20,], control = list(alpha = 0.1), k = 2)
topic_size(lda)
Index

coherence, 2
contain_equal_docs, 2

dist_from_corpus, 3
doc_prominence, 4

exclusivity, 10

mean_token_length, 5

n_topics, 5

semanticCoherence, 8

tf_df_dist, 6
tf_df_dist_diff, 7
topic_coherence, 7
topic_diagnostics, 8
topic_exclusivity, 9
topic_size, 10