06 Topic modeling

H. David Shea

04 Aug 2021

Contents

"Topic modeling is a method for unsupervised classification of such documents, similar to clustering on numeric data, which finds natural groups of items even when we're not sure what we're looking for."

"Latent Dirichlet allocation (LDA) is a particularly popular method for fitting a topic model. It treats each document as a mixture of topics, and each topic as a mixture of words. This allows documents to "overlap" each other in terms of content, rather than being separated into discrete groups, in a way that mirrors typical use of natural language."

Latent Dirichlet allocation

```
data("AssociatedPress")
AssociatedPress
#> <<DocumentTermMatrix (documents: 2246, terms: 10473)>>
#> Non-/sparse entries: 302031/23220327
#> Sparsity           : 99%
#> Maximal term length: 18
#> Weighting          : term frequency (tf)

ap_lda <- LDA(AssociatedPress, k = 2, control = list(seed = 1234))
ap_lda
#> A LDA_VEM topic model with 2 topics.
```

Word-topic probabilities

```
ap_topics <- tidy(ap_lda, matrix = "beta")
ap_topics %>%
    slice_sample(n = 10) %>%
    kable(caption = "Beta - per-topic-per-word probabilities - for selected AP articles.")
```

Table 1: Beta - per-topic-per-word probabilities - for selected AP articles.

topic	term	beta
1	raw	0.0001060
1	academy	0.0000002
2	contained	0.0001082
1	dismiss	0.0000000
2	kentucky	0.0000193
2	grande	0.0000000
1	anita	0.0000000
2	arnold	0.0000233
2	representing	0.0001345
2	detainees	0.0000740

It is simple enough to visualize the most common terms likely to be associated with the two topics.

```
ap_top_terms <- ap_topics %>%
    group_by(topic) %>%
    slice_max(beta, n = 10) %>%
    ungroup() %>%
    arrange(topic,-beta)

ap_top_terms %>%
    mutate(term = reorder_within(term, beta, topic)) %>%
    ggplot(aes(beta, term, fill = factor(topic))) +
    geom_col(show.legend = FALSE) +
    facet_wrap( ~ topic, scales = "free") +
    scale_y_reordered() +
    theme_light()
```

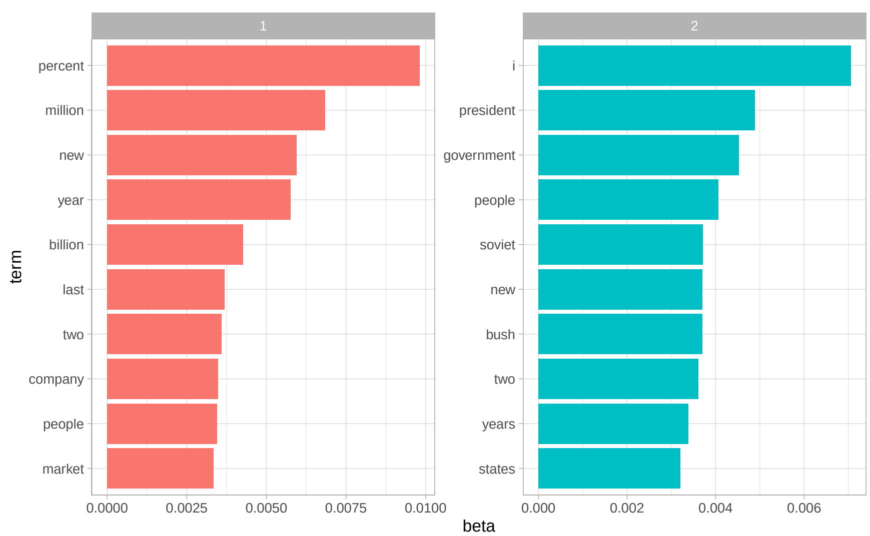

Figure 1: The terms that are most common within each topic

We can also consider the largest difference in beta between the two topics.

```r
beta_wide <- ap_topics %>%
    mutate(topic = paste0("topic", topic)) %>%
    pivot_wider(names_from = topic, values_from = beta) %>%
    filter(topic1 > .001 | topic2 > .001) %>%
    mutate(log_ratio = log2(topic2 / topic1))

beta_wide
#> # A tibble: 198 x 4
#>    term              topic1      topic2 log_ratio
#>    <chr>              <dbl>       <dbl>     <dbl>
#>  1 administration 0.000431  0.00138         1.68
#>  2 ago            0.00107   0.000842       -0.339
#>  3 agreement      0.000671  0.00104         0.630
#>  4 aid            0.0000476 0.00105         4.46
#>  5 air            0.00214   0.000297       -2.85
#>  6 american       0.00203   0.00168        -0.270
#>  7 analysts       0.00109   0.000000578   -10.9
#>  8 area           0.00137   0.000231       -2.57
#>  9 army           0.000262  0.00105         2.00
#> 10 asked          0.000189  0.00156         3.05
#> # ... with 188 more rows
```

```r
beta_wide %>%
    group_by(direction = log_ratio > 0) %>%
    slice_max(abs(log_ratio), n = 10) %>%
    ungroup() %>%
    mutate(term = reorder(term, log_ratio)) %>%
    ggplot(aes(log_ratio, term)) +
    geom_col() +
    labs(x = "Log2 ratio of beta in topic 2 / topic 1", y = NULL) +
    theme_light()
```

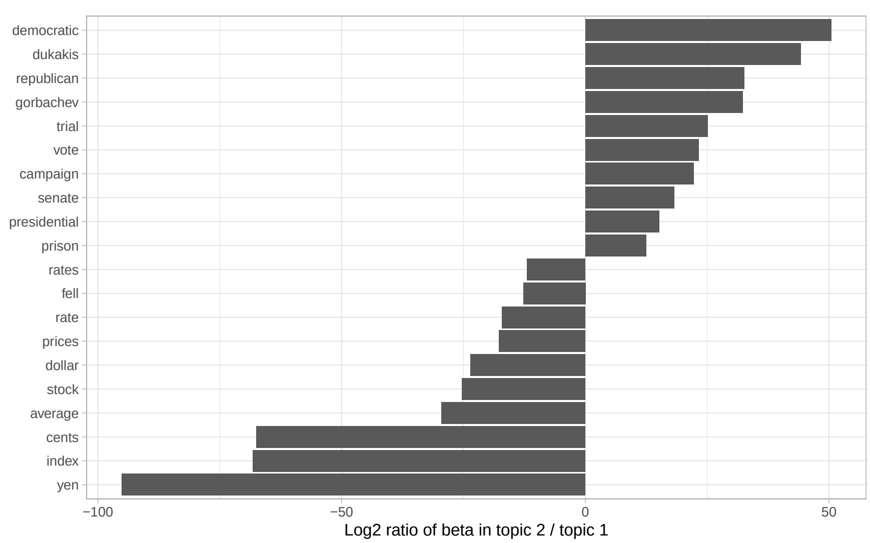

Figure 2: Words with the greatest difference in beta between topic 2 and topic 1

This helps confirm the two topics the algorithm identified as the difference terms are very to the high probability terms visualized above (i.e., topic 1 is "financial" and topic to is "political").

Document-topic probabilities

```
ap_documents <- tidy(ap_lda, matrix = "gamma")
ap_documents %>%
    slice_min(document, n = 10) %>%
    kable(caption = "Gamma - per-document-per-topic probabilities - for selected AP articles.")
```

Table 2: Gamma - per-document-per-topic probabilities - for selected AP articles.

document	topic	gamma
1	1	0.2480617
1	2	0.7519383
2	1	0.3615485
2	2	0.6384515
3	1	0.5265844
3	2	0.4734156
4	1	0.3566530
4	2	0.6433470
5	1	0.1812767
5	2	0.8187233

Interpretation: the model estimate is that about 25% of the words in document 1 were generated from topic 1.

Most documents have a mix of estimates for topic 1 and topic 2, but note the document 6 has a gamma of almost zero to topic 1.

```
tidy(AssociatedPress) %>%
  filter(document == 6) %>%
  arrange(desc(count)) %>%
    slice_max(count, n = 10) %>%
    kable(caption = "Most common words in document 6 correspond to topic 2.")
```

Table 3: Most common words in document 6 correspond to topic 2.

document	term	count
6	noriega	16
6	panama	12
6	jackson	6
6	powell	6
6	administration	5
6	economic	5
6	general	5
6	i	5
6	panamanian	5

document	term	count
6	american	4
6	letter	4
6	official	4
6	officials	4
6	president	4
6	reagan	4

Example: the great library heist

Run a test on "known" text to see how good the LDA works.

```
titles <- c(
    "Twenty Thousand Leagues under the Sea",
    "The War of the Worlds",
    "Pride and Prejudice",
    "Great Expectations"
)

books <- gutenberg_works(title %in% titles) %>%
    gutenberg_download(meta_fields = "title") %>%
    group_by(title) %>% # removing table of contents
    filter(!((title == "Great Expectations") & between(row_number(), 13, 74)),
           !((title == "Pride and Prejudice") & between(row_number(), 13, 135))) %>%
    ungroup()

# divide into documents, each representing one chapter
by_chapter <- books %>%
  group_by(title) %>%
  mutate(chapter = cumsum(str_detect(
    text, regex("^[ ]*chapter ", ignore_case = TRUE)
  ))) %>%
  ungroup() %>%
  filter(chapter > 0) %>%
  unite(document, title, chapter)

# split into words
by_chapter_word <- by_chapter %>%
  unnest_tokens(word, text)

# find document-word counts
word_counts <- by_chapter_word %>%
  anti_join(stop_words, by = "word") %>%
  count(document, word, sort = TRUE) %>%
  ungroup()

word_counts %>%
    slice_max(n, n = 10) %>%
    kable(caption = "Word counts for four novels.")
```

Table 4: Word counts for four novels.

document	word	n
Great Expectations_57	joe	88
Great Expectations_7	joe	70
Great Expectations_17	biddy	63
Great Expectations_27	joe	58
Great Expectations_38	estella	58
Great Expectations_2	joe	56
Great Expectations_23	pocket	53
Great Expectations_15	joe	50
Great Expectations_18	joe	50
The War of the Worlds_16	brother	50

LDA on chapters

```
chapters_dtm <- word_counts %>%
    cast_dtm(document, word, n)

chapters_dtm
#> <<DocumentTermMatrix (documents: 193, terms: 18314)>>
#> Non-/sparse entries: 105610/3428992
#> Sparsity           : 97%
#> Maximal term length: 19
#> Weighting          : term frequency (tf)

chapters_lda <- LDA(chapters_dtm, k = 4, control = list(seed = 1234))

chapters_lda
#> A LDA_VEM topic model with 4 topics.

chapter_topics <- tidy(chapters_lda, matrix = "beta")

chapter_topics %>%
    slice_head(n = 10) %>%
    kable(caption = "Beta - per-topic-per-term probabilities - for four novels.")
```

Table 5: Beta - per-topic-per-term probabilities - for four novels.

topic	term	beta
1	joe	0.0125618
2	joe	0.0000000
3	joe	0.0000000
4	joe	0.0000000
1	biddy	0.0041389
2	biddy	0.0000000
3	biddy	0.0000000
4	biddy	0.0000000
1	estella	0.0043022

topic	term	beta
2	estella	0.0000000

```
top_terms <- chapter_topics %>%
  group_by(topic) %>%
  slice_max(beta, n = 5) %>%
  ungroup() %>%
  arrange(topic, -beta)

top_terms %>%
    kable(caption = "Top terms within each topic for four novels.")
```

Table 6: Top terms within each topic for four novels.

topic	term	beta
1	joe	0.0125618
1	miss	0.0069131
1	time	0.0065458
1	pip	0.0059178
1	looked	0.0057931
2	captain	0.0153932
2	nautilus	0.0130383
2	sea	0.0088463
2	nemo	0.0087006
2	ned	0.0080236
3	elizabeth	0.0157928
3	darcy	0.0098672
3	bennet	0.0077773
3	miss	0.0075422
3	jane	0.0073007
4	people	0.0069482
4	martians	0.0067670
4	black	0.0054525
4	time	0.0053846
4	night	0.0045289

```
top_terms %>%
    mutate(term = reorder_within(term, beta, topic)) %>%
    ggplot(aes(beta, term, fill = factor(topic))) +
    geom_col(show.legend = FALSE) +
    facet_wrap( ~ topic, scales = "free") +
    scale_y_reordered() +
    theme_light()
```

"These topics are pretty clearly associated with the four books!"

- 1 - "pip" and "joe" from *Great Expectations*
- 2 - "captain" and "nautilus" from *Twenty Thousand Leagues under the Sea*
- 3 - "elizabeth" and "darcy" from *Pride and Prejudice*
- 4 - "martians" and "black" and "night" from *The War of the Worlds*

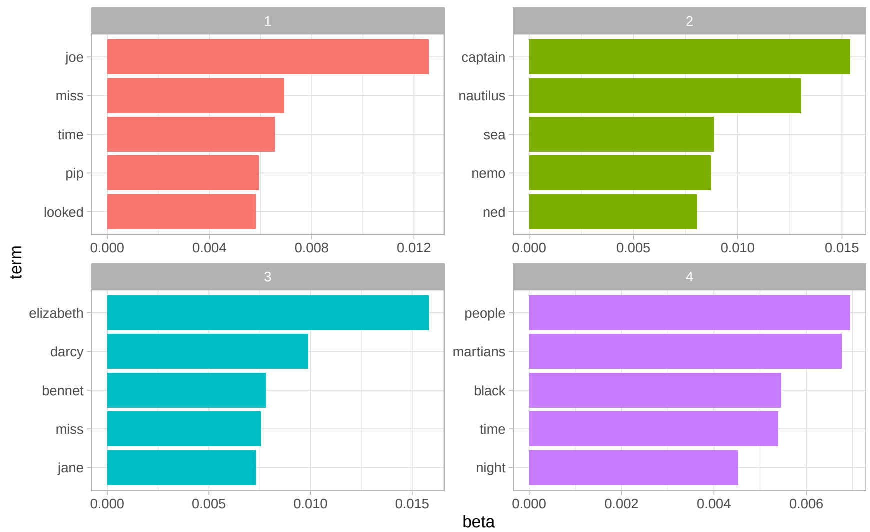

Figure 3: The terms that are most common within each topic

Per-document classification

```
chapters_gamma <- tidy(chapters_lda, matrix = "gamma")

chapters_gamma %>%
    slice_head(n = 10) %>%
    kable(caption = "Gamma - per-document-per-topic probabilities - for four novels.")
```

Table 7: Gamma - per-document-per-topic probabilities - for four novels.

document	topic	gamma
Great Expectations_57	1	0.9999463
Great Expectations_7	1	0.9999433
Great Expectations_17	1	0.9999169
Great Expectations_27	1	0.9999242
Great Expectations_38	1	0.9999486
Great Expectations_2	1	0.9999331
Great Expectations_23	1	0.8866287
Great Expectations_15	1	0.9999435
Great Expectations_18	1	0.9999499
The War of the Worlds_16	1	0.0000147

"Now that we have these topic probabilities, we can see how well our unsupervised learning did at distinguishing the four books. We'd expect that chapters within a book would be found to be mostly (or entirely), generated from the corresponding topic."

```
chapters_gamma <- chapters_gamma %>%
    separate(document,
             c("title", "chapter"),
```

```
             sep = "_",
             convert = TRUE)

chapters_gamma %>%
    slice_head(n = 10) %>%
    kable(caption = "Gamma - per-document-per-topic probabilities - for four novels.")
```

Table 8: Gamma - per-document-per-topic probabilities - for four novels.

title	chapter	topic	gamma
Great Expectations	57	1	0.9999463
Great Expectations	7	1	0.9999433
Great Expectations	17	1	0.9999169
Great Expectations	27	1	0.9999242
Great Expectations	38	1	0.9999486
Great Expectations	2	1	0.9999331
Great Expectations	23	1	0.8866287
Great Expectations	15	1	0.9999435
Great Expectations	18	1	0.9999499
The War of the Worlds	16	1	0.0000147

```
# reorder titles in order of topic 1, topic 2, etc before plotting
chapters_gamma %>%
    mutate(title = reorder(title, gamma * topic)) %>%
    ggplot(aes(factor(topic), gamma)) +
    geom_boxplot() +
    facet_wrap( ~ title) +
    labs(x = "topic", y = expression(gamma)) +
    theme_light()
```

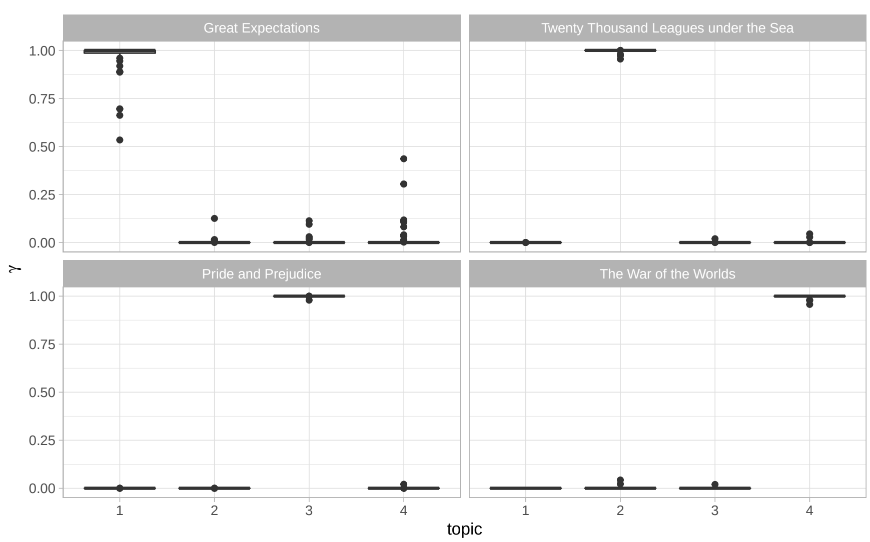

Figure 4: The gamma probabilities for each chapter within each book

Indeed, *Twenty Thousand Leagues under the Sea*, *Pride and Prejudice*, *The War of the Worlds* were almost uniquely identified as a single topic per book.

It does look like *Great E#xpectations* (which should be identified with topic 1) has chapters associated with other topics.

We can also look by chapter to see if there are cases of entire chapters associated with the topic most closely associated with a different book.

```
chapter_classifications <- chapters_gamma %>%
    group_by(title, chapter) %>%
    slice_max(gamma, n = 1) %>%
    ungroup()

chapter_classifications %>%
    slice_head(n = 10) %>%
    kable()
```

title	chapter	topic	gamma
Great Expectations	1	1	0.6953004
Great Expectations	2	1	0.9999331
Great Expectations	3	1	0.9596901
Great Expectations	4	1	0.9999342
Great Expectations	5	1	0.9999422
Great Expectations	6	1	0.9996857
Great Expectations	7	1	0.9999433
Great Expectations	8	1	0.9999467
Great Expectations	9	1	0.9999113
Great Expectations	10	1	0.9999077

```
book_topics <- chapter_classifications %>%
    count(title, topic) %>%
    group_by(title) %>%
    slice_max(n, n = 1) %>%
    ungroup() %>%
    transmute(consensus = title, topic)

chapter_classifications %>%
    inner_join(book_topics, by = "topic") %>%
    filter(title != consensus)
#> # A tibble: 0 x 5
#> # ... with 5 variables: title <chr>, chapter <int>, topic <int>, gamma <dbl>,
#> #   consensus <chr>
```

By word assignments: `augment()`

Here we look at which words in the document were assigned to which topic by the LDA.

```
assignments <- augment(chapters_lda, data = chapters_dtm)

assignments %>%
    slice_head(n = 10) %>%
    kable()
```

document	term	count	.topic
Great Expectations_57	joe	88	1
Great Expectations_7	joe	70	1
Great Expectations_17	joe	5	1
Great Expectations_27	joe	58	1
Great Expectations_2	joe	56	1
Great Expectations_23	joe	1	1
Great Expectations_15	joe	50	1
Great Expectations_18	joe	50	1
Great Expectations_9	joe	44	1
Great Expectations_13	joe	40	1

```
assignments <- assignments %>%
    separate(document,
             c("title", "chapter"),
             sep = "_",
             convert = TRUE) %>%
    inner_join(book_topics, by = c(".topic" = "topic"))

assignments %>%
    slice_head(n = 10) %>%
    kable()
```

title	chapter	term	count	.topic	consensus
Great Expectations	57	joe	88	1	Great Expectations
Great Expectations	7	joe	70	1	Great Expectations
Great Expectations	17	joe	5	1	Great Expectations
Great Expectations	27	joe	58	1	Great Expectations
Great Expectations	2	joe	56	1	Great Expectations
Great Expectations	23	joe	1	1	Great Expectations
Great Expectations	15	joe	50	1	Great Expectations
Great Expectations	18	joe	50	1	Great Expectations
Great Expectations	9	joe	44	1	Great Expectations
Great Expectations	13	joe	40	1	Great Expectations

We can use this data to visualize a **confusion matrix**, showing how often words from one book were assigned to the consensus topic of another book.

```
assignments %>%
    count(title, consensus, wt = count) %>%
    mutate(across(c(title, consensus), ~ str_wrap(., 20))) %>%
    group_by(title) %>%
    mutate(percent = n / sum(n)) %>%
    ggplot(aes(consensus, title, fill = percent)) +
    geom_tile() +
    scale_fill_gradient2(high = "darkred", label = percent_format()) +
    theme_minimal() +
    theme(axis.text.x = element_text(angle = 90, hjust = 1),
          panel.grid = element_blank()) +
    labs(x = "Book words were assigned to",
```

```
        y = "Book words came from",
        fill = "% of assignments") +
    theme_light()
```

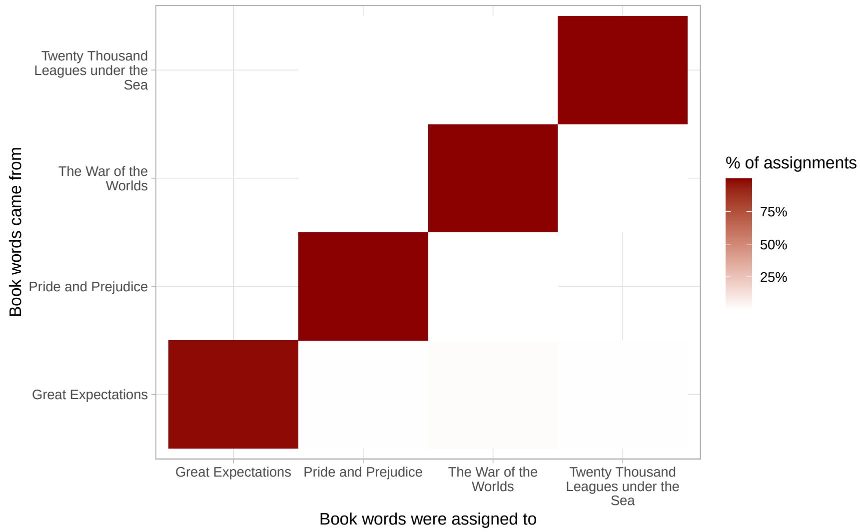

Figure 5: Confusion matrix showing where LDA assigned the words from each book. Each row of this table represents the true book each word came from, and each column represents what book it was assigned to.

```
wrong_words <- assignments %>%
    filter(title != consensus)

wrong_words %>%
    slice_head(n = 10) %>%
    kable()
```

title	chapter	term	count	.topic	consensus
Great Expectations	46	captain	1	2	Twenty Thousand Leagues under the Sea
Great Expectations	32	captain	1	2	Twenty Thousand Leagues under the Sea
The War of the Worlds	17	captain	5	2	Twenty Thousand Leagues under the Sea
Great Expectations	54	sea	2	2	Twenty Thousand Leagues under the Sea
Great Expectations	54	water	15	2	Twenty Thousand Leagues under the Sea
Great Expectations	1	water	1	4	The War of the Worlds
Great Expectations	54	road	1	4	The War of the Worlds
Great Expectations	54	people	1	4	The War of the Worlds
Great Expectations	1	people	1	4	The War of the Worlds
Great Expectations	21	people	1	4	The War of the Worlds

```
wrong_words %>%
    count(title, consensus, term, wt = count) %>%
    ungroup() %>%
    arrange(desc(n)) %>%
    slice_max(n, n = 10) %>%
    kable()
```

title	consensus	term	n
Great Expectations	Pride and Prejudice	galley	16
Great Expectations	Twenty Thousand Leagues under the Sea	water	15
Great Expectations	Pride and Prejudice	jane	13
Great Expectations	The War of the Worlds	steamer	12
Great Expectations	Twenty Thousand Leagues under the Sea	board	8
Great Expectations	The War of the Worlds	smoke	7
Great Expectations	The War of the Worlds	sun	7
Great Expectations	The War of the Worlds	black	6
Great Expectations	The War of the Worlds	sky	6
Great Expectations	The War of the Worlds	tilted	6

```
word_counts %>%
  filter(word == "flopson")
#> # A tibble: 3 x 3
#>   document              word        n
#>   <chr>                 <chr>   <int>
#> 1 Great Expectations_22 flopson    10
#> 2 Great Expectations_23 flopson     7
#> 3 Great Expectations_33 flopson     1
```

Alternative LDA implementations

Still problems with Java on Mac M1 machines and the mallet package requires Java.