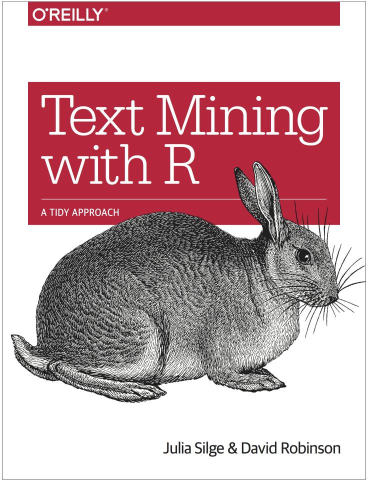
Socio-Informatics 348

Text Analysis
Topic Modelling with LDA

Dr Lisa Martin

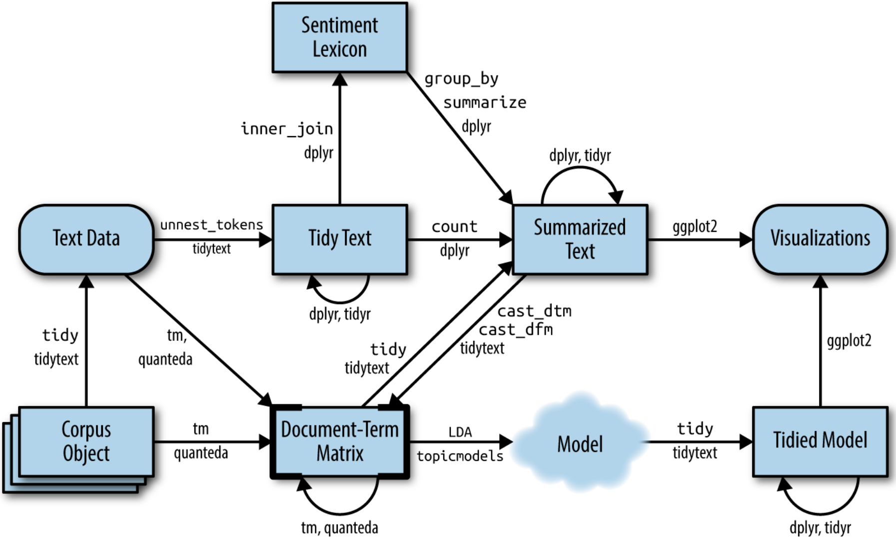
Department of Information Science
Stellenbosch University

Today's Reading



Text Mining with R, Chapter 6

Topic Modeling



What is Topic Modeling?

Problem: Collections of documents (blog posts, news articles) that need grouping

Topic modeling: Unsupervised classification method

- Similar to clustering on numeric data
- Finds natural groups without predefined categories
- Discovers patterns we're not sure we're looking for

Key advantage: Documents can "overlap" in content

- Not separated into discrete groups
- Mirrors typical use of natural language

Latent Dirichlet Allocation (LDA)

Most popular topic modeling algorithm

Core concept:

- Each document = mixture of topics
- Each topic = mixture of words

Example: Two-topic model of news

- **Politics topic:** "President", "Congress", "government"
- **Entertainment topic:** "movies", "television", "actor"
- **Shared words:** "budget" might appear in both

Two Guiding Principles of LDA

Principle 1: Every document is a mixture of topics

Example in 2-topic model:

- Document 1: 90% topic A, 10% topic B
- Document 2: 30% topic A, 70% topic B

Principle 2: Every topic is a mixture of words

Example topics:

- Topic A: High probability for "President", "Congress"
- Topic B: High probability for "movies", "actor"

LDA estimates both simultaneously

Topic Modeling Workflow

Text Data

$\downarrow$

Process to Document-Term Matrix

$\downarrow$

LDA Algorithm (topicmodels package)

$\downarrow$

`tidy()` model output

$\downarrow$

Analysis & Visualisation (dplyr, ggplot2)

Tidy principles apply to topic modeling!

Example: Associated Press News Articles

Dataset:

- 2,246 news articles
- Mostly from 1988
- Already in DocumentTermMatrix format

```
# set a seed so that the output of the model is predictable
ap_lda <- LDA(AssociatedPress, k = 2, control = list(seed = 1234))
ap_lda
#> A LDA_VEM topic model with 2 topics.
```

Extracting Per-Topic-Per-Word Probabilities

Beta (β): Probability of word being generated from topic

```
library(tidytext)

ap_topics <- tidy(ap_lda, matrix = "beta")
ap_topics
#> # A tibble: 20,946 × 3
#>    topic term          beta
#>    <int> <chr>        <dbl>
#>  1     1 aaron      1.69e-12
#>  2     2 aaron      3.90e- 5
#>  3     1 abandon    2.65e- 5
#>  4     2 abandon    3.99e- 5
#>  5     1 abandoned  1.39e- 4
#>  6     2 abandoned  5.88e- 5
#>  7     1 abandoning 2.45e-33
#>  8     2 abandoning 2.34e- 5
#>  9     1 abbott     2.13e- 6
```

Finding Top Terms in Each Topic

```r
library(ggplot2)
library(dplyr)

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()
```

Finding Top Terms in Each Topic

Use `reorder_within()` **and** `scale_y_reordered()`

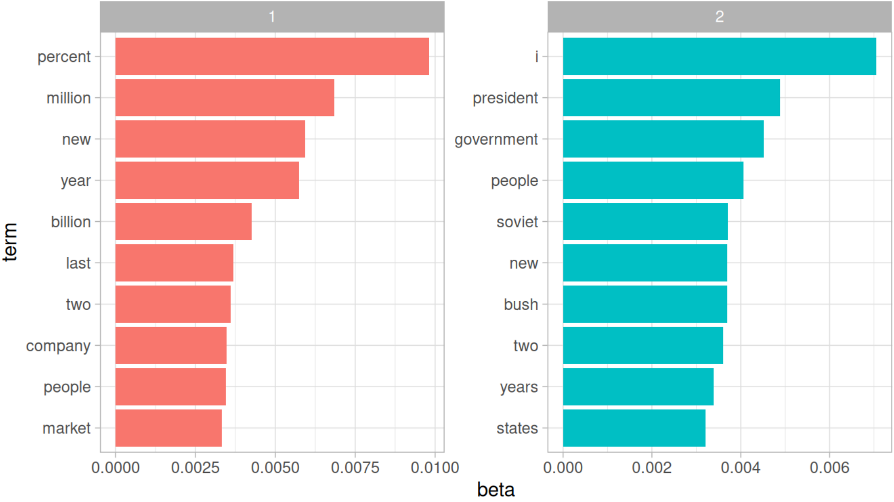
- Handles faceted plots with independent axes
- Orders terms by beta within each topic

Topic 1 (Financial): percent, million, billion, company, bank
Topic 2 (Political): president, government, soviet, party, war
Some overlap: "new", "people" appear in both topics

Insights from visualisation:

- Clear distinction between topics
- Topic 1: business/financial news
- Topic 2: political/national news
- Natural language overlap is preserved

Finding Top Terms in Each Topic

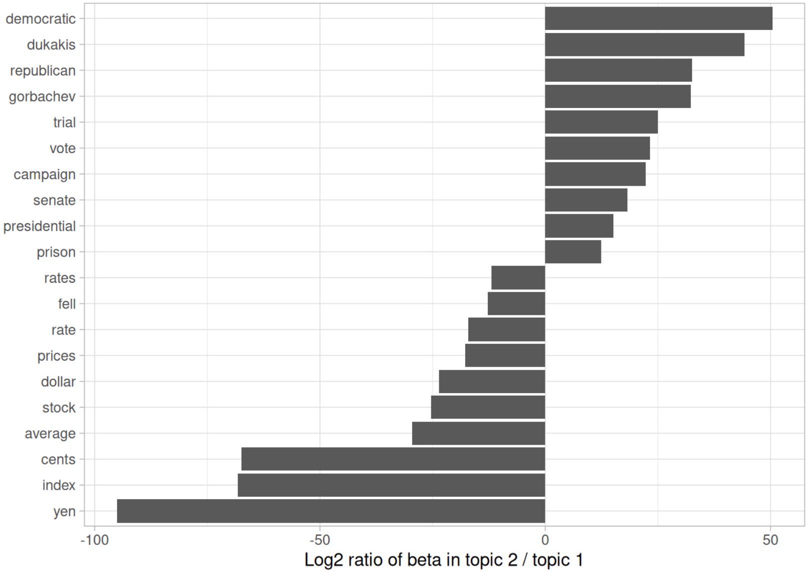


Greatest Differences in Beta Between Topics

Log ratio: $\log_2(\frac{\beta_2}{\beta_1})$

```
library(tidyr)

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 × 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
```

Greatest Differences in Beta Between Topics



Per-Document-Per-Topic Probabilities

Gamma (γ): Proportion of document from each topic
Document 6 is almost entirely Topic 2 (political)

```
ap_documents <- tidy(ap_lda, matrix = "gamma")
ap_documents
#> # A tibble: 4,492 × 3
#>    document topic    gamma
#>       <int> <int>    <dbl>
#>  1        1     1 0.248
#>  2        2     1 0.362
#>  3        3     1 0.527
#>  4        4     1 0.357
#>  5        5     1 0.181
#>  6        6     1 0.000588
#>  7        7     1 0.773
#>  8        8     1 0.00445
#>  9        9     1 0.967
#> 10       10     1 0.147
```

Checking Document Classification

Content: Article about US-Panama relations and Manuel Noriega
Validation: Algorithm correctly identified as political news!
Document 6 most common words:

```
tidy(AssociatedPress) %>%
  filter(document == 6) %>%
  arrange(desc(count))
#> # A tibble: 287 × 3
#>    document term           count
#>       <int> <chr>          <dbl>
#>  1        6 noriega           16
#>  2        6 panama            12
#>  3        6 jackson            6
#>  4        6 powell             6
#>  5        6 administration     5
#>  6        6 economic           5
#>  7        6 general            5
#>  8        6 i                  5
```

Case Study: The Great Library Heist

Scenario: Four books torn into chapters and mixed together

1. *Great Expectations* - Charles Dickens
2. *The War of the Worlds* - H.G. Wells
3. *Twenty Thousand Leagues Under the Sea* - Jules Verne
4. *Pride and Prejudice* - Jane Austen

Challenge: Restore chapters to original books
Approach: Use LDA to cluster unlabeled chapters
Advantage: We know the “right answer” for validation

Pre-Processing the Books

After downloading using `gutenbergr`:

```
library(stringr)

# 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)
```

Pre-Processing the Books

```
# 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) %>%
  count(document, word, sort = TRUE)

word_counts
#> # A tibble: 104,704 × 3
#>    document                 word      n
#>    <chr>                    <chr> <int>
#>  1 Great Expectations_57    joe      88
#>  2 Great Expectations_7     joe      70
#>  3 Great Expectations_17    biddy    63
```

Casting to DocumentTermMatrix

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

chapters_dtm
#> <<DocumentTermMatrix (documents: 193, terms: 18202)>>
#> Non-/sparse entries: 104704/3408282
#> Sparsity           : 97%
#> Maximal term length: 19
#> Weighting          : term frequency (tf)
```

Result:

- 193 documents (chapters)
- 18,202 terms
- 97% sparse
- Four topics (one per book)

Performing LDA on Chapters

```r
chapters_lda <- LDA(chapters_dtm, k = 4, control = list(seed = 1234))
chapters_lda
#> A LDA_VEM topic model with 4 topics.
```

```r
chapter_topics <- tidy(chapters_lda, matrix = "beta")
chapter_topics
#> # A tibble: 72,808 × 3
#>    topic term        beta
#>    <int> <chr>      <dbl>
#>  1     1 joe     1.41e- 16
#>  2     2 joe     5.13e- 54
#>  3     3 joe     1.40e-  2
#>  4     4 joe     2.75e- 39
#>  5     1 biddy   3.96e- 22
#>  6     2 biddy   5.72e- 62
#>  7     3 biddy   4.63e-  3
#>  8     4 biddy   3.24e- 47
#>  9     1 estella 4.19e- 18
```

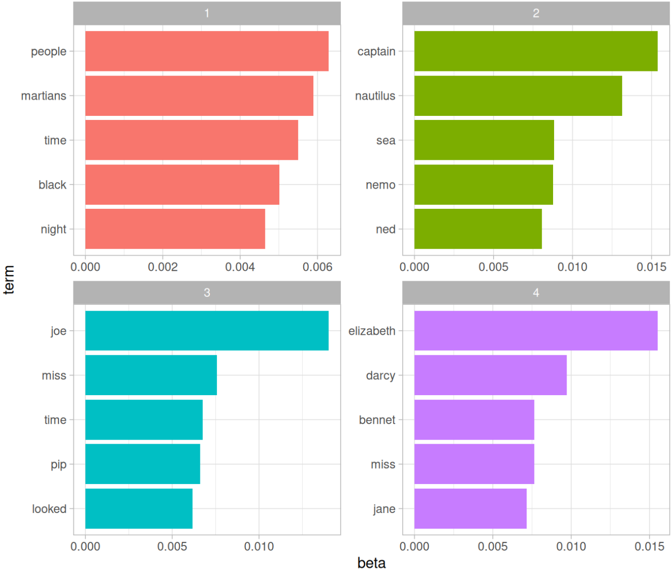
Top Terms in Each Topic

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

top_terms
#> # A tibble: 20 × 3
#>    topic term        beta
#>    <int> <chr>      <dbl>
#>  1     1 people   0.00629
#>  2     1 martians 0.00590
#>  3     1 time     0.00550
#>  4     1 black    0.00501
#>  5     1 night    0.00464
#>  6     2 captain  0.0154
```

Top Terms in Each Topic

```
library(ggplot2)

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()
```

Top Terms in Each Topic



Top Terms in Each Topic

Topic 1	people, martians, time, black, night $\rightarrow$ *War of the Worlds*
Topic 2	captain, nautilus, sea, nemo, ned $\rightarrow$ *Twenty Thousand Leagues*
Topic 3	joe, miss, time, pip, looked $\rightarrow$ *Great Expectations*
Topic 4	elizabeth, darcy, bennet, miss, jane $\rightarrow$ *Pride and Prejudice*

Topics clearly distinguish the four books!

LDA as Fuzzy Clustering

Hard clustering: Each item belongs to one cluster
Fuzzy clustering (LDA): Items can partially belong to multiple clusters

Evidence in results:

- Word "miss" appears in topics 3 and 4
- Word "time" appears in topics 1 and 3
- Reflects natural language overlap

Advantage: More realistic representation of language use

Examining Document-Topic Probabilities

```
chapters_gamma <- tidy(chapters_lda, matrix = "gamma")
chapters_gamma
#> # A tibble: 772 × 3
#>    document                 topic     gamma
#>    <chr>                    <int>     <dbl>
#>  1 Great Expectations_57        1 0.0000134
#>  2 Great Expectations_7         1 0.0000146
#>  3 Great Expectations_17        1 0.0000210
#>  4 Great Expectations_27        1 0.0000190
#>  5 Great Expectations_38        1 0.0000127
#>  6 Great Expectations_2         1 0.0000171
#>  7 Great Expectations_23        1 0.290
#>  8 Great Expectations_15        1 0.0000143
#>  9 Great Expectations_18        1 0.0000126
#> 10 The War of the Worlds_16     1 1.00
#> # ℹ 762 more rows
```

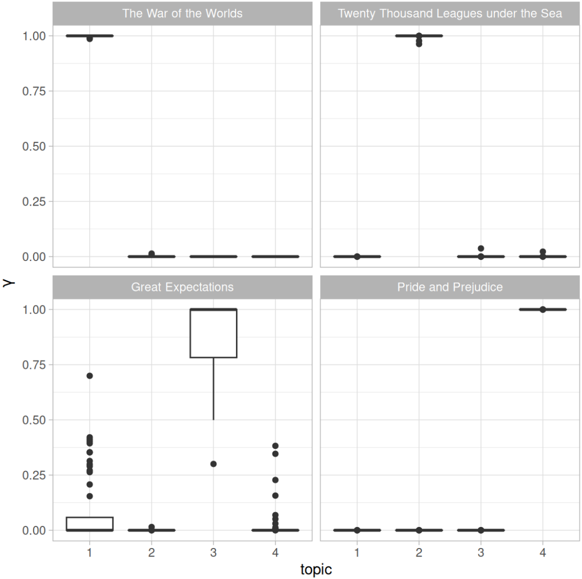
Examining Document-Topic Probabilities

```
chapters_gamma <- chapters_gamma %>%
  separate(document, c("title", "chapter"), sep = "_", convert = TRUE)

chapters_gamma
#> # A tibble: 772 × 4
#>    title                 chapter topic     gamma
#>    <chr>                   <int> <int>     <dbl>
#>  1 Great Expectations         57     1 0.0000134
#>  2 Great Expectations          7     1 0.0000146
#>  3 Great Expectations         17     1 0.0000210
#>  4 Great Expectations         27     1 0.0000190
#>  5 Great Expectations         38     1 0.0000127
#>  6 Great Expectations          2     1 0.0000171
#>  7 Great Expectations         23     1 0.290
#>  8 Great Expectations         15     1 0.0000143
#>  9 Great Expectations         18     1 0.0000126
#> 10 The War of the Worlds      16     1 1.00
#> # ℹ 762 more rows
```

Examining Document-Topic Probabilities

```
# 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))
```

Examining Document-Topic Probabilities



Examining Document-Topic Probabilities

Results:

- Most chapters strongly associated with one topic
- Some Great Expectations chapters show mixed membership
- Other three books almost perfectly classified

Measuring Classification Success

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

chapter_classifications
#> # A tibble: 193 × 4
#>    title              chapter topic gamma
#>    <chr>                <int> <int> <dbl>
#>  1 Great Expectations       1     3 0.597
#>  2 Great Expectations       2     3 1.00
#>  3 Great Expectations       3     3 0.579
#>  4 Great Expectations       4     3 1.00
```

Measuring Classification Success

Find "consensus" topic for each book (most common topic)

```
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: 1 × 5
#>   title                chapter topic gamma consensus
#>   <chr>                  <int> <int> <dbl> <chr>
#> 1 Great Expectations        54     1 0.700 The War of the Worlds
```

Measuring Classification Success

Misclassifications:

- Great Expectations chapter 23: assigned to Topic 1
- Great Expectations chapter 54: assigned to Topic 3

Overall: Only 2 chapters misclassified out of 193!

The augment() Function

Purpose: See which topic each word was assigned to

Output: Original document-word pairs + `.topic` column

Use case: Find which words were misassigned

- Combine with consensus topics
- Filter for `title != consensus`
- Analyse patterns in errors

The augment() Function

```
assignments <- augment(chapters_lda, data = chapters_dtm)
assignments
#> # A tibble: 104,704 × 4
#>    document                term  count .topic
#>    <chr>                   <chr> <dbl>  <dbl>
#>  1 Great Expectations_57 joe      88      3
#>  2 Great Expectations_7  joe      70      3
#>  3 Great Expectations_17 joe       5      3
#>  4 Great Expectations_27 joe      58      3
#>  5 Great Expectations_2  joe      56      3
#>  6 Great Expectations_23 joe       1      3
#>  7 Great Expectations_15 joe      50      3
#>  8 Great Expectations_18 joe      50      3
#>  9 Great Expectations_9  joe      44      3
#> 10 Great Expectations_13 joe      40      3
#> # ℹ 104,694 more rows
```

Visualising Classification Errors

Confusion matrix: Shows word assignment patterns

- Rows: true book words came from
- Columns: book words were assigned to
- Color: percentage of assignments

Findings:

- Pride and Prejudice: nearly perfect
- Twenty Thousand Leagues: nearly perfect
- War of the Worlds: nearly perfect
- Great Expectations: significant misassignments

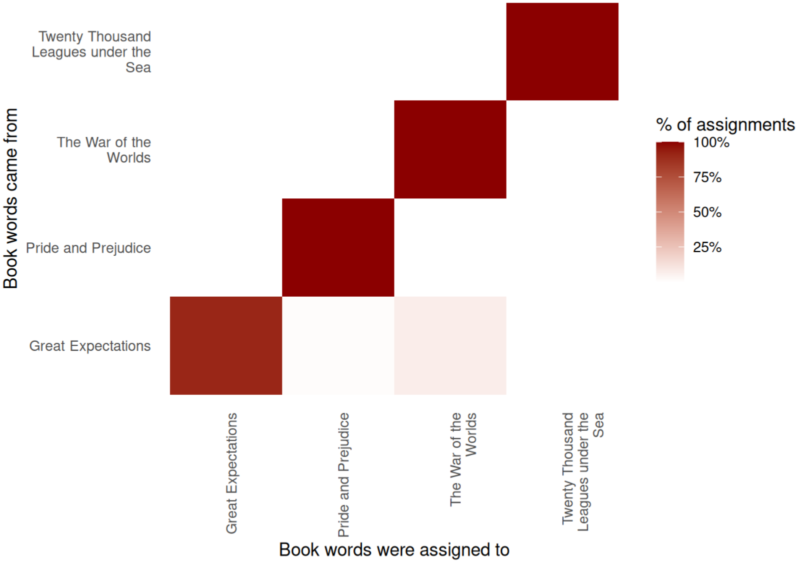
Visualising Classification Errors

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

assignments
#> # A tibble: 104,704 × 6
#>   title              chapter term  count .topic consensus
#>   <chr>                <int> <chr> <dbl>  <dbl> <chr>
#> 1 Great Expectations      57 joe      88      3 Great Expectations
#> 2 Great Expectations       7 joe      70      3 Great Expectations
#> 3 Great Expectations      17 joe       5      3 Great Expectations
#> 4 Great Expectations      27 joe      58      3 Great Expectations
#> 5 Great Expectations       2 joe      56      3 Great Expectations
#> 6 Great Expectations      23 joe       1      3 Great Expectations
```

Visualising Classification Errors

```r
library(scales)

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")
```

Visualising Classification Errors



Most Commonly Mistaken Words

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

wrong_words
#> # A tibble: 3,641 × 6
#>    title                                 chapter term     count .topic con
#>    <chr>                                   <int> <chr>    <dbl>  <dbl> <ch
#>  1 Great Expectations                         20 brother      1      1 The
#>  2 Great Expectations                         37 brother      2      4 Pri
#>  3 Great Expectations                         22 brother      4      4 Pri
#>  4 Twenty Thousand Leagues under the Sea       8 miss         1      3 Gre
#>  5 Great Expectations                          5 sergeant    37      1 The
#>  6 Great Expectations                         46 captain      1      1 The
#>  7 Great Expectations                         32 captain      1      1 The
#>  8 The War of the Worlds                      17 captain      5      2 Twe
#>  9 Great Expectations                         54 sea          2      1 The
#> 10 Great Expectations                          1 sea          2      1 The
#> # ℹ 3,631 more rows
```

Most Commonly Mistaken Words

```
wrong_words %>%
  count(title, consensus, term, wt = count) %>%
  ungroup() %>%
  arrange(desc(n))
#> # A tibble: 2,820 × 4
#>    title              consensus            term          n
#>    <chr>              <chr>                <chr>     <dbl>
#>  1 Great Expectations The War of the Worlds boat         39
#>  2 Great Expectations The War of the Worlds sergeant     37
#>  3 Great Expectations The War of the Worlds river        34
#>  4 Great Expectations The War of the Worlds jack         28
#>  5 Great Expectations The War of the Worlds tide         28
#>  6 Great Expectations The War of the Worlds water        25
#>  7 Great Expectations The War of the Worlds black        19
#>  8 Great Expectations The War of the Worlds soldiers     19
#>  9 Great Expectations The War of the Worlds london       18
#> 10 Great Expectations The War of the Worlds people       18
#> # ℹ 2,810 more rows
```

Most Commonly Mistaken Words

Words from Great Expectations misassigned to War of the Worlds:

- boat (39), sergeant (37), river (34)
- water (25), black (19), soldiers (19)
- These have thematic similarity!

Interesting case: "flopson"

- Appears only in Great Expectations
- Assigned to Pride and Prejudice cluster
- LDA is stochastic—can make mistakes

Lesson: Algorithm isn't perfect, but performs well overall

Understanding Model Limitations

Reasons for errors:

1. **Word frequency:**
   - "love", "lady" more common in Pride and Prejudice
   - When they appear in Great Expectations, pull toward wrong topic
2. **Thematic overlap:**
   - "boat", "water" relevant to multiple books
   - Model must make probabilistic choices
3. **Stochastic nature:**
   - LDA uses random initialization
   - Can accidentally create topics spanning multiple books
   - Setting seed ensures reproducibility