CTA-ED Exercise 5: Unsupervised learning (topic models)

Nayah and Min

12/03/2024

Exercises

Setup

Before proceeding, we'll load the packages we will need for this tutorial.

```
library(tidyverse) # loads dplyr, ggplot2, and others
library(stringr) # to handle text elements
library(tidytext) # includes set of functions useful for manipulating text
```

```
## Warning: 程辑包'tidytext'是用R版本4.3.2 来建造的
```

```
library(topicmodels) # to estimate topic models
```

```
## Warning: 程辑包'topicmodels'是用R版本4.3.3 来建造的
```

```
library(gutenbergr) # to get text data
```

```
## Warning: 程辑包'gutenbergr'是用R版本4.3.3 来建造的
```

```
library(scales)
library(tm)
```

```
## Warning: 程辑包'tm'是用R版本4.3.3 来建造的
```

```
library(ggthemes) # to make your plots look nice
```

```
## Warning: 程辑包'ggthemes'是用R版本4.3.2 来建造的
```

```
library(readr)
library(quanteda)
```

```
## Warning: 程辑包'quanteda'是用R版本4.3.2 来建造的
```

```
## Warning in .recacheSubclasses(def@className, def, env):
## "replValueSp"类别的子类别"ndiMatrix"没有定义；因此没有更新
```

```
## Warning in stringi::stri_info(): Your current locale is not in the list of
## available locales. Some functions may not work properly. Refer to
## stri_locale_list() for more details on known locale specifiers.

## Warning in stringi::stri_info(): Your current locale is not in the list of
## available locales. Some functions may not work properly. Refer to
## stri_locale_list() for more details on known locale specifiers.
```

```
library(quanteda.textmodels)
```

```
## Warning: 程辑包'quanteda.textmodels'是用R版本4.3.3 来建造的
```

```
#install_package(devtools)
devtools::install_github("matthewjdenny/preText")
```

```
## Skipping install of 'preText' from a github remote, the SHA1 (4d40c44c) has not changed sinc
e last install.
##   Use `force = TRUE` to force installation
```

```
library(preText)
```

```
## Warning in .recacheSubclasses(def@className, def, env):
## "replValueSp"类别的子类别"ndiMatrix"没有定义；因此没有更新
```

```
## preText: Diagnostics to Assess the Effects of Text Preprocessing Decisions
## Version 0.7.2 created on 2021-07-25.
## copyright (c) 2021, Matthew J. Denny, Georgetown University
##                     Arthur Spirling, NYU
## Type vignette('getting_started_with_preText') to get started.
## Development website: https://github.com/matthewjdenny/preText
```

Question 1: War and Peace

1. Choose another book or set of books from Project Gutenberg We choose War and Peace, a novel composed of 15 books + epilogue.

```
tolstoy <- gutenberg_download(c(2600),
                          meta_fields = "author")
```

```
## Determining mirror for Project Gutenberg from https://www.gutenberg.org/robot/harvest
```

```
## Using mirror http://aleph.gutenberg.org
```

In order to create a dfm, we take out the lengthy table of contents, and divide it up by book number, before passing the resulting data to a dataframe.

```
# first we need to take out lines 1-796, which are table of contents, and the irrelevant info.
tolstoy_edit<- tolstoy %>%
  select(-gutenberg_id,-author)%>%
  mutate(row.id=1:nrow(tolstoy))%>%
  filter(row.id>797)

Book_Headings <- tolstoy_edit[grep("BOOK", tolstoy_edit$text), ]
Book_Headings[16:30,]
```

```
## # A tibble: 15 × 2
##    text  row.id
##    <chr>  <int>
##  1 <NA>      NA
##  2 <NA>      NA
##  3 <NA>      NA
##  4 <NA>      NA
##  5 <NA>      NA
##  6 <NA>      NA
##  7 <NA>      NA
##  8 <NA>      NA
##  9 <NA>      NA
## 10 <NA>      NA
## 11 <NA>      NA
## 12 <NA>      NA
## 13 <NA>      NA
## 14 <NA>      NA
## 15 <NA>      NA
```

```
book.name<-c(Book_Headings$text,"EPILOGUE")#  create book name vector

#We make a vector with the rirght length of each book/chapter id.
row_chapt<-c(Book_Headings$row.id,nrow(tolstoy_edit)) #the row.ids
prior<-c(0,row_chapt[1:15]) #number of rows already accounted for
repetitions<-row_chapt-prior #how many times to repeat

book.id<-rep(book.name,times=repetitions) #create vector for df
nrow(tolstoy_edit)==length(book.id)#check it worked lengthwise, true
```

```
## [1] TRUE
```

```
tolstoy_edit2<-cbind(tolstoy_edit,book.id) #we bind this to the text dataframe.



tolstoy_words <- tolstoy_edit2 %>% # we now run normal preprocessing to create a dtm.
  unnest_tokens(word, text) %>%
  filter(!is.na(word)) %>%
  count(book.id, word, sort = TRUE) %>%
  ungroup() %>%
  anti_join(stop_words)
```

```
## Joining with `by = join_by(word)`
```

```
tolstoy_dtm <-tolstoy_words %>% #create dtm
  cast_dtm(book.id, word, n)

tm::inspect(tolstoy_dtm) #look at dtm
```

```
## <<DocumentTermMatrix (documents: 16, terms: 17476)>>
## Non-/sparse entries: 63960/215656
## Sparsity           : 77%
## Maximal term length: 18
## Weighting          : term frequency (tf)
## Sample             :
##                       Terms
## Docs                   andrew eyes french moscow natásha pierre prince
##   BOOK ELEVEN: 1812       146   69    129    140       5    200    238
##   BOOK FOUR: 1806          87   74     62     21      39      7    214
##   BOOK NINE: 1812          50   59     29     44     234    115     85
##   BOOK SEVEN: 1810 - 11   122   36     11     19     169     82    149
##   BOOK SIX: 1808 - 10     184   52     39      6       3    125    232
##   BOOK TEN: 1812           75   45     55     42      39     67    138
##   BOOK THREE: 1805        133   76    112      6       0     82    277
##   BOOK TWELVE: 1812        29   68     87    174     115    258     62
##   BOOK TWO: 1805          129   92     26     30      54    216    276
##   EPILOGUE                 18   65     51     51     159    230     31
##                       Terms
## Docs                   princess rostóv time
##   BOOK ELEVEN: 1812         127     42  102
##   BOOK FOUR: 1806            98    195   57
##   BOOK NINE: 1812            45      9   60
##   BOOK SEVEN: 1810 - 11      27     26   62
##   BOOK SIX: 1808 - 10        40    140   53
##   BOOK TEN: 1812             34     68   54
##   BOOK THREE: 1805           24    113   55
##   BOOK TWELVE: 1812          19     18   84
##   BOOK TWO: 1805            159      8   63
##   EPILOGUE                  106      5  139
```

Question 2: Own topic model

2. Run your own topic model on these books, changing the k of topics, and evaluating accuracy.

Some of the plausible topic distinctions in War and Peace would be: distinguishing by books (16), or distinguishing by family subplots (5), or between narrative and philosophy (2). We will run them seperately, staring with the lowest k. Vis

```
tolstoy_lda_2 <- LDA(tolstoy_dtm, k = 2, control = list(seed = 1209)) #run model

tolstoy_lda_2 #view model
```

```
## A LDA_VEM topic model with 2 topics.
```

```
tolstoy_topics_2 <- tidy(tolstoy_lda_2, matrix = "beta") #extract the per-topic-per-word probab
ilities

head(tolstoy_topics_2, n = 10) #look at topics.
```

```
## # A tibble: 10 × 3
##    topic term        beta
##    <int> <chr>      <dbl>
##  1     1 prince  0.0146
##  2     2 prince  0.00362
##  3     1 pierre  0.0100
##  4     2 pierre  0.00759
##  5     1 natásha 0.00885
##  6     2 natásha 0.00173
##  7     1 rostóv  0.00491
##  8     2 rostóv  0.00208
##  9     1 andrew  0.000326
## 10     2 andrew  0.0104
```

We can also run it with family subplots.

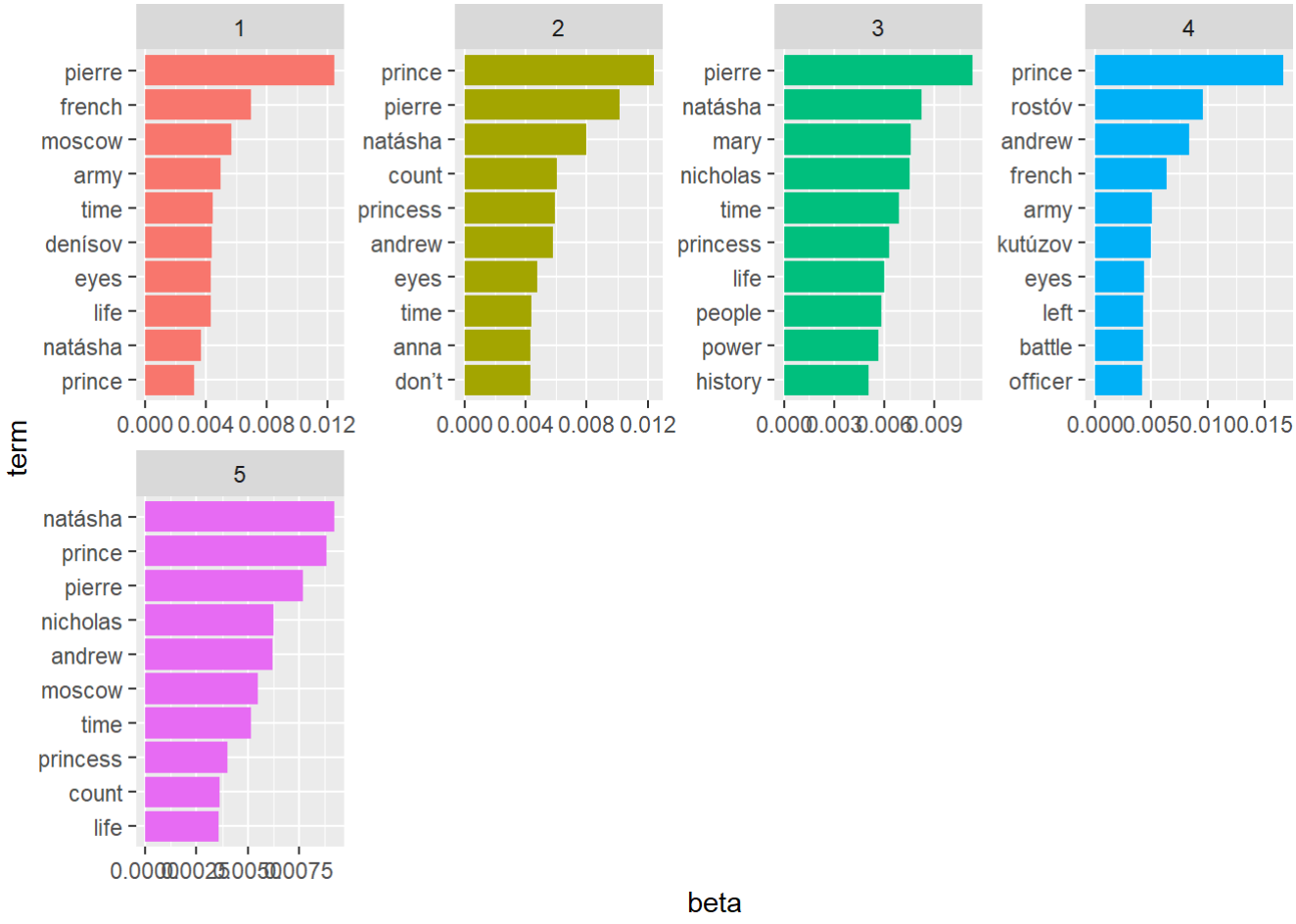
```
tolstoy_lda_5 <- LDA(tolstoy_dtm, k = 5, control = list(seed = 1920)) #run model

tolstoy_lda_5 #view model
```

```
## A LDA_VEM topic model with 5 topics.
```

```
tolstoy_topics_5<- tidy(tolstoy_lda_5, matrix = "beta")

head(tolstoy_topics_5, n = 10) #look at topics.
```

```
## # A tibble: 10 × 3
##    topic term      beta
##    <int> <chr>    <dbl>
##  1     1 prince 0.00320
##  2     2 prince 0.0124
##  3     3 prince 0.00161
##  4     4 prince 0.0167
##  5     5 prince 0.00883
##  6     1 pierre 0.0125
##  7     2 pierre 0.0102
##  8     3 pierre 0.0113
##  9     4 pierre 0.00334
## 10     5 pierre 0.00771
```

```
tolstoy_top_terms_5 <- tolstoy_topics_5 %>% #arrange topics
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)

tolstoy_top_terms_5 %>% # plot topics
  mutate(term = reorder_within(term, beta, topic)) %>%
  ggplot(aes(beta, term, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free", ncol = 4) +
  scale_y_reordered()
```



We are also looking at 10.

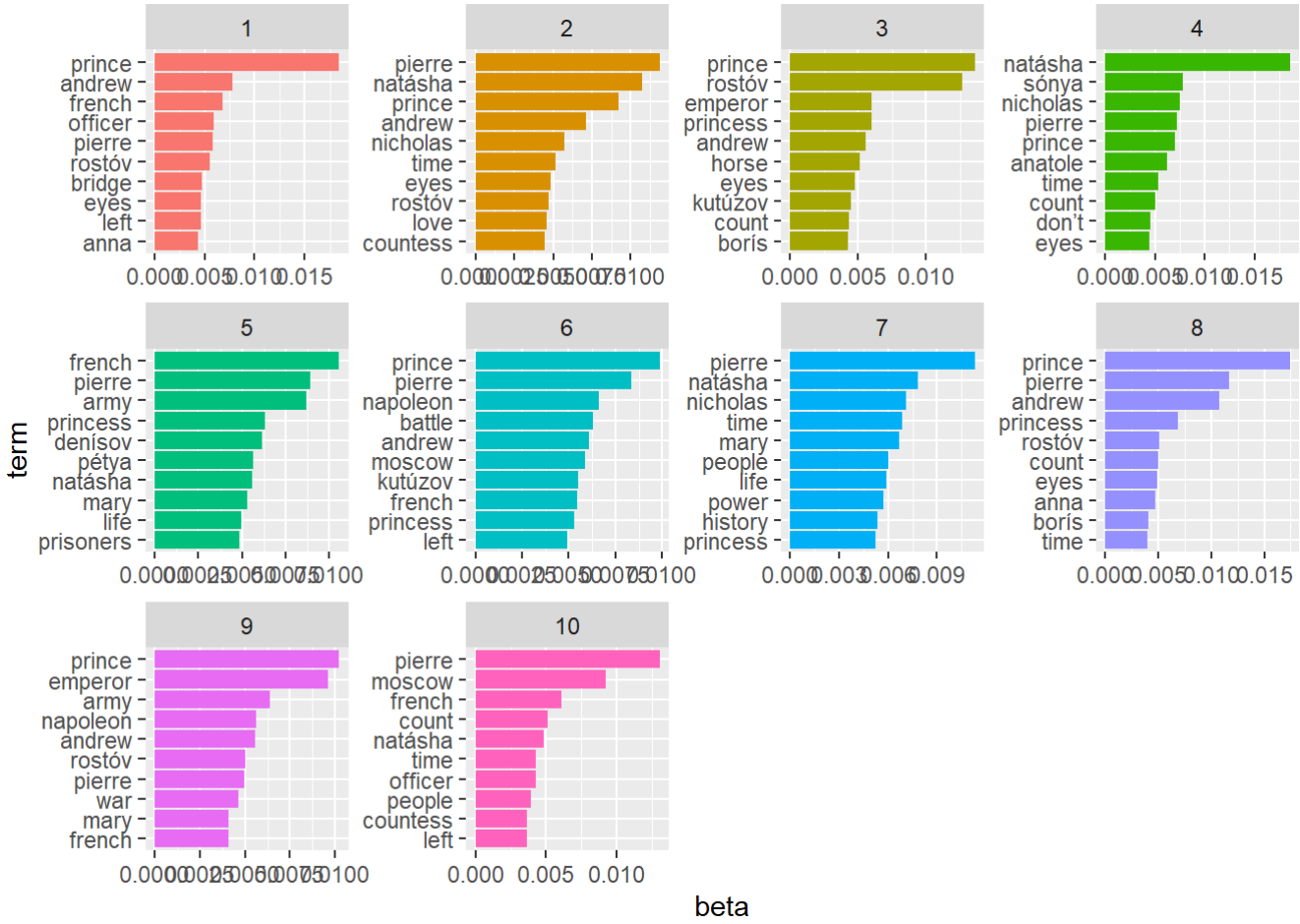
```
tolstoy_lda_10 <- LDA(tolstoy_dtm, k = 10, control = list(seed = 1230)) #run model

tolstoy_lda_10 #view model
```

```
## A LDA_VEM topic model with 10 topics.
```

```
tolstoy_topics_10<- tidy(tolstoy_lda_10, matrix = "beta")

head(tolstoy_topics_10, n = 10) #look at topics.
```

```
## # A tibble: 10 × 3
##    topic term      beta
##    <int> <chr>    <dbl>
##  1     1 prince 0.0185
##  2     2 prince 0.00922
##  3     3 prince 0.0136
##  4     4 prince 0.00701
##  5     5 prince 0.00244
##  6     6 prince 0.00988
##  7     7 prince 0.00152
##  8     8 prince 0.0174
##  9     9 prince 0.0102
## 10    10 prince 0.00271
```

```
tolstoy_top_terms_10 <- tolstoy_topics_10 %>% #arrange topics
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)

tolstoy_top_terms_10 %>% # plot topics
  mutate(term = reorder_within(term, beta, topic)) %>%
  ggplot(aes(beta, term, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free", ncol = 4) +
  scale_y_reordered()
```



And finally, 16.

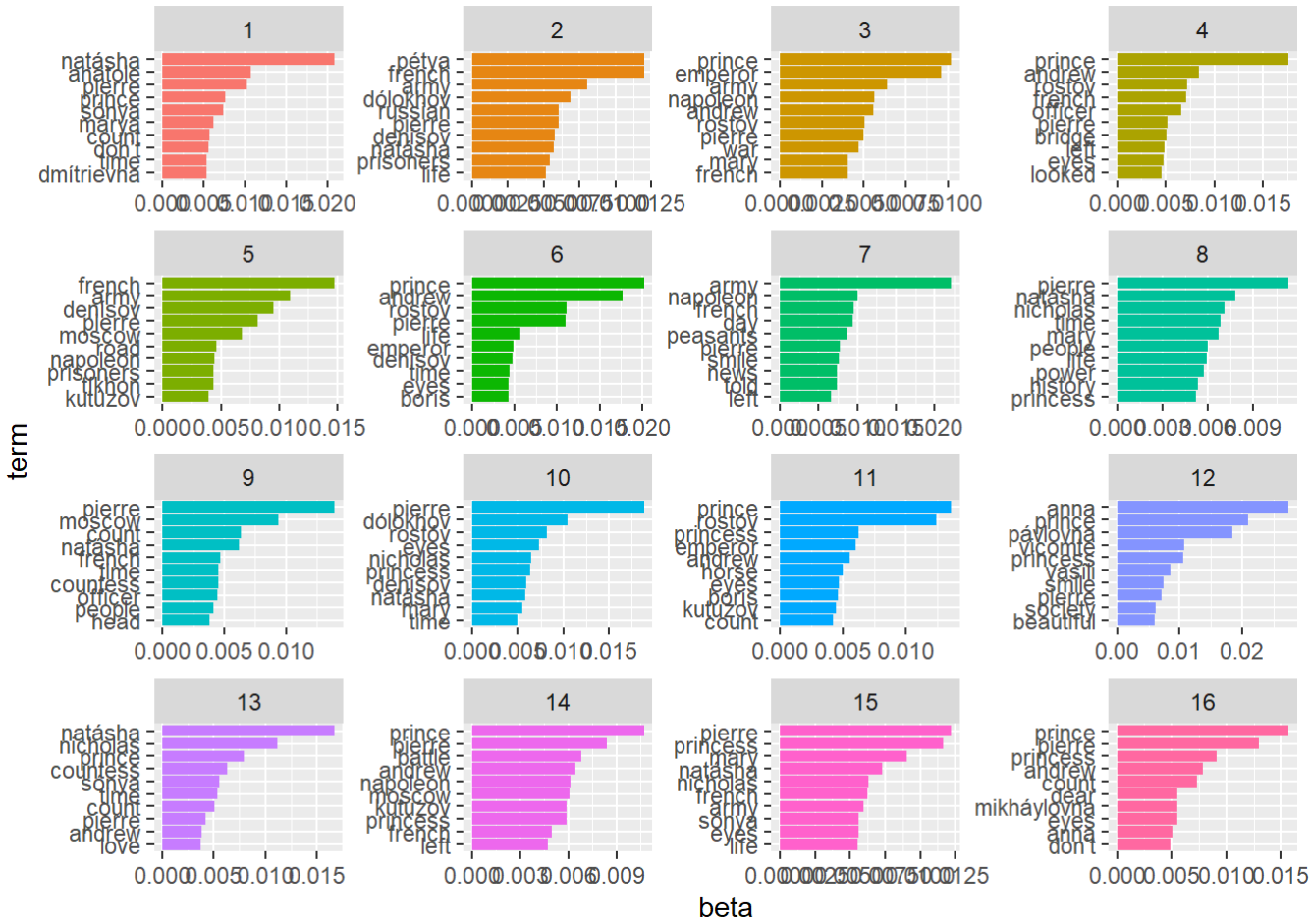
```
tolstoy_lda_16 <- LDA(tolstoy_dtm, k = 16, control = list(seed = 1203)) #run model

tolstoy_lda_16 #view model
```

```
## A LDA_VEM topic model with 16 topics.
```

```
tolstoy_topics_16<- tidy(tolstoy_lda_16, matrix = "beta")

head(tolstoy_topics_16, n = 10) #look at topics.
```

```
## # A tibble: 10 × 3
##    topic term       beta
##    <int> <chr>     <dbl>
##  1     1 prince 0.00755
##  2     2 prince 0.000527
##  3     3 prince 0.0102
##  4     4 prince 0.0177
##  5     5 prince 0.000338
##  6     6 prince 0.0202
##  7     7 prince 0.00284
##  8     8 prince 0.00152
##  9     9 prince 0.00334
## 10    10 prince 0.00450
```

```
tolstoy_top_terms_16 <- tolstoy_topics_16 %>% #arrange topics
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)

tolstoy_top_terms_16 %>% # plot topics
  mutate(term = reorder_within(term, beta, topic)) %>%
  ggplot(aes(beta, term, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free", ncol = 4) +
  scale_y_reordered()
```



These all provide some form of topic distinction, with some topics easy to distinguish: such as army vs love vs philosophical topics, or the key personages in each of them. However, in order to formally evaluate them, we need to look closer.

Split into chapter documents

In the below, we first separate the volumes into chapters, then we repeat the same procedure as above. The only difference now is that instead of one document per book, we have one document per chapter, increasing sparsity

```
# Make a check if there are any NAs in "tolstoy_edit2"
any(is.na(tolstoy_edit2))
```

```
## [1] FALSE
```

```
# Divide into documents, each representing one chapter
tolstoy_chapter <- tolstoy_edit2 %>%
  group_by(book.id) %>%
  mutate(chapter = cumsum(str_detect(text, regex("^chapter ", ignore_case = TRUE)))) %>%
  ungroup() %>%
  filter(chapter > 0) %>%
  unite(document, book.id, chapter)

# Split into words
tolstoy_chapter_word <- tolstoy_chapter %>%
  unnest_tokens(word, text)

# Find document-word counts
tolstoy_word_counts <- tolstoy_chapter_word %>%
  anti_join(stop_words) %>%
  count(document, word, sort = TRUE) %>%
  ungroup()
```

```
## Joining with `by = join_by(word)`
```

```
tolstoy_word_counts
```

```
## # A tibble: 129,946 × 3
##    document               word         n
##    <chr>                  <chr>    <int>
##  1 BOOK TWELVE: 1812_22   pierre      62
##  2 BOOK THREE: 1805_1     rostóv      53
##  3 BOOK NINE: 1812_10     sónya       51
##  4 BOOK ELEVEN: 1812_27   pierre      49
##  5 BOOK NINE: 1812_10     natásha     49
##  6 BOOK THREE: 1805_20    pierre      45
##  7 BOOK FOUR: 1806_1      prince      44
##  8 BOOK TEN: 1812_21      alpátych    41
##  9 BOOK TWO: 1805_23      andrew      41
## 10 BOOK EIGHT: 1811 - 12_2 uncle      39
## # ℹ 129,936 more rows
```

```
# Cast into DTM format for LDA analysis

tolstoy_chapters_dtm <- tolstoy_word_counts %>%
  cast_dtm(document, word, n)

tm::inspect(tolstoy_chapters_dtm)
```

```
## <<DocumentTermMatrix (documents: 365, terms: 17241)>>
## Non-/sparse entries: 129946/6163019
## Sparsity           : 98%
## Maximal term length: 18
## Weighting          : term frequency (tf)
## Sample             :
##                        Terms
## Docs                    andrew eyes french moscow natásha pierre prince
##   BOOK EIGHT: 1811 - 12_2      1    2      2      0      36      0      1
##   BOOK EIGHT: 1811 - 12_5      1    5      0      0      27      0      1
##   BOOK ELEVEN: 1812_4          5   11      4      3       0      0     17
##   BOOK FOUR: 1806_4           11    5      1      0       0      1     12
##   BOOK THREE: 1805_1           0   14      0      0       0      0      0
##   BOOK THREE: 1805_20          0    7      0      1       0     45     29
##   BOOK THREE: 1805_5           0    3      7      0       0      0      3
##   BOOK TWELVE: 1812_18         0    5      4      8       0      0      0
##   BOOK TWELVE: 1812_22         0    9     12     10       2     62      1
##   BOOK TWO: 1805_20            1    8      2      5       0      4     19
##                        Terms
## Docs                    princess rostóv time
##   BOOK EIGHT: 1811 - 12_2        0      3    7
##   BOOK EIGHT: 1811 - 12_5        0      0    3
##   BOOK ELEVEN: 1812_4           37      0    5
##   BOOK FOUR: 1806_4              0     32    3
##   BOOK THREE: 1805_1             0     53    1
##   BOOK THREE: 1805_20            9      0    7
##   BOOK THREE: 1805_5             0     24    6
##   BOOK TWELVE: 1812_18           0      0    7
##   BOOK TWELVE: 1812_22           0      0    3
##   BOOK TWO: 1805_20             26      1    3
```

We then re-estimate the topic model with this new DocumentTermMatrix object, specifying k equal to 16. This will enable us to evaluate whether a topic model is able to generatively assign to volume with accuracy.

```
tolstoy_chapters_lda <- LDA(tolstoy_chapters_dtm, k = 16, control = list(seed = 1249))
```

After this, it is worth looking at another output of the latent dirichlet allocation procedure. The γ probability represents the per-document-per-topic probability or, in other words, the probability that a given document (here: chapter) belongs to a particular topic (and here, we are assuming these topics represent volumes).

The gamma values are therefore the estimated proportion of words within a given chapter allocated to a given volume.

```
tolstoy_chapters_gamma <- tidy(tolstoy_chapters_lda, matrix = "gamma")
tolstoy_chapters_gamma
```

```
## # A tibble: 5,840 × 3
##    document                topic     gamma
##    <chr>                   <int>     <dbl>
##  1 BOOK TWELVE: 1812_22        1 0.0000118
##  2 BOOK THREE: 1805_1          1 0.0000140
##  3 BOOK NINE: 1812_10          1 0.0000244
##  4 BOOK ELEVEN: 1812_27        1 1.00
##  5 BOOK THREE: 1805_20         1 0.0000144
##  6 BOOK FOUR: 1806_1           1 0.0000154
##  7 BOOK TEN: 1812_21           1 0.590
##  8 BOOK TWO: 1805_23           1 0.0000147
##  9 BOOK EIGHT: 1811 - 12_2     1 0.0000133
## 10 BOOK FIVE: 1806 - 07_16     1 0.0000178
## # ℹ 5,830 more rows
```

Examine consensus

Now that we have these topic probabilities, we can see how well our unsupervised learning did at distinguishing the books generatively just from the words contained in each chapter.

```
# First separate the document name into title and chapter

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

tolstoy_chapter_classifications <- tolstoy_chapters_gamma %>%
  group_by(title, chapter) %>%
  top_n(1, gamma) %>%
  ungroup()

tolstoy_book_topics <- tolstoy_chapter_classifications %>%
  count(title, topic) %>%
  group_by(title) %>%
  top_n(1, n) %>%
  ungroup() %>%
  transmute(consensus = title, topic)

tolstoy_chapter_classifications %>%
  inner_join(tolstoy_book_topics, by = "topic") %>%
  filter(title != consensus)
```

```
## Warning in inner_join(., tolstoy_book_topics, by = "topic"): Detected an unexpected many-to-
many relationship between `x` and `y`.
## ℹ Row 22 of `x` matches multiple rows in `y`.
## ℹ Row 3 of `y` matches multiple rows in `x`.
## ℹ If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```

```
## # A tibble: 357 × 5
##    title                   chapter topic gamma consensus
##    <chr>                     <int> <int> <dbl> <chr>
##  1 BOOK TEN: 1812               21     1 0.590 BOOK ELEVEN: 1812
##  2 BOOK NINE: 1812              11     1 0.875 BOOK ELEVEN: 1812
##  3 BOOK FOURTEEN: 1812          15     1 0.732 BOOK ELEVEN: 1812
##  4 BOOK FOURTEEN: 1812          16     1 1.00  BOOK ELEVEN: 1812
##  5 BOOK FOURTEEN: 1812          14     1 0.946 BOOK ELEVEN: 1812
##  6 BOOK THREE: 1805             15     1 0.326 BOOK ELEVEN: 1812
##  7 BOOK FOUR: 1806              13     1 0.322 BOOK ELEVEN: 1812
##  8 BOOK NINE: 1812              12     1 0.757 BOOK ELEVEN: 1812
##  9 BOOK FIFTEEN: 1812 - 13       1     1 0.569 BOOK ELEVEN: 1812
## 10 BOOK FOUR: 1806              11     1 1.00  BOOK ELEVEN: 1812
## # ℹ 347 more rows
```

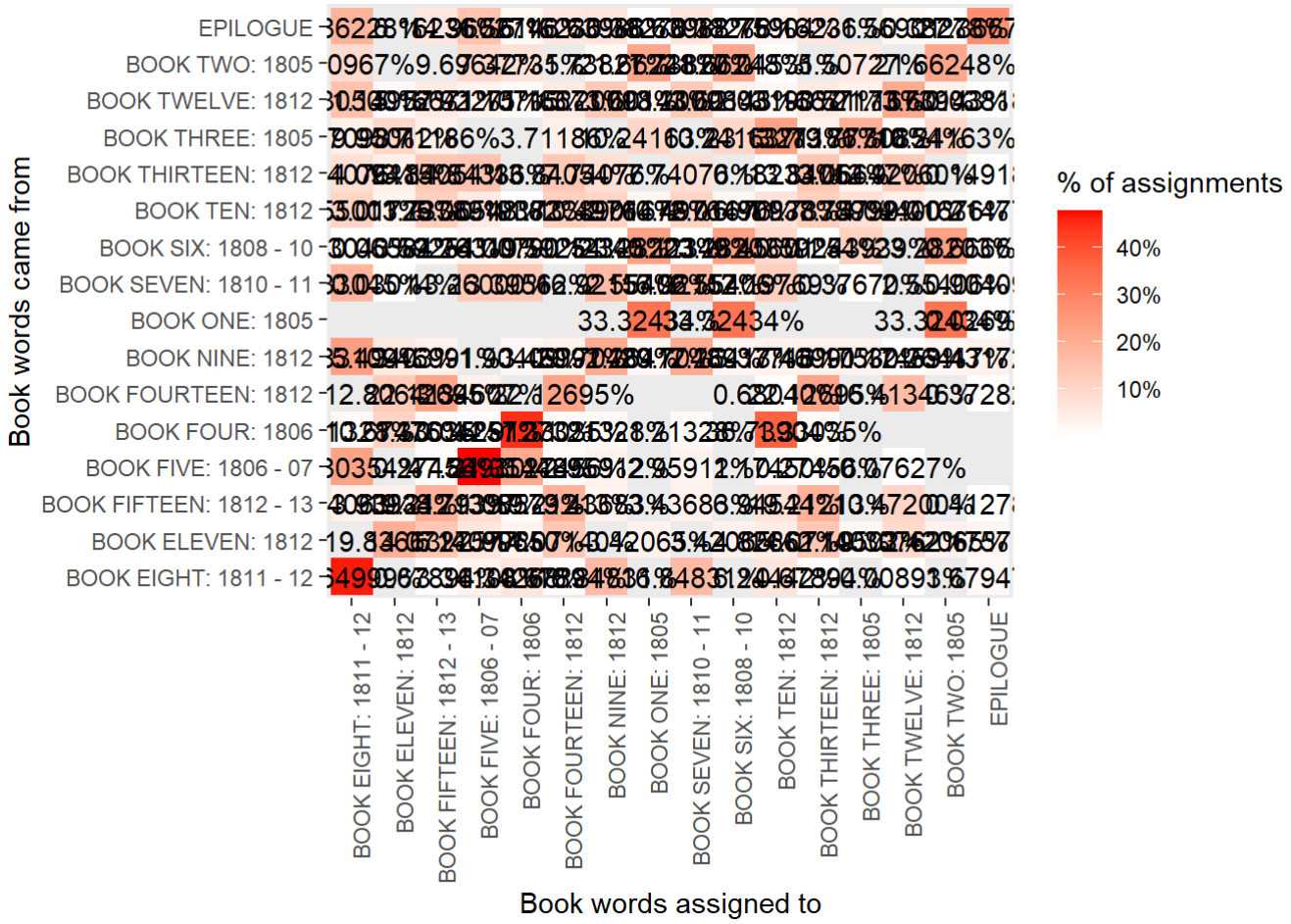
```
# Look document-word pairs were to see which words in each documents were assigned
# to a given topic

assignments <- augment(tolstoy_chapters_lda, data = tolstoy_chapters_dtm)
assignments
```

```
## # A tibble: 129,946 × 4
##    document                term   count .topic
##    <chr>                   <chr>  <dbl>  <dbl>
##  1 BOOK TWELVE: 1812_22    pierre    62     12
##  2 BOOK NINE: 1812_10      pierre     1      8
##  3 BOOK ELEVEN: 1812_27    pierre    49      1
##  4 BOOK THREE: 1805_20     pierre    45      2
##  5 BOOK FIVE: 1806 - 07_16 pierre    39      4
##  6 BOOK TEN: 1812_2        pierre     3      4
##  7 BOOK THREE: 1805_19     pierre    37      2
##  8 BOOK TWELVE: 1812_26    pierre    37     14
##  9 BOOK ELEVEN: 1812_21    pierre    18     12
## 10 BOOK SIX: 1808 - 10_7   pierre    33      2
## # ℹ 129,936 more rows
```

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

```
## Warning in inner_join(., tolstoy_book_topics, by = c(.topic = "topic")): Detected an unexpec
ted many-to-many relationship between `x` and `y`.
## ℹ Row 2 of `x` matches multiple rows in `y`.
## ℹ Row 5 of `y` matches multiple rows in `x`.
## ℹ If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```

```
assignments %>%
  count(title, consensus, wt = count) %>%
  group_by(title) %>%
  mutate(percent = n / sum(n)) %>%
  ggplot(aes(consensus, title, fill = percent)) +
  geom_tile() +
  scale_fill_gradient2(high = "red", label = percent_format()) +
  geom_text(aes(x = consensus, y = title, label = scales::percent(percent))) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1),
        panel.grid = element_blank()) +
  labs(x = "Book words assigned to",
       y = "Book words came from",
       fill = "% of assignments")
```




```
?geom_text
```

```
## 打开httpd帮助服务器… 好了
```

Not bad! We see that the diagonal, with few exceptions, shows the highest correlation, implying that the model did a good job.

Question 3: Validation using `preText`

In this section, we'll be using the `preText` package mentioned in @denny_text_2018 to see the impact of different pre-processing choices on our text.

First we need to reformat our text into a `quanteda` corpus object.

```
# load in corpus of the text data.
corp_tolstoy <- corpus(tolstoy_edit2, text_field = "text")
# use first 10 documents for example
documents_tolstoy <- corp_tolstoy[sample(1:30000,1000)]
# take a look at the document names
print(names(documents_tolstoy[1:10]))
```

```
##  [1] "text27030" "text9855"  "text14663" "text29665" "text1974"  "text15400"
##  [7] "text201"   "text19789" "text16368" "text14345"
```

And now we are ready to preprocess in different ways. Here, we are including n-grams so we are preprocessing the text in 128 different ways. This takes about ten minutes to run on a machine with 8GB RAM.

```
preprocessed_documents_tolstoy <- factorial_preprocessing(
    documents_tolstoy,
    use_ngrams = TRUE,
    infrequent_term_threshold = 0.2,
    verbose = FALSE)
```

```
## Preprocessing 1000 documents 128 different ways...
```

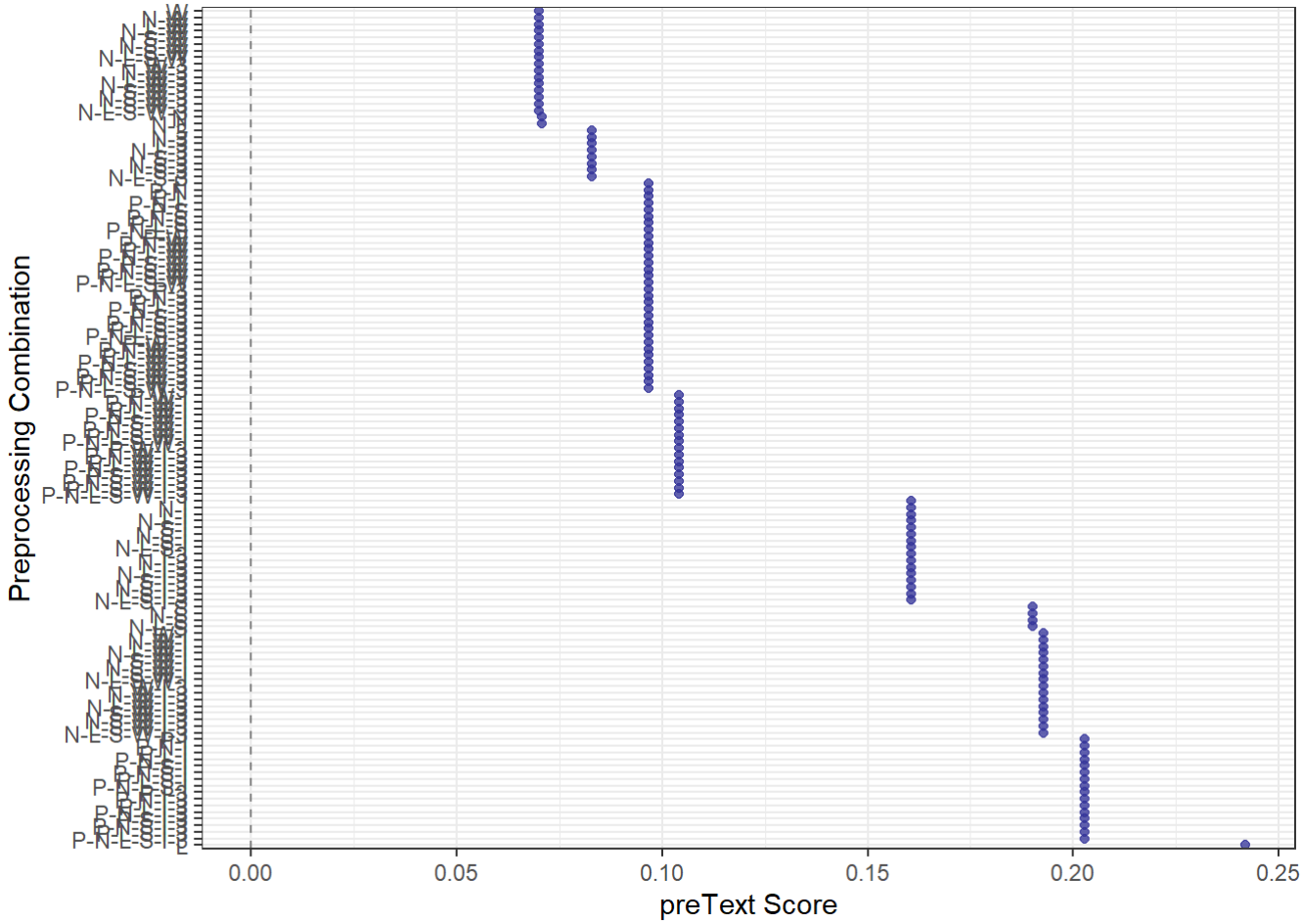
We can then get the results of our pre-processing, comparing the distance between documents that have been processed in different ways.

```
preText_results1 <- preText(
    preprocessed_documents_tolstoy,
    dataset_name = " new text",
    distance_method = "cosine",
    num_comparisons = 20,
    verbose = FALSE)
```

```
## Generating document distances...
## Generating preText Scores...
## Generating regression results..
## The R^2 for this model is: 0.5899617
## Regression results (negative coefficients imply less risk):
##                  Variable Coefficient    SE
## 1               Intercept       0.119 0.009
## 2      Remove Punctuation      -0.011 0.006
## 3          Remove Numbers      -0.003 0.006
## 4               Lowercase       0.002 0.006
## 5                Stemming       0.004 0.006
## 6        Remove Stopwords      -0.029 0.006
## 7 Remove Infrequent Terms       0.069 0.006
## 8              Use NGrams      -0.009 0.006
## Complete in: 557.66 seconds...
```

And we can plot these accordingly.

```
preText_score_plot(preText_results1)
```


Preprocessing Combination
0.00
0.05
0.10
0.15
0.20
0.25
preText Score