Intro to Text Mining: Tidy Text

Filippo Chiarello, Ph.D.

Tidy text

- tidytext is an R package for analysing text with the tidyverse philosophy
- treating **text as data frames** of individual words allows us to manipulate, summarize, and visualize the characteristics of text easily and integrate natural language processing into effective workflows of the tidyverse

One-token-per-row

- we define the tidy text format as being a table with **one-token-per-row**
- a **token** is a meaningful unit of text, such as a word, a sentence, or paragraph, that we are interested in using for analysis
- **tokenization** is the process of splitting text into tokens

unnest_tokens

- `unnest_tokens` is the main verb of tidytext
- it splits text into tokens and outputs a one-token-per-row table
- takes 3 main parameters:
  1. tbl: a data frame containing the text to tokenize
  2. output: the output column to be created
  3. input: the input column that gets split
- punctuation is stripped
- tokens are converted to lowercase
- other columns, such as the line number each word came from, are retained

unnest_tokens

```
# the tidy tools
library(tidyverse)

## Warning: package 'tidyr' was built under R version 4.0.5

## Warning: package 'readr' was built under R version 4.0.5

## Warning: package 'dplyr' was built under R version 4.0.5

# the tidy tools for text
library(tidytext)

# Emily Dickinson wrote some lovely text in her time
text <- c("Because I could not stop for Death -",
          "He kindly stopped for me -",
          "The Carriage held but just Ourselves -",
          "and Immortality")


# a data frame with one row per sentence
text_df <- data_frame(line = 1:4, text = text)

## Warning: `data_frame()` was deprecated in tibble 1.1.0.
```

Let's print the table

```
text_df
```

```
## # A tibble: 4 × 2
##    line text
##   <int> <chr>
## 1     1 Because I could not stop for Death –
## 2     2 He kindly stopped for me –
## 3     3 The Carriage held but just Ourselves –
## 4     4 and Immortality
```

tokenization: one row per word

```
unnest_tokens(tbl = text_df, output = word, input = text)
```

```
## # A tibble: 20 × 2
##    line word
##   <int> <chr>
## 1     1 because
## 2     1 i
## 3     1 could
## 4     1 not
## 5     1 stop
## 6     1 for
## # … with 14 more rows
```

Stop words and stems

- **stop words** are words which are filtered out before processing of natural language data (text), such as such as *the*, *is*, *at*, *which*, *for*, *an* and *on*
- **stemming** is the process of reducing inflected words to their word stem, base or root form. For instance, a stemming algorithm might reduce the words *fishing*, *fished*, and *fisher* to the stem *fish*
- a popular stemmer is Porter's stemming algorithm

Jane Austen's novels

- Let's use the text of Jane Austen's 6 completed, published novels from the janeaustenr package, and transform them into a tidy format
- The janeaustenr package provides these texts in a **one-row-per-line** format, where a line is this context is analogous to a literal printed line in a physical book

Jane Austen's novels

```
library(janeaustenr)
library(stringr)

# one sentence per row
austen_books()
```

```
## # A tibble: 73,422 × 2
##   text                    book
## * <chr>                   <fct>
## 1 "SENSE AND SENSIBILITY" Sense & Sensibility
## 2 ""                      Sense & Sensibility
## 3 "by Jane Austen"        Sense & Sensibility
## 4 ""                      Sense & Sensibility
## 5 "(1811)"                Sense & Sensibility
## 6 ""                      Sense & Sensibility
## # … with 73,416 more rows
```

Add line and chapter numbers relative to books

```
original_books <- austen_books() %>%
  group_by(book) %>%
  mutate(linenumber = row_number(),
         chapter = cumsum(
           str_detect(text, regex("^chapter [\\divxlc]", ignore_case = TRUE)))) %>%
  ungroup()

original_books
```

```
## # A tibble: 73,422 × 4
##   text                    book                linenumber chapter
##   <chr>                   <fct>                    <int>   <int>
## 1 "SENSE AND SENSIBILITY" Sense & Sensibility          1       0
## 2 ""                      Sense & Sensibility          2       0
## 3 "by Jane Austen"        Sense & Sensibility          3       0
## 4 ""                      Sense & Sensibility          4       0
## 5 "(1811)"                Sense & Sensibility          5       0
## 6 ""                      Sense & Sensibility          6       0
## # … with 73,416 more rows
```

tokenize: one work per row

```
tidy_books <- original_books %>%
  unnest_tokens(word, text)

tidy_books
```

```
## # A tibble: 725,055 × 4
##   book                linenumber chapter word
##   <fct>                    <int>   <int> <chr>
## 1 Sense & Sensibility          1       0 sense
## 2 Sense & Sensibility          1       0 and
## 3 Sense & Sensibility          1       0 sensibility
## 4 Sense & Sensibility          3       0 by
## 5 Sense & Sensibility          3       0 jane
## 6 Sense & Sensibility          3       0 austen
## # … with 725,049 more rows
```

remove stop words

```
stop_words
```

```
## # A tibble: 1,149 × 2
##   word      lexicon
##   <chr>     <chr>
## 1 a         SMART
## 2 a's       SMART
## 3 able      SMART
## 4 about     SMART
## 5 above     SMART
## 6 according SMART
## # … with 1,143 more rows
```

```
tidy_books <- tidy_books %>%
  anti_join(stop_words)
```

word frequency

```
tidy_books %>%
  count(word, sort = TRUE)
```

```
## # A tibble: 13,914 × 2
##   word      n
##   <chr> <int>
## 1 miss   1855
## 2 time   1337
## 3 fanny   862
## 4 dear    822
## 5 lady    817
## 6 sir     806
## # … with 13,908 more rows
```

plot word frequency

```
tidy_books %>%
  count(word, sort = TRUE) %>%
  filter(n > 600) %>%
  # reorder levels of factor word wrt n
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n)) +
  geom_col() +
  xlab(NULL) +
  coord_flip() +
  theme_bw()
```

wordcloud

```
library(wordcloud)

tidy_books %>%
  count(word) %>%
  # evaluate an R expression in an environment constructed from data
  with(wordcloud(word, n, max.words = 100))
```

Porter's word stemming

```
library(SnowballC)
tidy_books <- tidy_books %>%
  mutate(word = wordStem(word)) # stemming

tidy_books
```

```
## # A tibble: 217,609 × 4
##   book                linenumber chapter word
##   <fct>                    <int>   <int> <chr>
## 1 Sense & Sensibility          1       0 sens
## 2 Sense & Sensibility          1       0 sensibl
## 3 Sense & Sensibility          3       0 jane
## 4 Sense & Sensibility          3       0 austen
## 5 Sense & Sensibility          5       0 1811
## 6 Sense & Sensibility         10       1 chapter
## # … with 217,603 more rows
```

plot word frequency

```
tidy_books %>%
  count(word, sort = TRUE) %>%
  filter(n > 600) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n)) +
  geom_col() +
  xlab(NULL) +
  coord_flip() +
  theme_bw()
```

tokenize by pattern (with regular expression)

```
austen_chapters <- austen_books() %>%
  group_by(book) %>%
  unnest_tokens(chapter, text,
                token = "regex",
                pattern = "(Chapter|CHAPTER) [\\dIVXLC]{1,8}") %>%
  ungroup()
```

how many chapters in each book?

```
austen_chapters %>%
  group_by(book) %>%
  summarise(chapters = n()) %>%
  arrange(-chapters)
```

```
## # A tibble: 6 × 2
##   book                chapters
##   <fct>                  <int>
## 1 Pride & Prejudice         62
## 2 Emma                      56
## 3 Sense & Sensibility       51
## 4 Mansfield Park            49
## 5 Northanger Abbey          32
## 6 Persuasion                25
```

Project Gutenberg

- now that we've used the `janeaustenr` package to explore tidying text, let's introduce the gutenbergr package
- the `gutenbergr` package provides access to the public domain works from the Project Gutenberg collection
- we will mostly use the function `gutenberg_download()` that downloads one or more works from Project Gutenberg by ID

Project Gutenberg - H.G. Wells

Let's look at some science fiction and fantasy novels by H.G. Wells, who lived in the late 19th and early 20th centuries. Let's get:

> - *The Time Machine*
> - *The War of the Worlds*
> - *The Invisible Man*
> - *The Island of Doctor Moreau*

Download the RDS file.

```
library(gutenbergr)

# run once and save the result as RDS
#hgwells <- gutenberg_download(c(35, 36, 5230, 159))
#write_rds(hgwells, "hgwells.rds")

# read from RDS
hgwells = read_rds("hgwells.rds")
hgwells
```

```
## # A tibble: 20,020 × 2
##   gutenberg_id text
##          <int> <chr>
## 1           35 "The Time Machine"
## 2           35 ""
## 3           35 "An Invention"
## 4           35 ""
## 5           35 "by H. G. Wells"
## 6           35 ""
## # … with 20,014 more rows
```

```
tidy_hgwells <- hgwells %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words)

tidy_hgwells %>%
  count(word, sort = TRUE)
```

```
## # A tibble: 11,811 × 2
##   word       n
##   <chr>  <int>
## 1 time     461
## 2 people   302
## 3 door     260
## 4 heard    249
## 5 black    232
## 6 stood    229
## # … with 11,805 more rows
```

Project Gutenberg - Bronte sisters

Now let's get some well-known works of the Bronte sisters, whose lives overlapped with Jane Austen's somewhat but who wrote in a rather different style. Let's get:

> - *Jane Eyre*
> - *Wuthering Heights*
> - *The Tenant of Wildfell Hall*
> - *Villette*
> - *Agnes Grey*

Download the RDS file.

```
# run once and save the result as RDS
# bronte <- gutenberg_download(c(1260, 768, 969, 9182, 767))
# write_rds(bronte, "bronte.rds")

# read from RDS
bronte = read_rds("bronte.rds")
bronte
```

```
## # A tibble: 80,117 × 2
##   gutenberg_id text
##          <int> <chr>
## 1          767 "Agnes Grey"
## 2          767 "A NOVEL,"
## 3          767 ""
## 4          767 "by ACTON BELL."
## 5          767 ""
## 6          767 "LONDON:"
## # … with 80,111 more rows
```

```
tidy_bronte <- bronte %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words)

tidy_bronte %>%
  count(word, sort = TRUE)
```

```
## # A tibble: 23,297 × 2
##   word      n
##   <chr> <int>
## 1 time   1065
## 2 miss    854
## 3 day     825
## 4 hand    767
## 5 eyes    714
## 6 don’t   666
## # … with 23,291 more rows
```

Interesting that "time", "eyes", and "hand" are in the top 10 for both H.G. Wells and the Bronte sisters.

Compare words used by Jane Austen and the Bronte sisters

```
frequency <-
  bind_rows(mutate(tidy_bronte, author = "Bronte Sisters"),
                    mutate(tidy_hgwells, author = "H.G. Wells"),
                    mutate(tidy_books, author = "Jane Austen")) %>%
  mutate(word = str_extract(word, "[a-z']+")) %>%
  count(author, word) %>%
  group_by(author) %>%
  mutate(proportion = n / sum(n)) %>%
  select(-n) %>%
  spread(author, proportion)

frequency
```

```
## # A tibble: 30,292 × 4
##   word      `Bronte Sisters` `H.G. Wells` `Jane Austen`
##   <chr>                <dbl>        <dbl>         <dbl>
## 1 a               0.0000587    0.0000147    0.0000138
## 2 a'n't          NA            NA           0.00000460
## 3 aback           0.00000391   0.0000147   NA
## 4 abaht           0.00000391  NA           NA
## 5 abandon         0.0000313    0.0000147    0.00000460
```

About str_extract()

We use `str_extract()` here because the UTF-8 encoded texts from Project Gutenberg have some examples of words with underscores around them to indicate emphasis (like italics).

Compare words used by Jane Austen, the Bronte sisters, and H.G. Wells

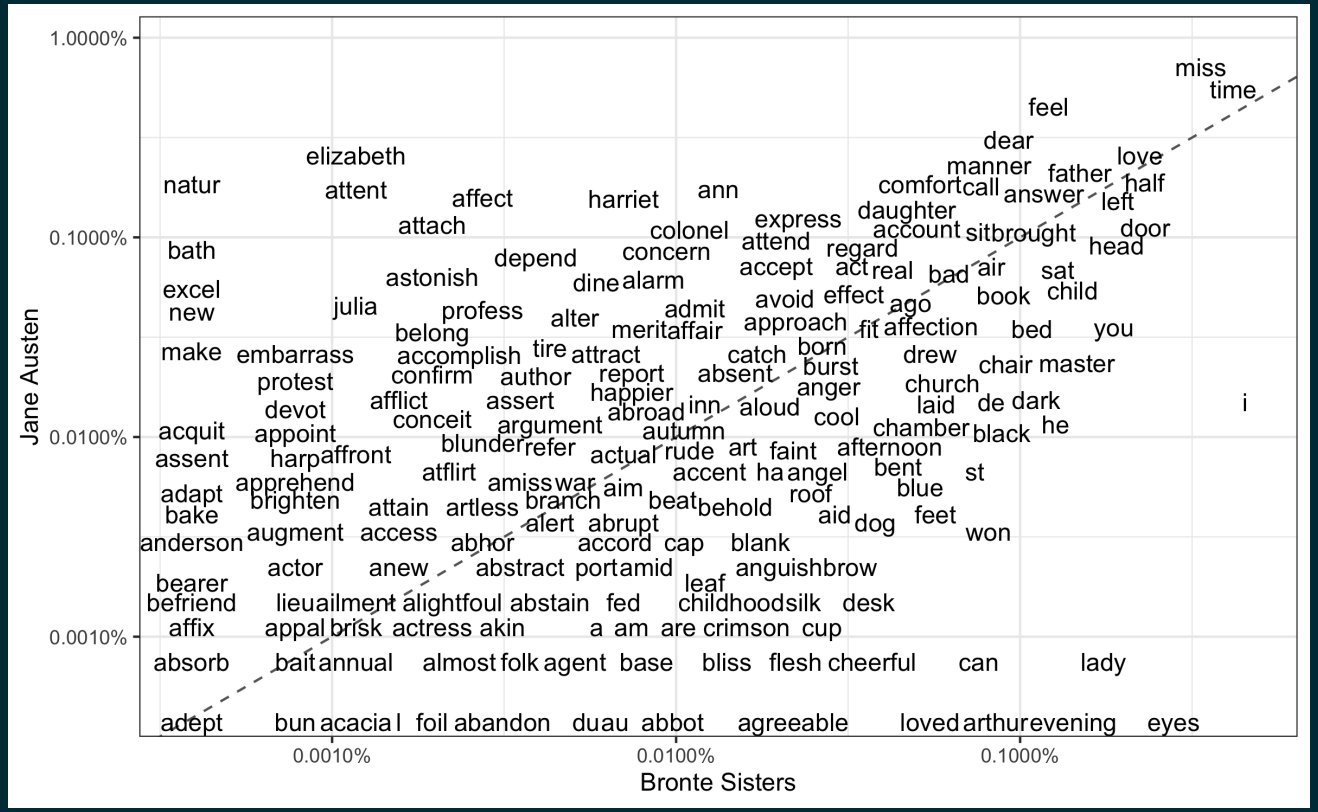
Let's comparing the word frequencies of Jane Austen, the Bronte sisters, and H.G. Wells:

```
library(scales)

# correlate frequencies of words in `BrontÃ« Sisters` and `Jane Austen` books
# expect a warning about rows with missing values being removed
graph <- ggplot(frequency, aes(x = `Bronte Sisters`, y = `Jane Austen`)) +
  geom_abline(color = "gray40", lty = 2) +
  geom_text(aes(label = word), check_overlap = TRUE, vjust = 1.5) +
  scale_x_log10(labels = percent_format()) +
  scale_y_log10(labels = percent_format()) +
  labs(y = "Jane Austen", x = "Bronte Sisters") +
  theme_bw()
```

```
graph

## Warning: Removed 26810 rows containing missing values
## (geom_text).
```



Compare words used by Jane Austen, the Bronte sisters, and H.G. Wells

Let's quantify how similar and different these sets of word frequencies are using a correlation test. How correlated are the word frequencies between Austen and the Bronte sisters, and between Austen and Wells?

```
# quantify correlation
cor.test(frequency$`Bronte Sisters`,  frequency$`Jane Austen`)
```

```
##
##     Pearson's product-moment correlation
##
## data:  frequency$`Bronte Sisters` and frequency$`Jane Austen`
## t = 50.908, df = 3481, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.6338151 0.6719097
## sample estimates:
##       cor
## 0.6532757
```

```
cor.test(frequency$`H.G. Wells`,  frequency$`Jane Austen`)
```

```
##
##     Pearson's product-moment correlation
##
## data:  frequency$`H.G. Wells` and frequency$`Jane Austen`
## t = 17.393, df = 2296, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.3045576 0.3768308
## sample estimates:
##       cor
## 0.3411984
```

Just as we saw in the plots, the word frequencies are more correlated between the Austen and Bronte novels than between Austen and H.G. Wells.