01 The tidy text format

H. David Shea

29 Jul 2021

Contents

The `unnest_tokens` function

```
text <- c("Because I could not stop for Death -",
          "He kindly stopped for me -",
          "The Carriage held but just Ourselves -",
          "and Immortality")

text
#> [1] "Because I could not stop for Death -"
#> [2] "He kindly stopped for me -"
#> [3] "The Carriage held but just Ourselves -"
#> [4] "and Immortality"

text_df <- tibble(line = 1:4, text = text)

text_df
#> # A tibble: 4 x 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

text_df %>%
  unnest_tokens(word, text)
#> # A tibble: 20 x 2
#>     line word
#>    <int> <chr>
#>  1     1 because
```

```
#>  2     1 i
#>  3     1 could
#>  4     1 not
#>  5     1 stop
#>  6     1 for
#>  7     1 death
#>  8     2 he
#>  9     2 kindly
#> 10     2 stopped
#> 11     2 for
#> 12     2 me
#> 13     3 the
#> 14     3 carriage
#> 15     3 held
#> 16     3 but
#> 17     3 just
#> 18     3 ourselves
#> 19     4 and
#> 20     4 immortality
```

```
text_df %>%
  unnest_tokens(word, text, to_lower = FALSE)
#> # A tibble: 20 x 2
#>     line word
#>    <int> <chr>
#>  1     1 Because
#>  2     1 I
#>  3     1 could
#>  4     1 not
#>  5     1 stop
#>  6     1 for
#>  7     1 Death
#>  8     2 He
#>  9     2 kindly
#> 10     2 stopped
#> 11     2 for
#> 12     2 me
#> 13     3 The
#> 14     3 Carriage
#> 15     3 held
#> 16     3 but
#> 17     3 just
#> 18     3 Ourselves
#> 19     4 and
#> 20     4 Immortality
```

Tidying the works of Jane Austen

Get the text from all Jane Austen books, add fields for line number and chapter number. The line number is obtained by a simple `row_number()` call. The chapter number relies on a `cumsum` of each line that starts with the word 'chapter' followed by a space and then a number or any of the (smaller - i.e., no 'm' - not a lot of 1000 chapter books) Roman numeral letters - neat trick.

NOTE: The `austen_books()` data are in text only format - exactly what we want - so no pre-processing is required.

```
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 x 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
#>  7 ""                      Sense & Sensibility          7       0
#>  8 ""                      Sense & Sensibility          8       0
#>  9 ""                      Sense & Sensibility          9       0
#> 10 "CHAPTER 1"             Sense & Sensibility         10       1
#> # ... with 73,412 more rows

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

tidy_books
#> # A tibble: 725,055 x 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
#>  7 Sense & Sensibility          5       0 1811
#>  8 Sense & Sensibility         10       1 chapter
#>  9 Sense & Sensibility         10       1 1
#> 10 Sense & Sensibility         13       1 the
#> # ... with 725,045 more rows
```

Stop words are words that are not usually useful for analyses. These are the typically high frequency common words like 'the', 'of', 'to', etc. The package `tidytext` contains a dataset `stop_words` containing several lexicons' versions of stop words.

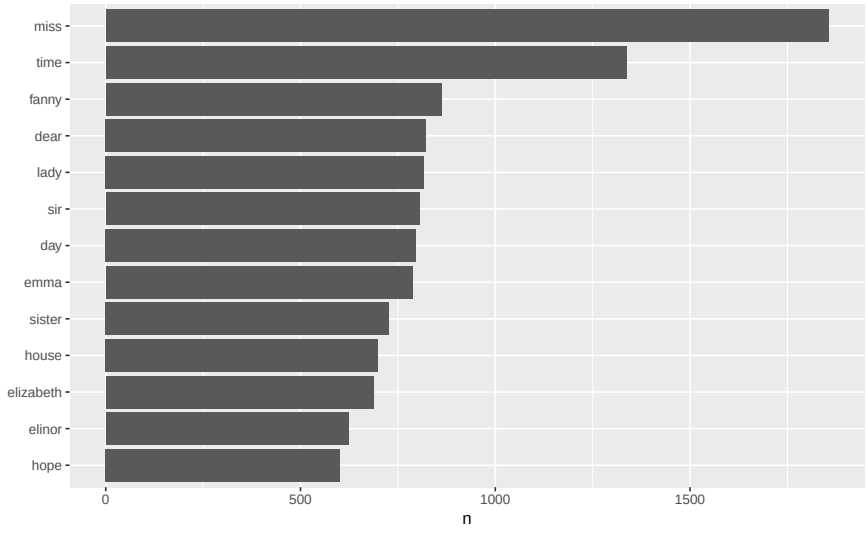
```
data(stop_words)

tidy_books <- tidy_books %>%
  anti_join(stop_words, by = "word")

tidy_books %>%
  count(word, sort = TRUE)
#> # A tibble: 13,914 x 2
#>    word       n
#>    <chr>  <int>
#>  1 miss    1855
#>  2 time    1337
#>  3 fanny    862
#>  4 dear     822
#>  5 lady     817
#>  6 sir      806
#>  7 day      797
#>  8 emma     787
#>  9 sister   727
#> 10 house    699
#> # ... with 13,904 more rows

library(ggplot2)

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



The gutenbergr package

Project Gutenberg is a library of over 60,000 free eBooks. The `gutenbergr` package provides access to these

books. Here, we pull the data for some H.G. Wells books: *The Time Machine* (ID = 35), *The War of the Worlds* (ID = 36), *The Invisible Man* (ID = 5230), and *The Island of Doctor Moreau* (ID = 159). Then we do the same for works from the Bronte Sisters: *Jane Eyre* (ID = 1260), *Wuthering Heights* (ID = 768), *The Tenant of Wildfell Hall* (ID = 969), *Villette* (ID = 9182), and *Agnes Grey* (ID = 767).

```
hgwells <- gutenberg_download(c(35, 36, 5230, 159))

tidy_hgwells <- hgwells %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words, by = "word")

tidy_hgwells %>%
  count(word, sort = TRUE)
#> # A tibble: 11,830 x 2
#>    word       n
#>    <chr>  <int>
#>  1 time     461
#>  2 people   302
#>  3 door     260
#>  4 heard    249
#>  5 black    232
#>  6 stood    229
#>  7 white    224
#>  8 hand     218
#>  9 kemp     213
#> 10 eyes     210
#> # ... with 11,820 more rows

bronte <- gutenberg_download(c(1260, 768, 969, 9182, 767))

tidy_bronte <- bronte %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words, by = "word")

tidy_bronte %>%
  count(word, sort = TRUE)
#> # A tibble: 23,303 x 2
#>    word       n
#>    <chr>  <int>
#>  1 time    1064
#>  2 miss     854
#>  3 day      826
#>  4 hand     767
#>  5 eyes     713
#>  6 don't    666
#>  7 night    648
#>  8 heart    638
#>  9 looked   601
#> 10 door     591
#> # ... with 23,293 more rows
```

Word frequencies

Now we calculate the frequency for each word for the collected work of the set of authors: Jane Austen, the Bronte sisters, and H.G. Wells. This makes good use of `tidverse` operations.

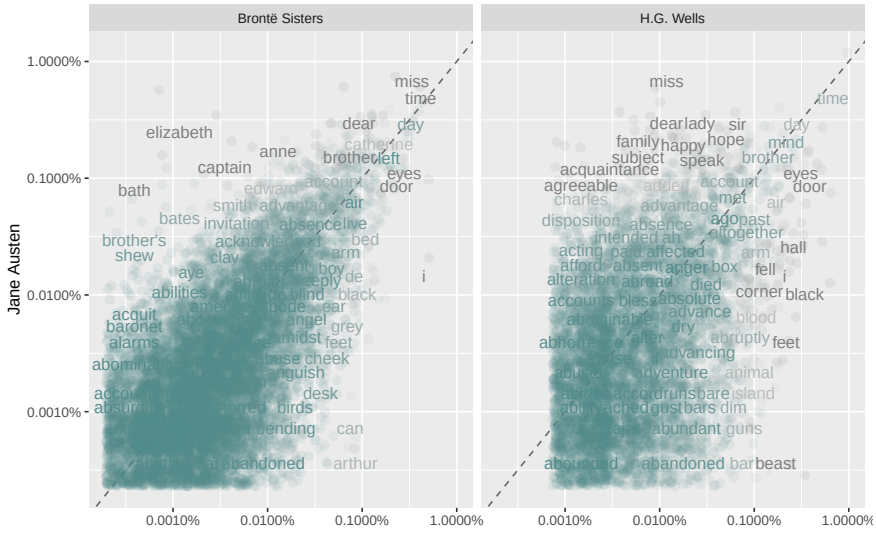
NOTE: The Project Gutenberg books have some examples of emphasized words indicated by underscores. The `str_extract` below, makes sure that only letters and apostrophes are sampled, not the special characters.

```
frequency <- bind_rows(mutate(tidy_bronte, author = "Brontë 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) %>%
  pivot_wider(names_from = author, values_from = proportion) %>%
  pivot_longer(`Brontë Sisters`:`H.G. Wells`,
               names_to = "author", values_to = "proportion")

frequency
#> # A tibble: 57,252 x 4
#>    word      `Jane Austen` author          proportion
#>    <chr>             <dbl> <chr>                <dbl>
#>  1 a            0.00000919 Brontë Sisters  0.0000587
#>  2 a            0.00000919 H.G. Wells      0.0000148
#>  3 aback       NA          Brontë Sisters  0.00000391
#>  4 aback       NA          H.G. Wells      0.0000148
#>  5 abaht       NA          Brontë Sisters  0.00000391
#>  6 abaht       NA          H.G. Wells     NA
#>  7 abandon     NA          Brontë Sisters  0.0000313
#>  8 abandon     NA          H.G. Wells      0.0000148
#>  9 abandoned    0.00000460 Brontë Sisters  0.0000900
#> 10 abandoned    0.00000460 H.G. Wells      0.000178
#> # ... with 57,242 more rows
```

And this can be used to make a frequency scatter plot to show words used at similar frequencies by the authors - words closer to the abline are similar in frequency.

```
ggplot(frequency, aes(x = proportion, y = `Jane Austen`,
                      color = abs(`Jane Austen` - proportion))) +
  geom_abline(color = "gray40", lty = 2) +
  geom_jitter(alpha = 0.1, size = 2.5, width = 0.3, height = 0.3, na.rm = TRUE) +
  geom_text(aes(label = word), check_overlap = TRUE, vjust = 1.5, na.rm = TRUE) +
  scale_x_log10(labels = percent_format()) +
  scale_y_log10(labels = percent_format()) +
  scale_color_gradient(limits = c(0, 0.001),
                       low = "darkslategray4", high = "gray75") +
  facet_wrap(~author, ncol = 2) +
  theme(legend.position="none") +
  labs(y = "Jane Austen", x = NULL)
```



Note the difference in shape between the two plots. The Austen-Bronte plots shows more data points, points that are generally closer to the abline and more lower frequency words in common versus the Austen-Wells plot. This indicates that Jane Austen and the Bronte sisters used more similar words than Jane Austen and H.G. Wells did.

This can be shown in correlation tests as well.

```
cor.test(data = frequency[frequency$author == "Brontë Sisters",],
         ~ proportion + `Jane Austen`)
#>
#>  Pearson's product-moment correlation
#>
#> data:  proportion and Jane Austen
#> t = 111.09, df = 10345, p-value < 2.2e-16
#> alternative hypothesis: true correlation is not equal to 0
#> 95 percent confidence interval:
#>  0.7286568 0.7462330
#> sample estimates:
#>       cor
#> 0.7375698

cor.test(data = frequency[frequency$author == "H.G. Wells",],
         ~ proportion + `Jane Austen`)
#>
#>  Pearson's product-moment correlation
#>
#> data:  proportion and Jane Austen
#> t = 36.083, df = 6046, p-value < 2.2e-16
#> alternative hypothesis: true correlation is not equal to 0
#> 95 percent confidence interval:
#>  0.3999815 0.4414612
#> sample estimates:
#>       cor
#> 0.4209414
```