R scripts can be rendered!

Stefan Veleski

May 13, 2021

Tidytext sentiment analysis Loading in the necessary packages

```
library(tidyverse)
library(syuzhet)
library(gutenbergr)
library(tidytext)
library(textdata)
```

```
hardy_meta <- gutenberg_works(author == "Hardy, Thomas") # Download metadata for Hardy's works
hardy_meta <- hardy_meta %>%
  slice(c(1,2,3,4,5,6,7,8,10,11,13,18,22,23,24)) # Manually selecting Hardy's novels
```

```
austen_meta <- gutenberg_works(author == "Austen, Jane") # Download metadata for Austen's works
austen_meta <- austen_meta %>%  # Cutting off two compilation works that are not necessary
  slice(1:8)
```

Download all of Hardy's and Austen's texts - retain a column with the titles of the novels

```
hardy_tidy_texts <- gutenberg_download(hardy_meta$gutenberg_id,
                                       meta_fields = "title")

austen_tidy_texts <- gutenberg_download(austen_meta$gutenberg_id,
                                       meta_fields = "title")
```

The following code tokenizes the full text by individual words, but also provides additional metadata about the line and the chapter that the word is located in

```
austen_tidy_texts <- austen_tidy_texts %>%
  group_by(title) %>%
  mutate(
    linenumber = row_number(),
    chapter = cumsum(str_detect(text,  #cumulative sum, detect string, column analyzed
                                regex("^chapter [\\divxlc]", # regular expression, chapter any char
                                      ignore_case = TRUE)))) %>% # include both lower and uppercase
  ungroup() %>% # removing the grouping by title set up above
  unnest_tokens(word, text) # tokenization of the text column by word (word per row)

nrc_joy <- get_sentiments("nrc") %>% # tidy text function that extracts the values of 3 sentiment lexic
  filter(sentiment == "joy") # filtering only joy
```

The "Text Mining With R" book uses the janeaustenr package, which is not really

```
# necessary, as we can represent the full workflow from downloading
# the books to visualization, which can be used with books from other authors as well.
```

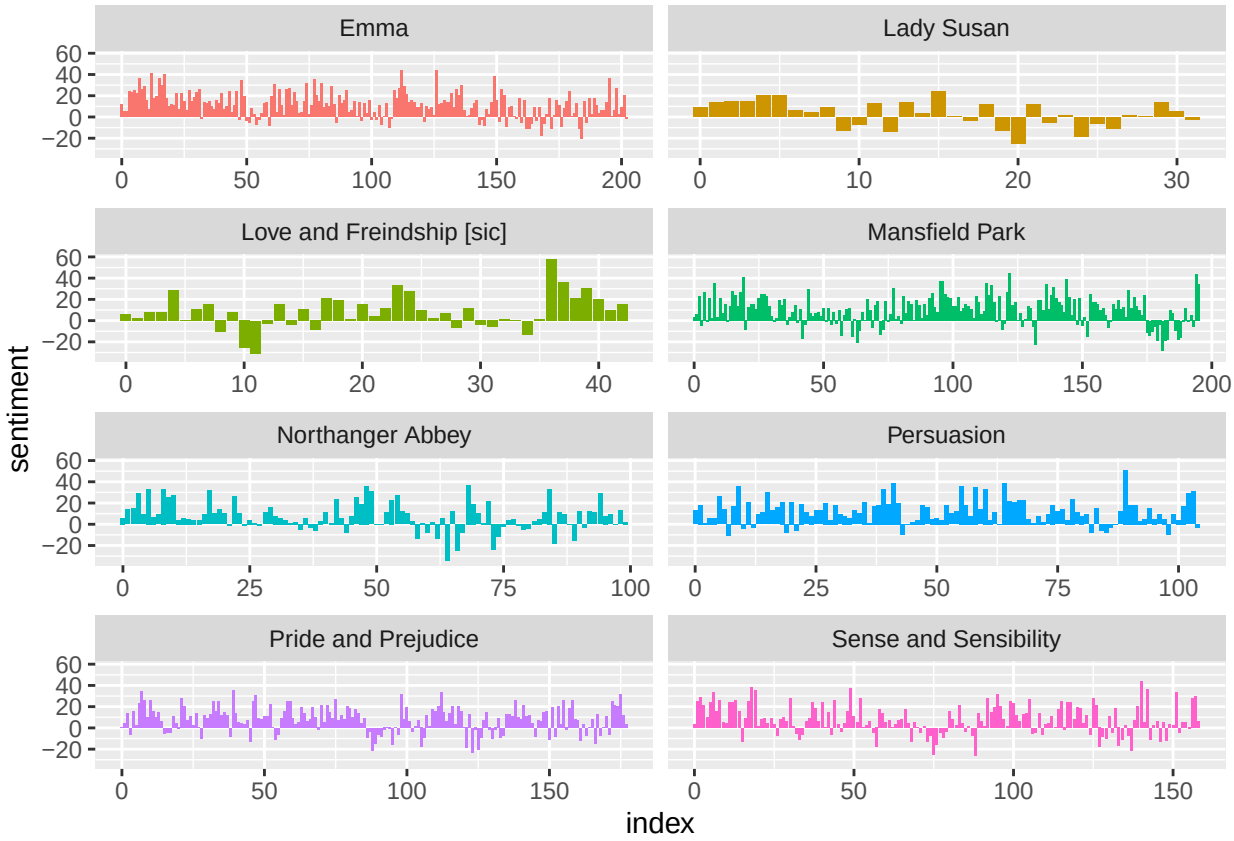
The code below simply counts the most common words in Persuasion that correspond to the joy

```r
# emotion in the NRC sentiment lexicon
```

```r
austen_tidy_texts %>%
  filter(title == "Persuasion") %>% # only selecting Persuasion from the rest of the books
  inner_join(nrc_joy) %>% # combining two tables together (see data wrangling cheat sheet)
  count(word, sort = TRUE) # counting the words
```

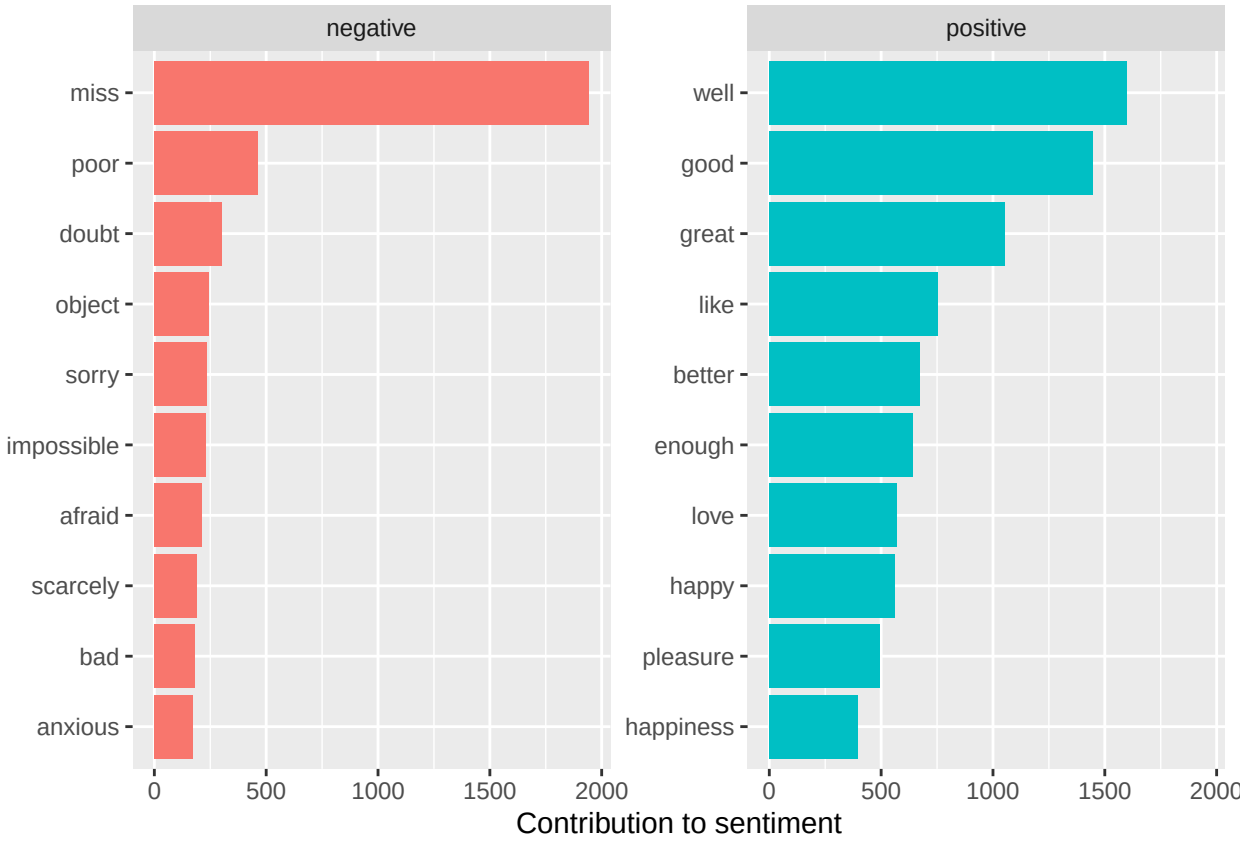
```
## # A tibble: 258 x 2
##    word        n
##    <chr>   <int>
##  1 good      187
##  2 young      84
##  3 found      83
##  4 friend     77
##  5 present    65
##  6 happy      64
##  7 hope       53
##  8 deal       45
##  9 love       42
## 10 spirits    41
## # ... with 248 more rows
```

```r
jane_austen_sentiment <- austen_tidy_texts %>%
  inner_join(get_sentiments("bing")) %>% # change th
  count(title, index = linenumber %/% 80, sentiment) %>%
  pivot_wider(names_from = sentiment, values_from = n, values_fill = 0) %>%
  mutate(sentiment = positive - negative)

ggplot(jane_austen_sentiment, aes(index, sentiment, fill = title)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~title, ncol = 2, scales = "free_x")
```



Most common positive and negative words

```
bing_word_counts <- austen_tidy_texts %>%
  inner_join(get_sentiments("bing")) %>% #retain only words that exist in both
  count(word, sentiment, sort = TRUE) %>%
  ungroup()

bing_word_counts %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10) %>%  # slice n to retain top 10
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word, fill = sentiment)) + # n x axis, word y axis, color according to sentiment
  geom_col(show.legend = FALSE) + # barplot, no legends
  facet_wrap(~sentiment, scales = "free_y") + # facet wrapped along sentiment, free y scale
  labs(x = "Contribution to sentiment", # x label
       y = NULL) # no y label
```




```
# Export in 5x10
```

Adding miss as a custom stopword to the stopwords lexicon contained in the tidytext package. This will be removed from the visualization later on

```
custom_stop_words <- bind_rows(tibble(word = c("miss"),
                                      lexicon = c("custom")),
                               stop_words)
```

Wordclouds clear plot cache

```
library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

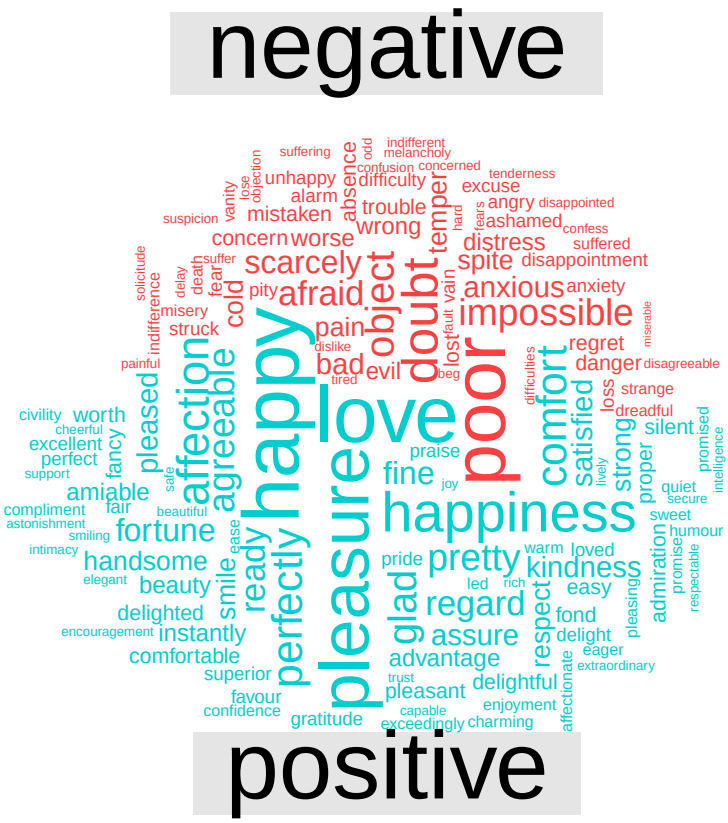
```
library(reshape2)
```

```
##
## Attaching package: 'reshape2'
```

```
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
set.seed(1234) # reproducibility
austen_tidy_texts %>%
  anti_join(custom_stop_words) %>% # remove all words that have a match in custom_stop_words
  inner_join(get_sentiments("bing")) %>% # retain only words that exist in both
  count(word, sentiment, sort = TRUE) %>% #
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
```

```
comparison.cloud(colors = c("brown1", "cyan3"),
                 max.words = 150, # maximum number of words in the visualization
                 scale = c(2.5, 0.05), # largest and smallest words in the word cloud
                 min.freq = 1, # minimal frequency of the words in the visualizaion
                 random.order = FALSE, # no random order, size dictated by frequency
                 rot.per = 0.35) # percentage of words rotated (35%)
```



Syuzhet package

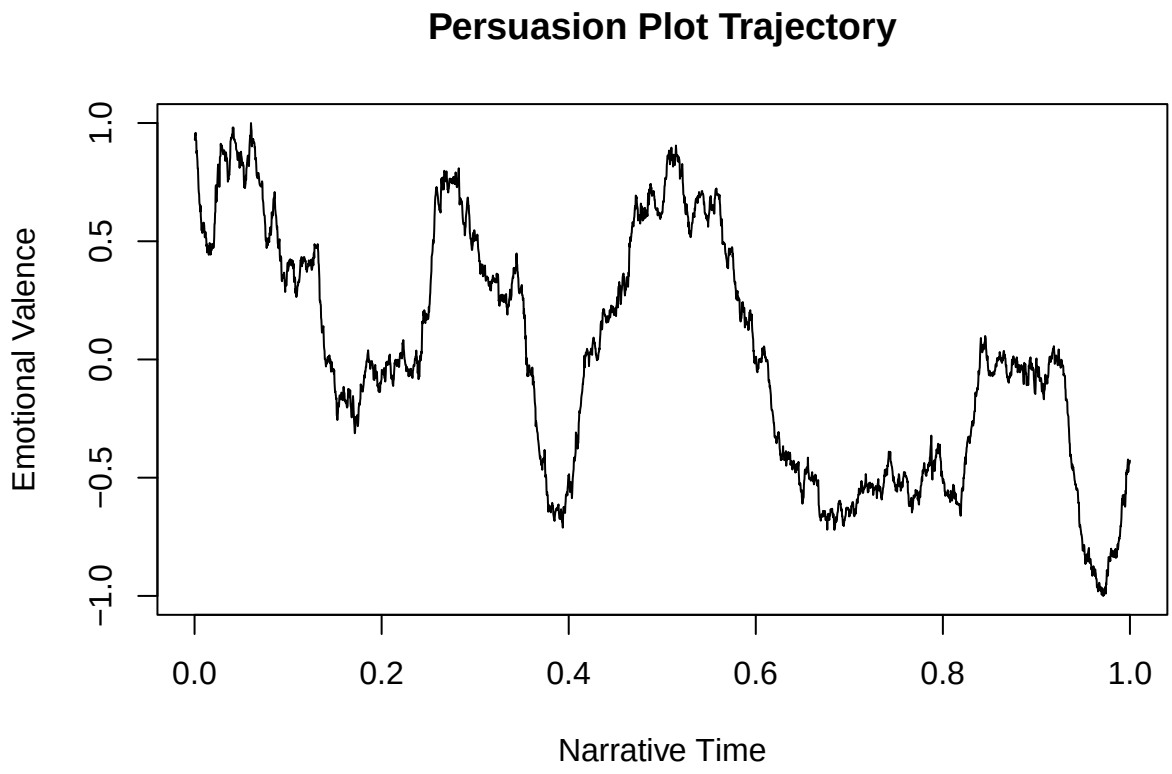
```
library(zoo)
library(syuzhet)
```

One of the more popular specialized sentiment analysis packages. Available dictionaries: bing, afinn, nrc, syuzhet Getting a particular novel as a single character string, ready for syuzhet analysis. Let's first load the Austen novels from scratch, so that we have a clean slate

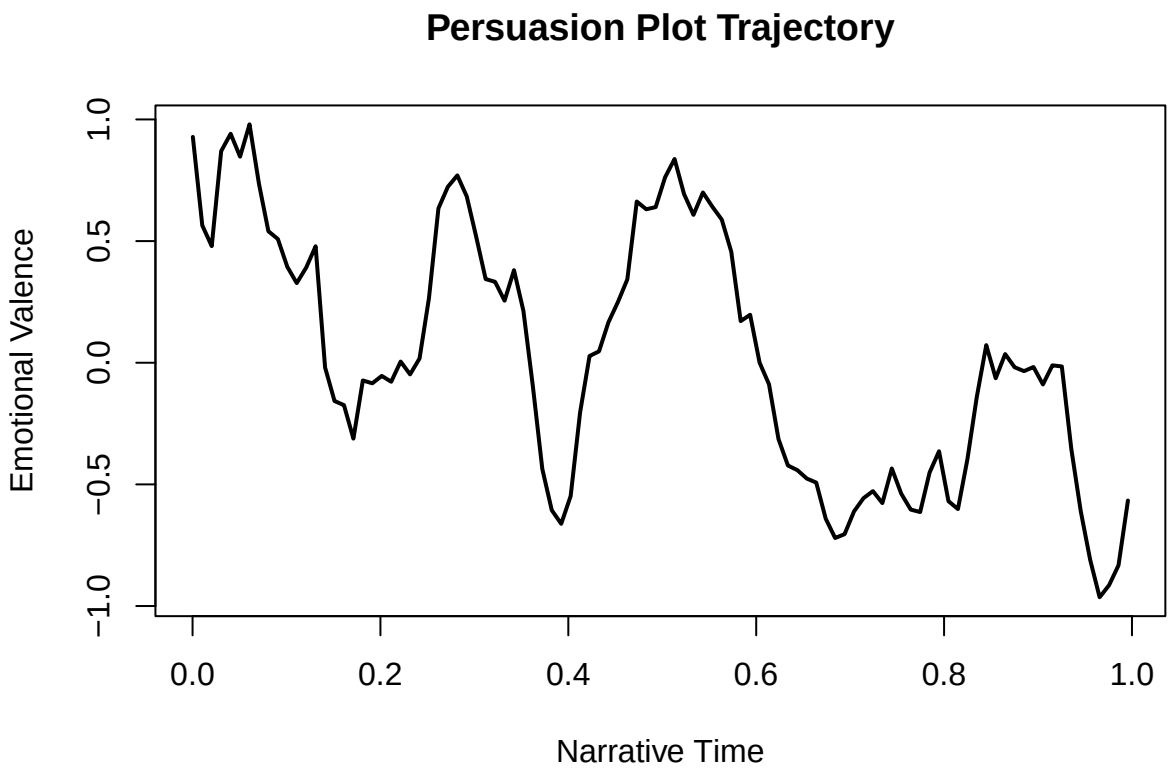
```
austen_tidy_texts <- gutenberg_download(austen_meta$gutenberg_id,
                                        meta_fields = "title")

persuasion_tidy <- austen_tidy_texts %>% #this extracts only the text of Persuasion
  filter(title == "Persuasion")

persuasion_string <- paste(persuasion_tidy$text, collapse = " ") # Extracting the text column as
#character string

persuasion_sentences <- tolower(get_sentences(persuasion_string)) # Extract sentences & lowercase
```

```
persuasion_sentiment <- get_sentiment(persuasion_sentences, method = "syuzhet") # Get sentiment for eac
```

```
pwdw <- round(length(persuasion_sentiment)*.1)
persuasion_rolled <- rollmean(persuasion_sentiment, k=pwdw) #moving/rolling average (1/10 window)
persuasion_list <- rescale_x_2(persuasion_rolled) # rescaled so another novel can be compared
plot(persuasion_list$x,
     persuasion_list$z,
     type="l", # line plot
     main ="Persuasion Plot Trajectory", # title
     col="black", # color of the line
     xlab="Narrative Time", #
     ylab="Emotional Valence") #This is almost perfect, but it's not smoothed out just right.
```




```
persuasion_sample <- seq(1, length(persuasion_list$x),
                         by=round(length(persuasion_list$x)/100)) # Taking 100 equal chunks of the text
plot(persuasion_list$x[persuasion_sample],
     persuasion_list$z[persuasion_sample],
     type="l", # line plot
     lwd = 2, # line width
     main = "Persuasion Plot Trajectory",# title
     col="black", # color of the line
     xlab="Narrative Time",#x axis name
     ylab="Emotional Valence",#y axis name
)
```



Package of the week Sentimentr A sentiment analysis package that is more sophisticated in several ways than both tidytext and

```
# syuzhet. Available dictionaries:
```

```
library(sentimentr)
library(magrittr)

austen_sentiment <- austen_tidy_texts %>%
  mutate(sentences = get_sentences(text)) %$%
  sentiment_by(sentences, title)

hardy_sentiment <- hardy_tidy_texts %>%
  mutate(sentences = get_sentences(text)) %$%
  sentiment_by(sentences, title)

author_column <- factor(c("Austen", "Austen","Austen","Austen","Austen","Austen",
                          "Austen","Austen","Hardy", "Hardy","Hardy","Hardy",
                          "Hardy","Hardy","Hardy","Hardy","Hardy","Hardy",
                          "Hardy","Hardy","Hardy","Hardy","Hardy"))
```

Combining the austen and hardy sentiment dataframes

```
hardy_vs_austen <- rbind(austen_sentiment, hardy_sentiment)
```

Adding the author column, which is already a factor vector

```
hardy_vs_austen <- hardy_vs_austen %>%
  cbind(author_column)
```

The resulting dataframe is an ideal use case for a boxplot/violin plot visualization. Let's use the ggstatsplot package we used in week 5.

```
library(ggstatsplot)
```

Final visualization

```
options(scipen = 10000)

hardy_vs_austen_plot <- ggbetweenstats(
  data = hardy_vs_austen, # data
  x = author_column, # data for x axis
  y = ave_sentiment, # data for y axis
  title = "Comparison of the mean sentiment of Hardy's and Austen's novels", # Title
  xlab = "Author", # x axis label
  ylab = "Sentiment" # y axis label
)
hardy_vs_austen_plot
```