

Assignment 10A

James Chun

October 31, 2025

Sentiment Analysis

Primary Code Example

We will first incorporate the primary example code from <https://www.tidytextmining.com/sentiment.html> (Robinson) as it appears on the site. However, some edits have to be made to make the code functional in this markdown. As such, all such edits will be indicated with the comment “NEW”.

2.1 The sentiments datasets

```
library(textdata) #NEW

library(tidytext)

get_sentiments("afinn")

## # A tibble: 2,477 x 2
##   word      value
##   <chr>     <dbl>
## 1 abandon    -2
## 2 abandoned  -2
## 3 abandons   -2
## 4 abducted   -2
## 5 abduction  -2
## 6 abductions -2
## 7 abhor      -3
## 8 abhorred   -3
## 9 abhorrent  -3
## 10 abhors    -3
## # i 2,467 more rows

get_sentiments("bing")

## # A tibble: 6,786 x 2
##   word      sentiment
##   <chr>     <chr>
## 1 2-faces   negative
```

```

##  2 abnormal    negative
##  3 abolish     negative
##  4 abominable   negative
##  5 abominably   negative
##  6 abominante   negative
##  7 abomination  negative
##  8 abort        negative
##  9 aborted      negative
## 10 aborts       negative
## # i 6,776 more rows

```

```
get_sentiments("nrc")
```

```

## # A tibble: 13,872 x 2
##   word      sentiment
##   <chr>     <chr>
## 1 abacus    trust
## 2 abandon   fear
## 3 abandon   negative
## 4 abandon   sadness
## 5 abandoned anger
## 6 abandoned fear
## 7 abandoned negative
## 8 abandoned sadness
## 9 abandonment anger
## 10 abandonment fear
## # i 13,862 more rows

```

2.2 Sentiment analysis with inner join

```

library(janeaustenr)
library(dplyr)
library(stringr)

tidy_books <- austen_books() %>%
  group_by(book) %>%
  mutate(
    linenum = row_number(),
    chapter = cumsum(str_detect(text,
                                 regex("^chapter [\\d\\w]+")),
                     ignore_case = TRUE))) %>%
  ungroup() %>%
  unnest_tokens(word, text)

nrc_joy <- get_sentiments("nrc") %>%
  filter(sentiment == "joy")

tidy_books %>%
  filter(book == "Emma") %>%
  inner_join(nrc_joy) %>%
  count(word, sort = TRUE)

```

```

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

## # A tibble: 301 x 2
##   word      n
##   <chr>    <int>
## 1 good     359
## 2 friend   166
## 3 hope     143
## 4 happy    125
## 5 love     117
## 6 deal     92
## 7 found    92
## 8 present   89
## 9 kind     82
## 10 happiness 76
## # i 291 more rows

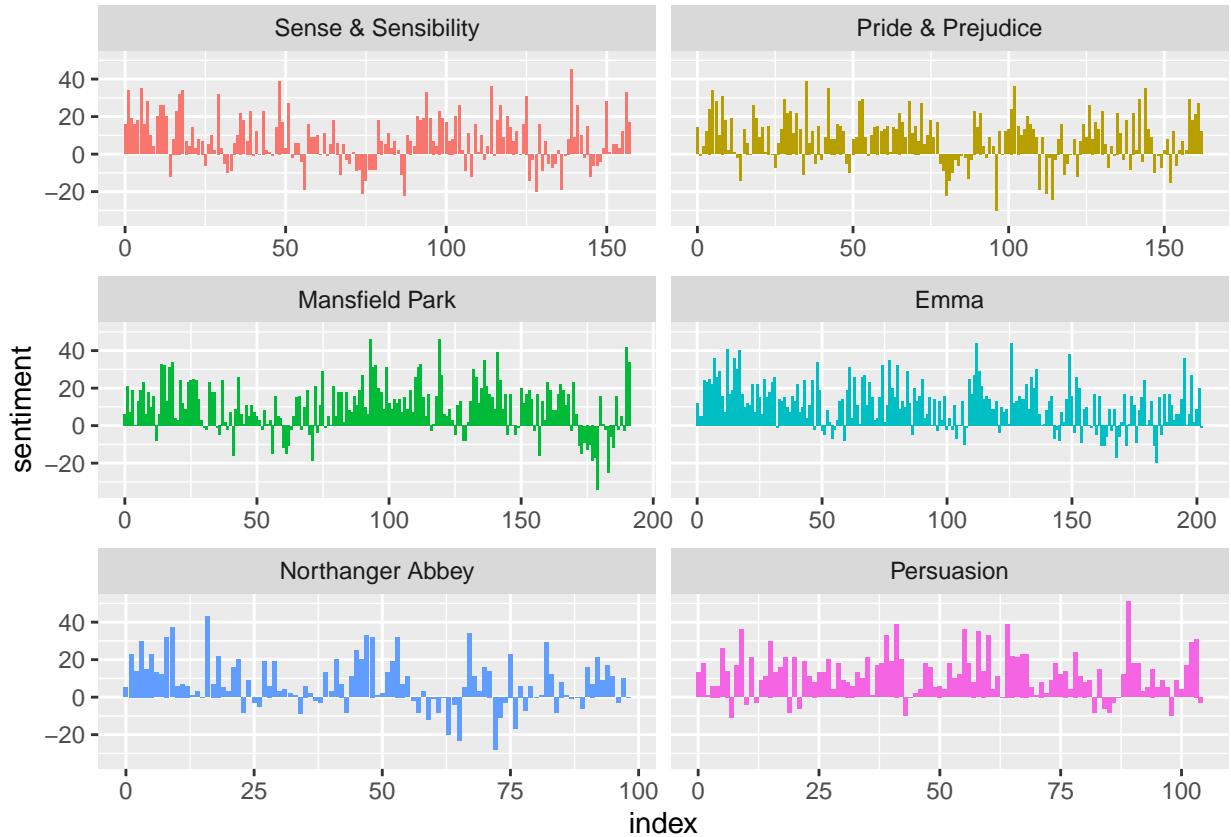
library(tidyr)

jane_austen_sentiment <- tidy_books %>%
  inner_join(get_sentiments("bing")) %>%
  count(book, index = linenumbers %/ 80, sentiment) %>%
  pivot_wider(names_from = sentiment, values_from = n, values_fill = 0) %>%
  mutate(sentiment = positive - negative)

library(ggplot2)

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

```



2.3 Comparing the three sentiment dictionaries

```
pride_prejudice <- tidy_books %>%
  filter(book == "Pride & Prejudice")

pride_prejudice

## # A tibble: 122,204 x 4
##   book           linenumber chapter word
##   <fct>          <int>     <int> <chr>
## 1 Pride & Prejudice      1        0 pride
## 2 Pride & Prejudice      1        0 and
## 3 Pride & Prejudice      1        0 prejudice
## 4 Pride & Prejudice      3        0 by
## 5 Pride & Prejudice      3        0 jane
## 6 Pride & Prejudice      3        0 austen
## 7 Pride & Prejudice      7        1 chapter
## 8 Pride & Prejudice      7        1 1
## 9 Pride & Prejudice     10        1 it
## 10 Pride & Prejudice     10        1 is
## # i 122,194 more rows
```

```

afinn <- pride_prejudice %>%
  inner_join(get_sentiments("afinn")) %>%
  group_by(index = linenumbers %/ 80) %>%
  summarise(sentiment = sum(value)) %>%
  mutate(method = "AFINN")

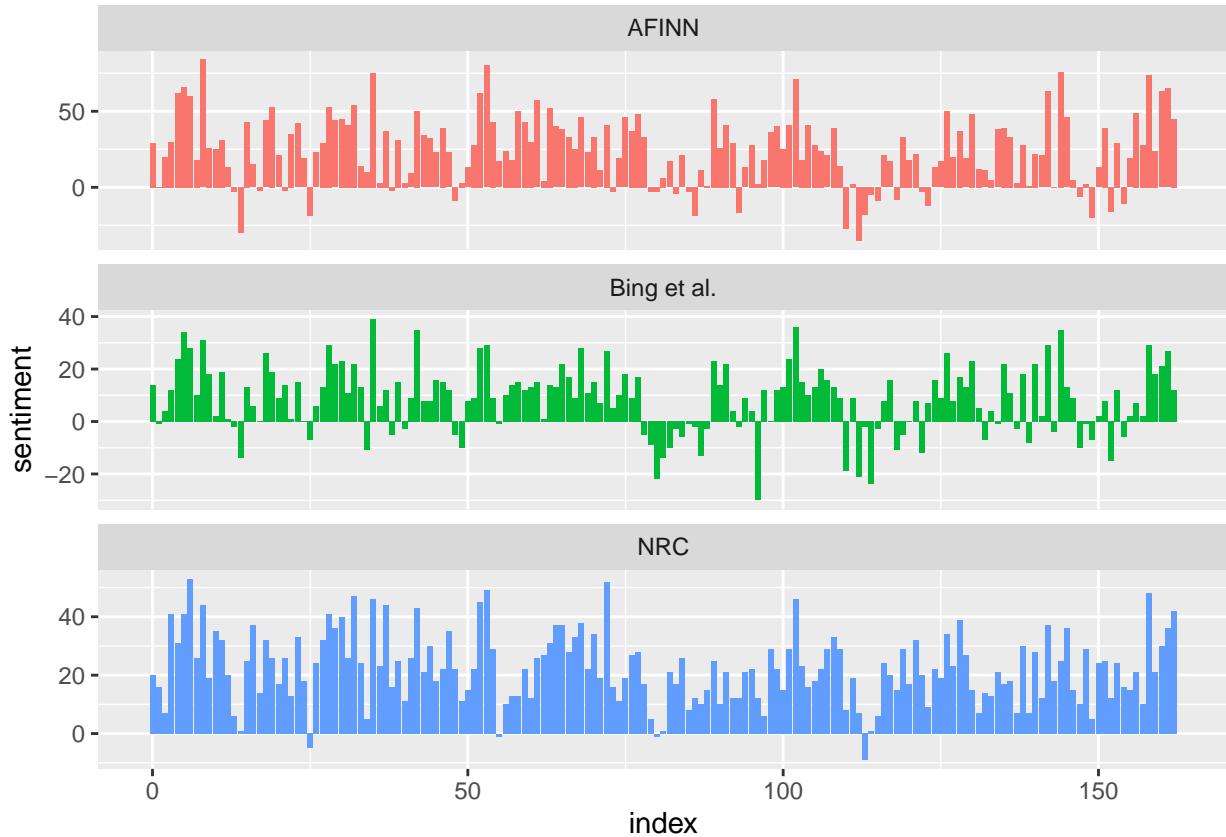
bing_and_nrc <- bind_rows(
  pride_prejudice %>%
    inner_join(get_sentiments("bing")) %>%
    mutate(method = "Bing et al."),
  pride_prejudice %>%
    inner_join(get_sentiments("nrc")) %>%
    filter(sentiment %in% c("positive",
                           "negative"))
  ) %>%
  mutate(method = "NRC")) %>%
count(method, index = linenumbers %/ 80, sentiment) %>%
pivot_wider(names_from = sentiment,
            values_from = n,
            values_fill = 0) %>%
mutate(sentiment = positive - negative)

```

```

bind_rows(afinn,
          bing_and_nrc) %>%
ggplot(aes(index, sentiment, fill = method)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~method, ncol = 1, scales = "free_y")

```



```
get_sentiments("nrc") %>%
  filter(sentiment %in% c("positive", "negative")) %>%
  count(sentiment)
```

```
## # A tibble: 2 x 2
##   sentiment     n
##   <chr>      <int>
## 1 negative    3316
## 2 positive    2308
```

```
get_sentiments("bing") %>%
  count(sentiment)
```

```
## # A tibble: 2 x 2
##   sentiment     n
##   <chr>      <int>
## 1 negative    4781
## 2 positive    2005
```

2.4 Most common positive and negative words

```
bing_word_counts <- tidy_books %>%
  inner_join(get_sentiments("bing")) %>%
```

```

count(word, sentiment, sort = TRUE) %>%
ungroup()

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

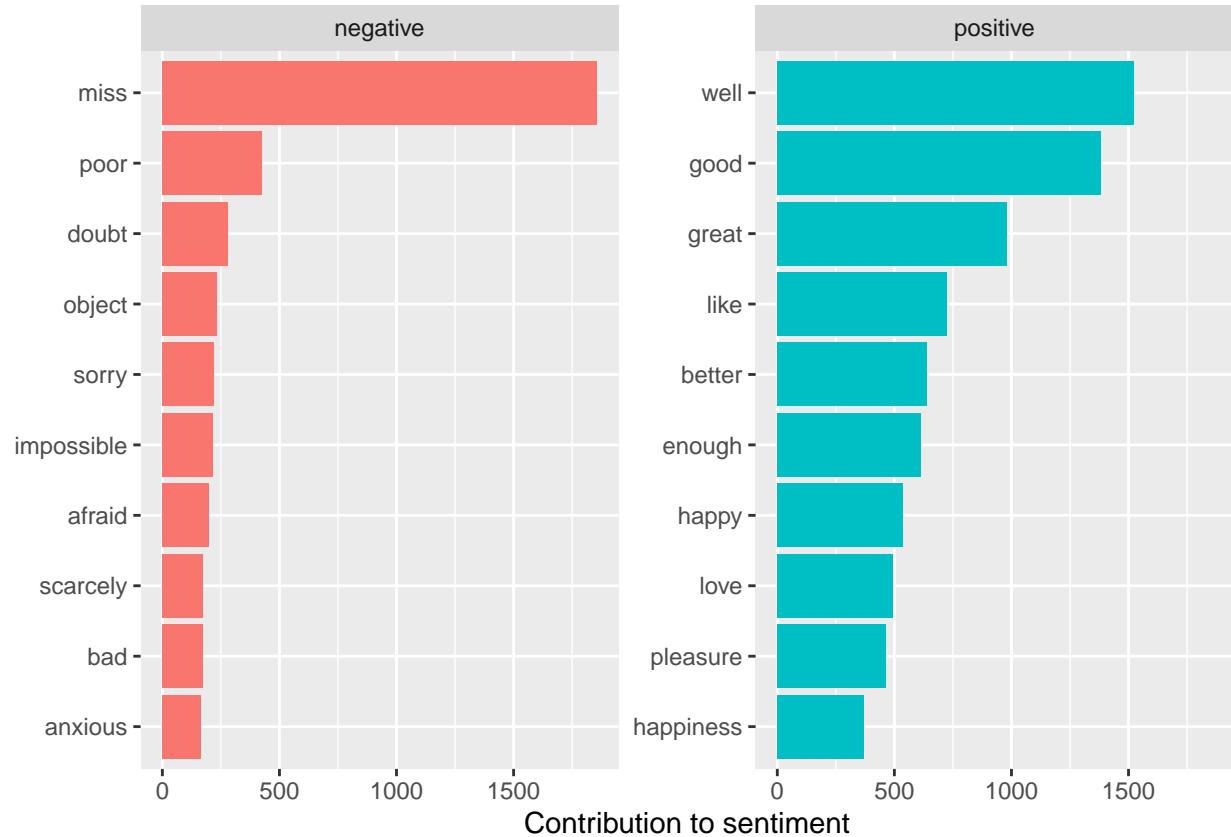
## Warning in inner_join(., get_sentiments("bing")): Detected an unexpected many-to-many relationship b
## i Row 435434 of 'x' matches multiple rows in 'y'.
## i Row 5051 of 'y' matches multiple rows in 'x'.
## i If a many-to-many relationship is expected, set 'relationship =
##   "many-to-many"' to silence this warning.

bing_word_counts

## # A tibble: 2,585 x 3
##   word      sentiment     n
##   <chr>    <chr>     <int>
## 1 miss     negative    1855
## 2 well      positive    1523
## 3 good      positive    1380
## 4 great     positive    981
## 5 like      positive    725
## 6 better     positive    639
## 7 enough     positive    613
## 8 happy      positive    534
## 9 love       positive    495
## 10 pleasure   positive   462
## # i 2,575 more rows

bing_word_counts %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(x = "Contribution to sentiment",
       y = NULL)

```



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

```
custom_stop_words
```

```
## # A tibble: 1,150 x 2
##   word      lexicon
##   <chr>     <chr>
## 1 miss     custom
## 2 a        SMART
## 3 a's      SMART
## 4 able     SMART
## 5 about    SMART
## 6 above    SMART
## 7 according SMART
## 8 accordingly SMART
## 9 across   SMART
## 10 actually SMART
## # ... with 1,140 more rows
```

2.5 Wordclouds

```
library(wordcloud)

tidy_books %>%
  anti_join(stop_words) %>%
  count(word) %>%
  with(wordcloud(word, n, max.words = 100))
```



```
library(reshape2)

tidy_books %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("gray20", "gray80"),
                    max.words = 100)
```

negative

misery
excused
disappointment
concern
absence
ashamed
mistaken
danger
trouble
alarm
worse
cold
wrong
pity
distress
spite
impossible
struck
strange
scarcely
afraid
sorry
anxious
vanity
angry
bad
poor
object
loss
fear
doubt
indifference
pain
evil
anxiety
instantly
glad
ready
strong
superior
happy
fancy
well
great
good
proper
enough
like
right
kindness
handsome
handsome
worth
pleasure
best
wonder
gratitude
silent
amiable
thank
happiness
fine
perfectly
fond
delight
advantage
affection
pleased
loved
pleasant
work
assure
easy
favour
admiration
greatest
delighted
comfortable

positive

2.6 Looking at units beyond just words

```
p_and_p_sentences <- tibble(text = prideprejudice) %>%
  unnest_tokens(sentence, text, token = "sentences")
```

```
p_and_p_sentences$sentence[2]
```

```
## [1] "by jane austen"
```

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

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

```
## # A tibble: 6 x 2
##   book           chapters
##   <fct>          <int>
```

```

## 1 Sense & Sensibility      51
## 2 Pride & Prejudice        62
## 3 Mansfield Park           49
## 4 Emma                      56
## 5 Northanger Abbey          32
## 6 Persuasion                 25

bingnegative <- get_sentiments("bing") %>%
  filter(sentiment == "negative")

wordcounts <- tidy_books %>%
  group_by(book, chapter) %>%
  summarize(words = n())

tidy_books %>%
  semi_join(bingnegative) %>%
  group_by(book, chapter) %>%
  summarize(negativewords = n()) %>%
  left_join(wordcounts, by = c("book", "chapter")) %>%
  mutate(ratio = negativewords/words) %>%
  filter(chapter != 0) %>%
  slice_max(ratio, n = 1) %>%
  ungroup()

## # A tibble: 6 x 5
##   book            chapter negativewords words   ratio
##   <fct>           <int>        <int> <int>   <dbl>
## 1 Sense & Sensibility     43         161  3405  0.0473
## 2 Pride & Prejudice       34         111  2104  0.0528
## 3 Mansfield Park          46         173  3685  0.0469
## 4 Emma                     15         151  3340  0.0452
## 5 Northanger Abbey         21         149  2982  0.0500
## 6 Persuasion                  4         62  1807  0.0343

```

Code Extension

This extension intends to expand on the primary example code by applying the same concepts to a different corpus, whilst introducing an additional sentiment lexicon.

Load Libraries

```

library(textdata)
library(dplyr)
library(gutenbergr)
library(tidyr)
library(ggplot2)

```

Corpus and Sentiment Lexicon

We will be using the loughran sentiment lexicon. We will also be using literary works from Project Gutenberg accessed by the R package ‘gutenbergr’. The specific works I’ll be looking at will be the novels “Tom Sawyer”

and “Huckleberry Finn” by Mark Twain. First we’ll import the necessary data and text.

```
# Sentiment lexicon
loughran <- get_sentiments("loughran")

# Corpora
twain_ids <- gutenberg_works(author == "Twain, Mark") #only need to see the ids
twain_books <- gutenberg_download(c(74, 76))

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

Tidy the novels

```
twain_books$book[twain_books$gutenberg_id == 74] <- "THE ADVENTURES OF TOM SAWYER"
twain_books$book[twain_books$gutenberg_id == 76] <- "ADVENTURES OF HUCKLEBERRY FINN"
twain_books$gutenberg_id <- NULL

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

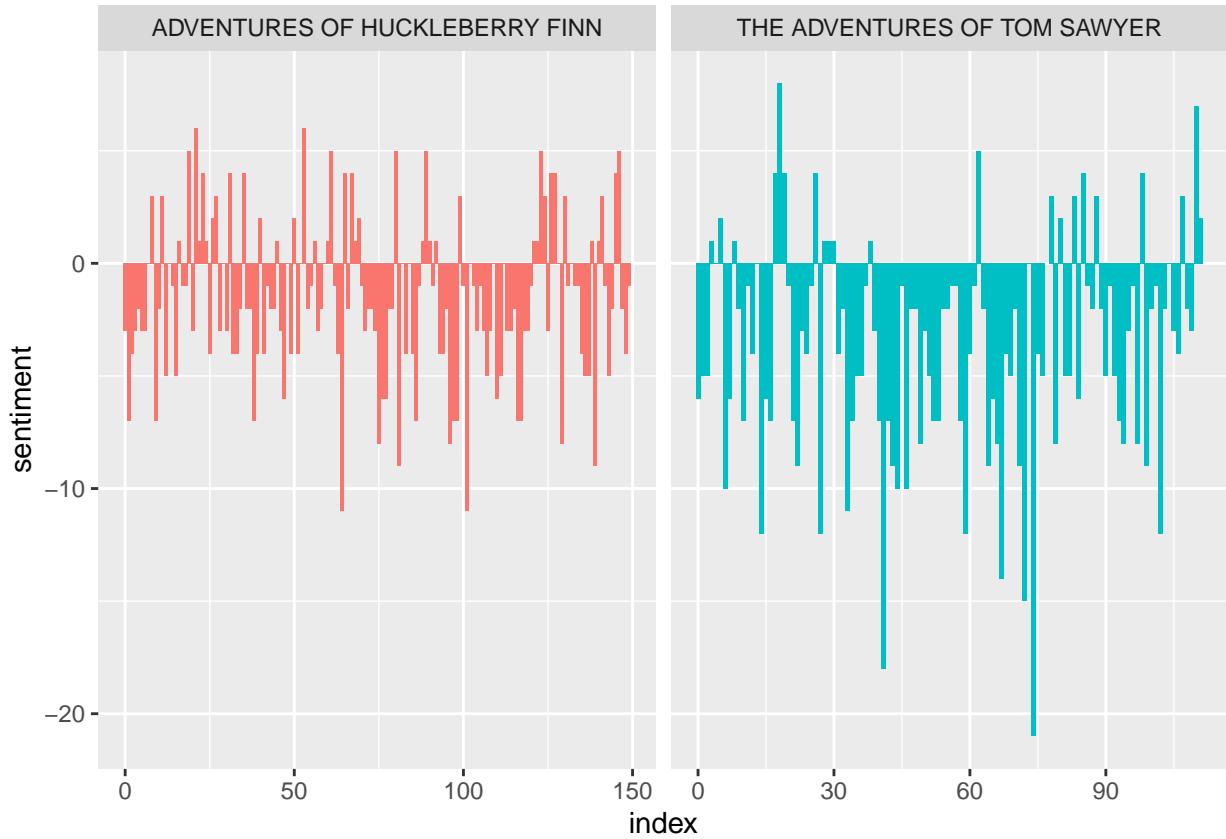
Sentiment Analysis

Perform sentiment analysis using the lexicon from ‘loughran’ on the two Twain novels.

```
mark_twain_sentiment <- twain_tidy_books %>%
  inner_join(get_sentiments("loughran")) %>%
  count(book, index = linenumber %/% 80, sentiment) %>%
  pivot_wider(names_from = sentiment, values_from = n, values_fill = 0) %>%
  mutate(sentiment = positive - negative)
```

Plot the results

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



Lexicon Comparisons

Like the example code, we will be comparing Loughran to the lexicons AFINN, NFC, and bing. Also like the example code, we will focus on one novel: "Tom Sawyer".

```
tom_sawyer <- twain_tidy_books %>%
  filter(book == "THE ADVENTURES OF TOM SAWYER")

twain_afinn <- tom_sawyer %>%
  inner_join(get_sentiments("afinn")) %>%
  group_by(index = linenumbers %/% 80) %>%
  summarise(sentiment = sum(value)) %>%
  mutate(method = "AFINN")

twain_bing_and_nrc <- bind_rows(
  tom_sawyer %>%
    inner_join(get_sentiments("bing")) %>%
    mutate(method = "Bing et al."),
  tom_sawyer %>%
    inner_join(get_sentiments("nrc")) %>%
    filter(sentiment %in% c("positive",
                           "negative"))
) %>%
  mutate(method = "NRC")) %>%
count(method, index = linenumbers %/% 80, sentiment) %>%
```

```

pivot_wider(names_from = sentiment,
            values_from = n,
            values_fill = 0) %>%
mutate(sentiment = positive - negative)

twain_loughran <- tom_sawyer %>%
  inner_join(get_sentiments("loughran")) %>%
  count(book, index = linenumbers %/ 80, sentiment) %>%
  pivot_wider(names_from = sentiment, values_from = n, values_fill = 0) %>%
  mutate(sentiment = positive - negative) %>%
  mutate(method = "Loughran")

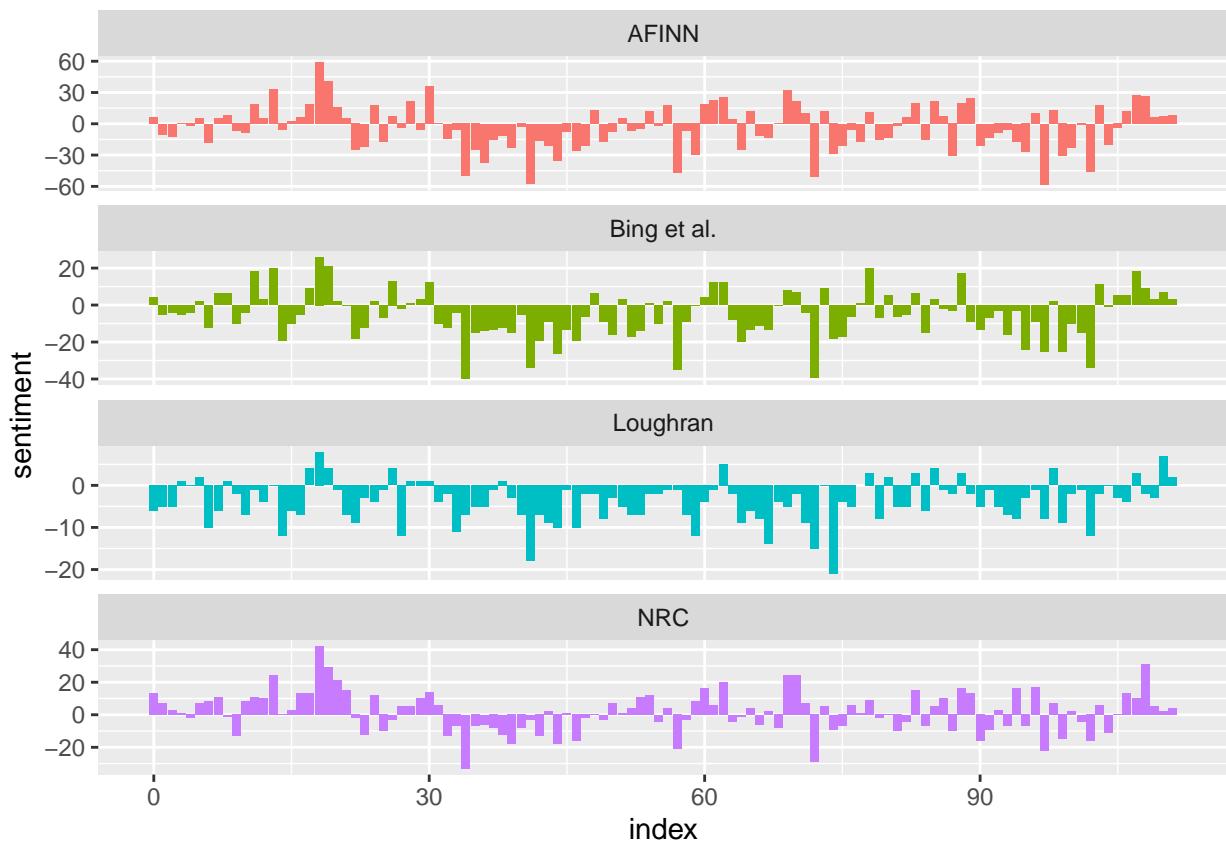
```

Plot the results

```

bind_rows(twain_afinn,
          twain_bing_and_nrc, twain_loughran) %>%
  ggplot(aes(index, sentiment, fill = method)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~method, ncol = 1, scales = "free_y")

```



Most of the plots suggest that Tom Sawyer is a surprisingly negative book – unless I am remembering it wrong. With that said, AFIN and NRC display the most positive outlook.

References

Robinson, J. S. and D. (n.d.). 2 sentiment analysis with Tidy Data: Text mining with R. A Tidy Approach.
<https://www.tidytextmining.com/sentiment.html>