WEEK 10

Heleine Fouda

2023-11-12

In this assignment, we're asked to replicate and then expand upon the sentiment analysis code provided in Chapter 2 looks at Sentiment Analysis.. We"ll start by getting and replicating the provided code and then we will extend the code in two ways: 1. by working with a different corpus, which in this case will be Ernest Hemingway's novel *The Sun also Rises*; 2 Then we'll incorporate one additional sentiment lexicon in addition to the three used by the textbook. And, since the R tidytext package contains 4 lexicons c("bing", "afinn", "ncr,"loughran") and that only the first three in the list were used in the textbook, we will add the "loughran" lexicon to our analysis tools.

Getting started: Loading the Libraries

Recreating Base Analysis from Textbook

The textbook used in this assignment is:Text Mining with R: A Tidy Approach, written by Julia Silge and David Robinson. The book was last built on 2022-11-02. .After first replicating some aspects of the book's code, I will the knowledge to then conduct my own sentiment analysis on Ernest Hemingway's book *The Sun also Rises* in order to discover and evaluate the main opinions or emotions of the book.

Getting specific sentiment lexicons with function get_sentiments().

afinn. AFINN lexicon measures sentiment with a numeric score between -5 and 5,

```
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
```

bing. The bing lexicon categorizes words in a binary fashion, either positive or negative.

```
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 abominate   negative
##  7 abomination negative
##  8 abort       negative
##  9 aborted     negative
## 10 aborts      negative
## # i 6,776 more rows
```

nrc. NRC lexicon categorizes words in a binary fashion, either positive or negative.

```
url <- ("http://saifmohammad.com/Webpages/lexicons.html")
nrc <- url
```

```
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
```

Sentiment analysis with inner join

1. Import Jane Austen Books

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

2. Find the common joy words in Emma by first taking the text of the novels and converting it to the tidy format using unnest_tokens()

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

3. Next, let's filter() the data frame for the words from Emma and then use inner_join() to perform the

sentiment analysis to find the most common joy words in Emma.

```
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
```

4. Use pivot_wider() to have negative and positive sentiment in separate columns, and lastly calculate a net sentiment (positive - negative).

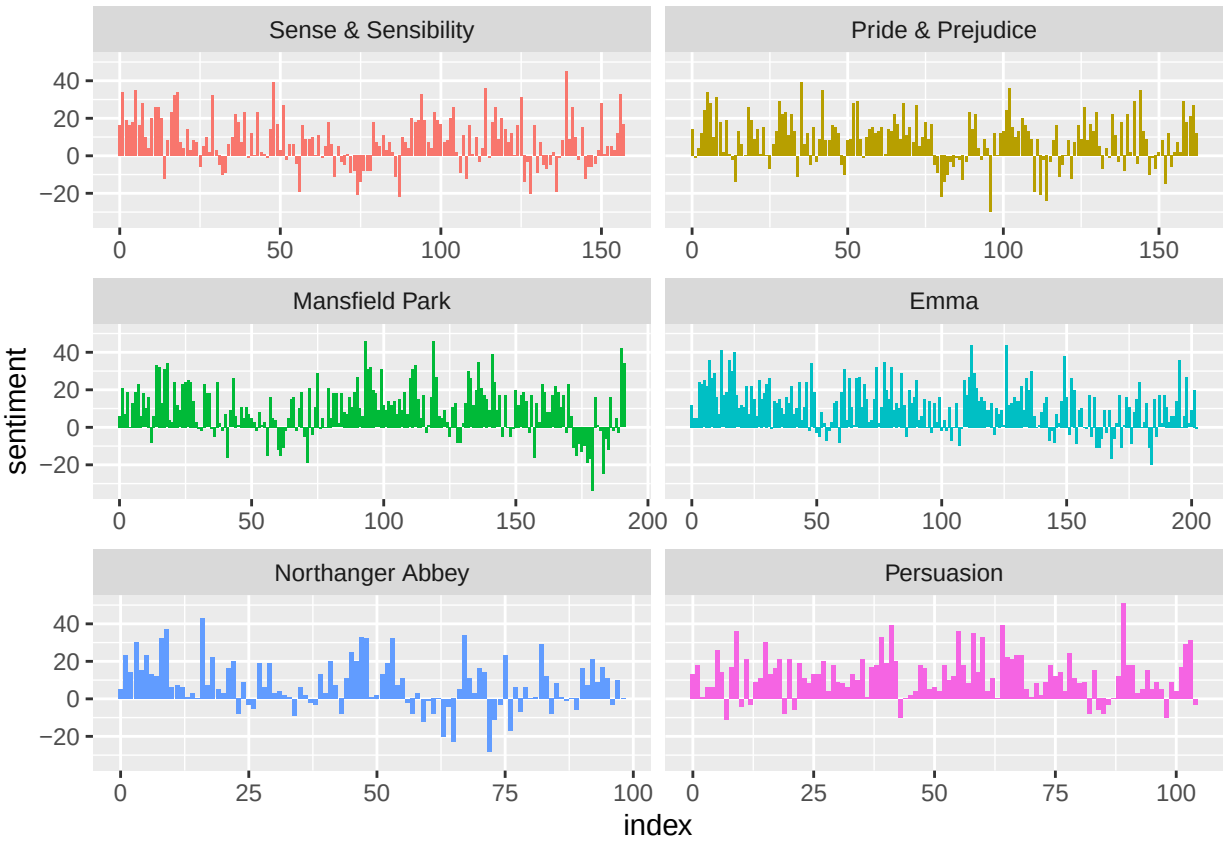
```
library(tidyr)

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

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

```
## Warning in inner_join(., get_sentiments("bing")): Detected an unexpected many-to-many relationship be
## 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.
```

5. Plot the sentiment scores (against the index on the x-axis that keeps track of narrative time in sections of text)across the plot trajectory of each novel.

```
library(ggplot2)

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



Comparing the three sentiment dictionaries

```
# Using filter() to choose the words from the one novel we are interested in.
pride_prejudice <- tidy_books %>%
  filter(book == "Pride & Prejudice")
```

Use inner_join() to calculate the sentiment in different ways.

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

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

```
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 = linenumber %/% 80, sentiment) %>%
  pivot_wider(names_from = sentiment,
```

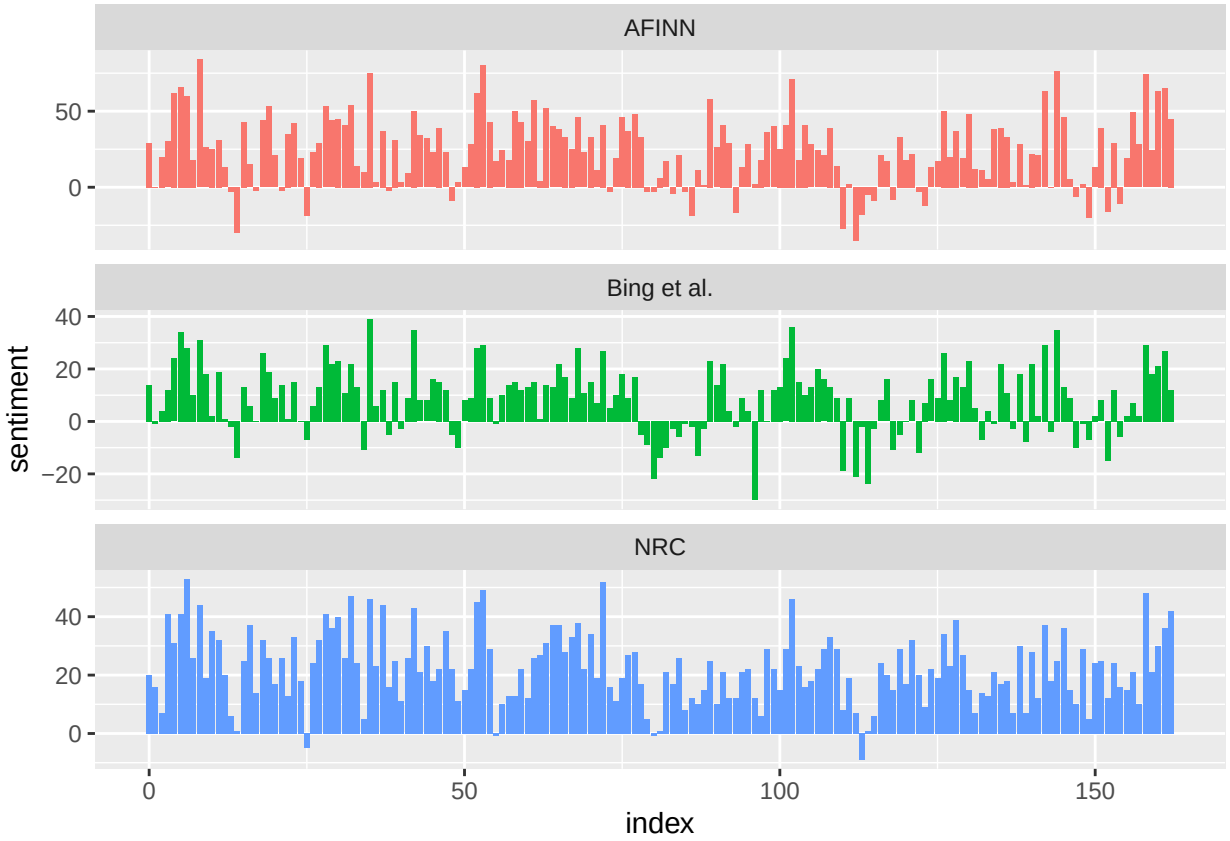
```
              values_from = n,
              values_fill = 0) %>%
  mutate(sentiment = positive - negative)
```

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

```
## Warning in inner_join(., get_sentiments("nrc") %>% filter(sentiment %in% : Detected an unexpected ma
## i Row 215 of `x` matches multiple rows in `y`.
## i Row 5178 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.
```

We now have an estimate of the net sentiment (positive - negative) in each chunk of the novel text for each sentiment lexicon. Let's bind them together and visualize them

```
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")
```



The three different lexicons for calculating sentiment give results that are different in an absolute sense but have similar relative trajectories through the novel. We see similar dips and peaks in sentiment at about the same places in the novel, but the absolute values are significantly different. The NRC sentiment is high, the AFINN sentiment has more variance, the Bing et al. sentiment appears to find longer stretches of similar text, but all three agree roughly on the overall trends in the sentiment through a narrative arc.

Finding the most positive and negative words by using the Bing lexicon to split the texts into 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)`
```

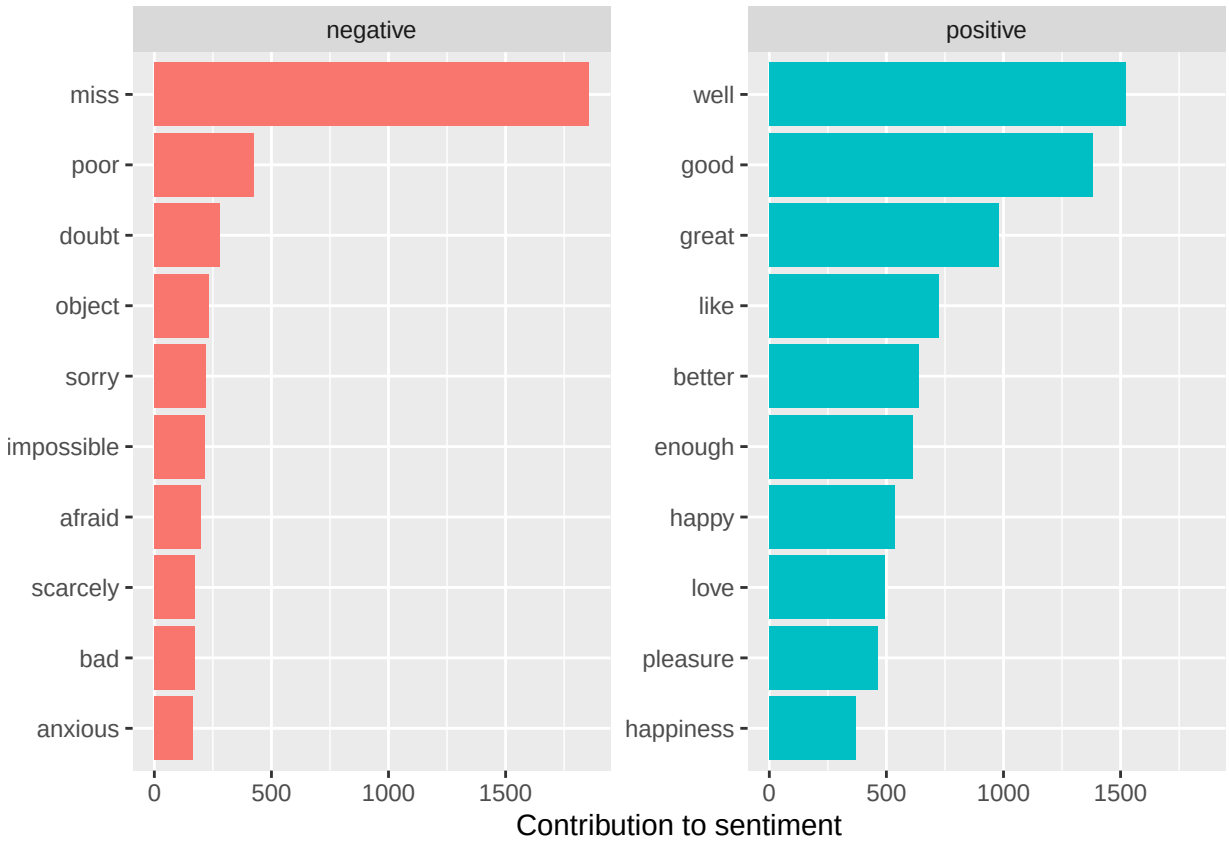
```
## Warning in inner_join(., get_sentiments("bing")): Detected an unexpected many-to-many relationship be
## i Row 435434 of `x` matches multiple rows in `y`.
## i Row 5051 of `y` matches multiple rows in `x`.
## i If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"`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

We can see how much each word contributes to the positive or negative sentiment and that can be shown visually using ggplot2

```
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)
```



Let's look at the most common words in Jane Austen's works as a whole again, but this time as a wordcloud, instead of ggplot2. The wordcloud package uses base R graphics.

```
library(wordcloud)

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

```
## Warning in wordcloud(word, n, max.words = 100): miss could not be fit on page.
## It will not be plotted.
```

We can also use reshape2's acast() function to turn the information into a matrix and then use comparison.cloud() to compare the positive and negative words.

```
library(reshape2)
```

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

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

```
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)
```

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

```
## Warning in inner_join(., get_sentiments("bing")): Detected an unexpected many-to-many relationship be
## 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.
```

The size of a word's text is in proportion to its frequency within its sentiment. We can use this visualization to see the most important positive and negative words, but the sizes of the words are not comparable across sentiments.

Extending with the "loughan"lexicon from the R tidytext package

```
get_sentiments("loughran")
```

```
## # A tibble: 4,150 x 2
##    word         sentiment
##    <chr>        <chr>
##  1 abandon      negative
##  2 abandoned    negative
##  3 abandoning   negative
##  4 abandonment  negative
##  5 abandonments negative
##  6 abandons     negative
##  7 abdicated    negative
##  8 abdicates    negative
##  9 abdicating   negative
## 10 abdication   negative
## # i 4,140 more rows
```

```
loughran<- tidy_books %>%
  inner_join(get_sentiments("loughran")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()
```

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

```
## Warning in inner_join(., get_sentiments("loughran")): Detected an unexpected many-to-many relationshi
## i Row 1252 of `x` matches multiple rows in `y`.
## i Row 2772 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.
```

```
loughran
```

```
## # A tibble: 1,374 x 3
##    word    sentiment       n
##    <chr>   <chr>       <int>
##  1 could   uncertainty  3613
##  2 miss    negative     1855
##  3 good    positive     1380
##  4 might   uncertainty  1369
##  5 great   positive      981
##  6 may     uncertainty   956
##  7 shall   litigious     834
##  8 better  positive      639
##  9 happy   positive      534
## 10 perhaps uncertainty   491
## # i 1,364 more rows
```

Bringing in a New Corpus -The Sun Also Rises by Ernest Hemingway

```
library(gutenbergr)
```

```
gutenberg_works(title == "The Sun Also Rises")
```

```
## # A tibble: 1 x 8
##   gutenberg_id title     author gutenberg_author_id language gutenberg_bookshelf
##          <int> <chr>     <chr>                <int> <chr>    <chr>
## 1        67138 The Sun ~ Hemin~               50533 en       ""
## # i 2 more variables: rights <chr>, has_text <lgl>
```

```
# Equipped with the gutenberg author id for the book we can now download it
the_sun_also_rises <- gutenberg_download(50533)
```

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

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

```
the_sun_also_rises
```

```
## # A tibble: 4,467 x 2
##    gutenberg_id text
##           <int> <chr>
##  1        50533 "  MOTOR STORIES"
##  2        50533 ""
##  3        50533 "  THRILLING"
##  4        50533 "  ADVENTURE"
##  5        50533 ""
##  6        50533 "  MOTOR"
##  7        50533 "  FICTION"
##  8        50533 ""
##  9        50533 "  NO. 21"
```

```
## 10        50533 "  JULY 17, 1909"
## # i 4,457 more rows
```

Tidying things up

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

```
## Rows: 33,064
## Columns: 4
## $ gutenberg_id <int> 50533, 50533, 50533, 50533, 50533, 50533, 50533, 50533, 5~
## $ linenumber   <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ chapter      <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ word         <chr> "motor", "stories", "thrilling", "adventure", "motor", "f~
```

```
tidy_book2  <- tidy_book %>%
  mutate(chapter = word)
tidy_book2
```

```
## # A tibble: 33,064 x 4
##    gutenberg_id linenumber chapter   word
##           <int>      <int> <chr>     <chr>
##  1        50533          1 motor     motor
##  2        50533          1 stories   stories
##  3        50533          1 thrilling thrilling
##  4        50533          1 adventure adventure
##  5        50533          1 motor     motor
##  6        50533          1 fiction   fiction
##  7        50533          1 no        no
##  8        50533          1 21        21
##  9        50533          1 july      july
## 10        50533          1 17        17
## # i 33,054 more rows
```

Let's find words related to happiness in the Sun Also Rises

```
library(dplyr)
nrc_joy2 <- get_sentiments("nrc") %>%
  filter(sentiment == "joy")

tidy_book2 %>%
  inner_join(nrc_joy2) %>%
  count(word, sort = TRUE)
```

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

```
## # A tibble: 125 x 2
##    word       n
##    <chr>  <int>
##  1 money     65
```

```
##  2 good      52
##  3 found     18
##  4 tree      14
##  5 cove      11
##  6 luck      11
##  7 pretty    11
##  8 safe      11
##  9 youth     10
## 10 friend     9
## # i 115 more rows
```

```
library(tidyr)


Hemingway_sentiment<- tidy_book2 %>%
  inner_join(get_sentiments("bing")) %>%
  count(gutenberg_id,index = linenumber %/% 80, sentiment) %>%
  pivot_wider(names_from = sentiment, values_from = n, values_fill = 0) %>%
  mutate(sentiment = positive - negative)
```

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

```
Hemingway_sentiment
```

```
## # A tibble: 1 x 5
##   gutenberg_id index negative positive sentiment
##          <int> <dbl>    <int>    <int>     <int>
## 1        50533     0      821      810       -11
```

Let's find the most common words in the Sun Also Rises

```
bing_word_counts2 <- tidy_book %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()
```

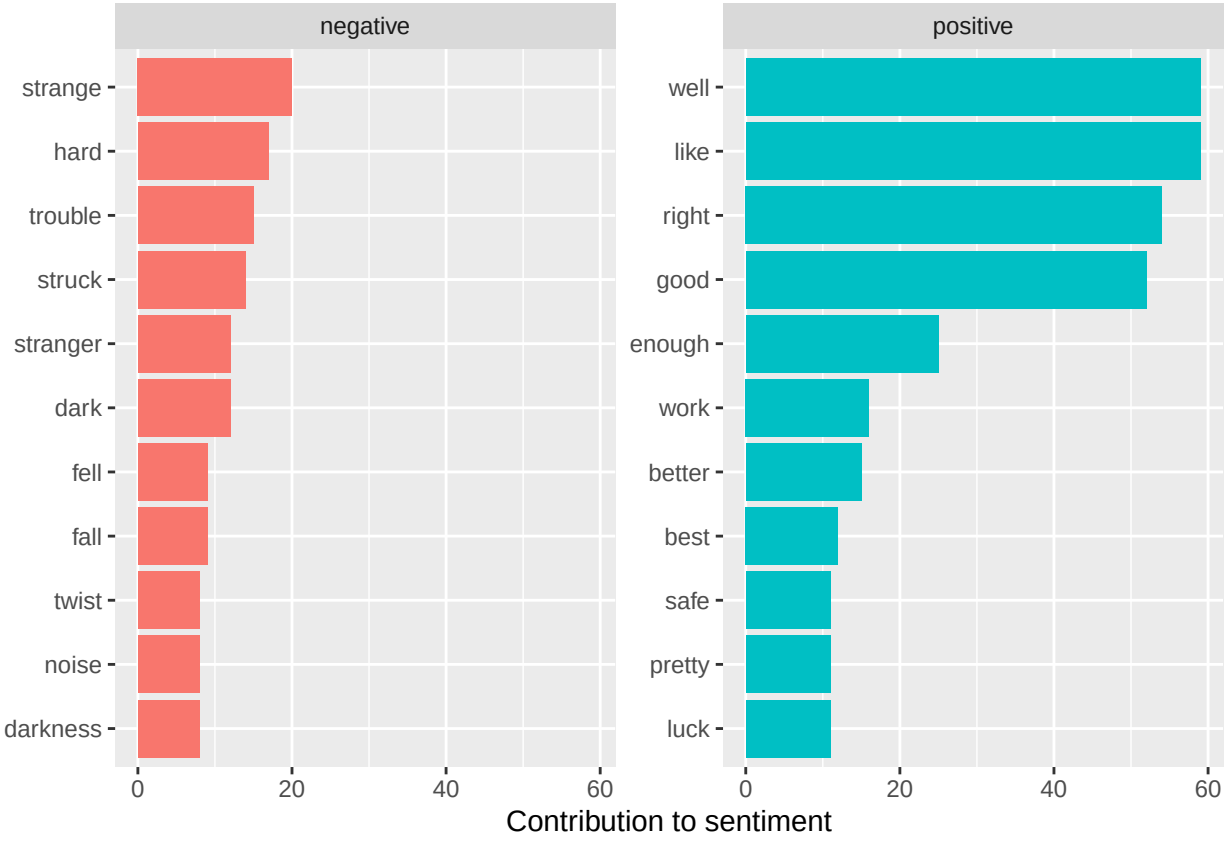
```
## Joining with `by = join_by(word)`
```

```
bing_word_counts2
```

```
## # A tibble: 625 x 3
##    word    sentiment     n
##    <chr>   <chr>     <int>
##  1 like    positive     59
##  2 well    positive     59
##  3 right   positive     54
##  4 good    positive     52
##  5 enough  positive     25
##  6 strange negative     20
##  7 hard    negative     17
##  8 work    positive     16
##  9 better  positive     15
## 10 trouble negative     15
## # i 615 more rows
```

Visualizing with ggplot2

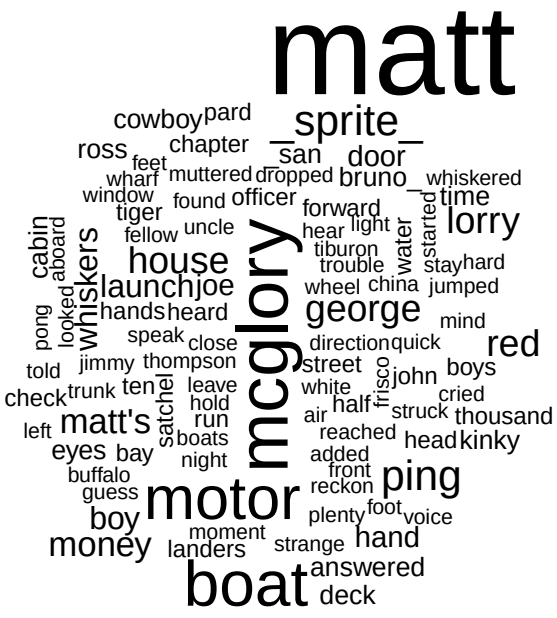
```
bing_word_counts2 %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(x = "Contribution to sentiment",
       y = NULL)
```



Let's visualize with a wordcloud

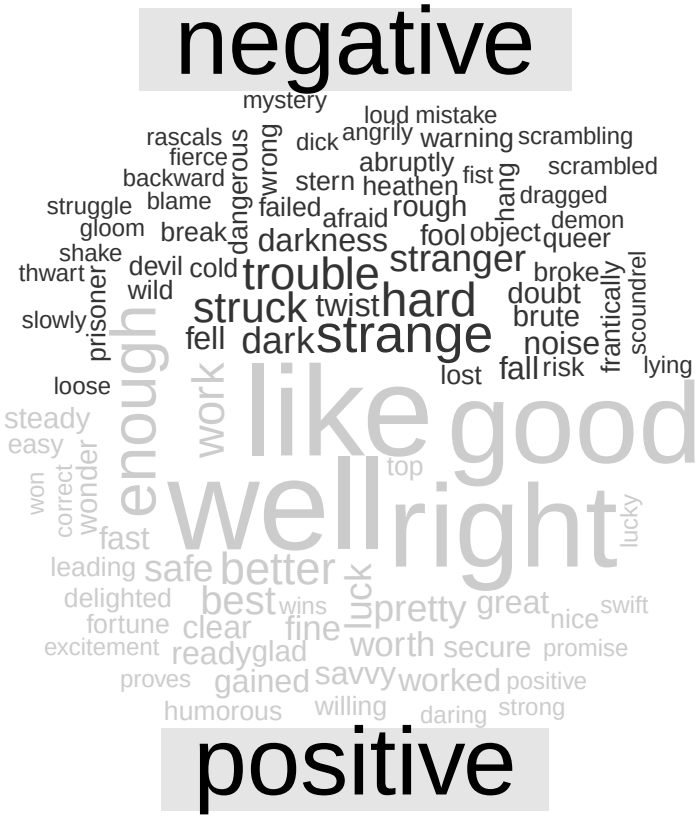
```
library(wordcloud)

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

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




```
tidy_book %>%
  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)
```

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



Let's use the cross_join() from dplyr instead of of inner_join() to see the amount of sentiments each lexiconn dictionary capture from Hemingway's novel.

```
# afinn
linenumber <-1
value <-count(sentiments)
afinn <- the_sun_also_rises %>%
  cross_join (get_sentiments("afinn")) %>%
  group_by(index = linenumber %/% 80) %>%
  summarise(sentiment = sum(value)) %>%
  mutate(method = "AFINN")
afinn
```

```
## # A tibble: 1 x 3
##   index sentiment method
##   <dbl>     <dbl> <chr>
## 1     0  -6521820 AFINN
```

```
# nrc
linenumber <-1
value <-count(sentiments)

nrc <- the_sun_also_rises %>%
  cross_join (get_sentiments("nrc")) %>%
  group_by(index = linenumber %/% 80) %>%
  summarise(sentiment = sum(value)) %>%
  mutate(method = "nrc")
nrc
```

```
## # A tibble: 1 x 3
##   index sentiment method
##   <dbl>     <int> <chr>
## 1     0      6786 nrc
```

```
# loughran

linenumber <-1
value <-count(sentiments)

loughran<- the_sun_also_rises %>%
  cross_join (get_sentiments("loughran")) %>%
  group_by(index = linenumber %/% 80) %>% summarise(sentiment = sum(value))%>%
  mutate(method = "loughran")

loughran
```

```
## # A tibble: 1 x 3
##   index sentiment method
##   <dbl>     <int> <chr>
## 1     0      6786 loughran
```

```
# bing
linenumber <-1
value <-count(sentiments)

bing<- the_sun_also_rises %>%
  cross_join (get_sentiments("bing")) %>%
```

```
  group_by(index = linenumber %/% 80) %>%
summarise(sentiment = sum(value)) %>%
mutate(method = "bing")
bing
```

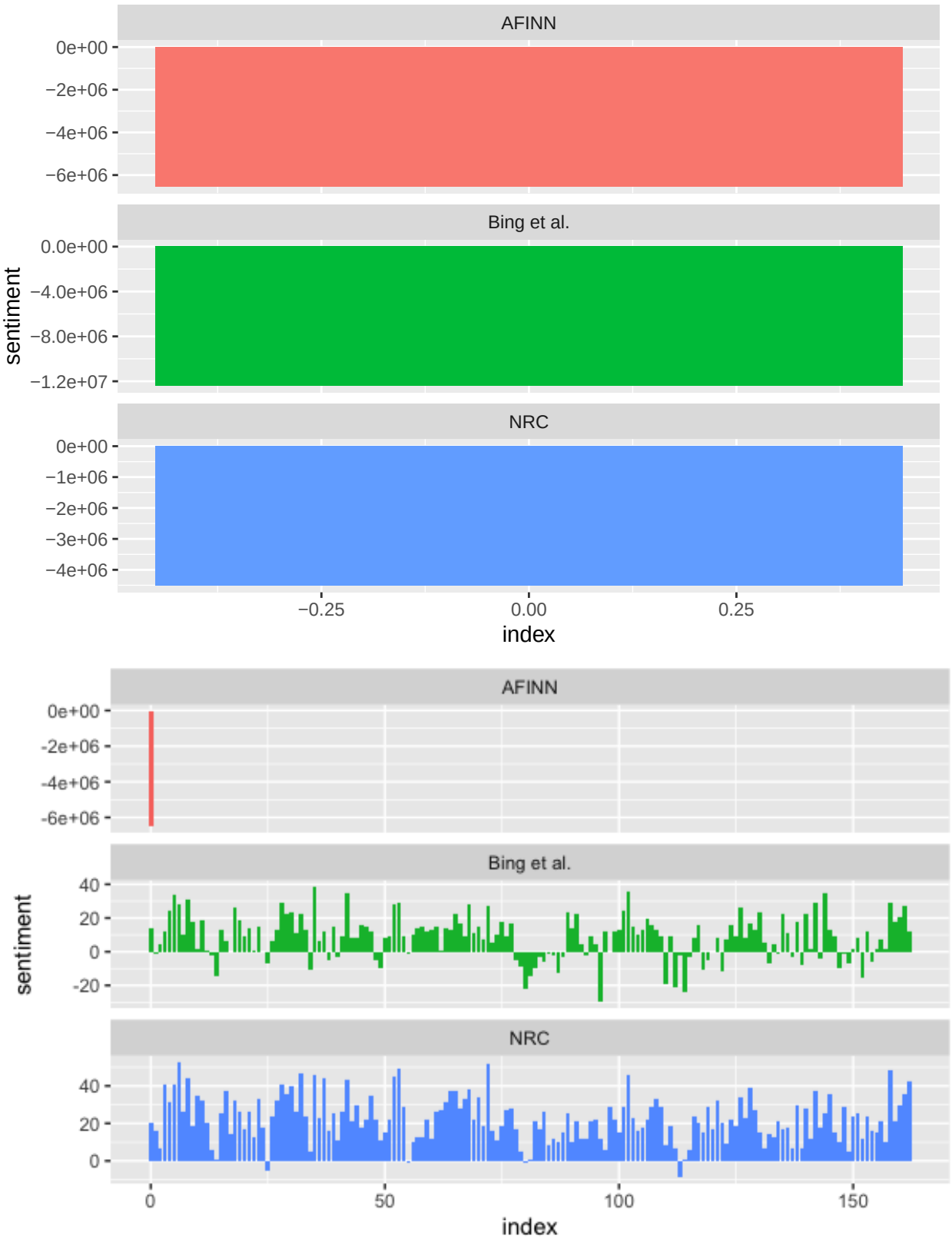
```
## # A tibble: 1 x 3
##   index sentiment method
##   <dbl>     <int> <chr>
## 1     0      6786 bing
```

Let's now use the cross_join() from dplyr to compare the following three sentiment dictionaries (AFINN, Bing et al,NRC) based on their opinion analysis of hemingway's novel

```
library (dplyr)
linenumber <-1

afinn <- the_sun_also_rises %>%
  cross_join (get_sentiments("afinn")) %>%
  group_by(index = linenumber %/% 80) %>%
  summarise(sentiment = sum(value)) %>%
  mutate(method = "AFINN")

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

Binding and visualizing the three methods

```
(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"))
```



We see a big difference in the behavior of the three lexicons. Bing provides a somehow balanced sentiment analysis of Hemingway's novel, while NRC's opinion account of the book is mostly positive and Afinn's is

restricted to one bar.

Conclusion

Like the texbook examples, our exploration of the afinn, bing et al. and nrc reveals that the three different lexicons for calculating sentiment give results that are different in an absolute sense. Common patterns are nonetheless visible especially between the Bing et al. and nrc lexicons as both dictionaries agree roughly on the overall trends in the sentiment through a narrative arc. [1]

[1] I discovered the syuzhet lexicon while working on this assignment. syuzhet](https://cran.r-project.org/web/packages/syuzhet/vignettes/syuzhet-vignette.html) It is worth exploring.