Week10_Assignment_Chunjie_Nan

Chunjie Nan

10/30/2021

Rererence#

Silge, J., & Robinson, D. (2017). Text mining with R: A tidy approach. O'Reilly Media.

1. Textbook Code

2.2 Sentiment analysis with inner join

```
get_sentiments("afinn")      # value from -5 to 5
```

```
## # 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
## # ... with 2,467 more rows
```

```
get_sentiments("bing")       # negative and positive
```

```
## # 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
## # ... with 6,776 more rows
```

```
get_sentiments("nrc")        # emotions etc
```

```
## # A tibble: 13,875 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
## # ... with 13,865 more rows
```

```
tidy_books <- austen_books() %>%    #from austen book
  group_by(book) %>%
  mutate(linenumber = row_number(), #setting line number
         chapter = cumsum(str_detect(text, regex("^chapter [\\divxlc]",
                                                 ignore_case = TRUE)))) %>%  # detect chapters
  ungroup() %>%
  unnest_tokens(word, text)         #unnest token by word

nrc_joy <- get_sentiments("nrc") %>%#using nrc method
  filter(sentiment == "joy")        #find out the word sentiment equals to joy

tidy_books %>%
  filter(book == "Emma") %>%        #from austen books get the book named Emma
  inner_join(nrc_joy) %>%           #apply the joy sentiment in nrc
  count(word, sort = TRUE) %>%
  head()
```

```
## Joining, by = "word"
```

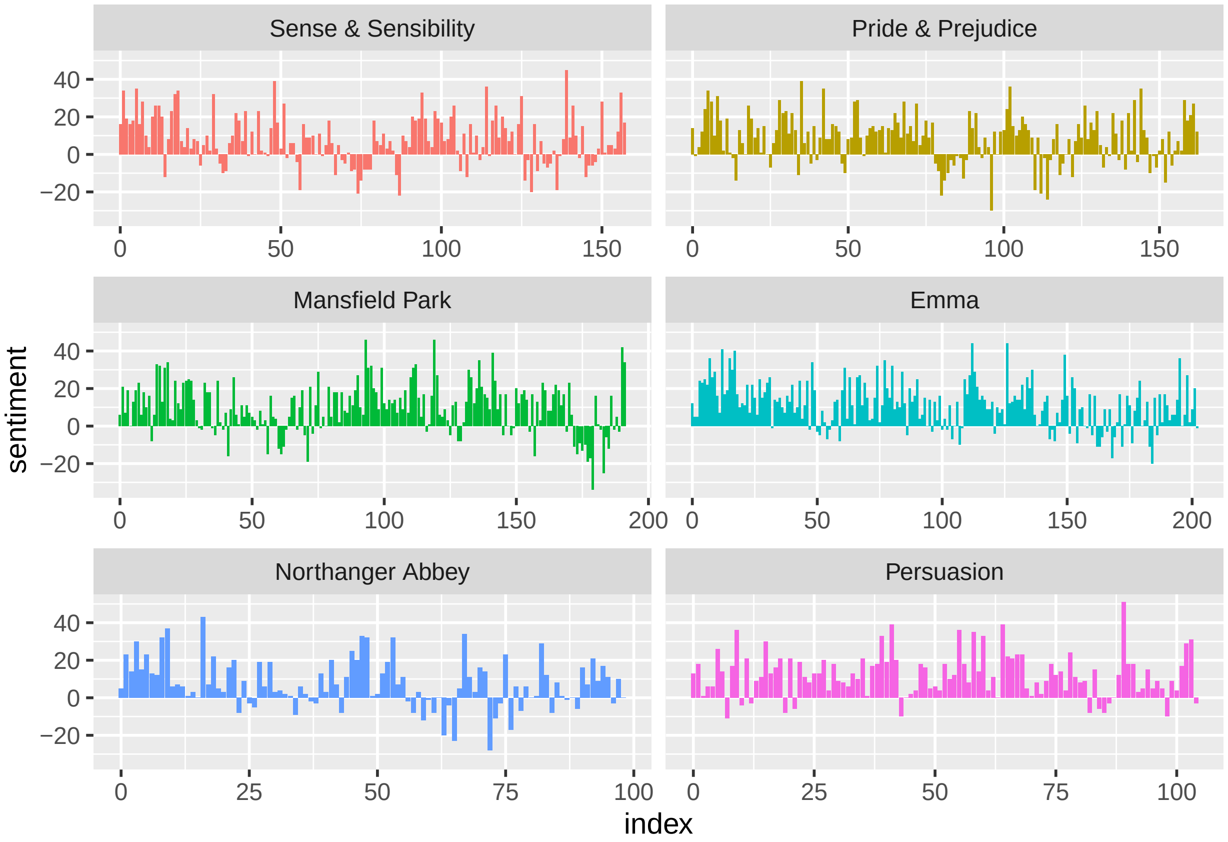
```
## # A tibble: 6 x 2
##   word       n
##   <chr>  <int>
## 1 good     359
## 2 friend   166
## 3 hope     143
## 4 happy    125
## 5 love     117
## 6 deal      92
```

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

```
## Joining, by = "word"
```

```
#jane_austen_sentiment
```

```
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%>%
  head()
```

```
## # A tibble: 6 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
```

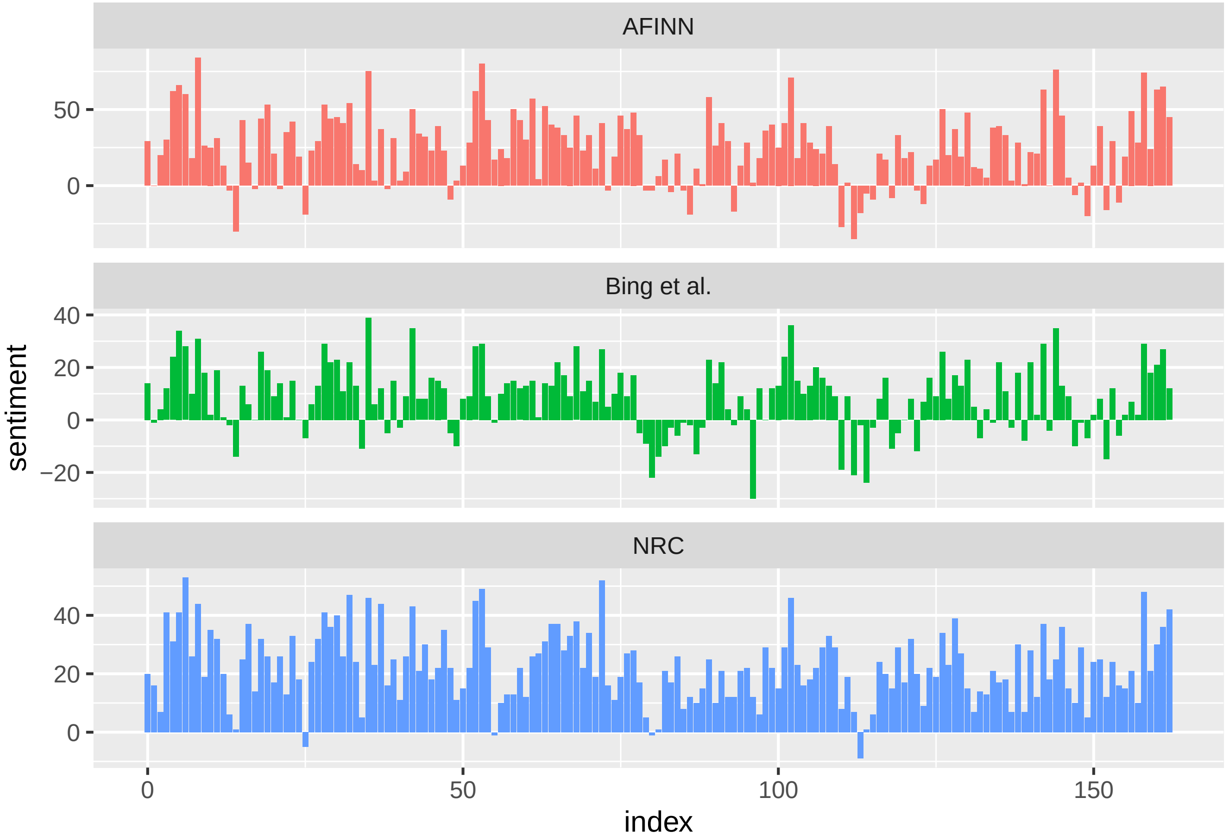
```
afinn <- pride_prejudice %>%
  inner_join(get_sentiments("afinn")) %>%
  group_by(index = linenumber %/% 80) %>%
  summarise(sentiment = sum(value)) %>%
  mutate(method = "AFINN")
```

```
## Joining, 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) %>%
  spread(sentiment, n, fill = 0) %>%
  mutate(sentiment = positive - negative)
```

```
## Joining, by = "word"
## Joining, by = "word"
```

```
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   3318
## 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()
```

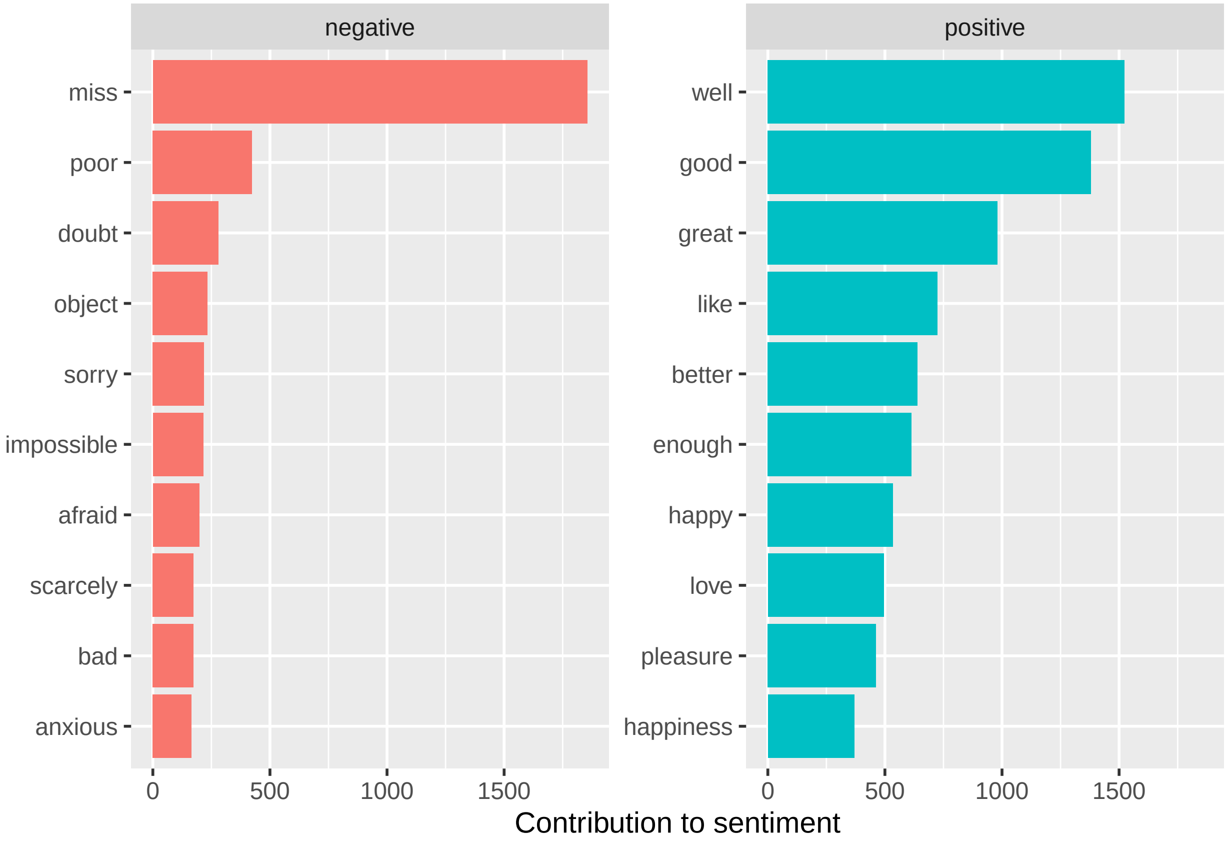
```
## Joining, by = "word"
```

```
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
## # ... with 2,575 more rows
```

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

```
## Selecting by n
```



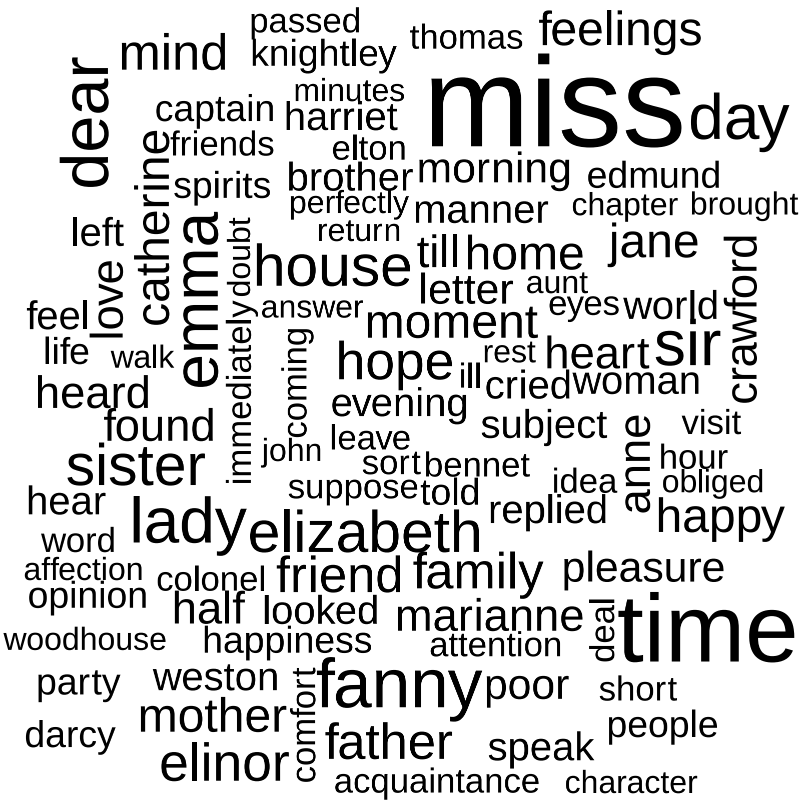
```
custom_stop_words <- bind_rows(tibble(word = c("miss"),
                                      lexicon = c("custom")),
                               stop_words)

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

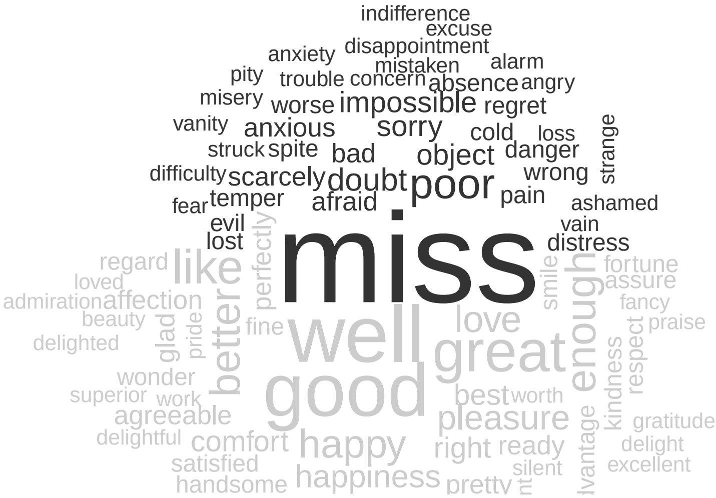
```
tidy_books %>%
  anti_join(stop_words) %>%
  count(word) %>%
  with(wordcloud(word, n, max.words = 100))
```

```
## Joining, by = "word"
```



```r
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, by = "word"
```



2.6 Looking at units beyond just words

```
PandP_sentences <- tibble(text = prideprejudice) %>%
  unnest_tokens(sentence, text, token = "sentences")
PandP_sentences$sentence[2]
```

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

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

```
## `summarise()` has grouped output by 'book'. You can override using the `.groups` argument.
```

```
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) %>%
  top_n(1) %>%
  ungroup()
```

```
## Joining, by = "word"
## `summarise()` has grouped output by 'book'. You can override using the `.groups` argument.
```

```
## Selecting by ratio
```

```
## # 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
```

My Own Choose from The Harry Potter Book - Half Blood Price.

Import the first

```
library(devtools)
```

```
## Loading required package: usethis
```

```
install_github("bradleyboehmke/harrypotter")
```

```
## Skipping install of 'harrypotter' from a github remote, the SHA1 (51f71461) has not changed since la
##   Use `force = TRUE` to force installation
```

```
library(harrypotter)
Title<- c("Half Blood Price")
Book<-list(half_blood_prince)
HBP <- tibble()

for(i in seq_along(Title)) {

        clean <- tibble(chapter = seq_along(Book[[i]]),
                        text = Book[[i]]) %>%
             unnest_tokens(word, text) %>%
             mutate(book = Title[i]) %>%
             select(book, everything())

            HBP<- rbind(HBP, clean)
}

HBP$book <- factor(HBP$book, levels = rev(Title))
head(HBP)
```

```
## # A tibble: 6 x 3
##   book             chapter word
##   <fct>              <int> <chr>
## 1 Half Blood Price       1 it
## 2 Half Blood Price       1 was
## 3 Half Blood Price       1 nearing
## 4 Half Blood Price       1 midnight
## 5 Half Blood Price       1 and
## 6 Half Blood Price       1 the
```

```
tail(HBP)
```

```
## # A tibble: 6 x 3
##   book             chapter word
##   <fct>              <int> <chr>
## 1 Half Blood Price      30 to
## 2 Half Blood Price      30 enjoy
## 3 Half Blood Price      30 with
## 4 Half Blood Price      30 ron
## 5 Half Blood Price      30 and
## 6 Half Blood Price      30 hermione
```

The Book Half Blood Price has total 30 Chapters.

Use Loughran as the new sentiment.

```
HBP %>%
        right_join(get_sentiments("loughran")) %>%
        filter(!is.na(sentiment)) %>%
        count(sentiment, sort = TRUE)
```

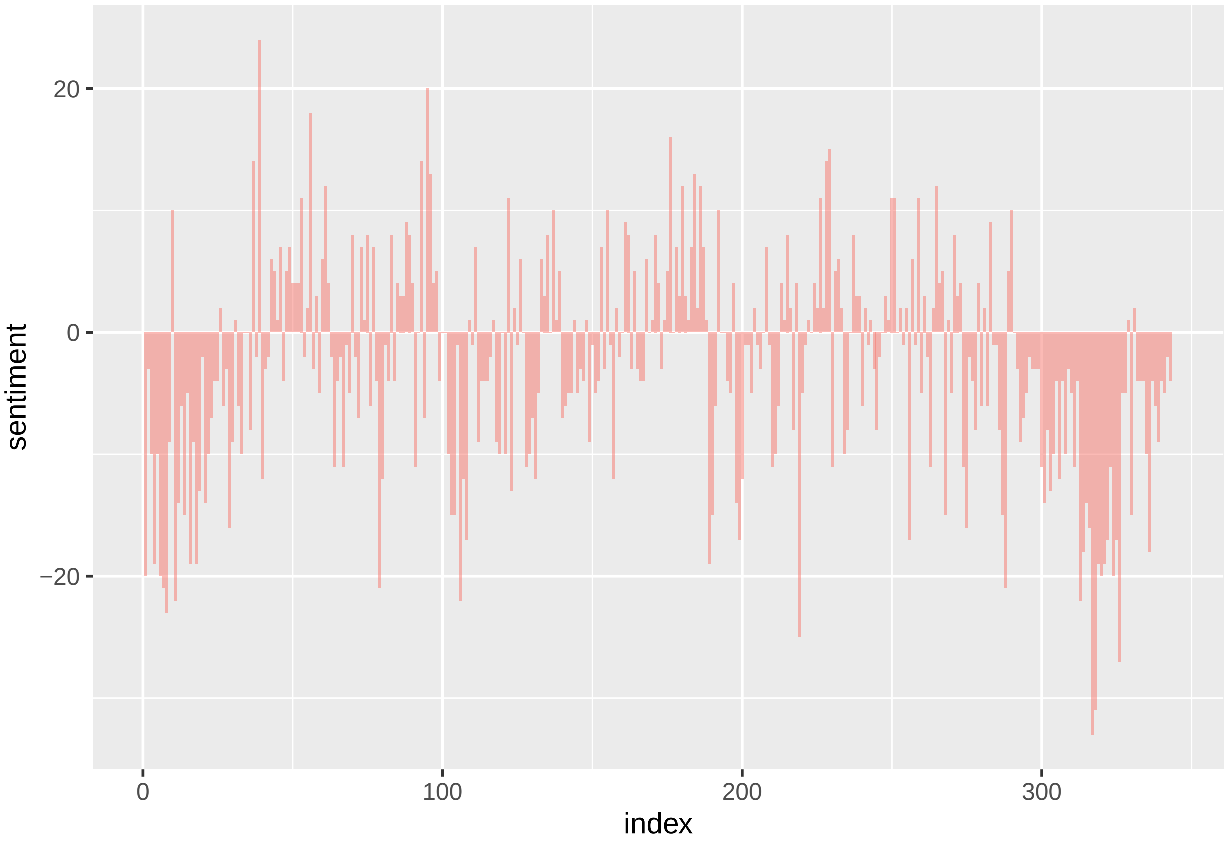
```
## Joining, by = "word"
```

```
## # A tibble: 6 x 2
##   sentiment        n
##   <chr>        <int>
## 1 negative      4289
## 2 uncertainty   1709
## 3 positive      1481
## 4 litigious     1034
## 5 constraining   272
## 6 superfluous     56
```

Plot Bing Sentiment

```
HBP %>%
        group_by(book) %>%
        mutate(word_count = 1:n(),
               index = word_count %/% 500 + 1) %>%
        inner_join(get_sentiments("bing")) %>%
        count(book, index = index , sentiment) %>%
        ungroup() %>%
        spread(sentiment, n, fill = 0) %>%
        mutate(sentiment = positive - negative,
               book = factor(book, levels = Title)) %>%
        ggplot(aes(index, sentiment, fill = book)) +
          geom_bar(alpha = 0.5, stat = "identity", show.legend = FALSE)
```

```
## Joining, by = "word"
```



The AFFIN and Loughran

```
AFINN <- HBP %>%
        group_by(book) %>%
        mutate(word_count = 1:n(),
               index = word_count %/% 500 + 1) %>%
        inner_join(get_sentiments("afinn")) %>%
        group_by(book, index) %>%
        summarise(sentiment = sum(value)) %>%
        mutate(method = "AFINN")
```

```
## Joining, by = "word"
```

```
## `summarise()` has grouped output by 'book'. You can override using the `.groups` argument.
```

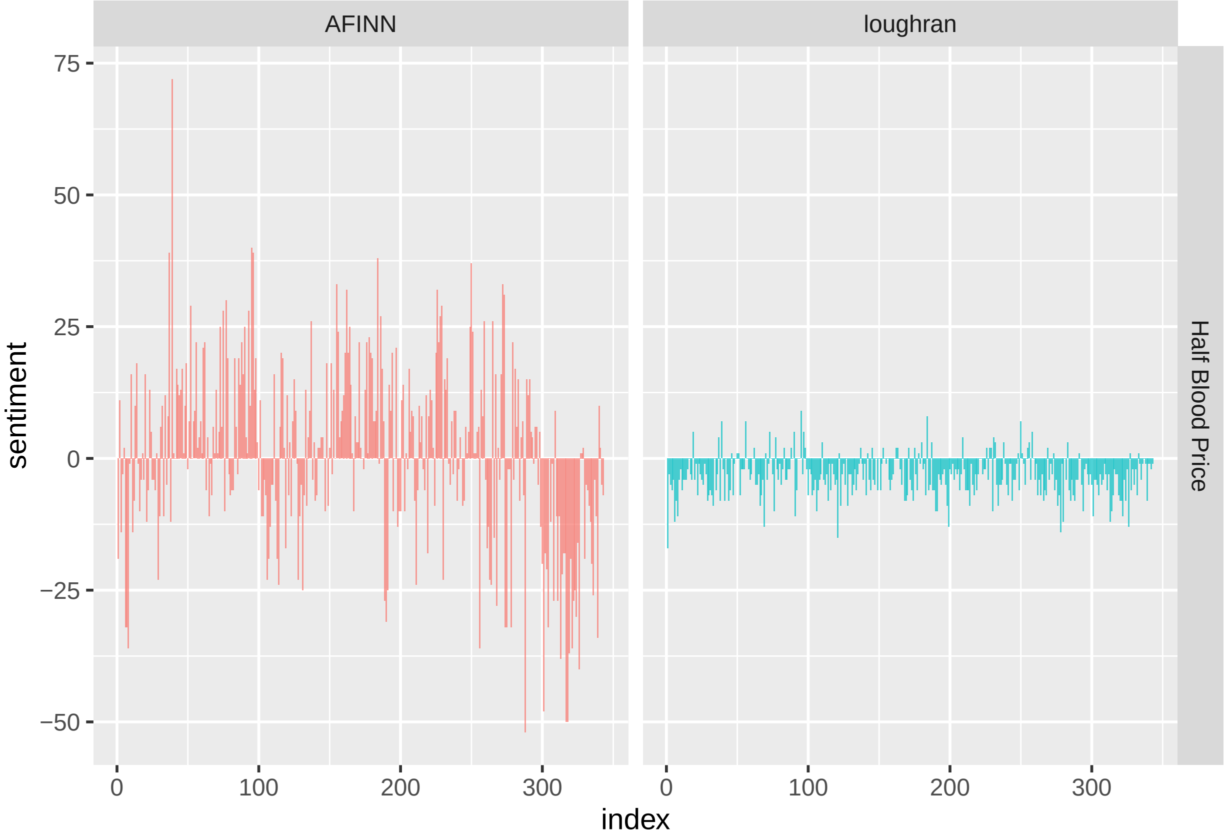
```
AFINN
```

```
## # A tibble: 343 x 4
## # Groups:   book [1]
##    book             index sentiment method
##    <fct>            <dbl>     <dbl> <chr>
##  1 Half Blood Price     1       -19 AFINN
##  2 Half Blood Price     2        11 AFINN
##  3 Half Blood Price     3       -14 AFINN
##  4 Half Blood Price     4        -3 AFINN
##  5 Half Blood Price     5         2 AFINN
##  6 Half Blood Price     6       -32 AFINN
##  7 Half Blood Price     7       -32 AFINN
##  8 Half Blood Price     8       -36 AFINN
##  9 Half Blood Price     9        -1 AFINN
## 10 Half Blood Price    10        16 AFINN
## # ... with 333 more rows
```

```
LOUG <- HBP %>%
            group_by(book) %>%
            mutate(word_count = 1:n(),
                  index = word_count %/% 500 + 1) %>%
            inner_join(get_sentiments("loughran")) %>%
            mutate(method = "loughran") %>%
count(book, method, index = index , sentiment) %>%
        ungroup() %>%
        spread(sentiment, n, fill = 0) %>%
        mutate(sentiment = positive - negative) %>%
        select(book, index, method, sentiment)
```

```
## Joining, by = "word"
```

```
bind_rows(LOUG, AFINN) %>%
        ungroup() %>%
        mutate(book = factor(book, levels = Title)) %>%
  ggplot(aes(index, sentiment, fill = method)) +
  geom_bar(alpha = 0.7, stat = "identity", show.legend = FALSE) +
  facet_grid(book ~ method)
```



In conclusion, compare the Afinn and Loughran, the AFinn looks more volatile than Loughran method due to most of the loughran sentiment in in between +12.5 and - 12.5. Also, the Analysis tells that loughran distributed more negative sentiment value than the Afinn. It looks Afinn outperformed the loughran in this case.