Chapter2. Sentiment analysis with tidy data

Geonwoo Ban

March 17th, 2021

One way to analyze the sentiment of a text is to consider the text as a combination of its individual words and the sentiment content of the whole text as the sum of the sentiment content of the individual words.

The **sentiments** datasets

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

- `AFINN` lexicon : A score between -5 and 5 is given to the word, a negative number represents a negative sentiment and a positive number represents a positive sentiment.

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

- `bing` lexicon : Categorize words into positive and negative categories according to binary format.

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

```
## # A tibble: 13,901 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 13,891 more rows
```

- `NRC` lexicon : Categorize words in a binary fashion (“yes”/“no”) into categories of positive, negative, anger, anticipation, disgust, fear, joy, sadness, surprise, and trust.

Sentiment analysis with inner join

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

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

First, let's use the NRC lexicon and `filter()` for the joy words.

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

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

```
## # A tibble: 303 x 2
##    word        n
##    <chr>   <int>
##  1 good      359
##  2 young     192
##  3 friend    166
##  4 hope      143
##  5 happy     125
##  6 love      117
##  7 deal       92
##  8 found      92
##  9 present    89
## 10 kind       82
## # ... with 293 more rows
```

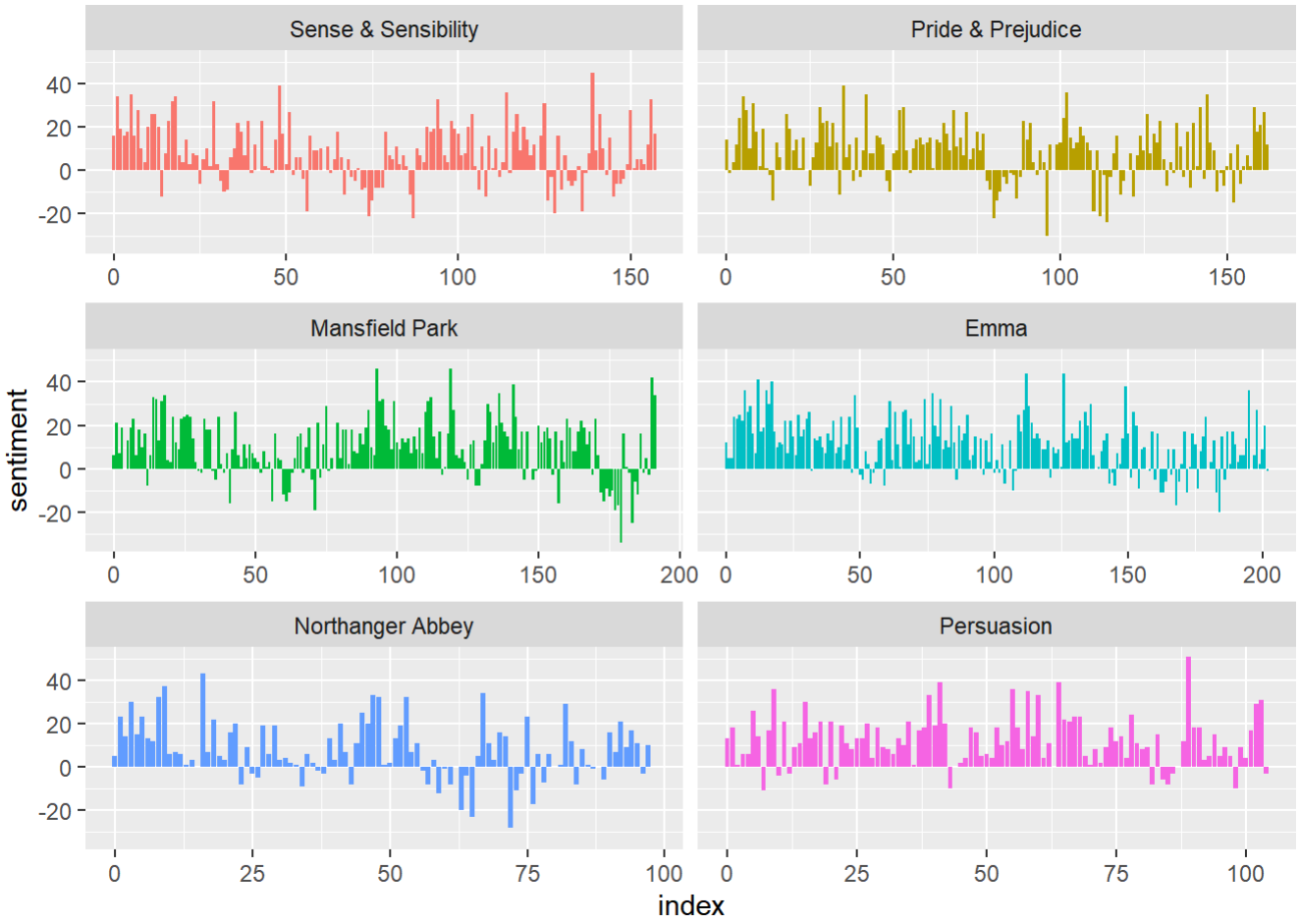
- Emma라는 책에서 "joy"를 가장 흔하게 나타내는 단어는 hope, love, friend 등으로 긍정적이고 행복감을 나타내는 단어가 많이 쓰였다.
- 이런 식으로 sentiment를 분석하는 방식 대신에 각 소설에서 sentiment가 전반적으로 어떻게 변하는지 확인할 수도 있다.
- 먼저 Bing lexicon을 통해 각 단어에 대한 정서 점수를 찾을 수 있다.
- 다음으로 각 도서에서 정의한 section별로 긍정 단어와 부정 단어가 몇 개인지를 세어본다.
- 텍스트의 80줄을 1개의 section으로 정의하여 시행할 것이다.
- 각 section별 positive sentiment - negative sentiment를 계산하여 section 별 sentiment score를 구하여 section이 지남에 따른 sentiment score에 대한 그래프를 그릴 수 있다.

```
library(tidyr)

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)

library(ggplot2)

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



- Jane auston 작가의 소설들은 초반부와 후반부에 긍정적인 단어를 많이 사용하는 것으로 알 수 있으며,
- 결말 직전에는 부정적인 단어를 더 많이 사용하여 대비적인 느낌을 사용하는 것으로 보인다.

Camparing the three sentiment dictionaries

Pride & Prejudice 책을 가지고 세 lexicon을 비교해보자.

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

pride_prejudice
```

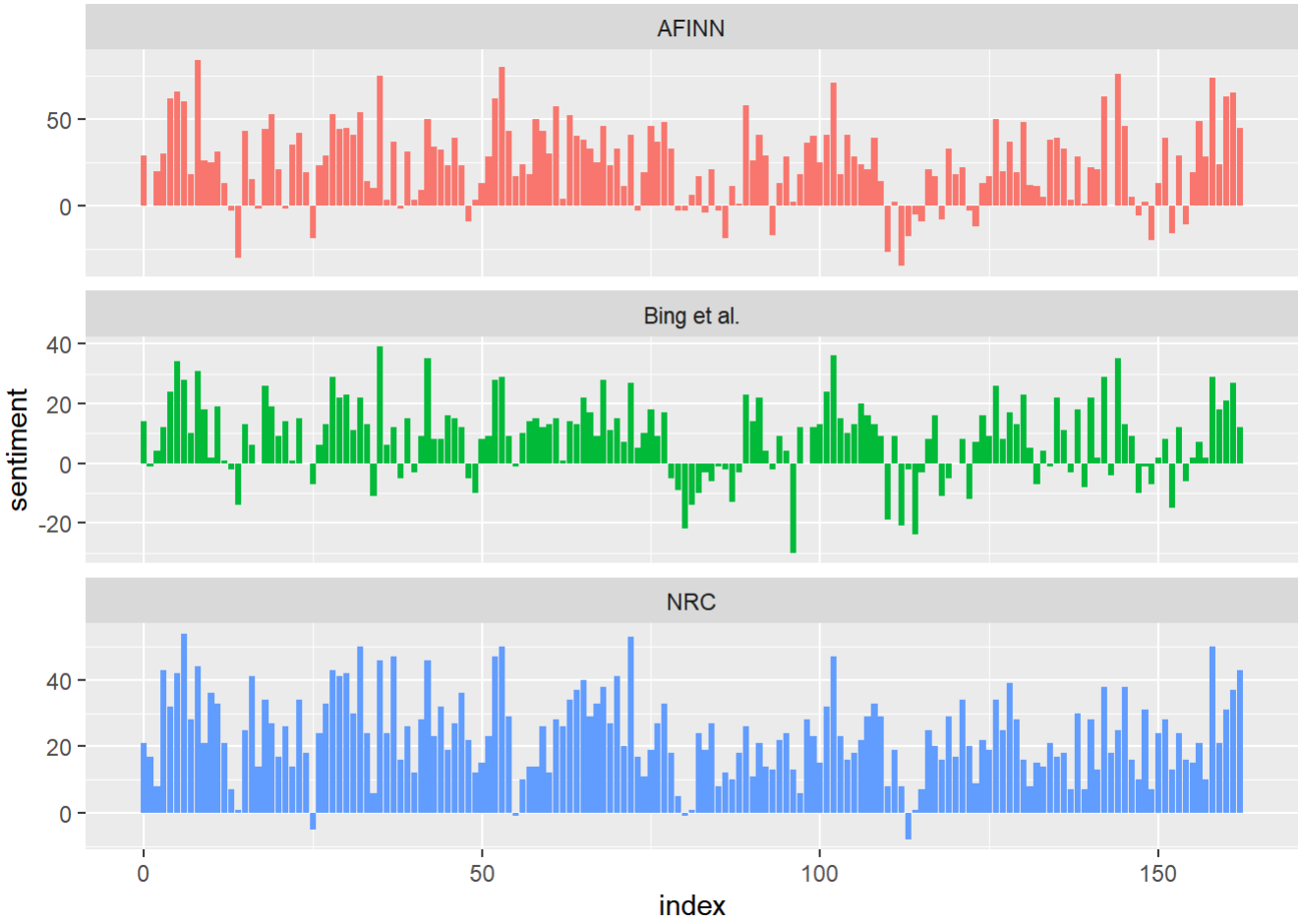
```
## # A tibble: 122,204 x 4
##    book              linenumber chapter word
##    <fct>                  <int>   <int> <chr>
##  1 Pride & Prejudice          1       0 pride
##  2 Pride & Prejudice          1       0 and
##  3 Pride & Prejudice          1       0 prejudice
##  4 Pride & Prejudice          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
## # ... with 122,194 more rows
```

```
afinn <- pride_prejudice %>%
  inner_join(get_sentiments("afinn")) %>%
  group_by(index = linenumber %/% 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 = linenumber %/% 80, sentiment) %>%
  spread(sentiment, n, fill = 0) %>%
  mutate(sentiment = positive - negative)
```

- `AFINN` 의 경우엔 단어의 부정과 긍정에 대해 -5에서 5사이의 연속적인 정수값을 주어 계산하였다.
- `Bing` 과 `NRC` 의 lexicon은 단어를 부정과 긍정, 이항으로 표현하였다.

```
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")
```



- 세 lexicon으로 소설을 분석해본 결과, 전반적으로 특정 부분에서 값이 크거나 값이 작아지며 그래프의 변화 또한 유사함을 볼 수있다.
- `Afinn` lexicon을 사용하였을 때, 점수의 절댓값이 가장 크고, positive 값이 큰 것을 확인 할 수 있다.
- `Bing` lexicon을 사용하였을 때는 부정과 긍정의 교차되는 부분에서 값의 변동이 큰 것을 볼 수 있다.
- 마지막으로 `NRC` lexicon을 사용하였을 때는 앞서 두 lexicon들 보다 상대적으로 음의 값들이 적어서 텍스트가 대부분 긍정이라고 처리하는 것처럼 보이지만 변동부분에 있어서는 두 방법과 비슷하다고 볼 수 있다.
- 이를 이해하기 위해 `NRC` lexicon에 몇 개의 positive 단어와 negative 단어가 있는지 확인해보자.

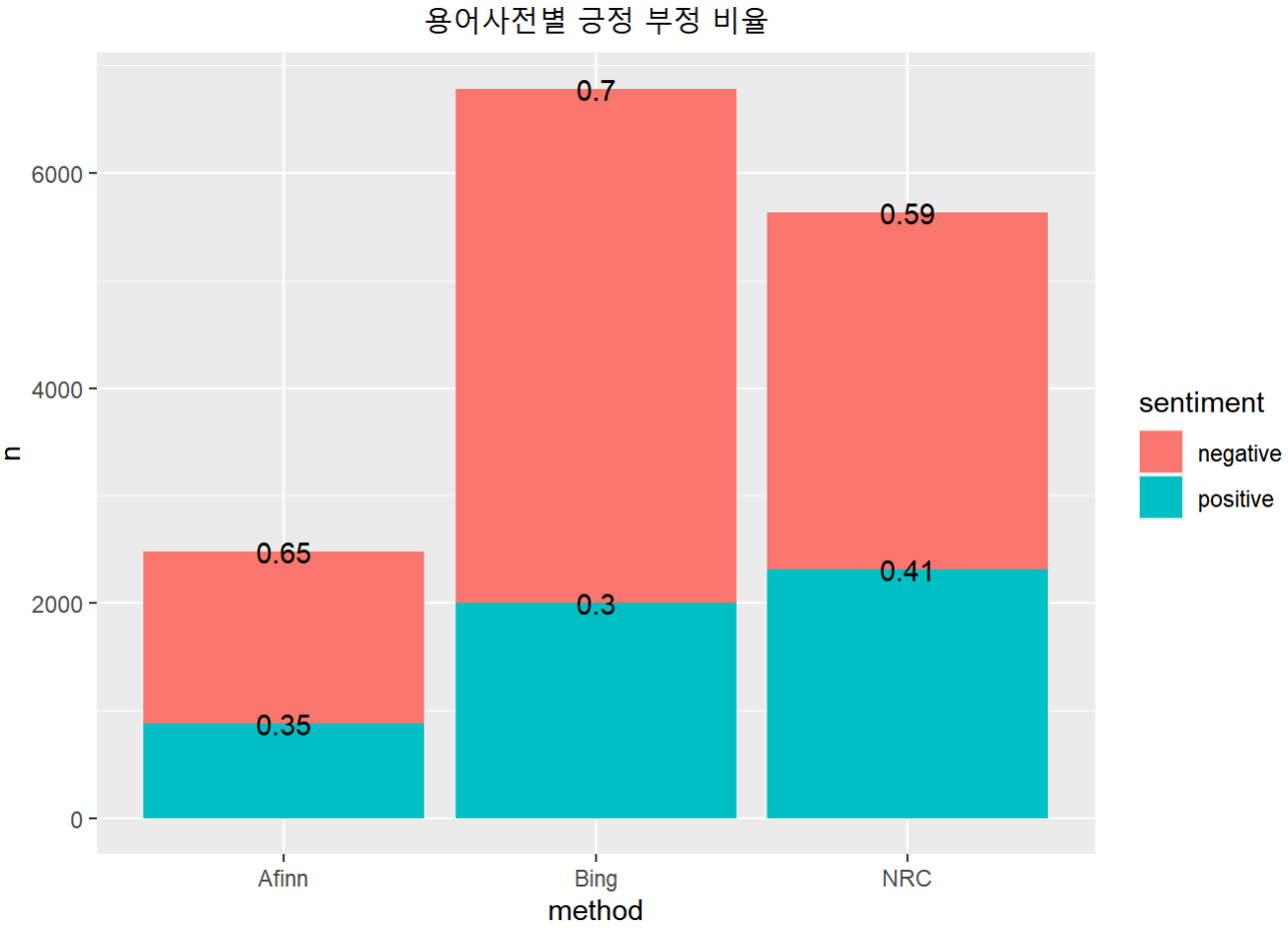
```
bind_rows(
  get_sentiments("afinn") %>% filter(value!=0) %>%
    mutate(sentiment=ifelse(value>=0, "positive", "negative")) %>%
    count(sentiment) %>% mutate(method="Afinn"),

  get_sentiments("bing") %>% count(sentiment) %>% mutate(method="Bing"),

  get_sentiments("nrc") %>% filter(sentiment %in% c("positive", "negative")) %>%
    count(sentiment) %>% mutate(method="NRC")) %>%

  group_by(method) %>%
  mutate(rate = round(n/sum(n),2)) %>%

  ggplot(aes(x=method, y=n, fill=sentiment)) + geom_bar(stat="identity")+
  geom_text(aes(label=rate), position = position_stack())+
  ggtitle("용어사전별 긍정 부정 비율")+ theme(plot.title = element_text(hjust=0.5))
```



- Bing lexicon의 전체 중 부정 단어 비율을 보면 다른 lexicon보다 더 높은 것을 알 수 있다.
- NRC 의 경우엔 긍정 단어 비율이 가장 높음을 알 수있다.
- 이러한 lexicon 별 긍정 부정 비율이 위의 sentiment 분석에 있어 영향을 미쳤을 것으로 파악된다.
- 전체적인 소설의 흐름과 각 lexicon을 사용해 sentiment 분석을 하여 그려본 그래프와는 연관성이 있을것이라고 예상되나,
- 정확한 긍정적인 section과 부정적인 section을 구분하기 위해서는 lexicon 선택을 신중하게 하여야한다고 볼 수 있다.

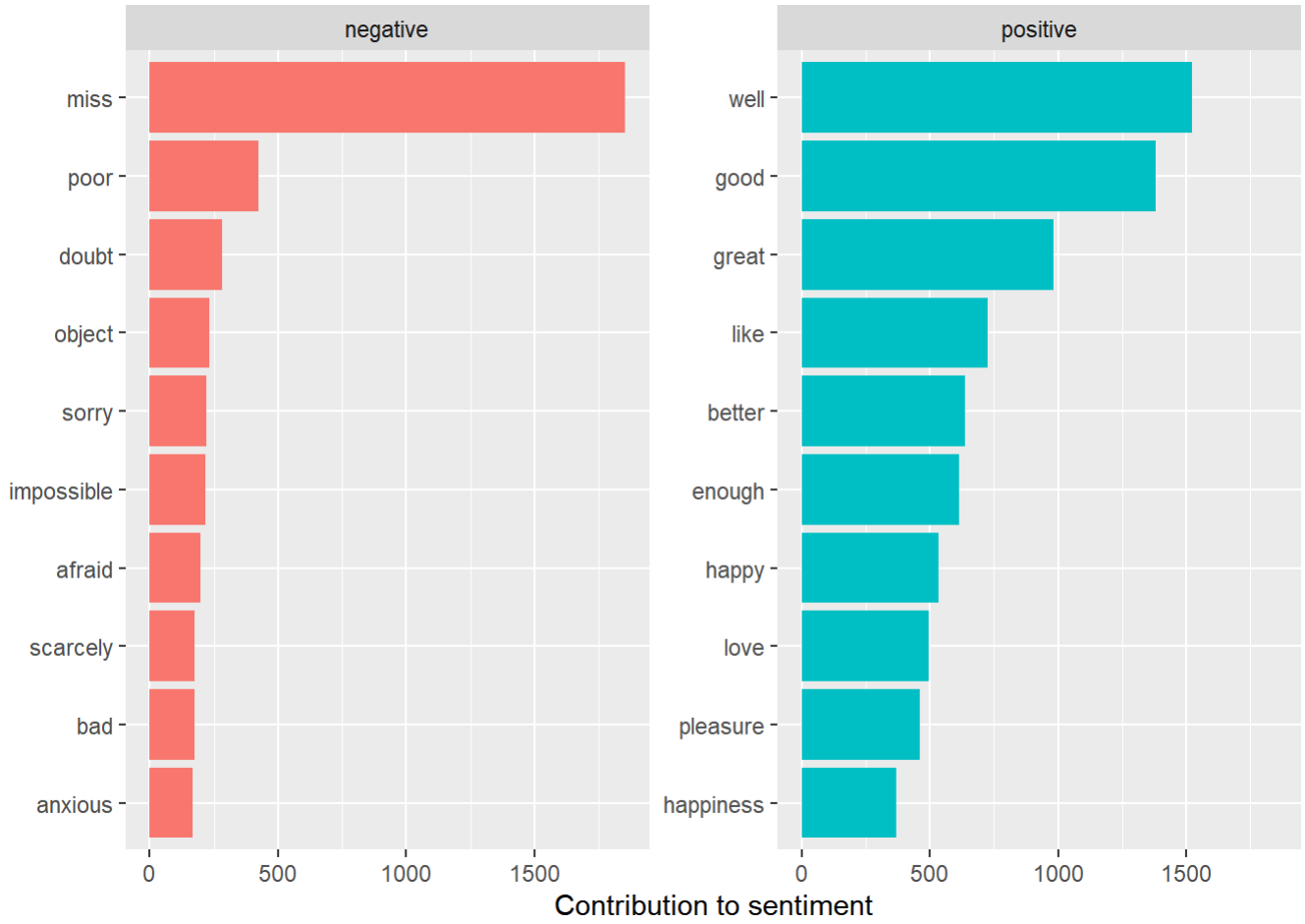
Most common positive and negative words

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

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(n, word, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(x = "Contribution to sentiment",
       y = NULL)
```



- Jane austen의 작품에서의 Bing lexicon을 가지고 sentiment 분석을 해보았을 때, 가장 많이 사용된 부정적인 단어와 긍정적인 단어를 나타낸 그림이다.
- 가장 많이 사용된 부정적인 단어 그래프를 보면, 'miss'라는 단어가 가장 많이 사용됨을 알 수 있다.
- 하지만 miss라는 것은 Bing lexicon에서는 부정적인 단어라고 분류가 되어있지만, 작품에서는 젊은 미혼 여성을 부르는 호칭으로 사용되어 잘못 분류된 것으로 볼 수 있다.

```
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
```

- 이렇게 제거되어야할 단어들을 stop words로 custom할 수 있다.

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("red", "blue"),
                   max.words = 100)
```

Looking at units beyond just words

단어 단위로 토큰화하여 분석을 할 수 있는 반면에, 때로는 단위를 달리하여 텍스트를 보는 것이 분석에 유용하기도 하고 필요하기도 하다. 예를 들어 “I’am not having a good day”라는 말이 “joy”가 아닌 “sad”를 나타내는 문장을 이해하고자 하는 것이다.

- 이를 시행하기위해 먼저 단어 단위가 아닌 Chapter 단위로 토큰화 시키기 위해 정규표현식 패턴을 사용하여 분할해보자.
- 이후 책 별로 분할 한 Chapter의 갯수를 파악해보자.

```
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
```

- 각 책별로 총 Chapter 수를 알 수 있다.
- Chapter별로 나누어진 데이터셋을 가지고 소설에서 가장 부정적인 Chapter가 어떤 것인지와 같은 질문에 답을 할 수 있다.

```
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) %>%
  top_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
```

- *Sense & Sensibility* 라는 책에서의 43장에서는 소설속 등장인물 중 한명이 부상때문에 위급한 상황을 묘사하며,
- *Pride & Prejudice* 책 중 34장에서는 등장인물이 청혼을 매우 서툴게하여 좋은 상황을 만들지 못했으며,
- 나머지 책들의 chapter 모두 전반적으로 부정적인 내용임을 확인 할 수 있었다.

Summary

- sentiment analysis를 통해 텍스트를 감정으로 표현할 수 있고, 이를 표현함으로써 텍스트에 대한 전반적인 이해를 도와주었다.
- lexicon에 있어서 분석 결과가 달라지는 것이 조금 신경써야하지만, lexicon을 선택만 잘 한다면 텍스트에 대해 분석을 더 자세하게 할 수 있을 것으로 보인다.
- wordcloud를 통해 사용빈도가 많았던 단어들을 시각화 할 수 있었으며, 이를 긍정과 부정으로 그룹을 나누어 표현도 해보았다.
- 이러한 wordcloud를 통해 주요 단어들을 파악하였지만 책이라는 어떤 텍스트 전체를 이해하는데에는 어려움이 있어 이를 문장 또는 chapter 단위로 쪼개어 sentiment analysis를 해보았을 때, 책 전체에 대한 특징을 파악하는데에 도움이 되었다.