Assignment 10

Alice Ding

Overview

To start with, I'll be copying over *Text Mining with R, Chapter 2's* code base in order to perform sentiment analysis on something of my choice.

Text Mining with R, Chapter 2

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

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(stringr)
library(tidytext)

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)

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

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

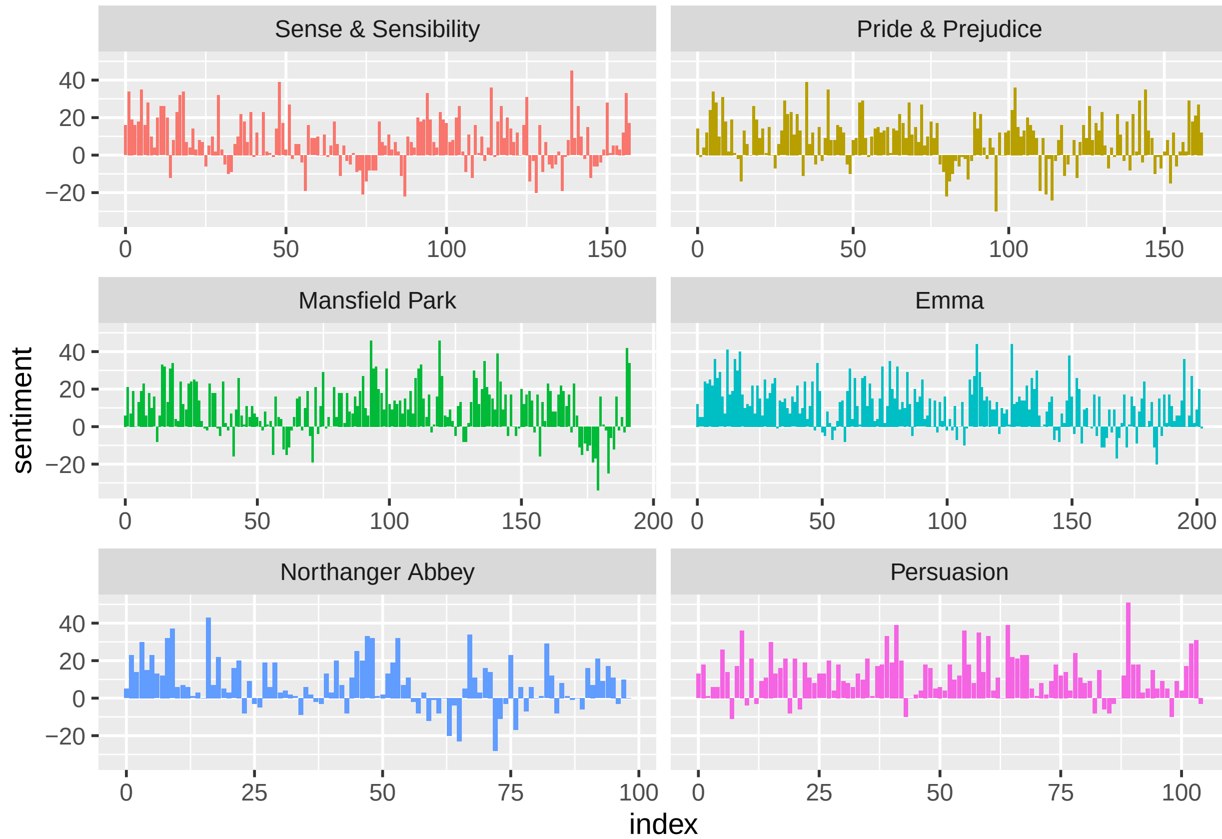
```
## Joining, by = "word"
```

```
## # 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
## # ... with 291 more rows
```

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

```r
library(ggplot2)

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



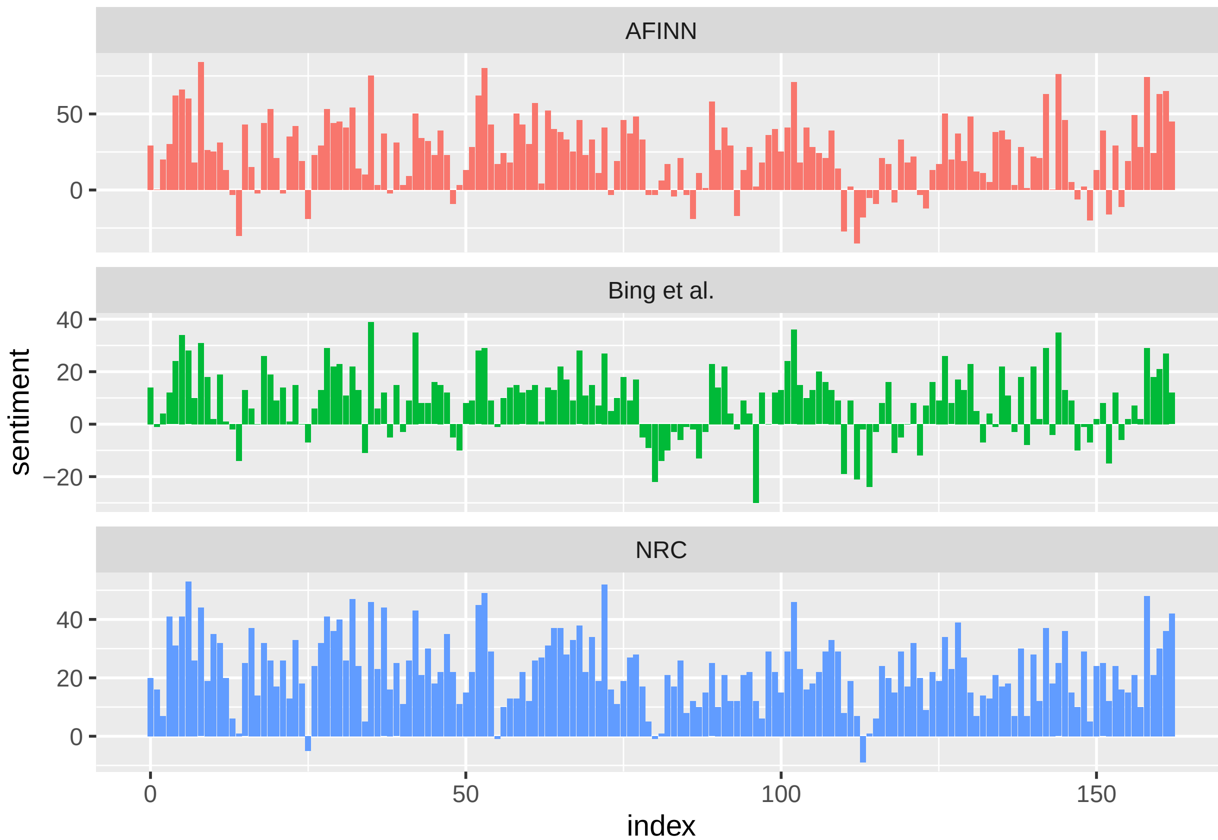

```
pride_prejudice <- tidy_books %>%
  filter(book == "Pride & Prejudice")

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) %>%
  pivot_wider(names_from = sentiment,
              values_from = n,
              values_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")
```



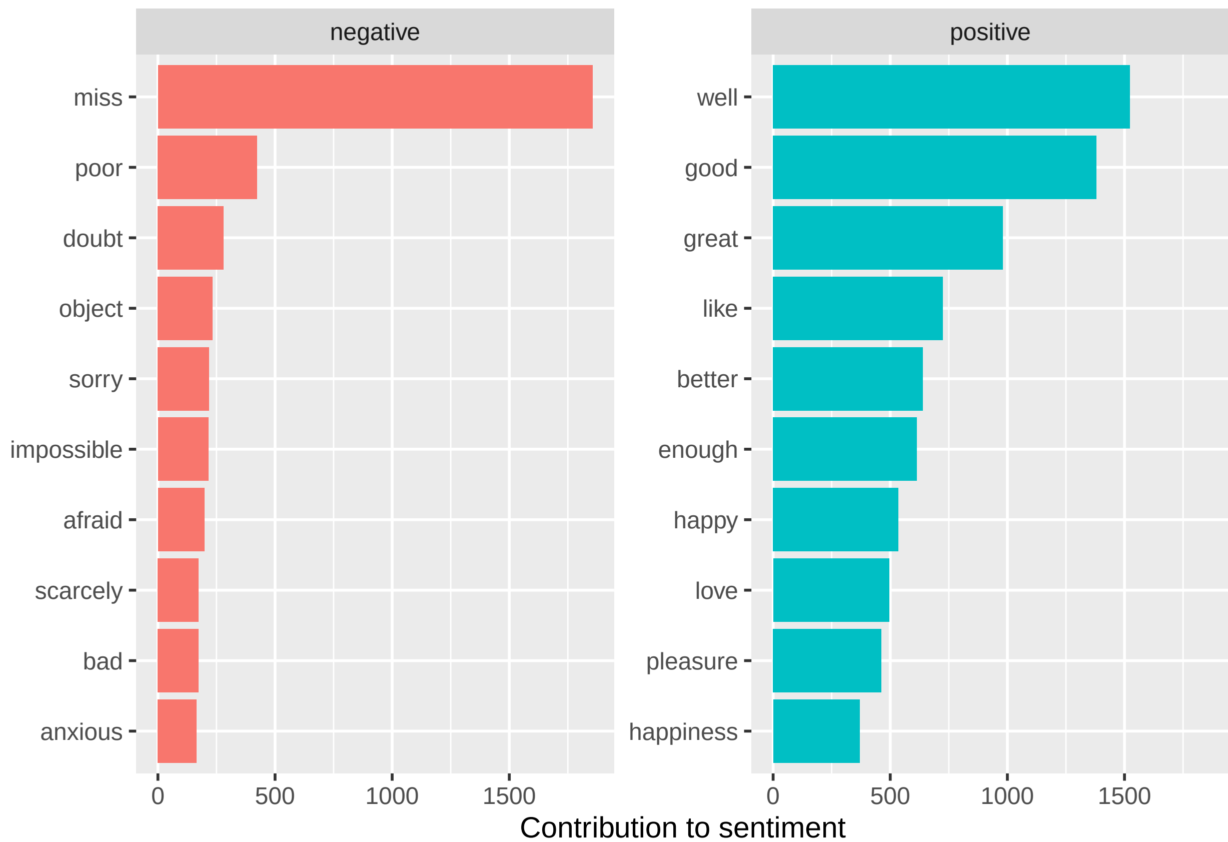

```
get_sentiments("nrc") %>%
  filter(sentiment %in% c("positive", "negative")) %>%
  count(sentiment)
```

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

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

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

```
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)

library(wordcloud)
```

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

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

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

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

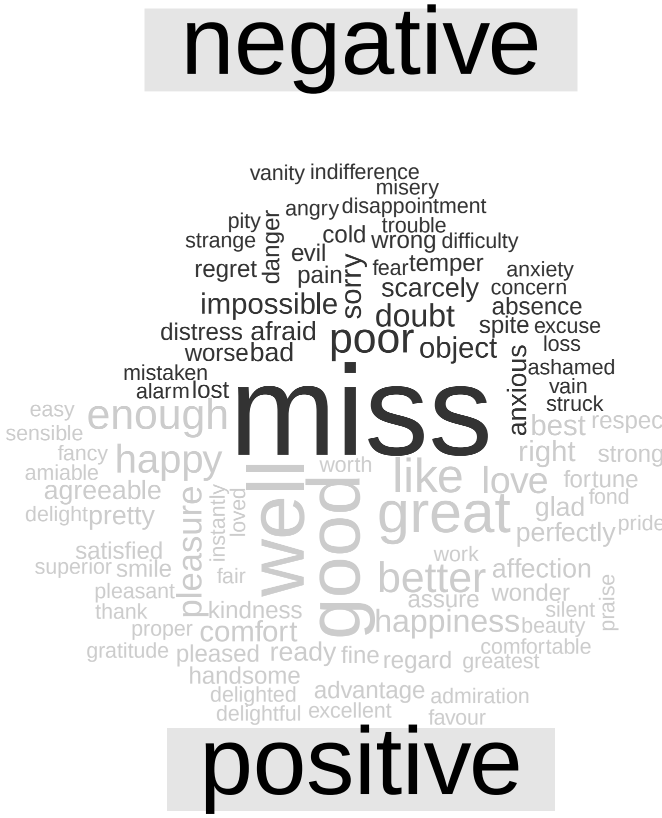

```r
library(reshape2)
```

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

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

```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"
```




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

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
```

```r
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) %>%
  slice_max(ratio, n = 1) %>%
  ungroup()
```

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

```
## # 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
```

Corpus of my Choosing: The Office

I've chosen to extend the assignment by analyzing the transcript from the TV show, *The Office*.

```
library(schrute)

glimpse(theoffice)
```

```
## Rows: 55,130
## Columns: 12
## $ index            <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16~
## $ season           <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,~
## $ episode          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,~
## $ episode_name     <chr> "Pilot", "Pilot", "Pilot", "Pilot", "Pilot", "Pilot",~
## $ director         <chr> "Ken Kwapis", "Ken Kwapis", "Ken Kwapis", "Ken Kwapis~
## $ writer           <chr> "Ricky Gervais;Stephen Merchant;Greg Daniels", "Ricky~
## $ character        <chr> "Michael", "Jim", "Michael", "Jim", "Michael", "Micha~
## $ text             <chr> "All right Jim. Your quarterlies look very good. How ~
## $ text_w_direction <chr> "All right Jim. Your quarterlies look very good. How ~
```

```
## $ imdb_rating      <dbl> 7.6, 7.6, 7.6, 7.6, 7.6, 7.6, 7.6, 7.6, 7.6, 7.6, 7.6~
## $ total_votes      <int> 3706, 3706, 3706, 3706, 3706, 3706, 3706, 3706, 3706,~
## $ air_date         <chr> "2005-03-24", "2005-03-24", "2005-03-24", "2005-03-24~
```

Each row seems to represent one line for one character in all episodes. Let's try analyzing lines by Michael Scott.

Positive Words

```r
tidy_office <- theoffice %>%
  group_by(character) %>%
  mutate(
    lines = row_number(),
    chapter = cumsum(str_detect(text,
                                regex("^chapter [\\divxlc]",
                                      ignore_case = TRUE)))) %>%
  ungroup() %>%
  unnest_tokens(word, text)

michael <- tidy_office |> filter(character == "Michael")

michael %>%
  inner_join(nrc_joy) %>%
  count(word, sort = TRUE)
```

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

```
## # A tibble: 282 x 2
##    word       n
##    <chr>  <int>
##  1 good     629
##  2 god      250
##  3 love     204
##  4 fun      128
##  5 kind      93
##  6 friend    82
##  7 happy     76
##  8 money     73
##  9 baby      72
## 10 pretty    52
## # ... with 272 more rows
```

Looks like he talks about god (not sure if this is used in a religious context or more like "oh my god"), love, fun, and friends.

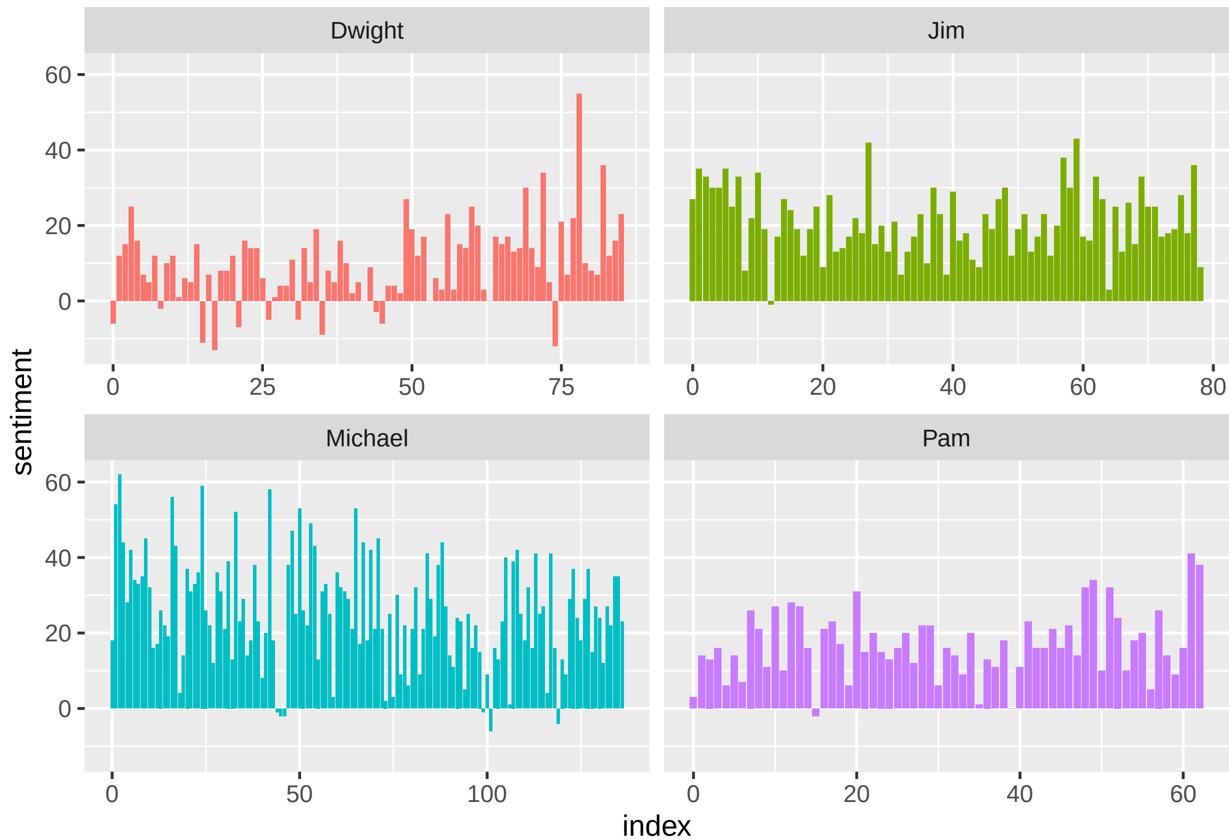
Positive and Negative Charts by Character

What if we look at a few characters and chart their lines?

```r
office_setiment <- tidy_office %>% filter(character %in% c("Michael", "Jim", "Pam", "Dwight")) %>%
  inner_join(get_sentiments("bing")) %>%
  count(character, index = lines %/% 80, sentiment) %>%
  pivot_wider(names_from = sentiment, values_from = n, values_fill = 0) %>%
  mutate(sentiment = positive - negative)
```

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

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



In general, it looks like Dwight has the most negative words, but even then it's not super negative. Jim and Pam both have pretty positive skewing dialogue while Michael does as well, albeit a few more negative words mixed in.

Different Sentiment Dictionaries

What does Michael's positive/negative distribution look like using three different dictionaries?

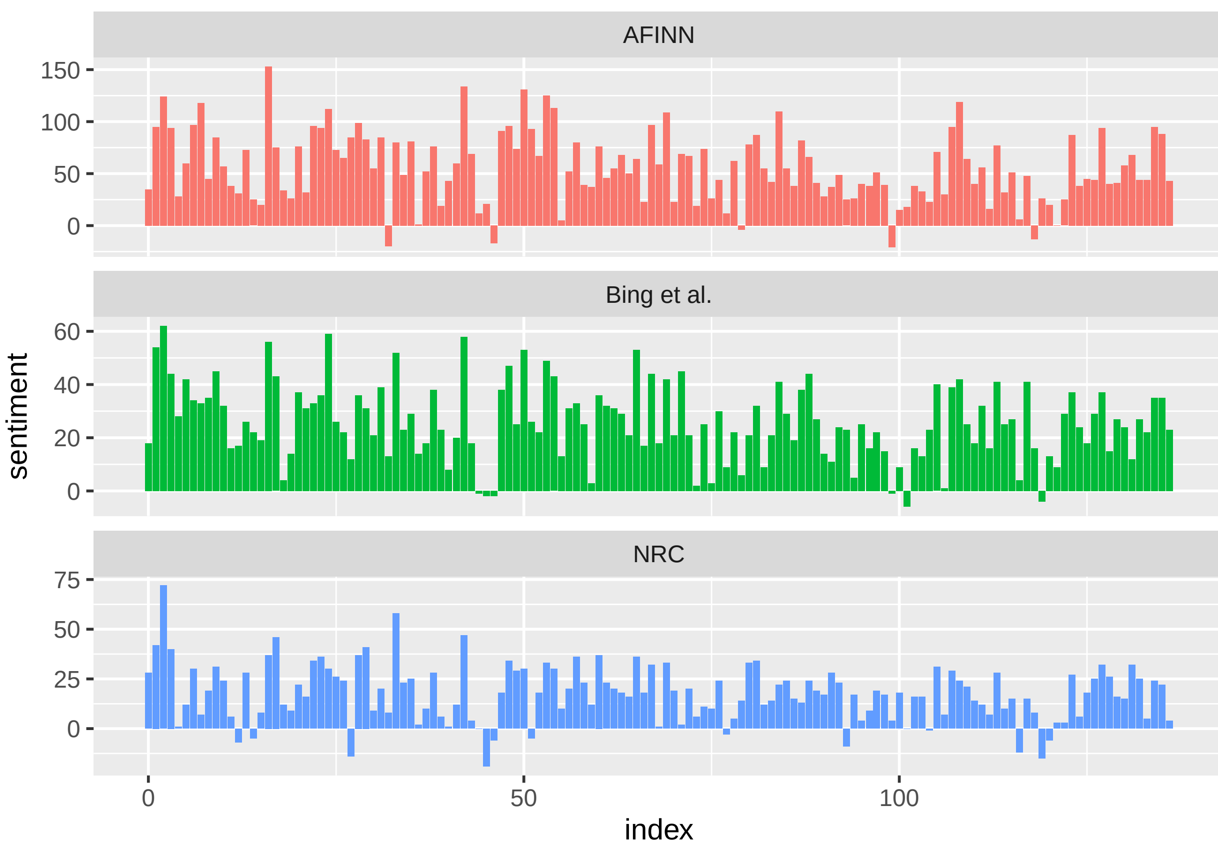
```r
afinn <- michael %>%
  inner_join(get_sentiments("afinn")) %>%
  group_by(index = lines %/% 80) %>%
  summarise(sentiment = sum(value)) %>%
  mutate(method = "AFINN")
```

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

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

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

```r
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")
```



A majority of words Michael uses is positive, however, it looks like NRC and AFINN have a few negative ones; more than Bing.

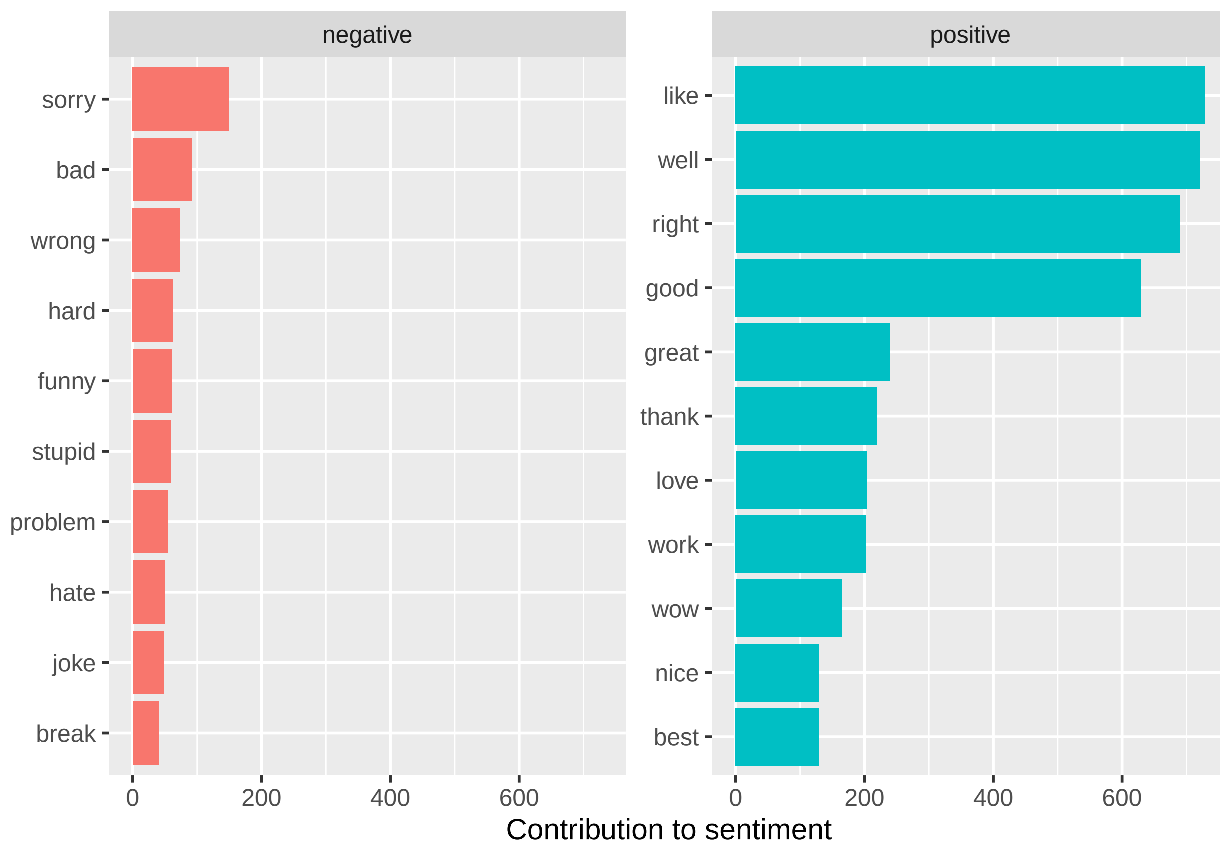
Most Common Words

What are the most common positive and negative words that Michael uses?

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

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

```
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)
```



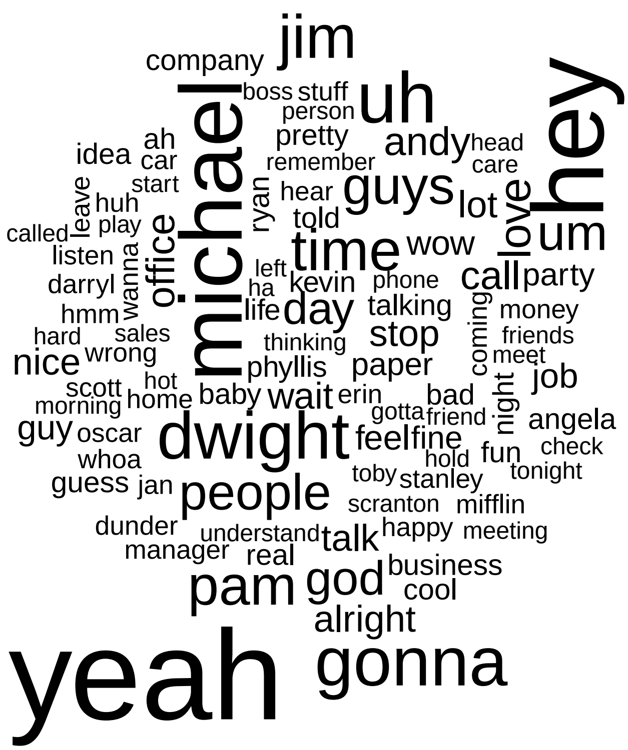
Here, we can see that the count of positive words is much higher – this makes sense as a majority of the words he uses are quite positive. We can see that for negative words, he uses sorry, bad, wrong, and hard the most; positive words he uses a lot are like, well, right, good.

Word Cloud

What would a word cloud look like for this script?

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

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



Words like yeah, hey, Michael, and Dwight all appear the most. How does this look with a positive and negative cut?

```
tidy_office %>%
  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"
```

Sorry really comes out here for the negative side while words like like, well, good, and right are all positive – very similar to Michael's word count which makes sense as he is the main character for a majority of the show.

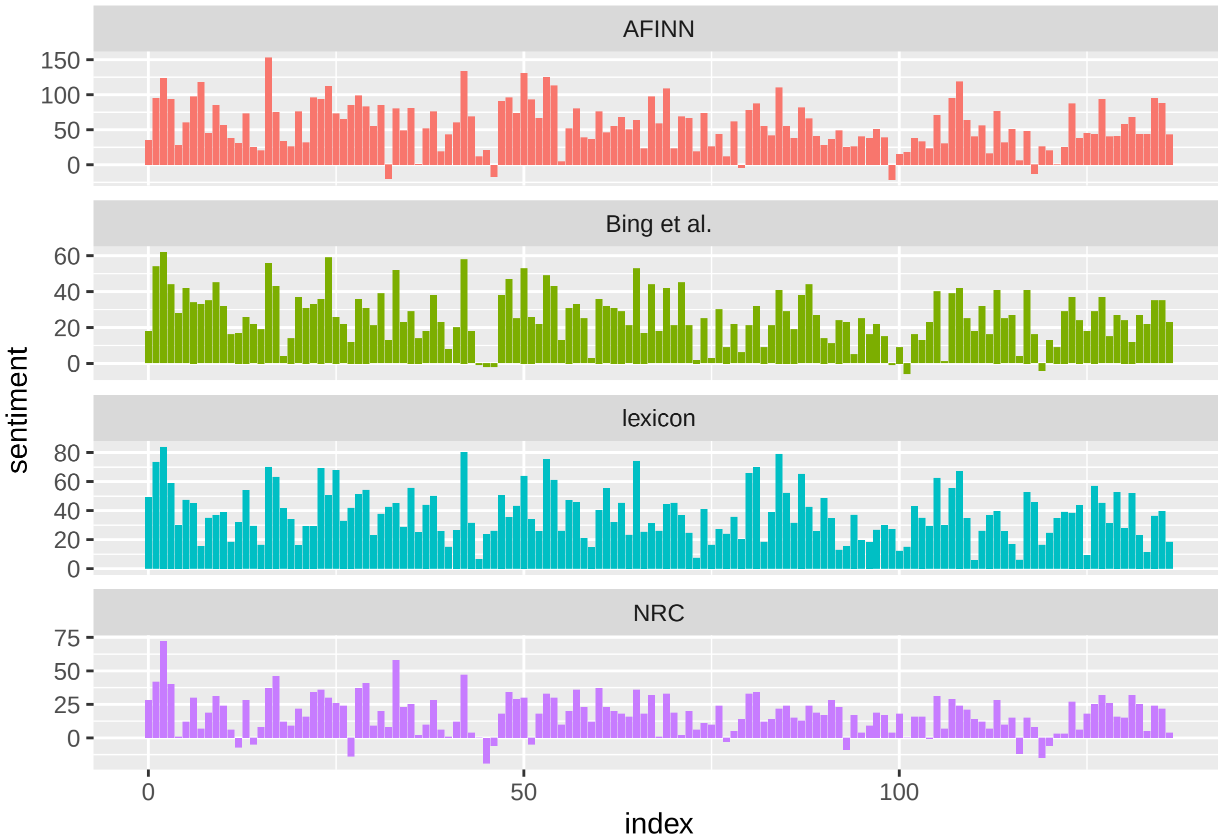
Another Lexicon

I'll be using the `lexicon` package's `hash_sentiment_senticnet` as another way to measure sentiment.

```
library(lexicon)

colnames(hash_sentiment_senticnet)[colnames(hash_sentiment_senticnet) == "x"] = "word"

lexicon_graph <- michael %>%
  inner_join(hash_sentiment_senticnet) %>%
  group_by(index = lines %/% 80) %>%
  summarise(sentiment = sum(y))%>%
  mutate(method = "lexicon")
```

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

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



Using `lexicon`, we can see that there are really not that many negative words here as the other dictionaries. It's probably due to the weight of negative to positive words, but that's interesting to see.

Conclusion

It's interesting how positive the 4 main characters' dialogue seems with this sentiment analysis given how The Office is very awkward and slightly inappropriate in humor. It makes sense though that Dwight's lines are more negative given his disposition and overall personality (he tends to be a little stranger and blunter with his words). Michael's dialogue also makes sense – he's generally a very positive guy, albeit he doesn't realize what he's saying or doing sometimes is pretty inappropriate.

Overall, this analysis was quite interesting and fun to do with R – it could definitely be replicated for other pieces of text and refined to look at more characters and perhaps break this down by season specifically for The Office since that could be an interesting cut.