Lab 7

Kiana Fields

Math 241, Week 9

```r
# Put all necessary libraries here
library(tidyverse)
library(tidytext)
library(dplyr)
library(ggplot2)
library(wordcloud)
# Ensure the textdata package is installed
if (!requireNamespace("textdata", quietly = TRUE)) {
  install.packages("textdata")
}
# Load the textdata package
library(textdata)

# Before knitting your document one last time, you will have to download the AFINN lexicon explicitly
lexicon_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
```

```r
lexicon_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
```

Due: Friday, March 29th at 5:30pm

Goals of this lab

1. Practice matching patterns with regular expressions.
2. Practice manipulating strings with `stringr`.
3. Practice tokenizing text with `tidytext`.
4. Practice looking at word frequencies.
5. Practice conducting sentiment analysis.

Problem 1: What's in a Name? (You'd Be Surprised!)

1. Load the `babynames` dataset, which contains yearly information on the frequency of baby names by sex and is provided by the US Social Security Administration. It includes all names with at least 5 uses per year per sex. In this problem, we are going to practice pattern matching!

```r
library(babynames)
data("babynames")
#?babynames
```

a. For 2000, find the ten most popular female baby names that start with the letter Z.

```r
Znames_2000 <- filter(babynames, year == 2000 & sex == "F" & startsWith(name, "Z"))
top_n(Znames_2000, 10, wt = n)
```

```
## # A tibble: 10 x 5
##     year sex   name       n      prop
##    <dbl> <chr> <chr>  <int>     <dbl>
##  1  2000 F     Zoe     3785 0.00190
##  2  2000 F     Zoey     691 0.000346
##  3  2000 F     Zaria    568 0.000285
##  4  2000 F     Zoie     320 0.000160
##  5  2000 F     Zariah   168 0.0000842
##  6  2000 F     Zion     156 0.0000782
##  7  2000 F     Zainab   142 0.0000712
##  8  2000 F     Zara     121 0.0000607
##  9  2000 F     Zahra    113 0.0000566
## 10  2000 F     Zaira    103 0.0000516
```

b. For 2000, find the ten most popular female baby names that contain the letter z.

```r
contZ_2000 <- babynames %>%
  filter(year == 2000 & sex == "F" & str_detect(name, "z"))
top_n(contZ_2000, 10, wt = n)
```

```
## # A tibble: 10 x 5
##     year sex   name          n     prop
##    <dbl> <chr> <chr>     <int>    <dbl>
##  1  2000 F     Elizabeth 15094 0.00757
##  2  2000 F     Mackenzie  6348 0.00318
##  3  2000 F     Mckenzie   2526 0.00127
##  4  2000 F     Makenzie   1613 0.000809
##  5  2000 F     Jazmin     1391 0.000697
##  6  2000 F     Jazmine    1353 0.000678
##  7  2000 F     Lizbeth     817 0.000410
##  8  2000 F     Eliza       759 0.000380
##  9  2000 F     Litzy       722 0.000362
## 10  2000 F     Esperanza   499 0.000250
```

c. For 2000, find the ten most popular female baby names that end in the letter z.

```
endZ_2000 <- babynames %>%
  filter(year == 2000 & sex == "F") %>%
  mutate(last_letter = str_sub(name, -1, -1)) %>%
  filter(last_letter == "z")

top_n(endZ_2000, 10, wt = n)
```

```
## # A tibble: 11 x 6
##     year sex   name         n       prop last_letter
##    <dbl> <chr> <chr>    <int>      <dbl> <chr>
##  1  2000 F     Luz        489 0.000245   z
##  2  2000 F     Beatriz    357 0.000179   z
##  3  2000 F     Mercedez   141 0.0000707  z
##  4  2000 F     Maricruz    96 0.0000481  z
##  5  2000 F     Liz         72 0.0000361  z
##  6  2000 F     Inez        69 0.0000346  z
##  7  2000 F     Odaliz      24 0.0000120  z
##  8  2000 F     Marycruz    23 0.0000115  z
##  9  2000 F     Cruz        19 0.00000952 z
## 10  2000 F     Deniz       16 0.00000802 z
## 11  2000 F     Taiz        16 0.00000802 z
```

d. Between your three tables in 1.a - 1.c, do any of the names show up on more than one list? If so, which ones? (Yes, I know you could do this visually but use some joins!)

```
joined_names <- inner_join(Znames_2000, contZ_2000, by = "name") %>%
  inner_join(endZ_2000, by = "name")

duplicate_znames <- joined_names %>%
  group_by(name) %>%
  filter(n() > 1) %>%
  distinct(name)

duplicate_znames
```

```
## # A tibble: 0 x 1
## # Groups:   name [0]
## # i 1 variable: name <chr>
```

e. Verify that none of the baby names contain a numeric (0-9) in them.

```
nonum_names <- babynames %>%
  filter(str_detect(name, "[0-9]"))

if(nrow(nonum_names) == 0) {
  print("None of the baby names contain a numeric digit.")
} else {
  print("Some baby names contain a numeric digit.")
}
```

```
## [1] "None of the baby names contain a numeric digit."
```

f. While none of the names contain 0-9, that doesn't mean they don't contain "one", "two", . . . , or "nine". Create a table that provides the number of times a baby's name contained the word "zero", the word "one", . . . the word "nine".

Notes:

- I recommend first converting all the names to lower case.
- If none of the baby's names contain the written number, there you can leave the number out of the table.
- Use `str_extract()`, not `str_extract_all()`. (We will ignore names where more than one of the words exists.)

Hint: You will have two steps that require pattern matching: 1. Subset your table to only include the rows with the desired words. 2. Add a column that contains the desired word.

```
babynames <- babynames %>%
  mutate(lowercase = tolower(name)) %>%
  mutate(num_word = str_extract(name, "zero|one|two|three|four|five|six|seven|eight|nine"))

babynames_numbers <- babynames %>%
  filter(!is.na(num_word)) %>%
  count(num_word)

babynames_numbers
```

```
## # A tibble: 8 x 2
##   num_word     n
##   <chr>    <int>
## 1 eight      356
## 2 four         2
## 3 nine       707
## 4 one       9258
## 5 six          1
## 6 three       58
## 7 two        286
## 8 zero         2
```

g. Which written number or numbers don't show up in any of the baby names?

```r
numbers <- c("zero", "one", "two", "three", "four", "five", "six", "seven", "eight", "nine")

missing_numbers <- setdiff(numbers, babynames_numbers$num_word)

print(paste("The following written numbers do not appear in any baby names:", paste(missing_numbers, co
```

```
## [1] "The following written numbers do not appear in any baby names: five, seven"
```

h. Create a table that contains the names and their frequencies for the two least common written numbers.

```r
least_numnamesnums <- babynames_numbers %>%
  arrange(n) %>%
  head(2)

least_numnames <- babynames %>%
  filter(num_word %in% least_numnamesnums$num_word) %>%
  select(name, num_word) %>%
  arrange(num_word)

least_numnames
```

```
## # A tibble: 3 x 2
##   name    num_word
##   <chr>   <chr>
## 1 Balfour four
## 2 Balfour four
## 3 Essix   six
```

i. List out the names that contain no vowels (consider "y" to be a vowel).

```r
vowels <- c("a", "e", "i", "o", "u", "y")

novowels_names <- babynames %>%
  filter(!str_detect(tolower(name), paste(vowels, collapse = "|"))) %>%
  distinct(name)

novowels_names$name
```

```
##  [1] "Wm"  "Ng"  "Mc"  "Jc"  "Jb"  "Jd"  "Jt"  "Jr"  "Lc"  "Tj"  "Wc"  "Ld"
## [13] "Rc"  "Jw"  "Jl"  "St"  "Rb"  "Lj"  "Lb"  "Rl"  "Rd"  "Lg"  "Mrk" "Kc"
## [25] "Jj"  "Dj"  "Bj"  "Cj"  "Pj"  "Rj"  "Jm"  "Jp"  "Tc"  "Kt"  "Kd"  "Md"
## [37] "Kj"  "Mr"  "Bg"  "Bb"  "Mj"  "Sj"  "Tr"
```

Problem 2: Tidying the "Call of the Wild"

Did you read "Call of the Wild" by Jack London? If not, read the first paragraph of its wiki page for a quick summary and then let's do some text analysis on this classic! The following code will pull the book into R using the `gutenbergr` package.

```r
library(gutenbergr)
wild <- gutenberg_download(215)
```

a. Create a tidy text dataset where you tokenize by words.

```r
tidy_wild <- wild %>%
  unnest_tokens(word, text)
```

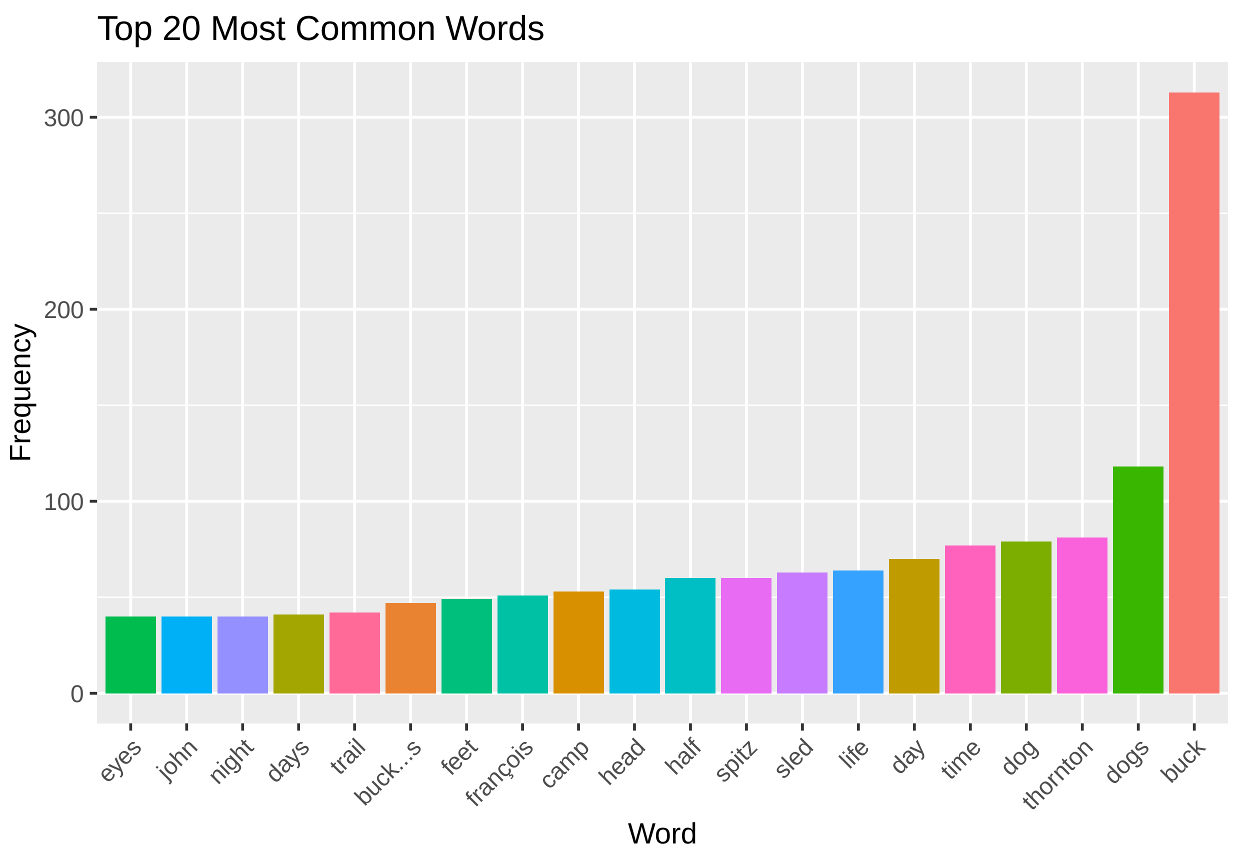
b. Find the frequency of the 20 most common words. First, remove stop words.

```r
tidy_wild_nostop <- tidy_wild %>%
  anti_join(stop_words)

top_wild <- tidy_wild_nostop %>%
  count(word, sort = TRUE) %>%
  head(20)

top_wild
```

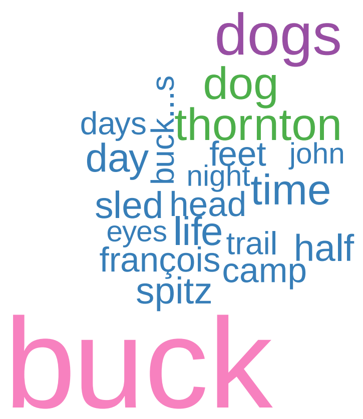
```
## # A tibble: 20 x 2
##    word         n
##    <chr>    <int>
##  1 buck       313
##  2 dogs       118
##  3 thornton    81
##  4 dog         79
##  5 time        77
##  6 day         70
##  7 life        64
##  8 sled        63
##  9 half        60
## 10 spitz       60
## 11 head        54
## 12 camp        53
## 13 françois    51
## 14 feet        49
## 15 buck's      47
## 16 trail       42
## 17 days        41
## 18 eyes        40
## 19 john        40
## 20 night       40
```

c. Create a bar graph and a word cloud of the frequencies of the 20 most common words.

```r
ggplot(top_wild, aes(x = reorder(word, n),
                     y = n,
                     fill = word)) +
  geom_bar(stat = "identity") +
  labs(title = "Top 20 Most Common Words",
       x = "Word",
       y = "Frequency") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
        legend.position = "none")
```



```
wordcloud(words = top_wild$word,
          freq = top_wild$n, max.words = 20,
          colors = brewer.pal(8, "Set1"))
```

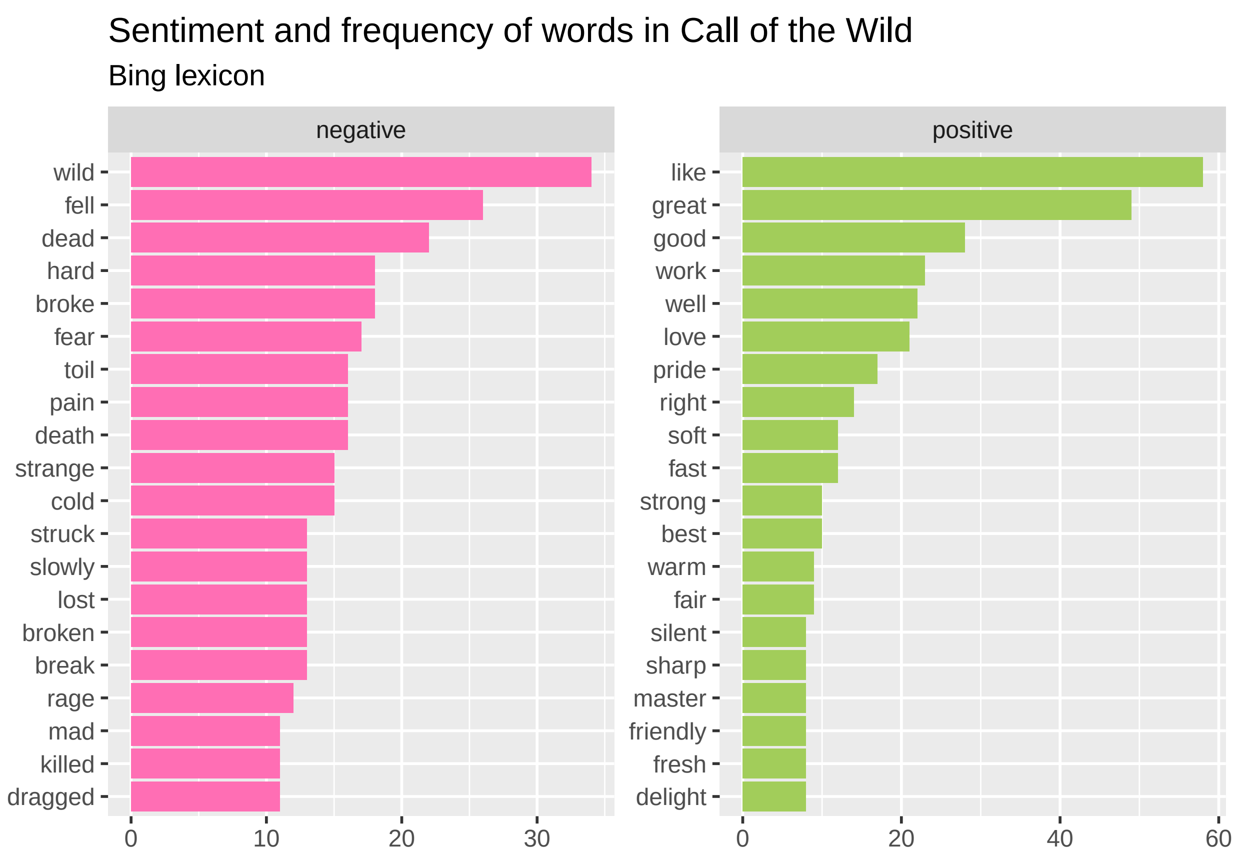


d. Explore the sentiment of the text using three of the sentiment lexicons in `tidytext`. What does your analysis say about the sentiment of the text?

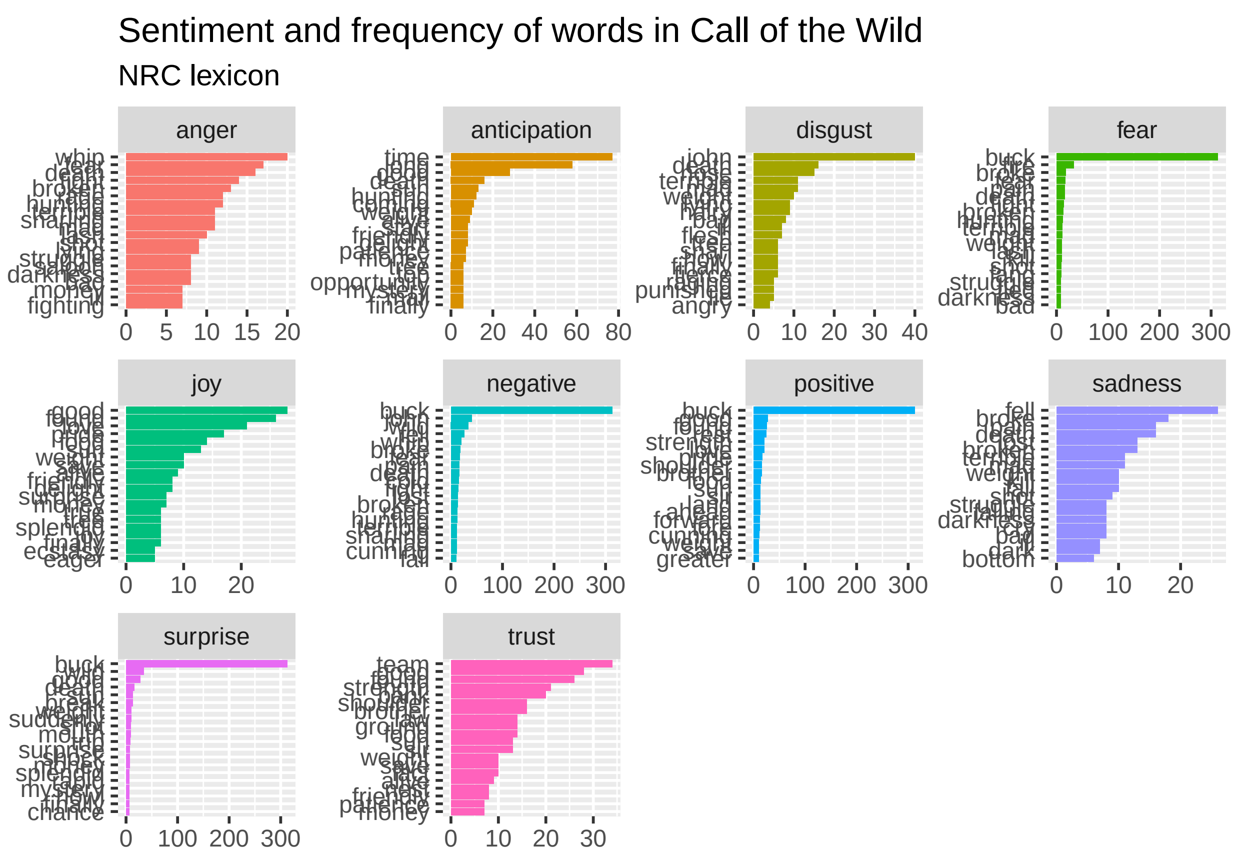
Notes:

- Make sure to NOT remove stop words this time.
- `afinn` is a numeric score and should be handled differently than the categorical scores.

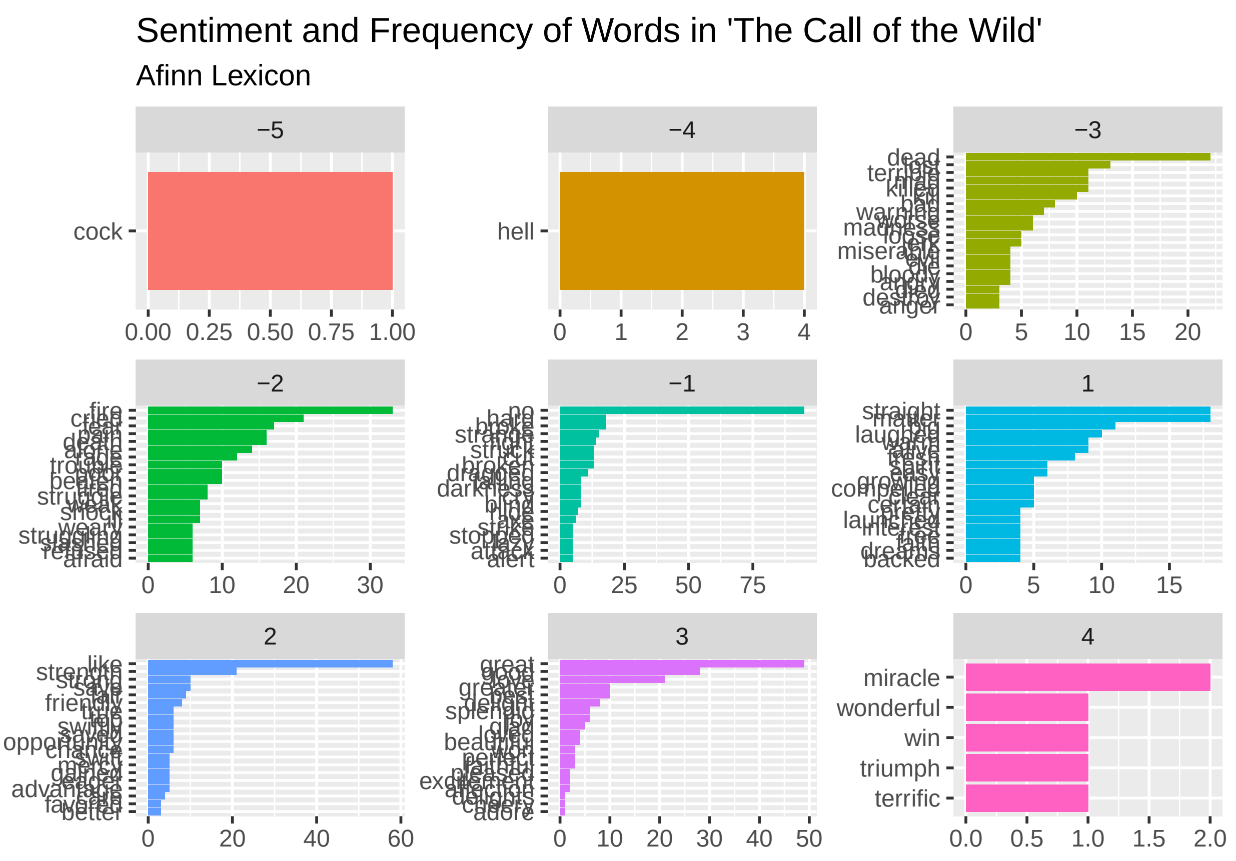
```r
#bing lexicon
tidy_wild %>%
  inner_join(get_sentiments("bing"), by = "word") %>%
  count(sentiment, word, sort = TRUE) %>%
  group_by(sentiment) %>%
  slice_head(n = 20) %>%
  ggplot(aes(y = fct_reorder(word, n), x = n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free") +
  labs(
    title = "Sentiment and frequency of words in Call of the Wild",
    subtitle = "Bing lexicon",
    y = NULL, x = NULL
  ) +
  scale_fill_manual(values = c("hotpink1", "darkolivegreen3"))
```



```r
#NRC lexicon
tidy_wild %>%
  inner_join(get_sentiments("nrc"), by = "word") %>%
  count(sentiment, word, sort = TRUE) %>%
  group_by(sentiment) %>%
  slice_head(n = 20) %>%
  ggplot(aes(y = fct_reorder(word, n), x = n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free") +
  labs(
    title = "Sentiment and frequency of words in Call of the Wild",
    subtitle = "NRC lexicon",
    y = NULL, x = NULL
  )
```



```r
#afinn lexicon
afinn <- get_sentiments("afinn")
affin_analysis <- tidy_wild %>%
  inner_join(afinn, by = "word") %>%
  count(sentiment = value, word, sort = TRUE) %>%
  group_by(sentiment) %>%
  slice_head(n=20)
ggplot(affin_analysis, aes(y = fct_reorder(word, n), x = n, fill = factor(sentiment))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free") +
  labs(title = "Sentiment and Frequency of Words in 'The Call of the Wild'",
       subtitle = "Afinn Lexicon",
       y = NULL, x = NULL
  )
```



e. If you didn't do so in 2.d, compute the average sentiment score of the text using `afinn`. Which positive words had the biggest impact? Which negative words had the biggest impact?

```
sentiment_scores <- tidy_wild %>%
  inner_join(afinn, by = "word") %>%
  summarise(average_score = sum(value) / n())
sentiment_scores
```

```
## # A tibble: 1 x 1
##   average_score
##           <dbl>
## 1        -0.322
```

```
positive_impact <- tidy_wild %>%
  inner_join(afinn, by = "word") %>%
  filter(value > 0) %>%
  arrange(desc(value))
head(positive_impact)
```

```
## # A tibble: 6 x 3
##   gutenberg_id word      value
##          <int> <chr>     <dbl>
## 1          215 miracle       4
## 2          215 miracle       4
## 3          215 wonderful     4
## 4          215 triumph       4
## 5          215 win           4
## 6          215 terrific      4
```

```
negative_impact <- tidy_wild %>%
  inner_join(afinn, by = "word") %>%
  filter(value < 0) %>%
  arrange(value)
head(negative_impact)
```

```
## # A tibble: 6 x 3
##   gutenberg_id word  value
##          <int> <chr> <dbl>
## 1          215 cock     -5
## 2          215 hell     -4
## 3          215 hell     -4
## 4          215 hell     -4
## 5          215 hell     -4
## 6          215 angry    -3
```

f. You should have found that "no" was an important negative word in the sentiment score. To know if that really makes sense, let's turn to the raw lines of text for context. Pull out all of the lines that have the word "no" in them. Make sure to not pull out extraneous lines (e.g., a line with the word "now").

```
lines_with_no <- wild$text[str_detect(wild$text, "\\bno\\b")]
print(lines_with_no)
```

```
##  [1] "solitary man, no one saw them arrive at the little flag station known"
##  [2] "that it was the club, but his madness knew no caution. A dozen times he"
##  [3] ""He's no slouch at dog-breakin', that's wot I say," one of the men on"
##  [4] "all, that he stood no chance against a man with a club. He had learned"
##  [5] "in the red sweater. "And seem' it's government money, you ain't got no"
##  [6] "animal. The Canadian Government would be no loser, nor would its"
##  [7] "while he developed no affection for them, he none the less grew"
##  [8] "The other dog made no advances, nor received any; also, he did not"
##  [9] "knew no law but the law of club and fang."
## [10] "full-grown wolf, though not half so large as she. There was no warning,"
## [11] "tail appeasingly, turned to run when he saw that appeasement was of no"
## [12] "his flank. But no matter how Spitz circled, Joe whirled around on his"
## [13] "their comradeship had no more trouble. His only apparent ambition, like"
## [14] "unduly civilized dog, and of his own experience knew no trap and so"
## [15] "no ice at all."
## [16] "him of his unfinished ration. There was no defending it. While he was"
## [17] "days, no matter what the odds, he had never run from a fight. But the"
## [18] "internal as well as external economy. He could eat anything, no matter"
## [19] "they ran it down. It was no task for him to learn to fight with cut and"
## [20] "he betrayed no impatience, shunned all offensive acts."
## [21] "as he circled back and forth for a chance to spring in. Buck was no"
## [22] "less eager, and no less cautious, as he likewise circled back and forth"
## [23] "terrifying, irresistible. There was no opposing them. The team-dogs"
## [24] "huskies, there was no hope for him. But he braced himself to the shock"
## [25] "nothing, no matter how remotely eatable, had escaped them. They had"
## [26] "Again, the rim ice broke away before and behind, and there was no"
## [27] "Things no longer went right. There was continual bickering and"
## [28] "had destroyed the solidarity of the team. It no longer was as one dog"
## [29] "There was no hope for him. Buck was inexorable. Mercy was a thing"
## [30] "make good time. No more Spitz, no more trouble, sure.""
## [31] "trail. There was no place for Buck save at the front. Once more"
## [32] "there was no new-fallen snow with which to contend. It was not too"
## [33] "Dawson. It was no light running now, nor record time, but heavy toil"
## [34] "dim and distant, and such memories had no power over him. Far more"
## [35] "ate, and no man sought his sleeping-robe till he had seen to the feet"
## [36] "but they could locate no broken bones, could not make it out."
## [37] "The half-breed tried to drive him away with the whip; but he paid no"
## [38] "slow and prolonged strength drainage of months of toil. There was no"
## [39] "power of recuperation left, no reserve strength to call upon. It had"
## [40] "manner, but no businesslike method. The tent was rolled into an awkward"
## [41] "lead. Hal cried "Whoa! whoa!" but they gave no heed. He tripped and was"
## [42] "nothing lively about it, no snap or go in him and his fellows. They"
## [43] "Buck felt vaguely that there was no depending upon these two men and"
## [44] "at all. And on no day did they succeed in making more than half the"
## [45] "that for love or money no additional dog-food was to be obtained. So he"
## [46] "the ration of the husky, so the six Outside dogs under Buck could do no"
## [47] "had no inkling of such a patience. They were stiff and in pain; their"
## [48] "sex-prerogative, she made their lives unendurable. She no longer"
## [49] "nightmare. He pulled when he could; when he could no longer pull, he"
## [50] "fresher; and Buck, still at the head of the team, but no longer"
## [51] "response to Thornton’s warning to take no more chances on the rotten"
## [52] "made no effort. He lay quietly where he had fallen. The lash bit into"
## [53] "pain left him. He no longer felt anything, though very faintly he could"
## [54] "hear the impact of the club upon his body. But it was no longer his"
## [55] "Thornton stood between him and Buck, and evinced no intention of"
## [56] "Hal had no fight left in him. Besides, his hands were full with his"
## [57] "To Buck’s surprise these dogs manifested no jealousy toward him. They"
## [58] "names. Buck knew no greater joy than that rough embrace and the sound"
## [59] "had come into the Northland had bred in him a fear that no master could"
## [60] "but the strange dog, no matter what the breed or valor, swiftly"
## [61] "knew there was no middle course. He must master or be mastered; while"
## [62] "the rapids, a stretch of wild water in which no swimmer could live."
## [63] "permitting no slack, while Pete kept it clear of coils. Buck held on"
## [64] "no thousand dollars; nor had Hans or Pete."
## [65] "There were no takers. Not a man believed him capable of the feat."
## [66] "heavy fore legs were no more than in proportion with the rest of the"
## [67] "frankly down his cheeks. "Sir," he said to the Skookum Bench king, "no,"
## [68] "But no living man had looted this treasure house, and the dead were"
## [69] "Being in no haste, Indian fashion, he hunted his dinner in the course"
## [70] "uncharted vastness, where no men were and yet where men had been if the"
## [71] "wildfowl had been, but where then there was no life nor sign of"
## [72] "packed flat, And that was all-no hint as to the man who in an early day"
## [73] "washing-pan. They sought no farther. Each day they worked earned them"
## [74] "He had made no noise, yet it ceased from its howling and tried to sense"
## [75] "that no harm was intended, finally sniffed noses with him. Then they"
## [76] "again he took to wandering in the woods, but the wild brother came no"
## [77] "behind who would quarrel no more."
## [78] "secrecy of the forest. He no longer marched. At once he became a thing"
## [79] "proceeded to cut the bull out from the herd. It was no slight task. He"
## [80] "next bound tearing wide the throat of a second man. There was no"
## [81] "from which no trace led away."
## [82] "was harder to kill a husky dog than them. They were no match at all,"
## [83] "claims of man no longer bound him."
```

g. Draw some conclusions about how "no" is used in the text. "no" does not really appear to be used negatively, but rather in a different context like "claims of man no longer bound him" rather than "no" as an answer or statement in and of itself.

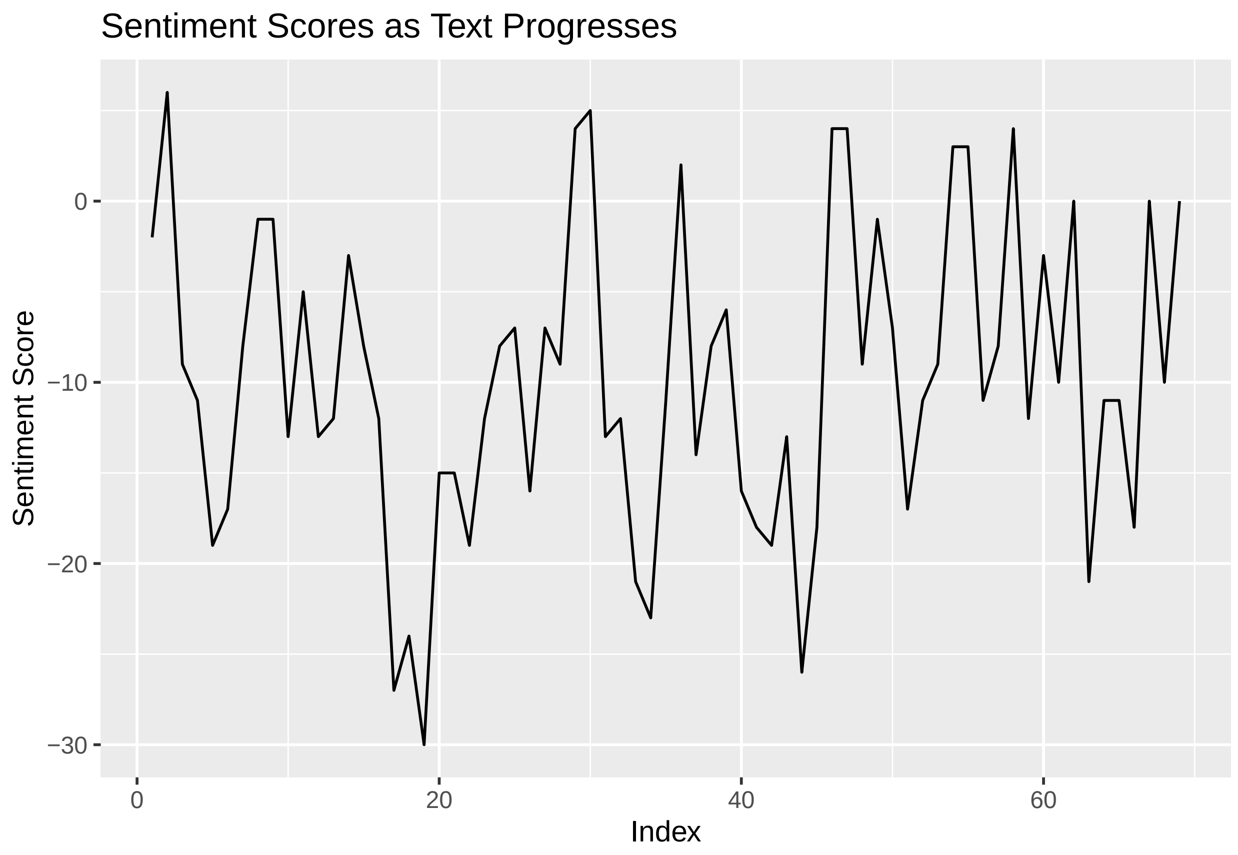
h. We can also look at how the sentiment of the text changes as the text progresses. Below, I have added two columns to the original dataset. Now I want you to do the following wrangling:

- Tidy the data (but don't drop stop words).
- Add the word sentiments using `bing`.
- Count the frequency of sentiments by index.
- Reshape the data to be wide with the count of the negative sentiments in one column and the positive in another, along with a column for index.
- Compute a sentiment column by subtracting the negative score from the positive.

```r
wild_time <- wild %>%
  mutate(line = row_number(), index = floor(line/45) + 1) %>%
  unnest_tokens(word, text) %>%
  inner_join(get_sentiments("bing"), by = "word") %>%
  count(index, sentiment) %>%
  spread(sentiment, n, fill = 0) %>%
  rename(negative = negative, positive = positive) %>%
  mutate(sentiment = positive - negative)
```

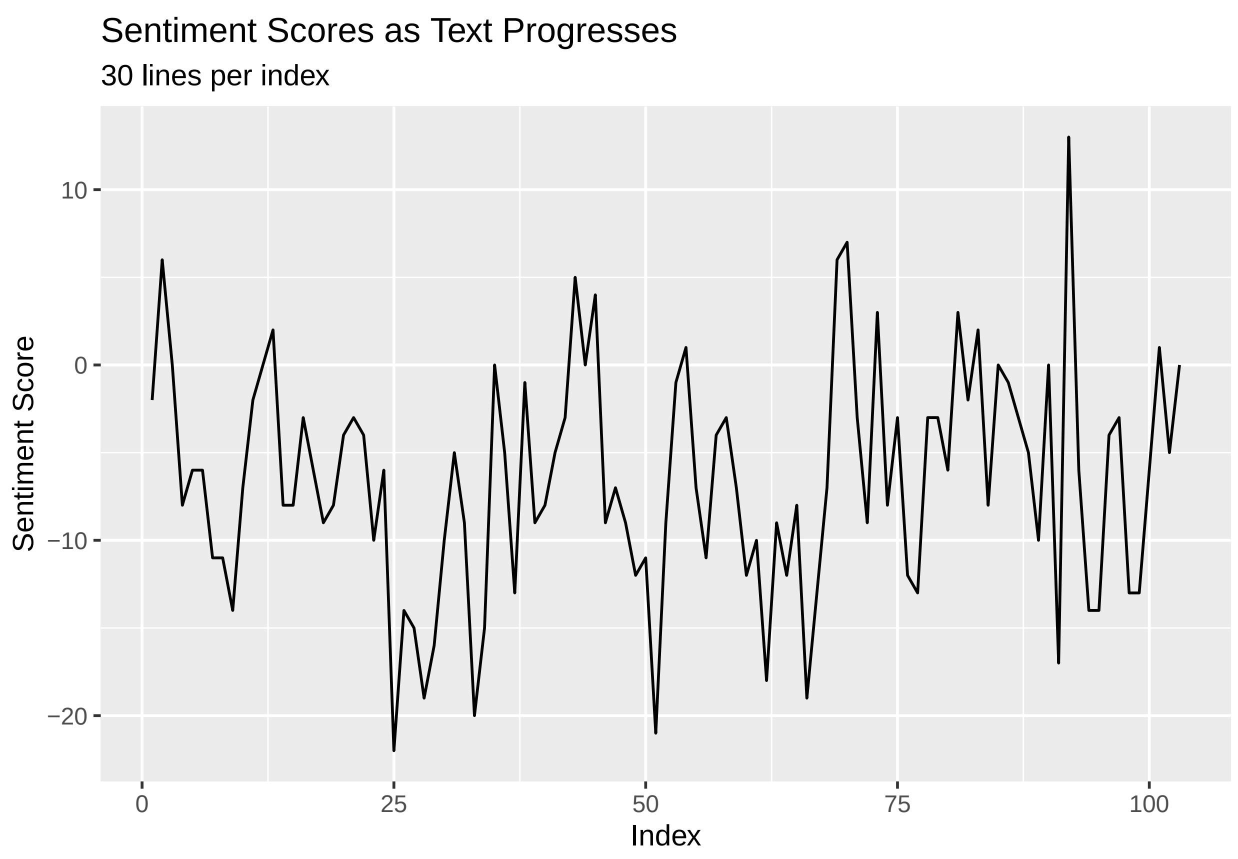
i. Create a plot of the sentiment scores as the text progresses.

```r
ggplot(wild_time, aes(x = index, y = sentiment)) +
  geom_line() +
  labs(title = "Sentiment Scores as Text Progresses",
       x = "Index", y = "Sentiment Score")
```

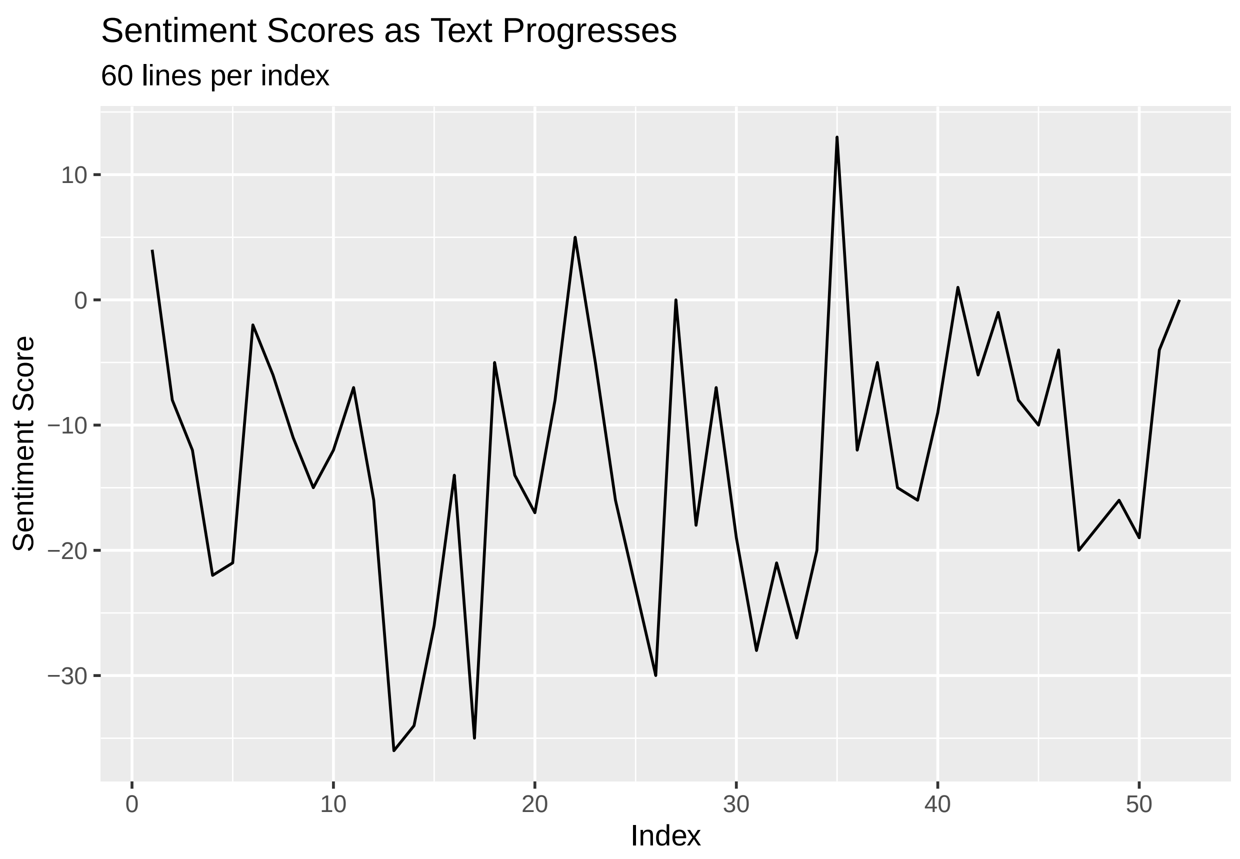


j. The choice of 45 lines per chunk was pretty arbitrary. Try modifying the index value a few times and recreating the plot in i. Based on your plots, what can you conclude about the sentiment of the novel as it progresses?

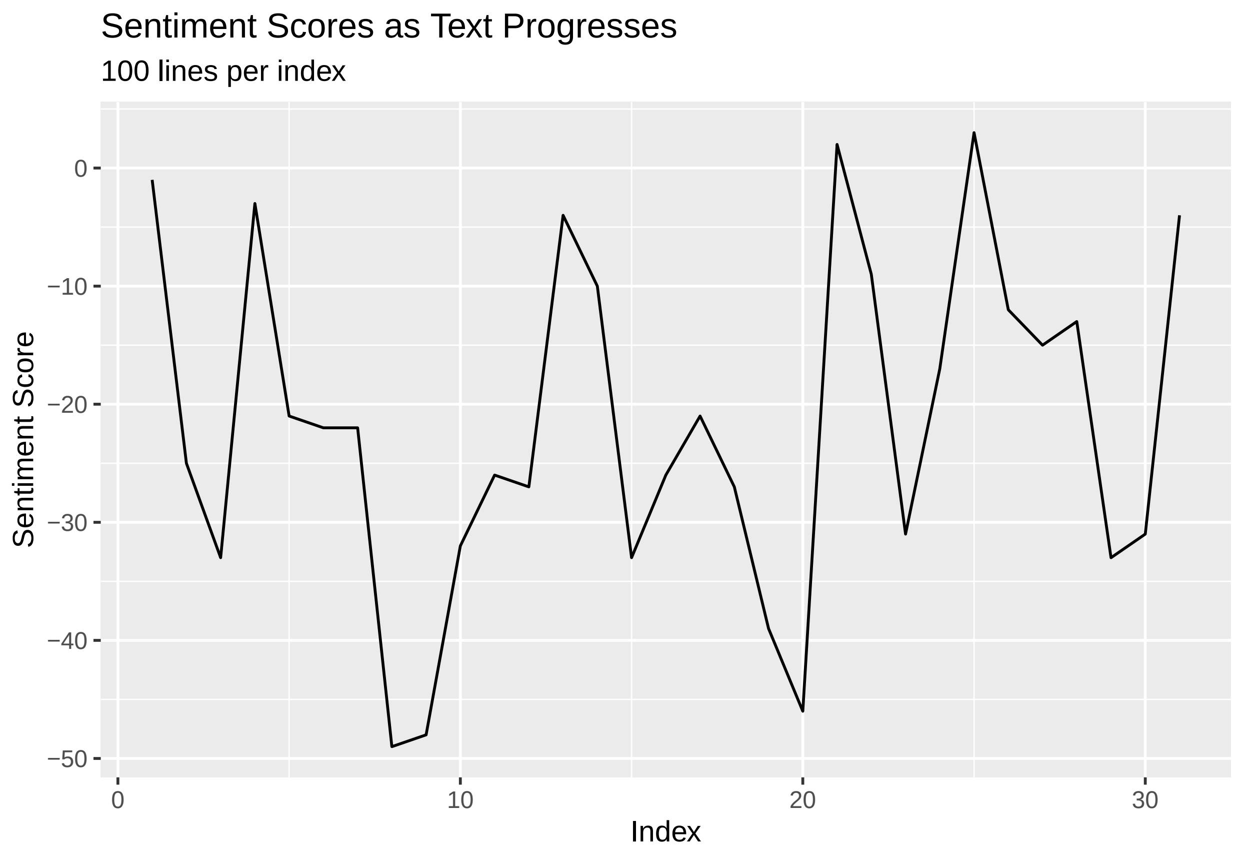
```
wild_time <- wild %>%
  mutate(line = row_number(), index = floor(line/30) + 1) %>%
  unnest_tokens(word, text) %>%
  inner_join(get_sentiments("bing"), by = "word") %>%
  count(index, sentiment) %>%
  spread(sentiment, n, fill = 0) %>%
  rename(negative = negative, positive = positive) %>%
  mutate(sentiment = positive - negative)
ggplot(wild_time, aes(x=index, y=sentiment)) +
  geom_line() +
  labs(title = "Sentiment Scores as Text Progresses",
       subtitle="30 lines per index",
       x="Index", y="Sentiment Score")
```



```r
wild_time <- wild %>%
  mutate(line = row_number(), index = floor(line/60) + 1) %>%
  unnest_tokens(word, text) %>%
  inner_join(get_sentiments("bing"), by = "word") %>%
  count(index, sentiment) %>%
  spread(sentiment, n, fill = 0) %>%
  rename(negative = negative, positive = positive) %>%
  mutate(sentiment = positive - negative)
ggplot(wild_time, aes(x=index, y=sentiment)) +
  geom_line() +
  labs(title = "Sentiment Scores as Text Progresses",
       subtitle="60 lines per index",
       x="Index", y="Sentiment Score")
```



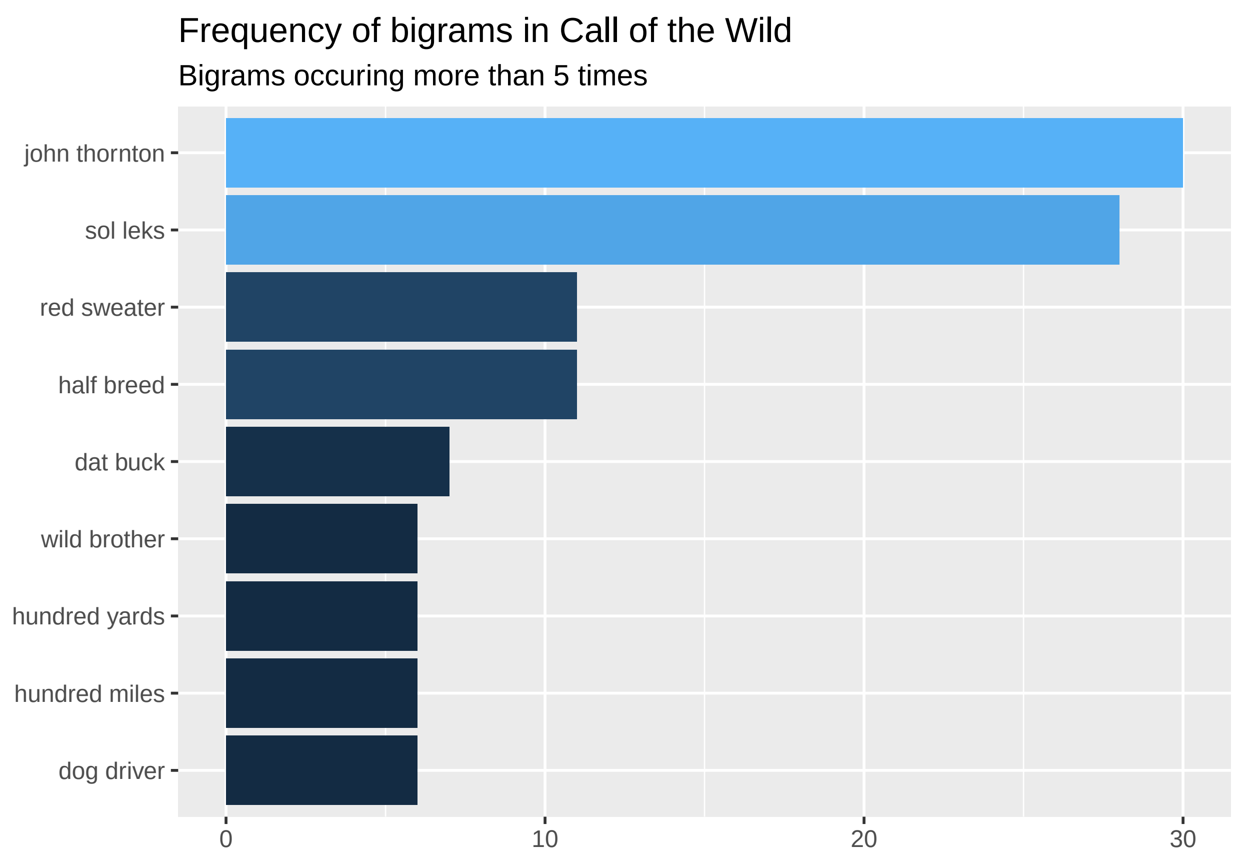
```
wild_time <- wild %>%
  mutate(line = row_number(), index = floor(line/100) + 1) %>%
  unnest_tokens(word, text) %>%
  inner_join(get_sentiments("bing"), by = "word") %>%
  count(index, sentiment) %>%
  spread(sentiment, n, fill = 0) %>%
  rename(negative = negative, positive = positive) %>%
  mutate(sentiment = positive - negative)
ggplot(wild_time, aes(x=index, y=sentiment)) +
  geom_line() +
  labs(title = "Sentiment Scores as Text Progresses",
       subtitle="100 lines per index",
       x="Index", y="Sentiment Score")
```



The middle of the story takes a dark turn before becoming more uplifting at the end.

k. Let's look at the bigrams (2 consecutive words). Tokenize the text by bigrams.

```
threshold <- 5
wild_bigrams <- wild %>%
  unnest_tokens(bigram, text, token = "ngrams", n = 2) %>%
  mutate(i = row_number()) %>%     # add index for later grouping
  unnest_tokens(word, bigram, drop = FALSE) %>%     # tokenize bigrams into words
  anti_join(stop_words) %>%     # drop rows with stop words
  group_by(i) %>%     # group by bigram index
  filter(n() == 2) %>%     # drop bigram instances where only one word left
  summarise(bigram = unique(bigram), .groups = "drop")
wild_bigrams %>%
  count(bigram, sort = TRUE) %>%
  filter(n > threshold) %>%
  ggplot(aes(y = fct_reorder(bigram, n), x = n, fill = n)) +
  geom_col() +
  guides(fill = FALSE) +
  labs(
    title = "Frequency of bigrams in Call of the Wild",
    subtitle ="Bigrams occuring more than 5 times",
    y = NULL, x = NULL
  )
```



1. Produce a sorted table that counts the frequency of each bigram and notice that stop words are still an issue.

```
threshold <- 5
wild_bigrams <- wild %>%
  unnest_tokens(bigram, text, token = "ngrams", n = 2) %>%
  mutate(i = row_number()) %>%     # add index for later grouping
  unnest_tokens(word, bigram, drop = FALSE) %>%     # tokenize bigrams into words
  anti_join(stop_words) %>%     # drop rows with stop words
  group_by(i) %>%     # group by bigram index
  filter(n() == 2) %>%     # drop bigram instances where only one word left
  summarise(bigram = unique(bigram), .groups = "drop")
wild_bigrams %>%
  count(bigram, sort = TRUE) %>%
  filter(n > threshold)
```

```
## # A tibble: 9 x 2
##   bigram            n
##   <chr>         <int>
## 1 john thornton    30
## 2 sol leks         28
## 3 half breed       11
## 4 red sweater      11
## 5 dat buck          7
## 6 dog driver        6
## 7 hundred miles     6
## 8 hundred yards     6
## 9 wild brother      6
```