**Sentiment Analysis**

We can use the sentiment analysis techniques we explored in [Chapter 2](http://my.safaribooksonline.com/9781491981641/sentiment_html#sentiment) to examine how often positive and negative words occur in these Usenet posts. Which newsgroups are the most positive or negative overall?

In this example we’ll use the AFINN sentiment lexicon, which provides numeric positivity scores for each word, and visualize it with a bar plot ([Figure 9-6](http://my.safaribooksonline.com/9781491981641/sentiment_analysis_html#average_afinn_score)).

newsgroup\_sentiments <- words\_by\_newsgroup %>%

inner\_join(get\_sentiments("afinn"), by = "word") %>%

group\_by(newsgroup) %>%

summarize(score = sum(score \* n) / sum(n))

newsgroup\_sentiments %>%

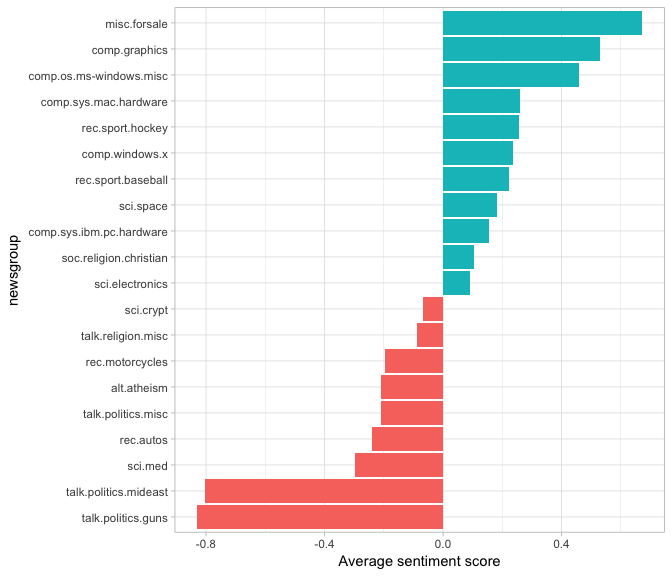
mutate(newsgroup = reorder(newsgroup, score)) %>%

ggplot(aes(newsgroup, score, fill = score > 0)) +

geom\_col(show.legend = **FALSE**) +

coord\_flip() +

ylab("Average sentiment score")



*Figure 9-6. Average AFINN score for posts within each newsgroup*

According to this analysis, the misc.forsale newsgroup is the most positive. This makes sense, since it likely includes many positive adjectives about the products that users want to sell!

## Sentiment Analysis by Word

It’s worth looking deeper to understand why some newsgroups end up more positive or negative than others. For that, we can examine the total positive and negative contributions of each word.

contributions <- usenet\_words %>%

inner\_join(get\_sentiments("afinn"), by = "word") %>%

group\_by(word) %>%

summarize(occurences = n(),

contribution = sum(score))

contributions

## # A tibble: 1,909 × 3

## word occurences contribution

## <chr> <int> <int>

## 1 abandon 13 -26

## 2 abandoned 19 -38

## 3 abandons 3 -6

## 4 abduction 2 -4

## 5 abhor 4 -12

## 6 abhorred 1 -3

## 7 abhorrent 2 -6

## 8 abilities 16 32

## 9 ability 177 354

## 10 aboard 8 8

## # ... with 1,899 more rows

Which words have the most effect on sentiment scores overall ([Figure 9-7](http://my.safaribooksonline.com/9781491981641/sentiment_analysis_by_word_html#words_with_greatest_contributions_to_sentiment))?

contributions %>%

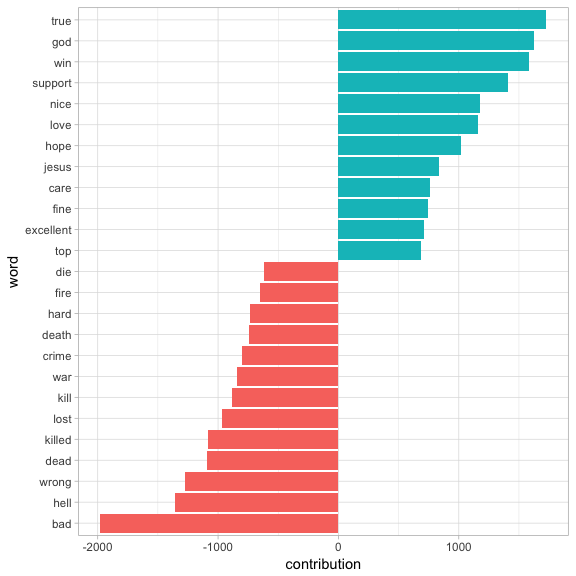
top\_n(25, abscontribution) %>%

mutate(word = reorder(word, contribution)) %>%

ggplot(aes(word, contribution, fill = contribution > 0)) +

geom\_col(show.legend = **FALSE**) +

coord\_flip()



###### *Figure 9-7. Words with the greatest contributions to positive/negative sentiment scores in the Usenet text*

These words look generally reasonable as indicators of each message’s sentiment, but we can spot possible problems with the approach. “True” could just as easily be a part of “not true” or a similar negative expression, and the words “God” and “Jesus” are apparently very common on Usenet but could easily be used in many contexts, positive or negative.

We may also care about which words contribute the most within each newsgroup, so that we can see which newsgroups might be incorrectly estimated. We can calculate each word’s contribution to each newsgroup’s sentiment score, and visualize the strongest contributors from a selection of the groups ([Figure 9-8](http://my.safaribooksonline.com/9781491981641/sentiment_analysis_by_word_html#contributed_the_most_within_each_newgroup)).

top\_sentiment\_words <- words\_by\_newsgroup %>%

inner\_join(get\_sentiments("afinn"), by = "word") %>%

mutate(contribution = score \* n / sum(n))

top\_sentiment\_words

## # A tibble: 13,063 × 5

## newsgroup word n score contribution

## <chr> <chr> <int> <int> <dbl>

## 1 soc.religion.christian god 917 1 0.014418012

## 2 soc.religion.christian jesus 440 1 0.006918130

## 3 talk.politics.guns gun 425 -1 -0.006682285

## 4 talk.religion.misc god 296 1 0.004654015

## 5 alt.atheism god 268 1 0.004213770

## 6 soc.religion.christian faith 257 1 0.004040817

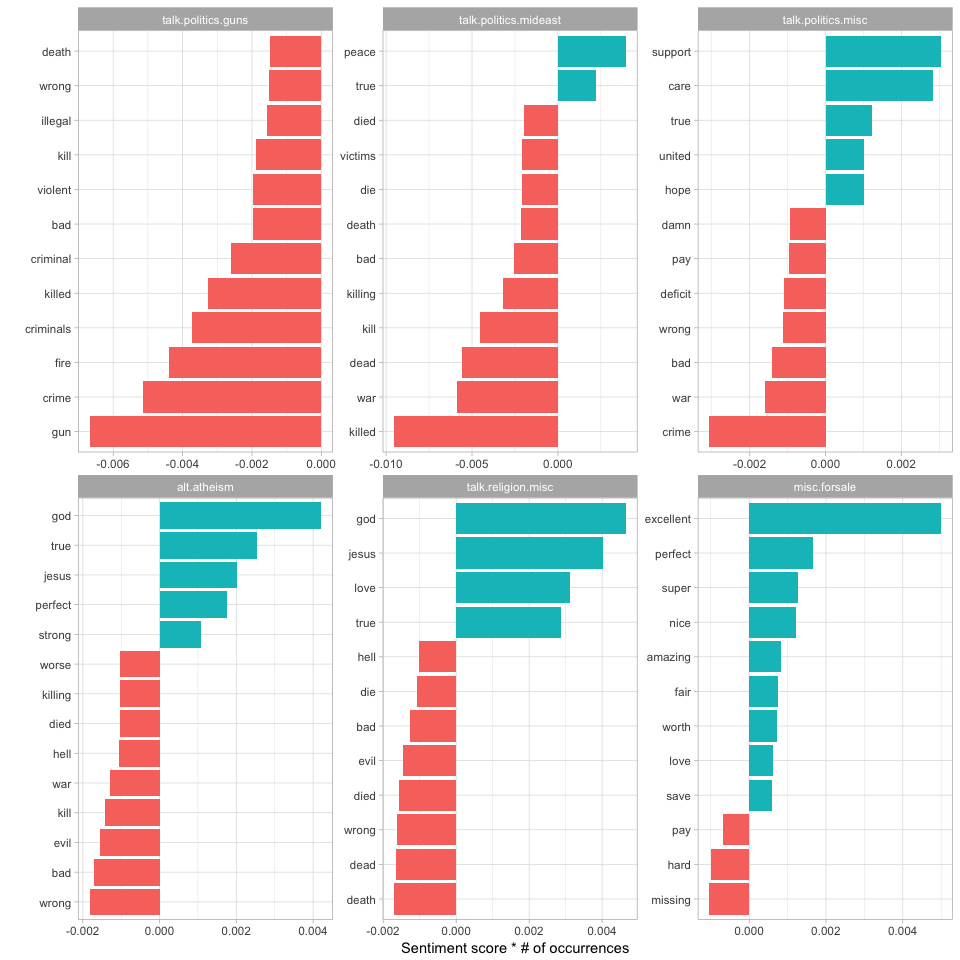
## 7 talk.religion.misc jesus 256 1 0.004025094

## 8 talk.politics.mideast killed 202 -3 -0.009528152

## 9 talk.politics.mideast war 187 -2 -0.005880411

## 10 soc.religion.christian true 179 2 0.005628842

## # ... with 13,053 more rows



###### *Figure 9-8. The 12 words that contributed the most to sentiment scores within each of 6 newsgroups*

This confirms our hypothesis about the misc.forsale newsgroup: most of the sentiment is driven by positive adjectives such as “excellent” and “perfect.” We can also see how much sentiment is confounded with topic. An atheism newsgroup is likely to discuss “god” in detail even in a negative context, and we can see that it makes the newsgroup look more positive. Similarly, the negative contribution of the word “gun” to the talk.politics.guns group will occur even when the members are discussing guns positively.

This helps remind us that sentiment analysis can be confounded by topic, and that we should always examine the influential words before interpreting the analysis too deeply.

## Sentiment Analysis by Message

We can also try finding the most positive and negative individual messages by grouping and summarizing by id rather than newsgroup.

sentiment\_messages <- usenet\_words %>%

inner\_join(get\_sentiments("afinn"), by = "word") %>%

group\_by(newsgroup, id) %>%

summarize(sentiment = mean(score),

words = n()) %>%

ungroup() %>%

filter(words >= 5)

###### NOTE

As a simple measure to reduce the role of randomness, we filtered out messages that had fewer than five words that contributed to sentiment.

What were the most positive messages?

sentiment\_messages %>%

arrange(desc(sentiment))

## # A tibble: 3,554 × 4

## newsgroup id sentiment words

## <chr> <chr> <dbl> <int>

## 1 rec.sport.hockey 53560 3.888889 18

## 2 rec.sport.hockey 53602 3.833333 30

## 3 rec.sport.hockey 53822 3.833333 6

## 4 rec.sport.hockey 53645 3.230769 13

## 5 rec.autos 102768 3.200000 5

## 6 misc.forsale 75965 3.000000 5

## 7 misc.forsale 76037 3.000000 5

## 8 rec.sport.baseball 104458 3.000000 11

## 9 rec.sport.hockey 53571 3.000000 5

## 10 comp.os.ms-windows.misc 9620 2.857143 7

## # ... with 3,544 more rows

Let’s check this by looking at the most positive message in the whole dataset. To assist in this, we could write a short function for printing a specified message.

print\_message <- function(group, message\_id) {

result <- cleaned\_text %>%

filter(newsgroup == group, id == message\_id, text != "")

cat(result$text, sep = "\n")

}

print\_message("rec.sport.hockey", 53560)

## Everybody. Please send me your predictions for the Stanley Cup Playoffs!

## I want to see who people think will win.!!!!!!!

## Please Send them in this format, or something comparable:

## 1. Winner of Buffalo-Boston

## 2. Winner of Montreal-Quebec

## 3. Winner of Pittsburgh-New York

## 4. Winner of New Jersey-Washington

## 5. Winner of Chicago-(Minnesota/St.Louis)

## 6. Winner of Toronto-Detroit

## 7. Winner of Vancouver-Winnipeg

## 8. Winner of Calgary-Los Angeles

## 9. Winner of Adams Division (1-2 above)

## 10. Winner of Patrick Division (3-4 above)

## 11. Winner of Norris Division (5-6 above)

## 12. Winner of Smythe Division (7-8 above)

## 13. Winner of Wales Conference (9-10 above)

## 14. Winner of Campbell Conference (11-12 above)

## 15. Winner of Stanley Cup (13-14 above)

## I will summarize the predictions, and see who is the biggest

## INTERNET GURU PREDICTING GUY/GAL.

## Send entries to Richard Madison

## rrmadiso@napier.uwaterloo.ca

## PS: I will send my entries to one of you folks so you know when I say

## I won, that I won!!!!!

It looks like this message was chosen because it uses the word “winner” many times. How about the most negative message? Turns out it’s also from the hockey site, but has a very different attitude.

sentiment\_messages %>%

arrange(sentiment)

## # A tibble: 3,554 × 4

## newsgroup id sentiment words

## <chr> <chr> <dbl> <int>

## 1 rec.sport.hockey 53907 -3.000000 6

## 2 sci.electronics 53899 -3.000000 5

## 3 talk.politics.mideast 75918 -3.000000 7

## 4 rec.autos 101627 -2.833333 6

## 5 comp.graphics 37948 -2.800000 5

## 6 comp.windows.x 67204 -2.700000 10

## 7 talk.politics.guns 53362 -2.666667 6

## 8 alt.atheism 51309 -2.600000 5

## 9 comp.sys.mac.hardware 51513 -2.600000 5

## 10 rec.autos 102883 -2.600000 5

## # ... with 3,544 more rows

print\_message("rec.sport.hockey", 53907)

## Losers like us? You are the fucking moron who has never heard of the Western

## Business School, or the University of Western Ontario for that matter. Why

## don't you pull your head out of your asshole and smell something other than

## shit for once so you can look on a map to see where UWO is! Back to hockey,

## the North Stars should be moved because for the past few years they have

## just been SHIT. A real team like Toronto would never be moved!!!

## Andrew--

Well, we can confidently say that the sentiment analysis worked!

## N-gram Analysis

In [Chapter 4](http://my.safaribooksonline.com/9781491981641/ngrams_html#ngrams), we considered the effect of words such as “not” and “no” on sentiment analysis of Jane Austen novels, such as considering whether a phrase like “don’t like” led to passages incorrectly being labeled as positive. The Usenet dataset is a much larger corpus of more modern text, so we may be interested in how sentiment analysis may be reversed in this text.

We’ll start by finding and counting all the bigrams in the Usenet posts.

usenet\_bigrams <- cleaned\_text %>%

unnest\_tokens(bigram, text, token = "ngrams", n = 2)

usenet\_bigram\_counts <- usenet\_bigrams %>%

count(newsgroup, bigram, sort = **TRUE**) %>%

ungroup() %>%

separate(bigram, c("word1", "word2"), sep = " ")

We could then define a list of six words that we suspect are used in negation, such as “no,” “not,” and “without,” and visualize the sentiment-associated words that most often follow them ([Figure 9-9](http://my.safaribooksonline.com/9781491981641/n_gram_analysis_html#when_they_followed_negating_word)). This shows the words that most often contribute in the “wrong” direction.

negate\_words <- c("not", "without", "no", "can't", "don't", "won't")

usenet\_bigram\_counts %>%

filter(word1 %in% negate\_words) %>%

count(word1, word2, wt = n, sort = **TRUE**) %>%

inner\_join(get\_sentiments("afinn"), by = c(word2 = "word")) %>%

mutate(contribution = score \* nn) %>%

group\_by(word1) %>%

top\_n(10, abs(contribution)) %>%

ungroup() %>%

mutate(word2 = reorder(paste(word2, word1, sep = "\_\_"), contribution)) %>%

ggplot(aes(word2, contribution, fill = contribution > 0)) +

geom\_col(show.legend = **FALSE**) +

facet\_wrap(~ word1, scales = "free", nrow = 3) +

scale\_x\_discrete(labels = function(x) gsub("\_\_.+$", "", x)) +

xlab("Words preceded by a negation") +

ylab("Sentiment score \* # of occurrences") +

theme(axis.text.x = element\_text(angle = 90, hjust = 1)) +

coord\_flip()



###### *Figure 9-9. Words that contribute the most to sentiment when they follow a “negating” word*

It looks like the largest sources of misidentifying a word as positive come from “don’t want/like/care,” and the largest source of incorrectly classified negative sentiment is “no problem.”

# Summary

In this analysis of Usenet messages, we’ve incorporated almost every method for tidy text mining described in this book, ranging from tf-idf to topic modeling, and from sentiment analysis to n-gram tokenization. Throughout the chapter, and indeed through all of our case studies, we’ve been able to rely on a small list of common tools for exploration and visualization. We hope that these examples show how much all tidy text analyses have in common with each other, and indeed with all tidy data analyses.