Sentiment_Analysis_2

Pricilla

2025-10-26

Sentiment Analysis with Tidy Data

#By Silge, J., & Robinson, D. (2017). *Text Mining with R: A Tidy Approach* (Chapter 2: Sentiment Analysis). #O'Reilly Media.

We will reproduce the base Jane Austen sentiment analysis, then extend it to a new corpus (Shakespeare) and include an extra lexicon (`sentimentr`) for sentence level sentiment.

Base Jane Austen Sentiment Analysis

```
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)
```

Most common joy words in Emma

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

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

```
## Joining with `by = join_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
## # i 291 more rows
```

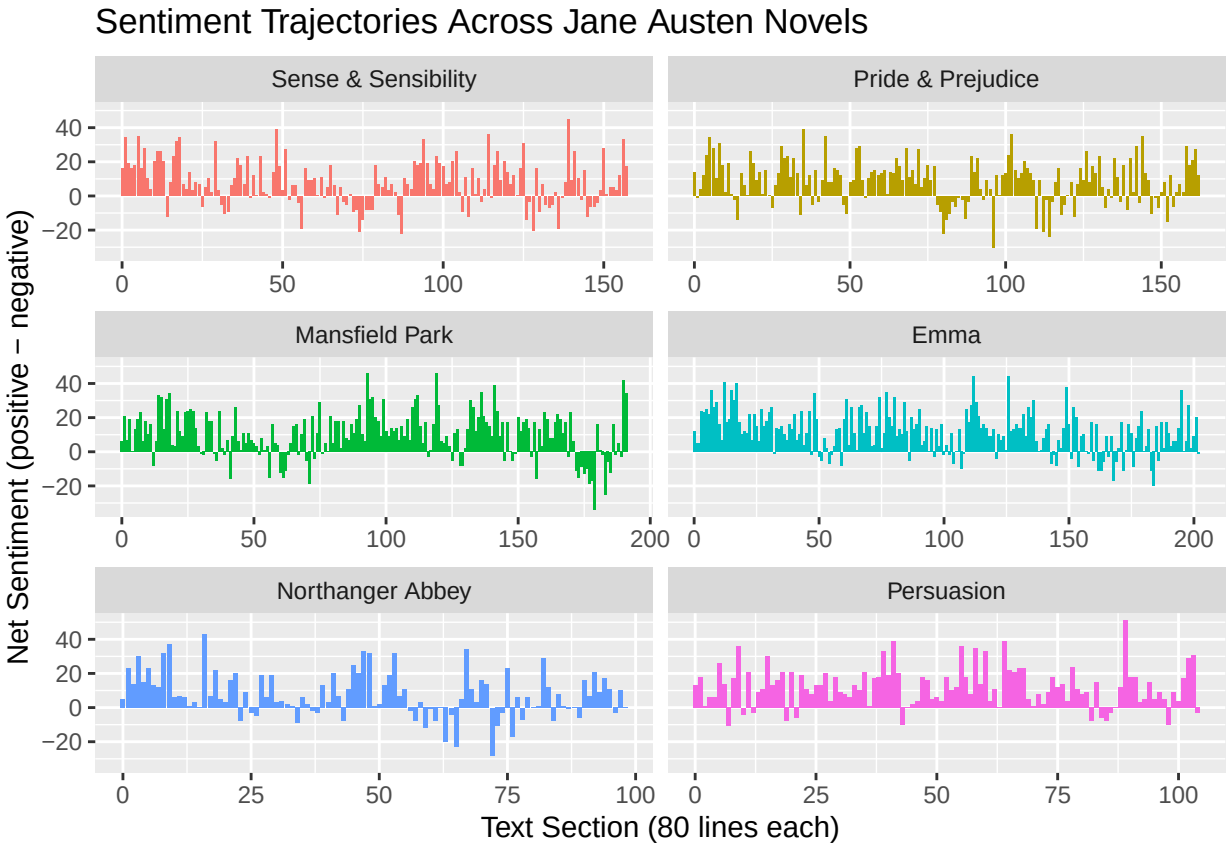
Sentiment trajectory per book (Bing lexicon)

```
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 with `by = join_by(word)`
```

```
## Warning in inner_join(., get_sentiments("bing")): Detected an unexpected many-to-many relationship be
## i Row 435434 of `x` matches multiple rows in `y`.
## i Row 5051 of `y` matches multiple rows in `x`.
## i If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```

```
ggplot(jane_austen_sentiment, aes(index, sentiment, fill = book)) +
geom_col(show.legend = FALSE) +
facet_wrap(~book, ncol = 2, scales = "free_x") +
labs(
x = "Text Section (80 lines each)",
y = "Net Sentiment (positive - negative)",
title = "Sentiment Trajectories Across Jane Austen Novels"
)
```



Pride & Prejudice. The sentiment for Pride & Prejudice is generally positive, with frequent fluctuations between positive and negative sentiments, but a consistently positive trend line. This suggests a narrative with a good balance of drama and conflict resolution, which is often considered a reason for its popularity.

Sense & Sensibility. Sense & Sensibility has a similar pattern to Pride & Prejudice but with lower points of sentiment, which some analyses suggest may have affected its reception. The plot features a dramatic contrast between the characters of Elinor and Marianne, which contributes to the tension and ambiguity in the novel. I enjoyed both the book and movie.

New Corpus Shakespeare (Gutenberg)

Download and tidy Shakespeare plays.

I start by pulling William Shakespeare's works from the Gutenberg Project dataset (via the {gutenbergr} R package), preparing them for text analysis, and adding useful indexing. The data includes gutenberg_id (unique ID for each text), title, author, language and downloads.

This leads to a structured dataset of Shakespeare's plays. Each row represents one line of text, labeled by play title and grouped into 80 line chunks. These chunks are the foundation for sentiment analysis steps (AFINN, Bing, NRC and sentimentr.).

```
shakespeare_meta <- gutenberg_works(author == "Shakespeare, William")
shakespeare_ids <- shakespeare_meta$gutenberg_id[1:6]

shakespeare_raw <- gutenberg_download(shakespeare_ids) %>%
left_join(shakespeare_meta %>% select(gutenberg_id, title), by = "gutenberg_id") %>%
group_by(title) %>%
mutate(linenumber = row_number(), index = linenumber %/% 80) %>%
ungroup()
```

```
## Determining mirror for Project Gutenberg from
## https://www.gutenberg.org/robot/harvest.
## Using mirror http://aleph.gutenberg.org.
```

```
tidy_shakespeare <- shakespeare_raw %>% unnest_tokens(word, text)
```

Apply Multiple Lexicons.

I use AFINN sentiment lexicon to calculate numerical sentiment scores for each chunk of Shakespeare's text. The AFINN lexicon is one of the most widely used sentiment dictionaries in text analysis. It was developed by Finn Årup Nielsen, and it assigns each word an integer sentiment score ranging from -5 to +5, where, Positive numbers indicate positive sentiment (e.g., love = +3, happy = +3) Negative numbers indicate negative sentiment (e.g., hate = -3, terrible = -4) 0 or missing indicates a neutral word or one not found in the lexicon

Unlike categorical lexicons (such as Bing or NRC), AFINN provides numeric scores, which makes it suitable for computing summed or averaged sentiment values across chunks of text. This is useful for comparing overall positivity or negativity over time or across works.

AFINN

```
# Compute AFINN sentiment per chunk

afinn_sh <- tidy_shakespeare %>%
inner_join(get_sentiments("afinn"), by = "word") %>%
group_by(title, index) %>%
summarise(sentiment_afinn = sum(value, na.rm = TRUE), .groups = "drop")
```

AFINN lexicon words in Shakespeare

```
afinn_sh <- tidy_shakespeare %>%
  inner_join(get_sentiments("afinn"), by = "word") %>%
  slice_head(n = 10)
  afinn_sh
```

```
## # A tibble: 10 x 6
##    gutenberg_id title                                linenumber index word  value
##           <int> <chr>                                     <int> <dbl> <chr> <dbl>
##  1          100 The Complete Works of William Shak~         12     0 trag~    -2
##  2          100 The Complete Works of William Shak~         13     0 like      2
##  3          100 The Complete Works of William Shak~         14     0 come~     1
##  4          100 The Complete Works of William Shak~         14     0 erro~    -2
##  5          100 The Complete Works of William Shak~         15     0 trag~    -2
##  6          100 The Complete Works of William Shak~         17     0 trag~    -2
##  7          100 The Complete Works of William Shak~         25     0 death    -2
##  8          100 The Complete Works of William Shak~         26     0 trag~    -2
##  9          100 The Complete Works of William Shak~         27     0 trag~    -2
## 10          100 The Complete Works of William Shak~         28     0 lost     -3
```

Bing & NRC (positive - negative)

Both Bing and NRC lexicons label words as "positive" or "negative" but not with numeric scores like AFINN. Instead of summing numeric values, we Count how many positive words appear, Count how many negative words appear, and Compute a difference score.

Bing and NRC methods both classify words as positive or negative. I count how many of each occur in 80 line text chunks, then compute the difference. This gives you a chunk-level sentiment trend across each Shakespeare play.

```
bing_sh <- tidy_shakespeare %>%
  inner_join(get_sentiments("bing"), by = "word", relationship = "many-to-many") %>%
  count(title, index, sentiment) %>%
  pivot_wider(names_from = sentiment, values_from = n, values_fill = 0) %>%
  mutate(sentiment_bing = positive - negative) %>%
  select(title, index, sentiment_bing)


# --- NRC sentiment (positive - negative) per chunk ---
nrc_sh <- tidy_shakespeare %>%
  inner_join(
    get_sentiments("nrc") %>% filter(sentiment %in% c("positive", "negative")),
    by = "word",
    relationship = "many-to-many"
  ) %>%
  count(title, index, sentiment) %>%
  pivot_wider(names_from = sentiment, values_from = n, values_fill = 0) %>%
  mutate(sentiment_nrc = positive - negative) %>%
  select(title, index, sentiment_nrc)
```

Bing lexicon words in Shakespeare.

Each row represents a single word in Shakespeare's text that was found in the Bing sentiment lexicon. The Bing lexicon has matched emotional polarity (positive/negative) to Shakespeare's words. The index = 0 means these all come from the very beginning of The Complete Works of William Shakespeare.

These word level matches are what get aggregated per chunk (in earlier code) to produce sentiment_bing positive - negative.

```
# Bing lexicon words in Shakespeare
bing_words <- tidy_shakespeare %>%
  inner_join(get_sentiments("bing"), by = "word", relationship = "many-to-many") %>%
  slice_head(n = 10)
bing_words
```

```
## # A tibble: 10 x 6
##    gutenberg_id title                              linenumber index word  sentiment
##           <int> <chr>                                   <int> <dbl> <chr> <chr>
##  1          100 The Complete Works of William ~          1     0 works positive
##  2          100 The Complete Works of William ~         11     0 well  positive
##  3          100 The Complete Works of William ~         11     0 well  positive
##  4          100 The Complete Works of William ~         12     0 trag~ negative
##  5          100 The Complete Works of William ~         13     0 like  positive
##  6          100 The Complete Works of William ~         14     0 erro~ negative
##  7          100 The Complete Works of William ~         15     0 trag~ negative
##  8          100 The Complete Works of William ~         17     0 trag~ negative
##  9          100 The Complete Works of William ~         25     0 death negative
## 10          100 The Complete Works of William ~         26     0 trag~ negative
```

NRC lexicon words in Shakespeare.

This output shows how words in Shakespeare's text being matched to the NRC sentiment lexicon, which is richer than AFINN or Bing. Unlike Bing or AFINN, which give each word one polarity (positive/negative or numeric score), the NRC Emotion Lexicon assigns multiple emotional categories to a single word. Multiple emotional categories may overlap for the same word.

```
nrc_words <- tidy_shakespeare %>%
  inner_join(get_sentiments("nrc"), by = "word", relationship = "many-to-many") %>%
  slice_head(n = 10)

nrc_words
```

```
## # A tibble: 10 x 6
##    gutenberg_id title                              linenumber index word  sentiment
##           <int> <chr>                                   <int> <dbl> <chr> <chr>
##  1          100 The Complete Works of William ~         12     0 trag~ fear
##  2          100 The Complete Works of William ~         12     0 trag~ negative
##  3          100 The Complete Works of William ~         12     0 trag~ sadness
##  4          100 The Complete Works of William ~         15     0 trag~ fear
##  5          100 The Complete Works of William ~         15     0 trag~ negative
##  6          100 The Complete Works of William ~         15     0 trag~ sadness
##  7          100 The Complete Works of William ~         17     0 trag~ fear
##  8          100 The Complete Works of William ~         17     0 trag~ negative
##  9          100 The Complete Works of William ~         17     0 trag~ sadness
## 10          100 The Complete Works of William ~         17     0 prin~ positive
```

View NRC lexicon structure

```
nrc_lexicon <- get_sentiments("nrc")
head(nrc_lexicon, 10)
```

```
## # A tibble: 10 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
```

Filter NRC lexicon for joy words.

This is a focused sentiment analysis example using the NRC lexicon, specifically isolating words associated with joy. I Filter NRC for joyful words, find which ones appear in Shakespeare and count their frequencies.

The output table reveals which positive or uplifting words Shakespeare used most often, giving a glimpse into the emotional tone of his writing.

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

#Count the most common joy words in Shakespeare plays

tidy_shakespeare %>%
inner_join(nrc_joy, by = "word") %>%
count(word, sort = TRUE) %>%
slice_head(n = 10)
```

```
## # A tibble: 10 x 2
##    word         n
##    <chr>    <int>
##  1 good      3145
##  2 love      2586
##  3 art       1135
##  4 sweet     1014
##  5 true       975
##  6 god        928
##  7 pray       814
##  8 peace      616
##  9 friend     610
## 10 daughter   572
```

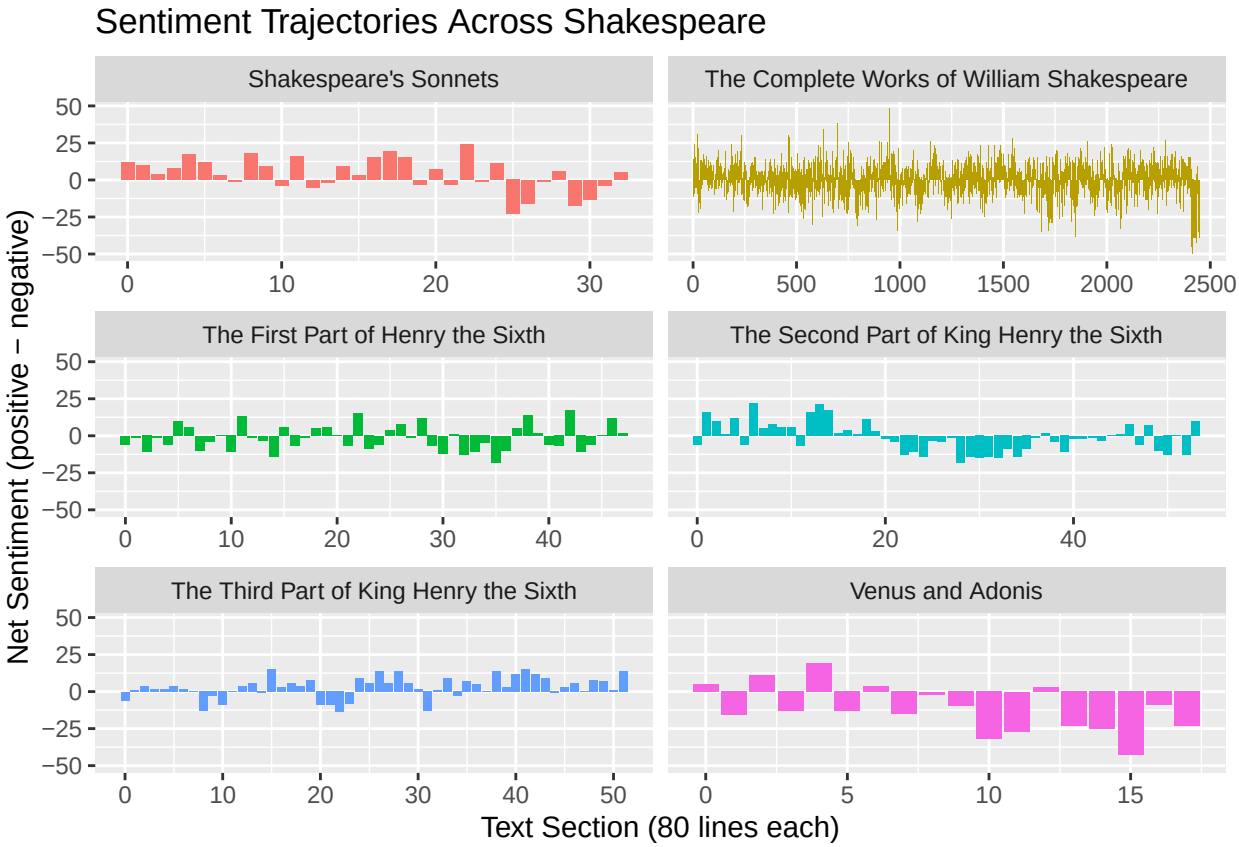
Sentiment trajectory using Bing lexicon.

This plot shows the sentiment trajectories (emotional ups and downs) across several of Shakespeare's works, using text-mined sentiment scores.

```r
shakespeare_sentiment <- tidy_shakespeare %>%
inner_join(get_sentiments("bing"), relationship = "many-to-many") %>%
count(title, index = linenumber %/% 80, sentiment) %>%
pivot_wider(names_from = sentiment, values_from = n, values_fill = 0) %>%
mutate(sentiment = positive - negative)
```

```
## Joining with `by = join_by(word)`
```

```r
ggplot(shakespeare_sentiment, aes(index, sentiment, fill = title)) +
geom_col(show.legend = FALSE) +
facet_wrap(~title, ncol = 2, scales = "free_x") +
labs(
x = "Text Section (80 lines each)",
y = "Net Sentiment (positive - negative)",
title = "Sentiment Trajectories Across Shakespeare"
)
```



What the plot shows.

Bars above 0 depict more positive sections (e.g., love, joy, success) Bars below 0 depict negative sections (e.g., death, anger, tragedy) Flat, near 0 are Neutral or balanced sentiment. Jagged pattern are frequent emotional changes or mood swings in text.

Shakespeare's Sonnets.

Bars hover near zero, with mild ups and downs. Sentiment fluctuates gently with a mix of joy, love, and melancholy.

The Complete Works of William Shakespeare

Very jagged pattern with wide swings. Reflects many works combined, highs and lows cancel out, showing overall variation in tone.

The First, Second, Third Part of Henry the Sixth.

Mostly near zero with small positive bursts. These historical plays have a relatively balanced tone with some scenes uplifting, others grim.

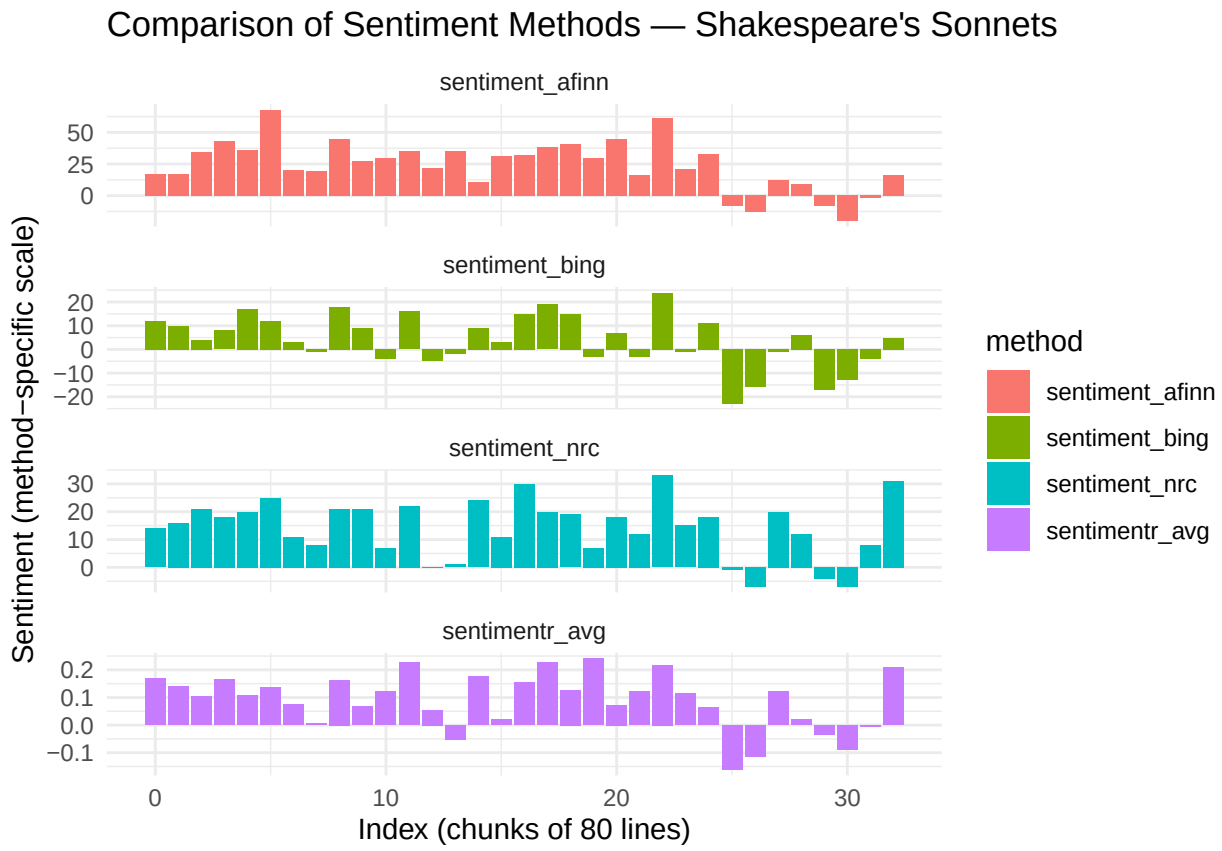
Venus and Adonis Bars skew below zero (mostly negative). This poem carries a more tragic or somber tone overall.

Below is a comparison of four different sentiment analysis methods applied to Shakespeare's Sonnets.

While the absolute values and scales differ, the overall trends of the peaks and troughs in sentiment appear to be roughly similar across the methods. Shakespeare's sonnets are known for exploring a range of themes, including love, time, death, and beauty, and contain both positive and negative emotions, which is reflected in the sentiment analysis.

```r
# --- AFINN Sentiment ---
afinn_sh <- tidy_shakespeare %>%
  inner_join(get_sentiments("afinn"), by = "word", relationship = "many-to-many") %>%
  group_by(title, index) %>%
  summarise(sentiment_afinn = sum(value, na.rm = TRUE), .groups = "drop")

# --- Bing Sentiment (positive - negative) ---
bing_sh <- tidy_shakespeare %>%
  inner_join(get_sentiments("bing"), by = "word", relationship = "many-to-many") %>%
  count(title, index, sentiment) %>%
  pivot_wider(names_from = sentiment, values_from = n, values_fill = 0) %>%
  mutate(sentiment_bing = positive - negative) %>%
  select(title, index, sentiment_bing)

# --- NRC Sentiment (positive - negative) ---
nrc_sh <- tidy_shakespeare %>%
  inner_join(
    get_sentiments("nrc") %>% filter(sentiment %in% c("positive", "negative")),
    by = "word",
    relationship = "many-to-many"
  ) %>%
  count(title, index, sentiment) %>%
  pivot_wider(names_from = sentiment, values_from = n, values_fill = 0) %>%
  mutate(sentiment_nrc = positive - negative) %>%
  select(title, index, sentiment_nrc)

# --- Sentence-level sentiment with sentimentr ---
# Combine lines into chunks first
chunks <- shakespeare_raw %>%
  group_by(title, index) %>%
  summarise(text_chunk = paste(text, collapse = " "), .groups = "drop")

# Use get_sentences() before sentiment_by() to avoid warnings
sentence_list <- get_sentences(chunks$text_chunk)
sentimentr_results <- sentiment_by(sentence_list)
chunks$sentimentr_avg <- sentimentr_results$ave_sentiment

# --- Combine all sentiment measures ---
shakespeare_sentiment_chunks <- afinn_sh %>%
  left_join(bing_sh, by = c("title", "index")) %>%
  left_join(nrc_sh, by = c("title", "index")) %>%
  left_join(chunks %>% select(title, index, sentimentr_avg), by = c("title", "index"))

# --- Compare methods for one play ---
example_title <- unique(shakespeare_sentiment_chunks$title)[1]

compare_df <- shakespeare_sentiment_chunks %>%
  filter(title == example_title) %>%
  pivot_longer(
    cols = c(sentiment_afinn, sentiment_bing, sentiment_nrc, sentimentr_avg),
    names_to = "method",
    values_to = "score"
  )
```

```
# --- Plot comparison ---
ggplot(compare_df, aes(index, score, fill = method)) +
  geom_col(position = "dodge") +
  facet_wrap(~method, ncol = 1, scales = "free_y") +
  labs(
    title = paste("Comparison of Sentiment Methods -", example_title),
    x = "Index (chunks of 80 lines)",
    y = "Sentiment (method-specific scale)"
  ) +
  theme_minimal()
```



‘‘‘

The graph uses four bar charts, one for each method,

Sentiment_afinn (Red/Salmon): This method's scores are all positive, with values generally ranging from around 10 to 60.

Sentiment_bing (Green): This method produces scores that fluctuate between positive and negative, mostly within the range of about -10 to 20.

Sentiment_nrc (Cyan/Teal): These scores are predominantly positive, spanning from just below 0 to about 30. There are a few instances of zero or near-zero scores.

Sentimentr_avg (Purple): This method's scores are on a much smaller scale, ranging from approximately -0.1 to 0.2, and also show both positive and negative values.

All four methods show fluctuations in sentiment across the different chunks of the sonnets.

In general, when one method shows a relatively high or low point, the others often show a corresponding, though not identically scaled, high or low point, suggesting some agreement on the relative sentiment of certain text chunks.

The most striking difference is the 'sentiment_afinn' and 'sentiment_nrc' consistently assign high positive scores, while 'sentiment_bing' and 'sentimentr_avg' frequently report negative sentiment for various chunks. This difference highlights that different sentiment dictionaries or algorithms measure sentiment differently (e.g., some might only measure positive,negative counts, while others might assign intensity scores and average them).

This visualization demonstrates the variability in results when different computational methods are used to measure the emotional or subjective content (sentiment) within the same body of text.

Summary. I found sentiment analysis interesting and instructive, particularly as an avid reader. The application of this method to literature, specifically classic novels like Sense and Sensibility and Pride and Prejudice, was a fascinating experience. I am impressed and somewhat amazed by how the computational interpretation of a text's positivity and negativity proved to be mostly accurate when compared to my own reading and understanding of Sense and Sensibility, and Pride and Prejudice.