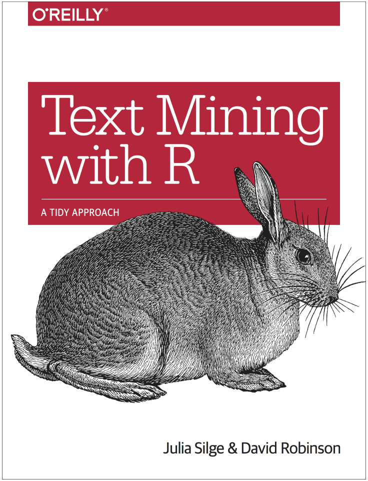
Socio-Informatics 348

Text Analysis
Sentiment Analysis

Dr Lisa Martin

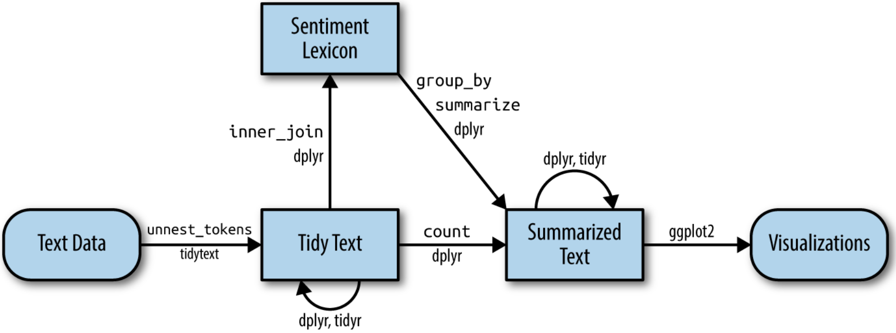
Department of Information Science
Stellenbosch University

Today’s Reading



Text Mining with R, Chapter 2

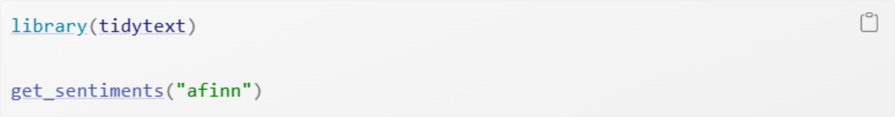
Today's Reading



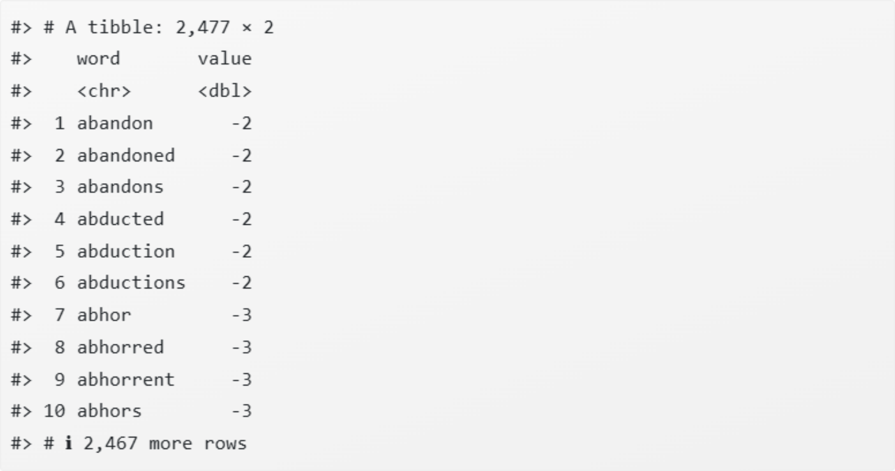
Sentiment Lexicons

- A lexicon is a dictionary of words with associated sentiment values
- Can be binary (positive/negative) or on a scale (e.g., -5 to +5)
- Examples:
  - AFINN (-5 to +5 scale)
  - Bing (positive/negative)
  - NRC (positive/negative categories: anger, fear, joy, etc.)
- All three of these lexicons are based on unigrams, i.e., single words.

Sentiment Lexicons

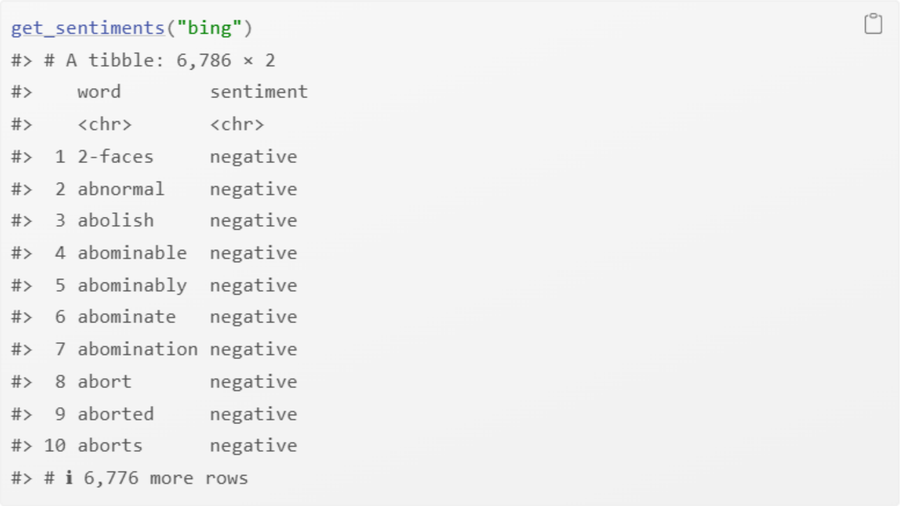


```r
library(tidytext)


get_sentiments("afinn")
```



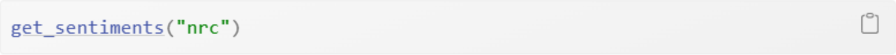
```
#> # A tibble: 2,477 × 2
#>    word       value
#>    <chr>      <dbl>
#>  1 abandon       -2
#>  2 abandoned     -2
#>  3 abandons      -2
#>  4 abducted      -2
#>  5 abduction     -2
#>  6 abductions    -2
#>  7 abhor         -3
#>  8 abhorred      -3
#>  9 abhorrent     -3
#> 10 abhors        -3
#> # ℹ 2,467 more rows
```

Sentiment Lexicons

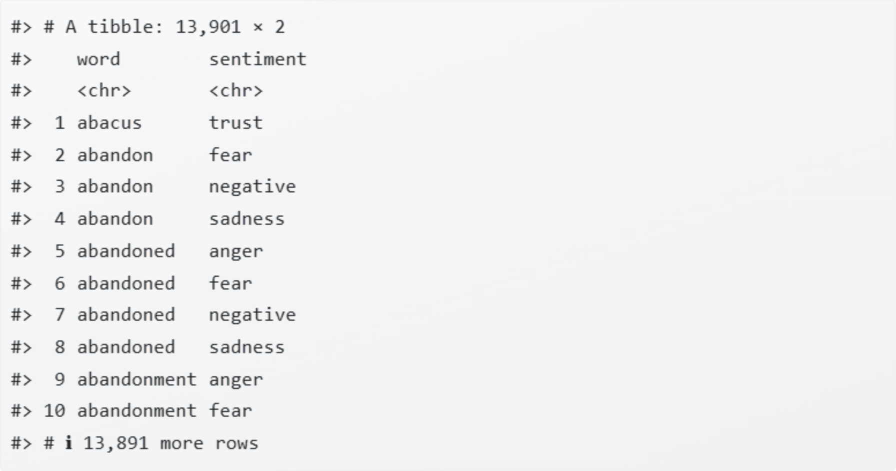


```
get_sentiments("bing")
#> # A tibble: 6,786 × 2
#>    word        sentiment
#>    <chr>       <chr>
#>  1 2-faces     negative
#>  2 abnormal    negative
#>  3 abolish     negative
#>  4 abominable  negative
#>  5 abominably  negative
#>  6 abominate   negative
#>  7 abomination negative
#>  8 abort       negative
#>  9 aborted     negative
#> 10 aborts      negative
#> # ℹ 6,776 more rows
```

Sentiment Lexicons

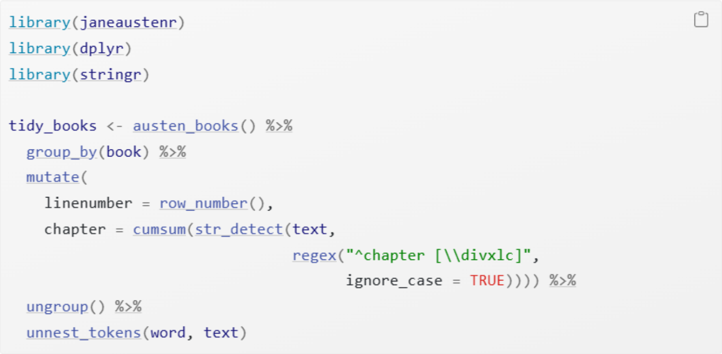


```r
get_sentiments("nrc")
```

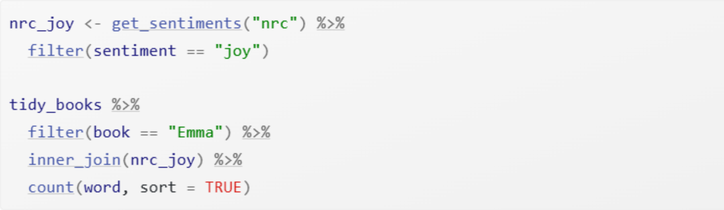


```
#> # A tibble: 13,901 × 2
#>    word        sentiment
#>    <chr>       <chr>
#>  1 abacus      trust
#>  2 abandon     fear
#>  3 abandon     negative
#>  4 abandon     sadness
#>  5 abandoned   anger
#>  6 abandoned   fear
#>  7 abandoned   negative
#>  8 abandoned   sadness
#>  9 abandonment anger
#> 10 abandonment fear
#> # ℹ 13,891 more rows
```

Example 1 with Inner Join

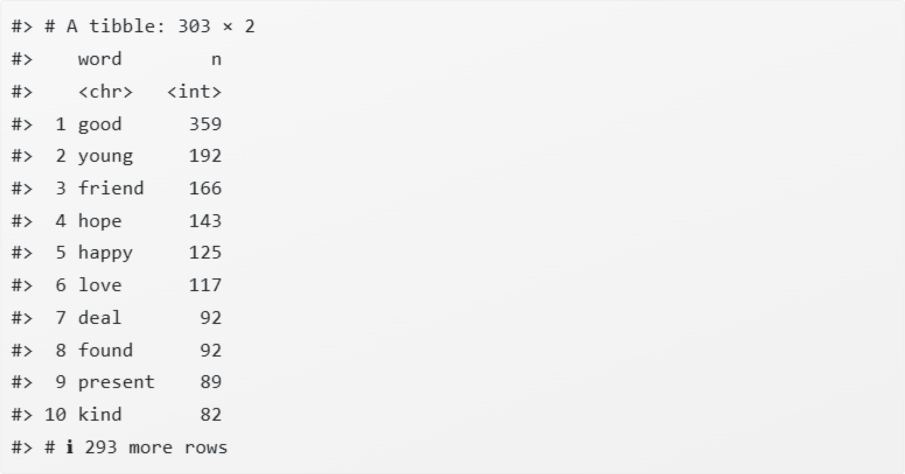


```r
library(janeaustenr)
library(dplyr)
library(stringr)

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)
```



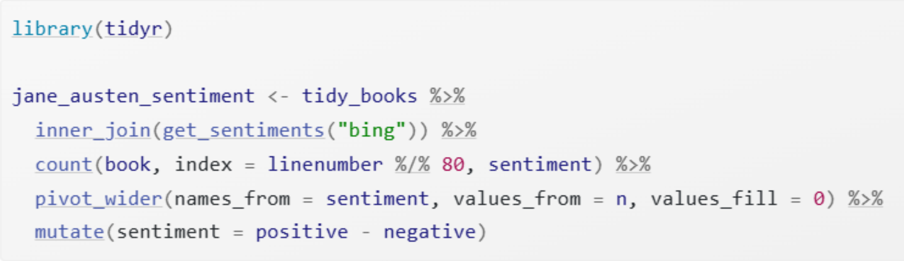
```r
nrc_joy <- get_sentiments("nrc") %>%
  filter(sentiment == "joy")

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

Example 1 with Inner Join

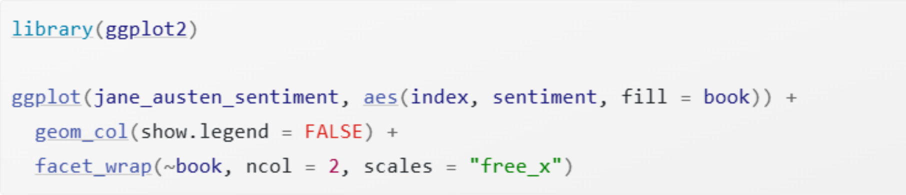


```
#> # A tibble: 303 × 2
#>    word        n
#>    <chr>   <int>
#>  1 good      359
#>  2 young     192
#>  3 friend    166
#>  4 hope      143
#>  5 happy     125
#>  6 love      117
#>  7 deal       92
#>  8 found      92
#>  9 present    89
#> 10 kind       82
#> # ℹ 293 more rows
```

Example 2 with Inner Join

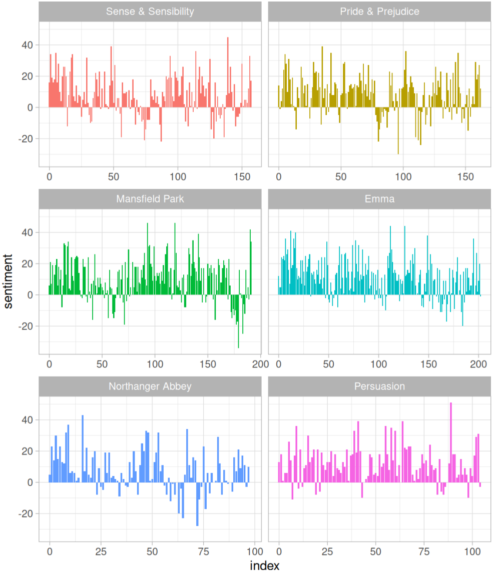


```r
library(tidyr)

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)
```

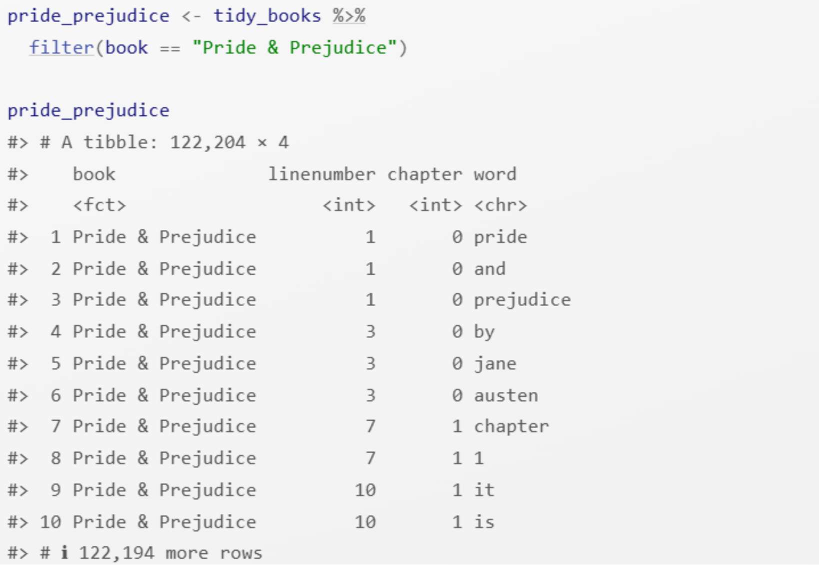


```r
library(ggplot2)

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

Example 2 with Inner Join

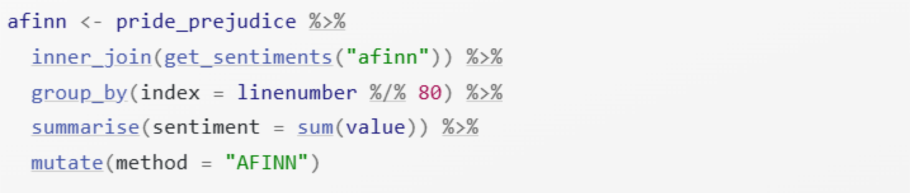


Comparing Sentiment Dictionaries



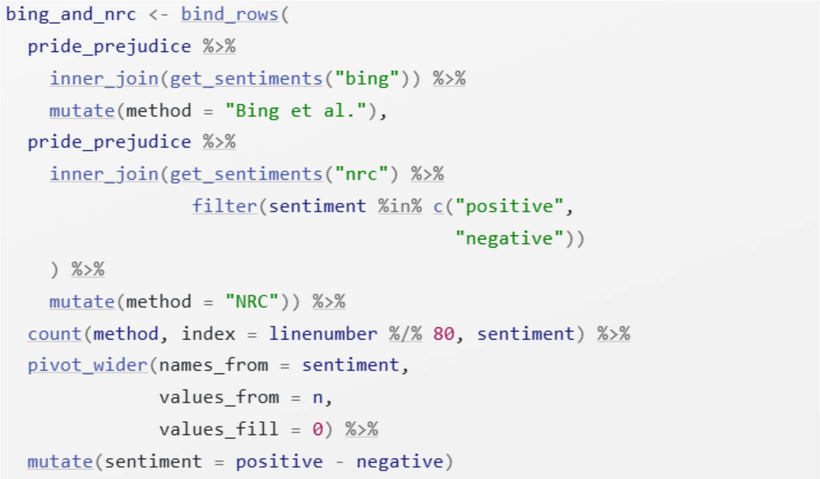
```
pride_prejudice <- tidy_books %>%
  filter(book == "Pride & Prejudice")

pride_prejudice
#> # A tibble: 122,204 × 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
#> # ℹ 122,194 more rows
```

Comparing Sentiment Dictionaries



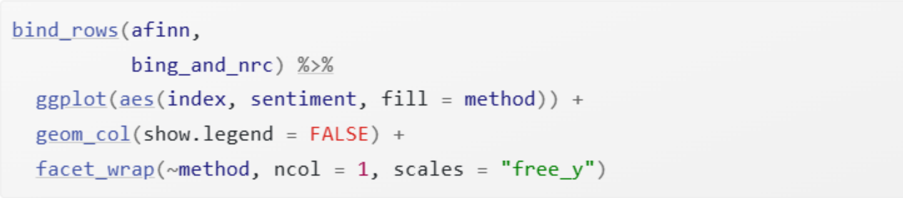
```r
afinn <- pride_prejudice %>%
  inner_join(get_sentiments("afinn")) %>%
  group_by(index = linenumber %/% 80) %>%
  summarise(sentiment = sum(value)) %>%
  mutate(method = "AFINN")
```

Comparing Sentiment Dictionaries



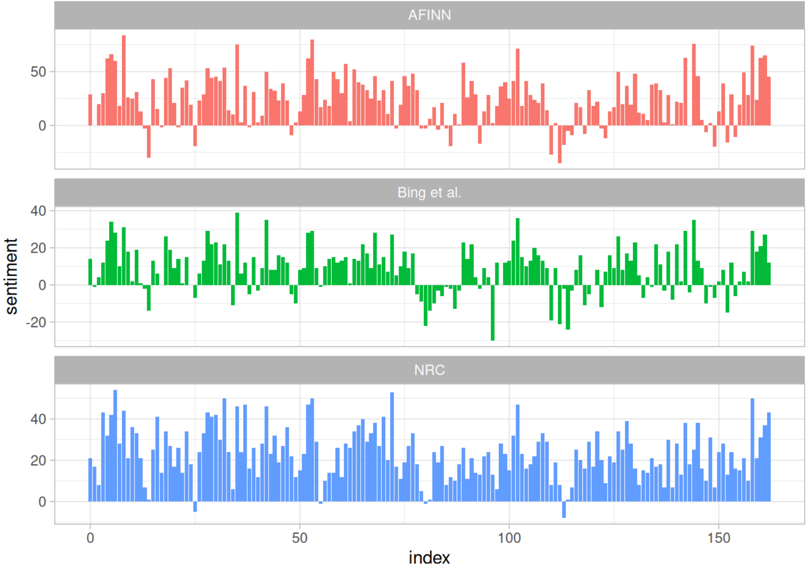
```
bing_and_nrc <- bind_rows(
  pride_prejudice %>%
    inner_join(get_sentiments("bing")) %>%
    mutate(method = "Bing et al."),
  pride_prejudice %>%
    inner_join(get_sentiments("nrc") %>%
                 filter(sentiment %in% c("positive",
                                         "negative"))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

Comparing Sentiment Dictionaries

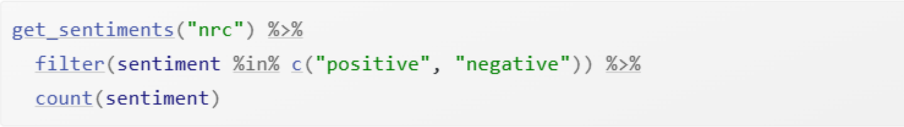


```r
bind_rows(afinn,
          bing_and_nrc) %>%
  ggplot(aes(index, sentiment, fill = method)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~method, ncol = 1, scales = "free_y")
```

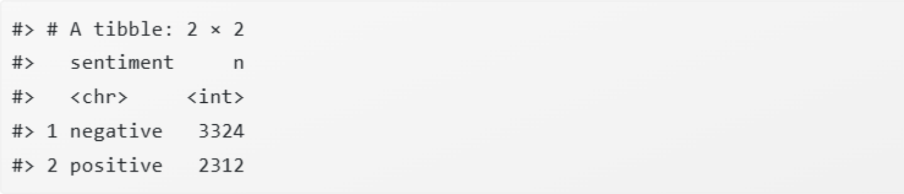
Comparing Sentiment Dictionaries



Comparing Sentiment Dictionaries

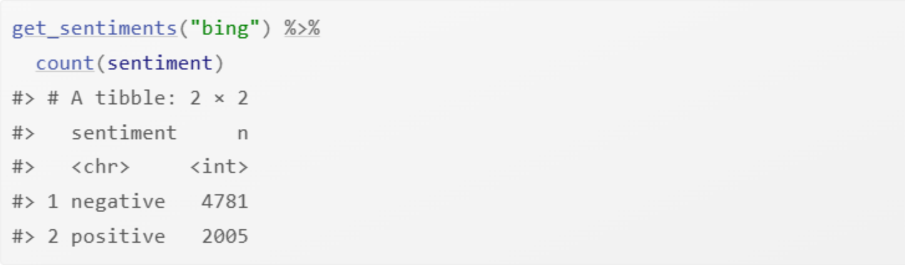


```r
get_sentiments("nrc") %>%
  filter(sentiment %in% c("positive", "negative")) %>%
  count(sentiment)
```

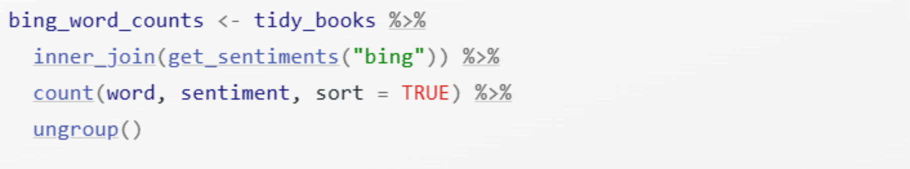


```
#> # A tibble: 2 × 2
#>   sentiment     n
#>   <chr>     <int>
#> 1 negative   3324
#> 2 positive   2312
```

Comparing Sentiment Dictionaries

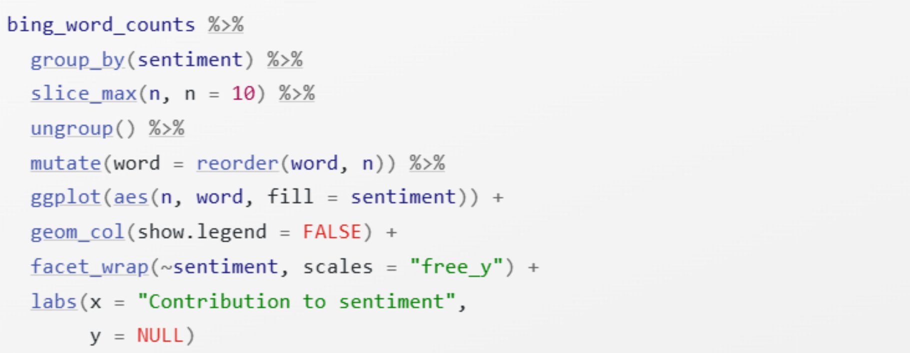


```
get_sentiments("bing") %>%
  count(sentiment)
#> # A tibble: 2 × 2
#>   sentiment     n
#>   <chr>     <int>
#> 1 negative   4781
#> 2 positive   2005
```



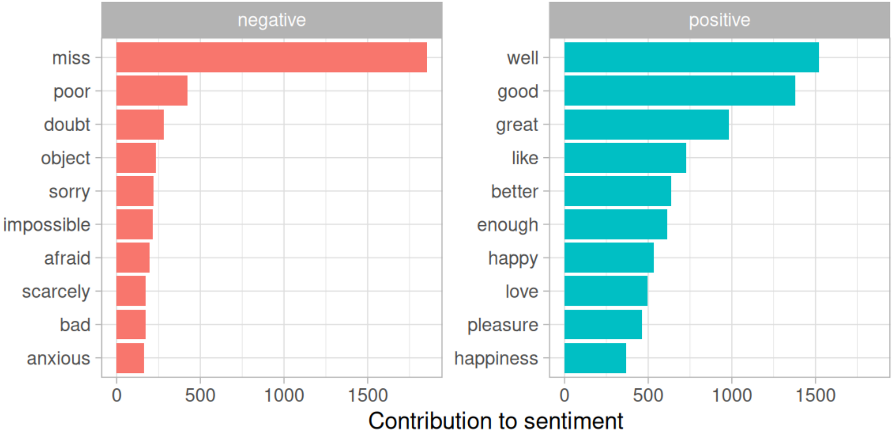
```
bing_word_counts <- tidy_books %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()
```

Most common positive and negative words



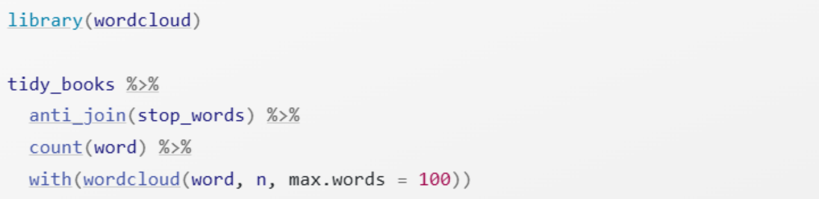
```
bing_word_counts %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(x = "Contribution to sentiment",
       y = NULL)
```

Most common positive and negative words



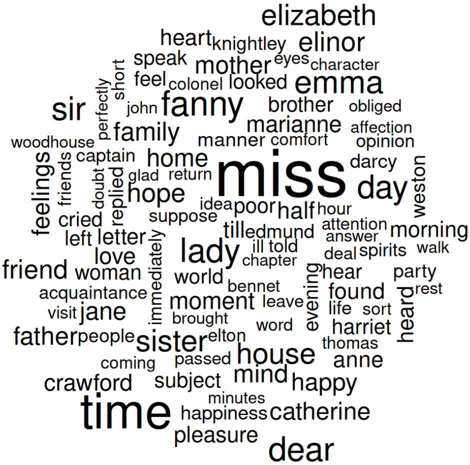
Note: 'miss' is captured as a negative word, but in this context it is not. We can remove it from the lexicon if we want to (by adding it to the stopwords).

Word Clouds

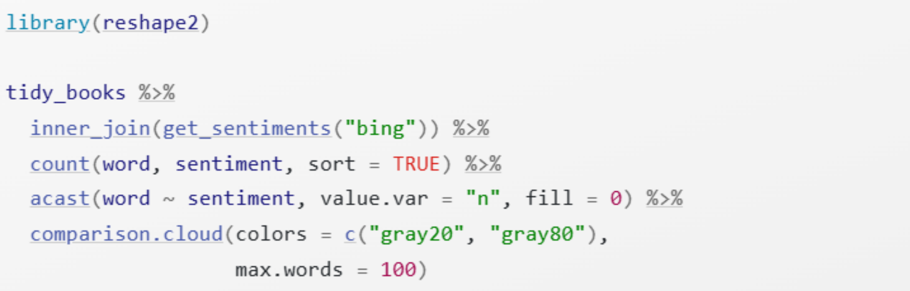


```
library(wordcloud)

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

Word Clouds

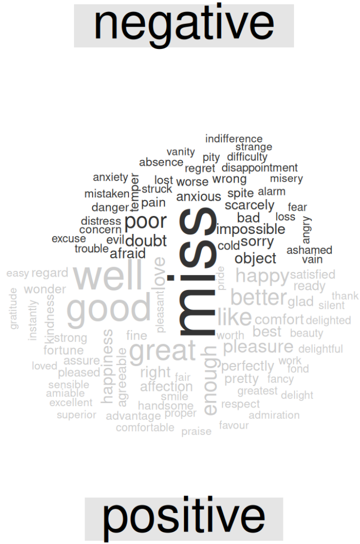

elizabeth
heart
knightley
elinor
speak
mother
eyes
character
short
feel
colonel
looked
emma
perfectly
sir
john
fanny
brother
obliged
family
marianne
affection
woodhouse
manner
comfort
opinion
captain
home
darcy
feelings
friends
doubt
replied
glad
return
miss
day
weston
hope
idea
poor
half
hour
cried
suppose
attention
morning
left
letter
till
edmund
answer
immediately
ill
told
walk
love
lady
chapter
deal
spirits
friend
woman
world
evening
hear
party
acquaintance
bennet
found
rest
moment
leave
heard
visit
jane
brought
life
sort
father
people
sister
elton
word
harriet
thomas
coming
passed
house
anne
crawford
subject
mind
happy
time
minutes
happiness
catherine
pleasure
dear

Word Clouds



```r
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("gray20", "gray80"),
                   max.words = 100)
```

Word Clouds



negative
indifference
strange
vanity
pity
difficulty
absence
regret
disappointment
anxiety
temper
lost
worse
wrong
misery
mistaken
struck
spite
alarm
danger
pain
anxious
scarcely
fear
distress
poor
bad
loss
concern
miss
impossible
angry
excuse
evil
doubt
cold
sorry
ashamed
trouble
afraid
object
vain
easy
regard
well
love
pride
happy
satisfied
wonder
ready
gratitude
instantly
kindness
good
pleasant
better
glad
thank
silent
like
comfort
delighted
strong
fine
worth
best
beauty
fortune
happiness
agreeable
great
enough
pleasure
delightful
loved
assure
perfectly
work
fond
pleased
right
fair
pretty
fancy
sensible
affection
greatest
amiable
smile
delight
excellent
handsome
respect
superior
advantage
proper
admiration
comfortable
favour
praise
positive