Text Analysis

Freelancing

12/21/2021

```
#Loading libraries

library(gutenbergr)
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(tidyr)
library(stringr)
library(tidytext)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.5     v forcats 0.5.1
## v readr   2.0.2
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(tm)
```

```
## Loading required package: NLP
```

```
##
## Attaching package: 'NLP'
```

```
## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```
library(ggplot2)
library(textdata)
library(wordcloud)
```

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

```
# Reading the books chosen for sentiment analysis in the gutenbergr library

# reading the Moby Dick

moby_book_ref <- gutenberg_works(title == "Moby Dick")
moby_book <- gutenberg_download(moby_book_ref$gutenberg_id)
```

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

```
## Using mirror http://aleph.gutenberg.org
```

```
moby_book
```

```
## # A tibble: 22,243 x 2
##    gutenberg_id text
##           <int> <chr>
##  1           15 "Moby-Dick"
##  2           15 ""
##  3           15 "or,"
##  4           15 ""
##  5           15 "THE WHALE."
##  6           15 ""
##  7           15 "by Herman Melville"
##  8           15 ""
##  9           15 ""
## 10           15 "Contents"
## # ... with 22,233 more rows
```

```
# reading The Wonderful Wizard of Oz Book
Oz_book_ref <- gutenberg_works(title == "The Wonderful Wizard of Oz")
Oz_book <- gutenberg_download(Oz_book_ref$gutenberg_id)
Oz_book
```

```
## # A tibble: 4,750 x 2
##    gutenberg_id text
##           <int> <chr>
##  1           55 "[Illustration]"
##  2           55 ""
##  3           55 ""
##  4           55 ""
##  5           55 ""
##  6           55 "The Wonderful Wizard of Oz"
##  7           55 ""
##  8           55 "by L. Frank Baum"
##  9           55 ""
## 10           55 ""
## # ... with 4,740 more rows
```

```
# Removing the parts of data we dont need using the slice function and then filtering the text data to

#Moby book filtering
moby_filtered = moby_book  %>%
  slice(-(1:199)) %>%
  filter(!text==str_to_upper(text),   # will remove THE PROLOGUE etc.
         !text==str_to_title(text),   # will remove names/single word lines
         !str_detect(text, pattern='^(Scene|SCENE)|^(Act|ACT)|^\\[')) %>%
```

```
  select(-gutenberg_id) %>%
  unnest_tokens(sentence, input=text, token='sentences') %>%
  mutate(Line_number = 1:n())

#The Wonderful Wizard of Oz filtering
Oz_book_filtered = Oz_book %>%
  slice(-(1:49)) %>%
  filter(!text==str_to_upper(text),   # will remove THE PROLOGUE etc.
         !text==str_to_title(text),   # will remove names/single word lines
         !str_detect(text, pattern='^(Scene|SCENE)|^(Act|ACT)|^\\[')) %>%
  select(-gutenberg_id) %>%
  unnest_tokens(sentence, input=text, token='sentences') %>%
  mutate(Line_number = 1:n())
```

```
 # Moby book analysis

stop_words$word[which(stop_words$word %in% sentiments$word)]
```

```
##   [1] "appreciate"    "appropriate"   "available"     "awfully"
##   [5] "best"          "better"        "clearly"       "enough"
##   [9] "like"          "liked"         "reasonably"    "right"
##  [13] "sensible"      "sorry"         "thank"         "unfortunately"
##  [17] "unlikely"      "useful"        "welcome"       "well"
##  [21] "willing"       "wonder"        "best"          "better"
##  [25] "clear"         "clearly"       "enough"        "evenly"
##  [29] "good"          "great"         "greatest"      "important"
##  [33] "interesting"   "interests"     "like"          "problem"
##  [37] "problems"      "right"         "right"         "well"
##  [41] "work"          "worked"        "works"
```

```
moby_filtered = moby_filtered %>%
  unnest_tokens(output=word, input=sentence, token='words') %>%
  anti_join(stop_words)
```

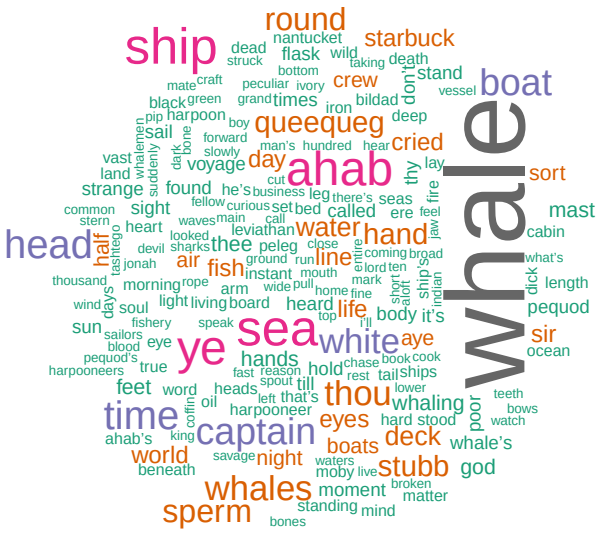
```
## Joining, by = "word"
```

```
summary(moby_filtered)
```

```
##   Line_number        word
##  Min.   :    1   Length:84823
##  1st Qu.: 6664   Class :character
##  Median :12835   Mode  :character
##  Mean   :12799
##  3rd Qu.:19036
##  Max.   :25292
```

```
# Fequency of words in the text (table 1)
moby_filtered_count <- moby_filtered %>%
                           count(word) %>%
                           arrange(desc(n))


# word count visualizing the most frequents words on the Moby book
 wordcloud(words =  moby_filtered_count$word, freq = moby_filtered_count   $n,scale=c(3.5,0.25),
          max.words=200, colors=brewer.pal(8, "Dark2"))
```




```
# Words and their contribution to the sentiments using the bing (positive or negative  lexicon
bing_word_counts <- moby_filtered%>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()
```

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

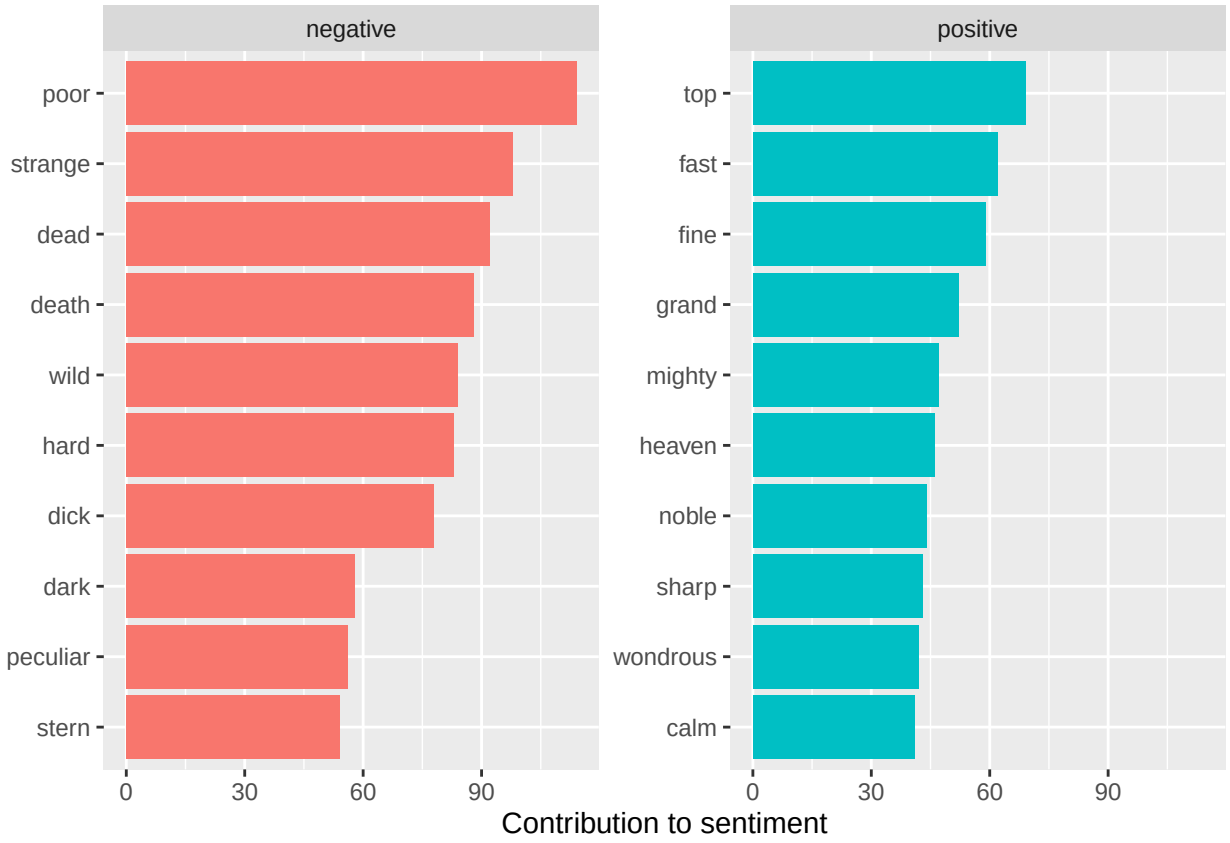
```
#Table 1
print(bing_word_counts)
```

```
## # A tibble: 2,362 x 3
##    word    sentiment     n
##    <chr>   <chr>     <int>
##  1 poor    negative    114
##  2 strange negative     98
##  3 dead    negative     92
##  4 death   negative     88
##  5 wild    negative     84
##  6 hard    negative     83
##  7 dick    negative     78
##  8 top     positive     69
##  9 fast    positive     62
## 10 fine    positive     59
## # ... with 2,352 more rows
```

```
summary(bing_word_counts)
```

```
##      word             sentiment                n          
##  Length:2362        Length:2362        Min.   :  1.000  
##  Class :character   Class :character   1st Qu.:  1.000  
##  Mode  :character   Mode  :character   Median :  2.000  
##                                        Mean   :  4.933  
##                                        3rd Qu.:  5.000  
##                                        Max.   :114.000
```

```
# words that contribute to the Sentiments (classified as Positive or negative )
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)
```




```
#Using the nrc lexicon
nrc_word_counts <- moby_filtered%>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()
```

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

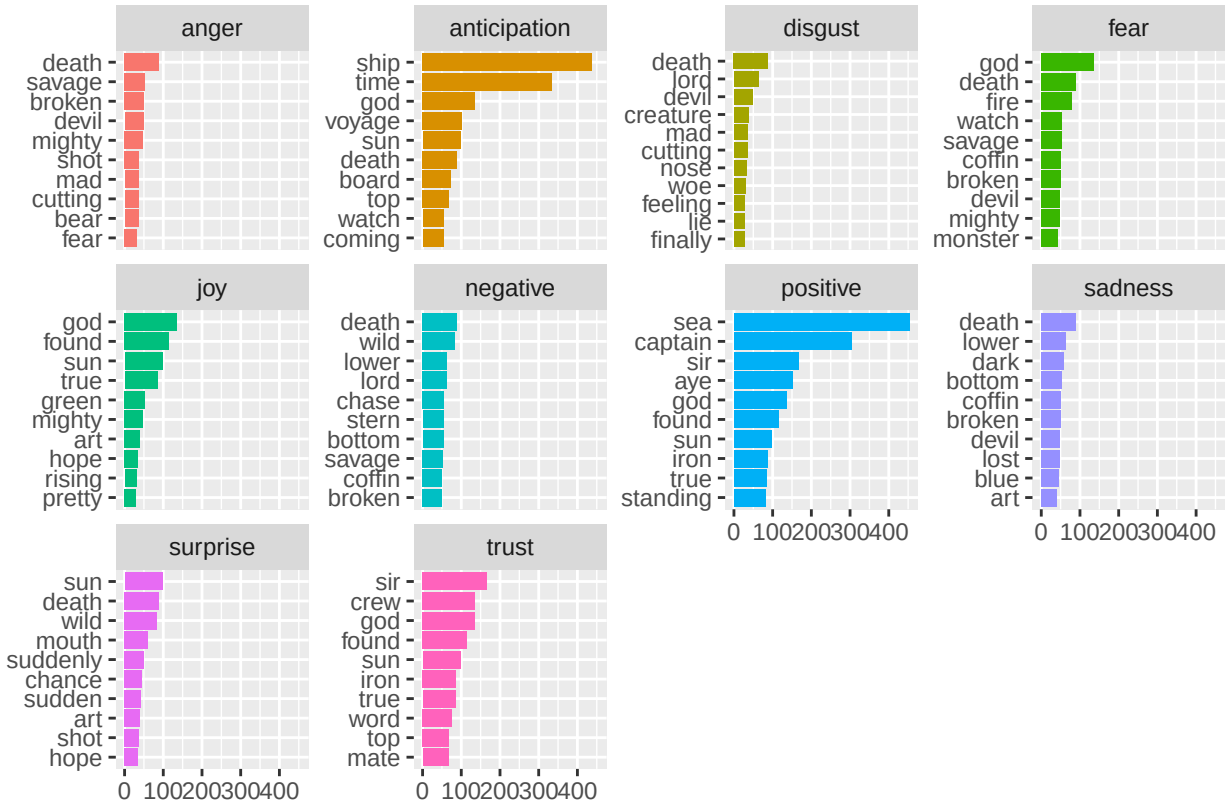
```r
#Table 2
print(nrc_word_counts)
```

```
## # A tibble: 6,789 x 3
##    word    sentiment        n
##    <chr>   <chr>        <int>
##  1 sea     positive       453
##  2 ship    anticipation   437
##  3 time    anticipation   334
##  4 captain positive       304
##  5 sir     positive       167
##  6 sir     trust          167
##  7 aye     positive       152
##  8 crew    trust          136
##  9 god     anticipation   135
## 10 god     fear           135
## # ... with 6,779 more rows
```

```r
summary(nrc_word_counts)
```

```
##      word            sentiment              n          
##  Length:6789        Length:6789        Min.   :  1.00  
##  Class :character   Class :character   1st Qu.:  1.00  
##  Mode  :character   Mode  :character   Median :  3.00  
##                                        Mean   :  6.26  
##                                        3rd Qu.:  6.00  
##                                        Max.   :453.00
```

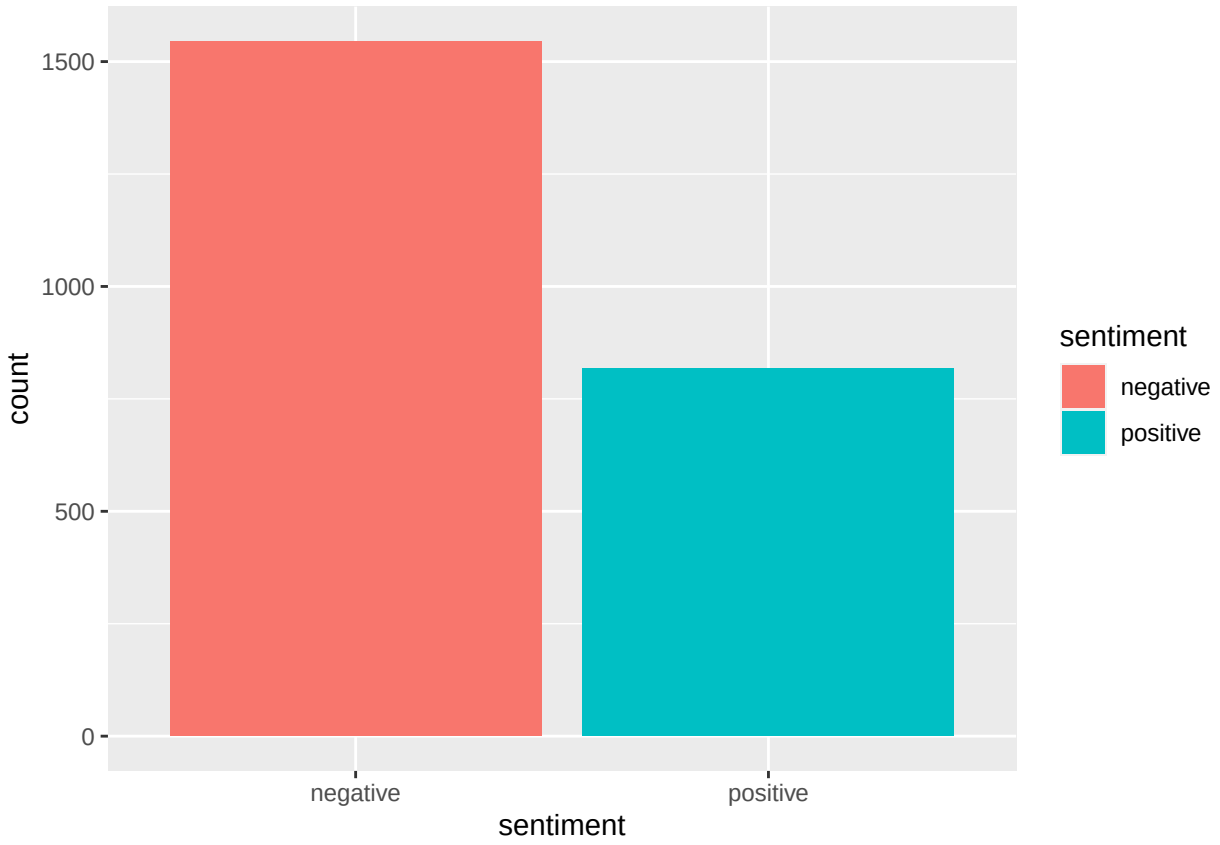
```r
nrc_word_counts %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(x = "Contribution to sentiment",
     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



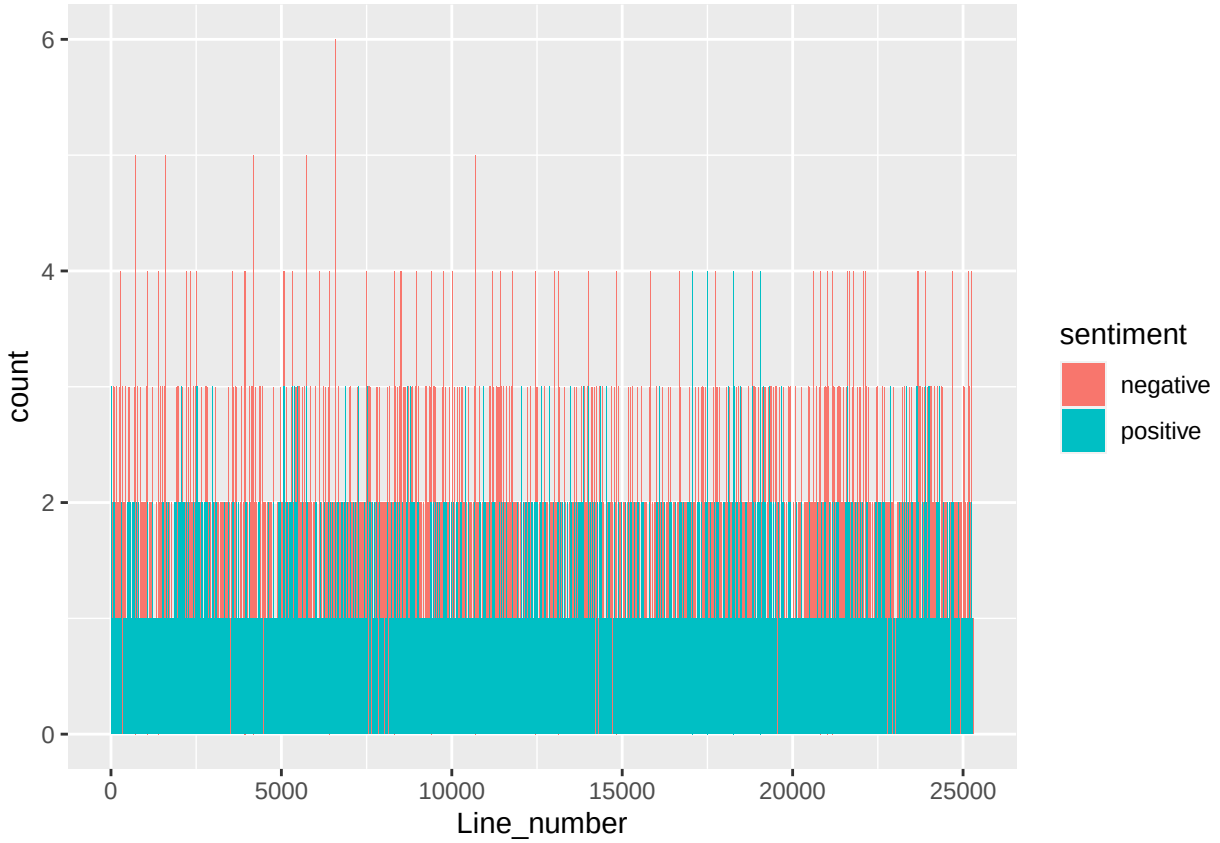

```
moby_sentiment = moby_filtered %>%
                      inner_join(sentiments)
```

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

```
#Plot showing the setiment difference in the book
ggplot(data = bing_word_counts) +
  geom_bar(mapping = aes(x = sentiment, fill = sentiment))
```




```
ggplot(data = moby_sentiment) +
  geom_bar(mapping = aes(x = Line_number, fill = sentiment))
```




```
# The Wonderful Wizard of Oz

stop_words$word[which(stop_words$word %in% sentiments$word)]
```

```
##   [1] "appreciate"    "appropriate"   "available"     "awfully"
##   [5] "best"          "better"        "clearly"       "enough"
##   [9] "like"          "liked"         "reasonably"    "right"
##  [13] "sensible"      "sorry"         "thank"         "unfortunately"
##  [17] "unlikely"      "useful"        "welcome"       "well"
##  [21] "willing"       "wonder"        "best"          "better"
##  [25] "clear"         "clearly"       "enough"        "evenly"
##  [29] "good"          "great"         "greatest"      "important"
##  [33] "interesting"   "interests"     "like"          "problem"
##  [37] "problems"      "right"         "right"         "well"
##  [41] "work"          "worked"        "works"
```

```
Oz_book_filtered <-Oz_book_filtered %>%
  unnest_tokens(output=word, input=sentence, token='words') %>%
  anti_join(stop_words)
```

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

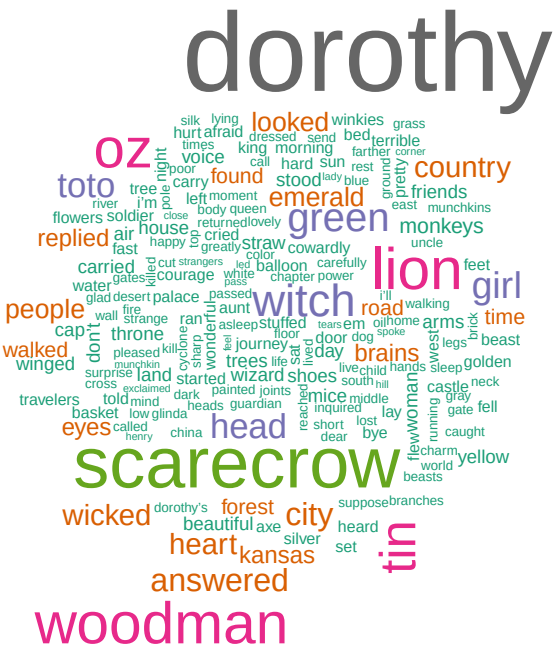
```
summary(Oz_book_filtered)
```

```
##   Line_number       word
##  Min.   :   1   Length:12350
##  1st Qu.:1143   Class :character
##  Median :2253   Mode  :character
```

```
##  Mean   :2250
##  3rd Qu.:3359
##  Max.   :4541
```

```
# Fequency of words in the text (table 1)
Oz_book_filtered_count <- Oz_book_filtered %>%
                            count(word) %>%
                            arrange(desc(n))

# word count visualization the most frequents words on the Moby book
 wordcloud(words =  Oz_book_filtered_count$word, freq = Oz_book_filtered_count  $n,scale=c(3.5,0.25),
          max.words=200, colors=brewer.pal(8, "Dark2"))
```




```
# Words and their contribution to the sentiments using the bing (positive or negative  lexicon
bing_word_counts <- Oz_book_filtered%>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()
```

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

```
#Table 1
print(bing_word_counts)
```

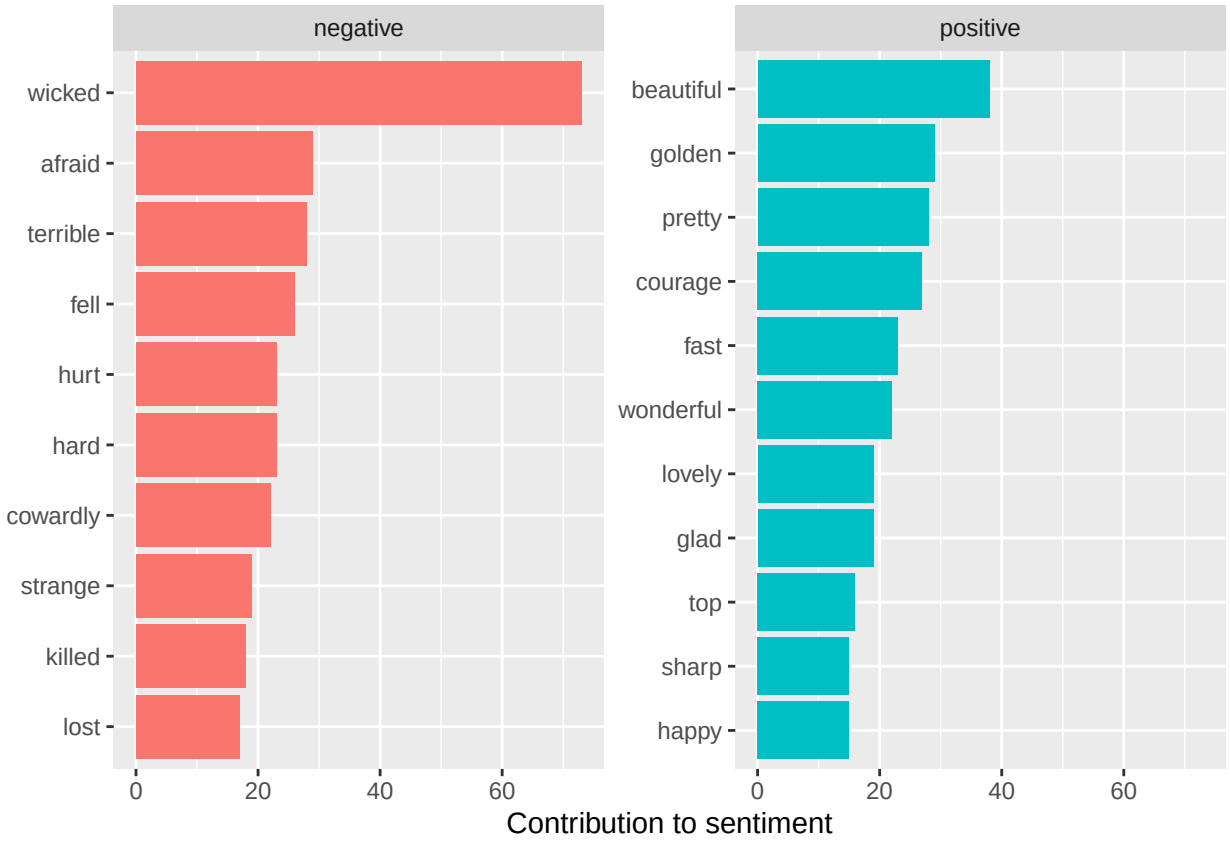
```
## # A tibble: 536 x 3
##    word      sentiment     n
##    <chr>     <chr>     <int>
##  1 wicked    negative     73
```

```
##  2 beautiful positive     38
##  3 afraid    negative     29
##  4 golden    positive     29
##  5 pretty    positive     28
##  6 terrible  negative     28
##  7 courage   positive     27
##  8 fell      negative     26
##  9 fast      positive     23
## 10 hard      negative     23
## # ... with 526 more rows
```

```
summary(bing_word_counts)
```

```
##      word            sentiment              n         
##  Length:536         Length:536         Min.   : 1.000  
##  Class :character   Class :character   1st Qu.: 1.000  
##  Mode  :character   Mode  :character   Median : 2.000  
##                                        Mean   : 3.612  
##                                        3rd Qu.: 3.000  
##                                        Max.   :73.000
```

```
# words that contribute to the Sentiments(  classified as Positive or negative )
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)
```




```
#Using the nrc lexicon
nrc_word_counts <- Oz_book_filtered%>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()
```

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

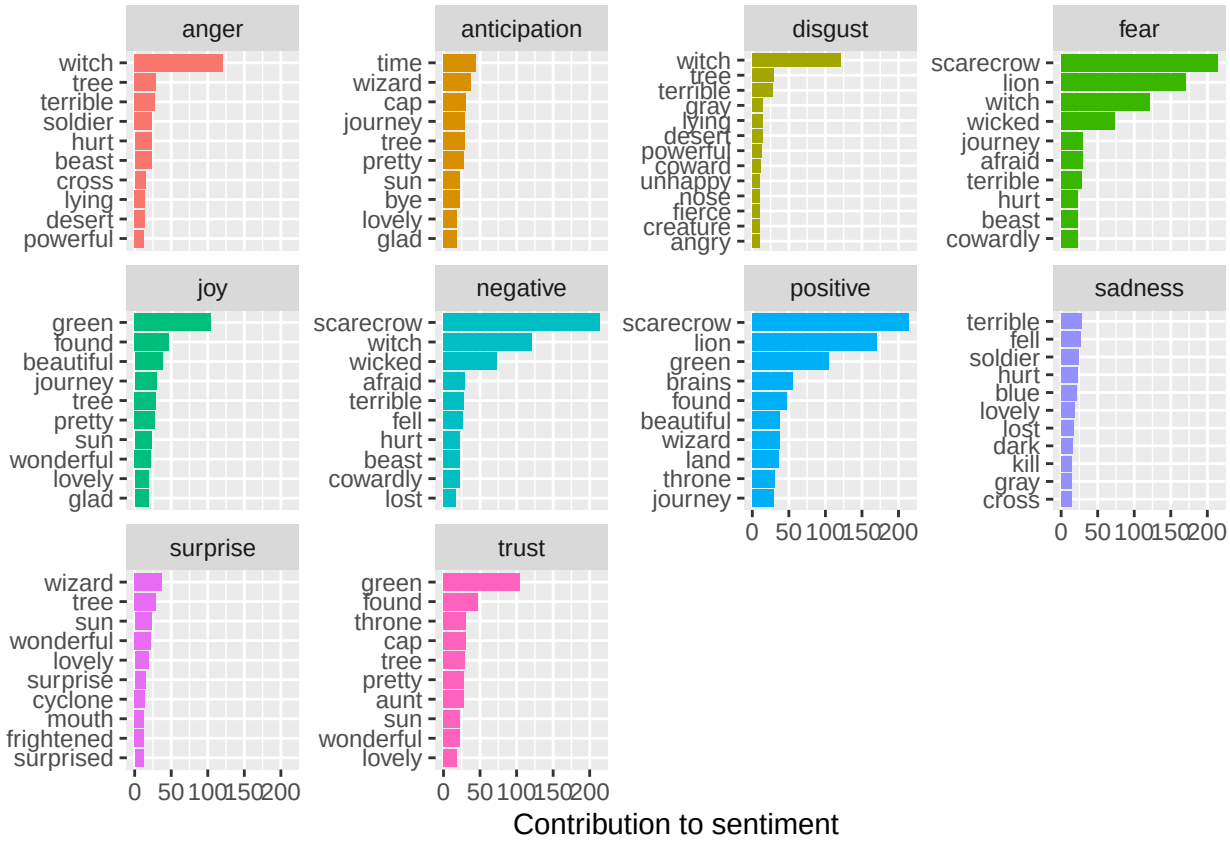
```
#Table 2
print(nrc_word_counts)
```

```
## # A tibble: 1,584 x 3
##    word      sentiment     n
##    <chr>     <chr>     <int>
##  1 scarecrow fear        214
##  2 scarecrow negative    214
##  3 scarecrow positive    214
##  4 lion      fear        170
##  5 lion      positive    170
##  6 witch     anger       121
##  7 witch     disgust     121
##  8 witch     fear        121
##  9 witch     negative    121
## 10 green     joy         104
## # ... with 1,574 more rows
```

```
summary(nrc_word_counts)
```

```
##      word             sentiment              n          
##  Length:1584        Length:1584        Min.   :  1.000  
##  Class :character   Class :character   1st Qu.:  1.000  
##  Mode  :character   Mode  :character   Median :  2.000  
##                                        Mean   :  5.479  
##                                        3rd Qu.:  4.000  
##                                        Max.   :214.000
```

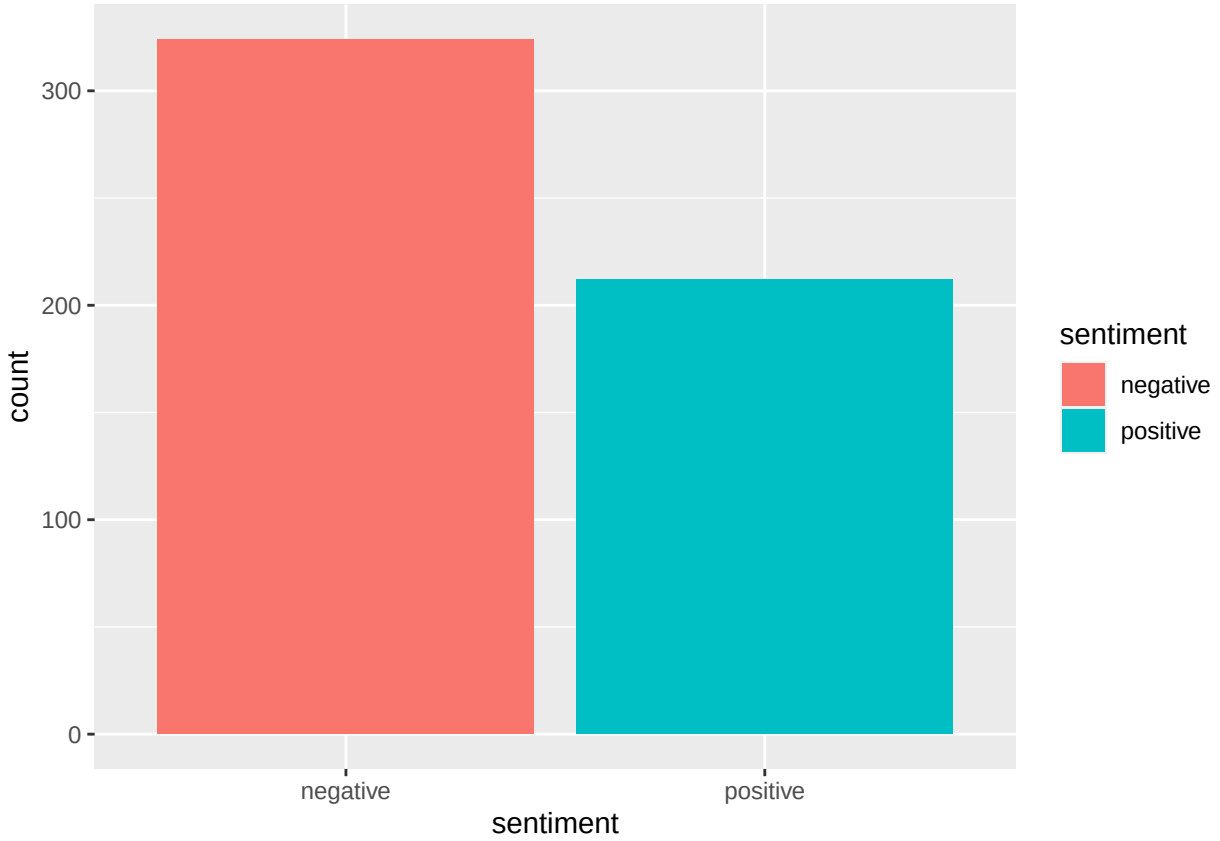
```
nrc_word_counts %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(x = "Contribution to sentiment",
      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



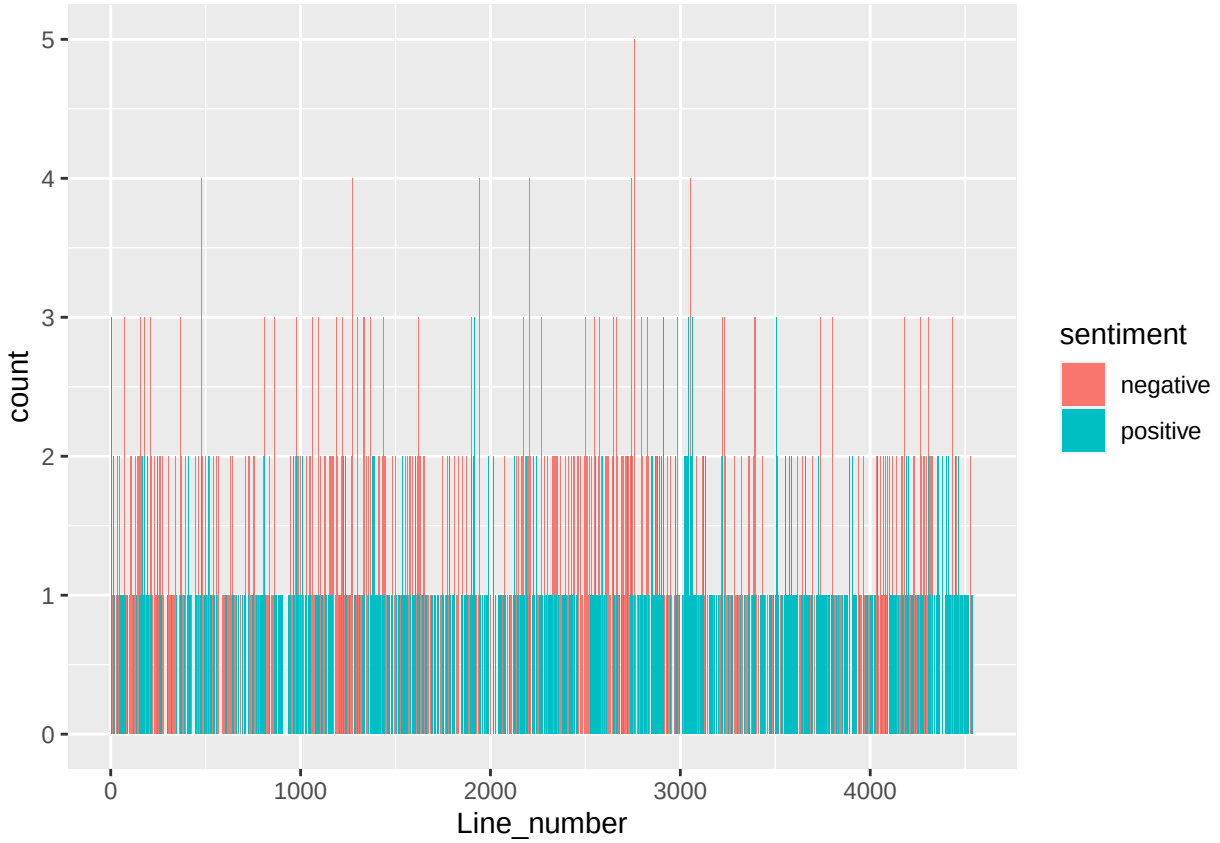

```
Oz_book_sentiment = Oz_book_filtered %>%
                      inner_join(sentiments)
```

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

```
#Plot showing the setiment difference in the book
ggplot(data = bing_word_counts) +
  geom_bar(mapping = aes(x = sentiment, fill = sentiment))
```




```
ggplot(data = Oz_book_sentiment) +
  geom_bar(mapping = aes(x = Line_number, fill = sentiment))
```



In this project, after working on the data processing, an inner join function was used to join the filtered text data of words with a sentiment lexicon of choice. This process will only retain words that are also in the particular lexicon chosen. In addition, I used the "bing" lexicon which classifies words into "Positive or Negative" and also I used the "nrc" lexicon which categorizes words into the following emotions (Anger, Joy, Surprise,Anticipation, Negative ,trust, disgust, positive, fear and sadness) which is much more detailed when compared to the bing lexicon.