Report onSentiment Analysis on two books 'Great Expectation and Treasure Island'

Registration # 2101142

Loading packages, libraries

```
library(dplyr)           # used for data manipulation such as to manipulate, clean and summarize unstruc
library(tidytext)        # used in conversion of text to and from tidy formats
library(stringr)         # contains a cohesive set of functions to manipulate strings i.e str_detect, st
library(tidyr)           # it contains tools for reshaping (pivoting) and hierarchy (nesting and 'unnest
library(ggplot2)         # it is used for plotting graphs, data visulaization
library(ggthemes)        # used for look and feel of graphs, visualization
library(gutenbergr)      # a library of many texts
library(tm)              # text mining package used for data wrangling
library(SnowballC)       # used for stemming of words, i.e changing words to its root elements
library(textstem)        # tools for Stemming and Lemmatizing Text
library(wordcloud)       # it helps to analyze text and visualize keywords/text
library(scales)          # gives tools to override default breaks, labels and transformations etc.
```

```
treasure_island  <- gutenberg_download(c(120))  # downloaded by number and assign contents to data fram
    treasure_island
```

```
## # A tibble: 7,491 x 2
##    gutenberg_id text
##           <int> <chr>
##  1          120 "TREASURE ISLAND"
##  2          120 ""
##  3          120 "by Robert Louis Stevenson"
##  4          120 ""
##  5          120 ""
##  6          120 ""
##  7          120 ""
##  8          120 "TREASURE ISLAND"
##  9          120 ""
## 10          120 "To S.L.O., an American gentleman in accordance with whose clas~
## # ... with 7,481 more rows
```

```
    treasure_island <- treasure_island[-c(1:110), ]  # skipping first few rows as those are irrelevant

    data(stop_words)  # a reference tibble of stop words in tidy format
    stop_words
```

```
## # A tibble: 1,149 x 2
##    word        lexicon
##    <chr>       <chr>
##  1 a           SMART
##  2 a's         SMART
##  3 able        SMART
##  4 about       SMART
##  5 above       SMART
##  6 according   SMART
##  7 accordingly SMART
##  8 across      SMART
##  9 actually    SMART
## 10 after       SMART
## # ... with 1,139 more rows
```

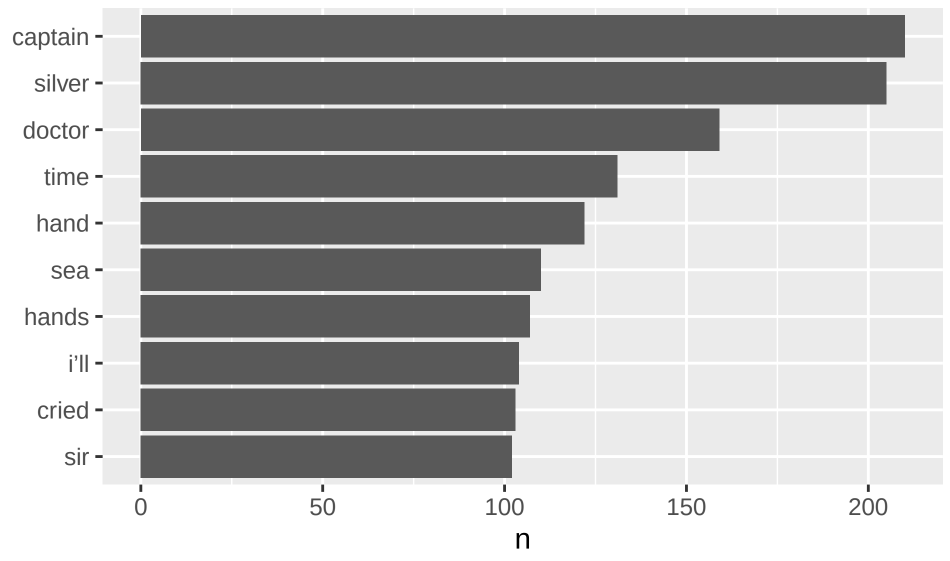
```
 treasure_island_tidy <- treasure_island %>%  # unnest i.e. convert to tidy format
      unnest_tokens(word, text) %>%     #this function will create collection of words and convert text
      anti_join(stop_words) # removal of stop words which are not relevant to sentiment analysis
    treasure_island_tidy
```

```
## # A tibble: 22,978 x 2
##    gutenberg_id word
##           <int> <chr>
##  1          120 buccaneer
##  2          120 1
##  3          120 sea
##  4          120 dog
##  5          120 admiral
##  6          120 benbow
##  7          120 squire
##  8          120 trelawney
##  9          120 dr
## 10          120 livesey
## # ... with 22,968 more rows
```

```
    treasure_island_tidy %>%count(word, sort = TRUE)  # count most common words
```

```
## # A tibble: 5,527 x 2
##    word        n
##    <chr>   <int>
##  1 captain   210
##  2 silver    205
##  3 doctor    159
##  4 time      131
##  5 hand      122
##  6 sea       110
##  7 hands     107
##  8 i'll      104
##  9 cried     103
## 10 sir       102
## # ... with 5,517 more rows
```

```
#visualizing most frequent words of treasure island after converting to tidy format
    treasure_island_tidy %>%
      count(word, sort = TRUE) %>%
      filter(n > 100) %>%
      mutate(word = reorder(word, n)) %>%
      ggplot(aes(word, n)) +
      geom_col() +
      xlab(NULL) +
      coord_flip()
```



```
  # custom words to be removed from text
  custom_stop_words <- bind_rows(tibble(word = c("light", "food","don't", "it's", "you'll", "i'll",
                  "i'm","hand","hands",  "till", "word", "dr", "you're"),  # to add own extra stop wo
                                        lexicon = c("custom")),
                                            stop_words)
  custom_stop_words
```

```
## # A tibble: 1,162 x 2
##    word   lexicon
##    <chr>  <chr>
##  1 light  custom
##  2 food   custom
##  3 don't  custom
##  4 it's   custom
##  5 you'll custom
##  6 i'll   custom
##  7 i'm    custom
##  8 hand   custom
##  9 hands  custom
## 10 till   custom
## # ... with 1,152 more rows
```

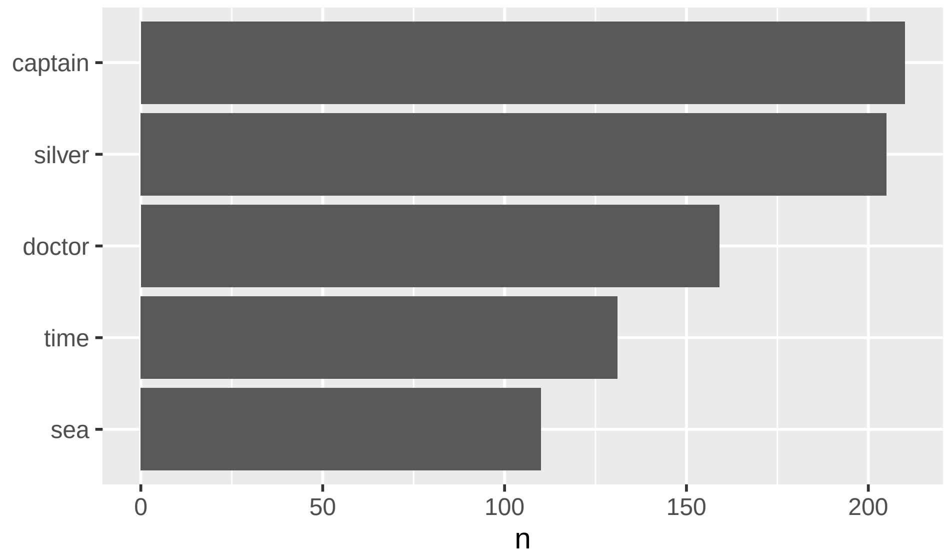
```
  treasure_island_tidy <- treasure_island_tidy %>%
    anti_join(custom_stop_words) # removal of custom stop words
  treasure_island_tidy %>%count(word, sort = TRUE)  # count most frequent words
```

```
## # A tibble: 5,520 x 2
##    word        n
##    <chr>   <int>
##  1 captain   210
##  2 silver    205
##  3 doctor    159
##  4 time      131
##  5 sea       110
##  6 i’ll      104
##  7 cried     103
##  8 sir       102
##  9 jim        97
## 10 squire     95
## # ... with 5,510 more rows
```

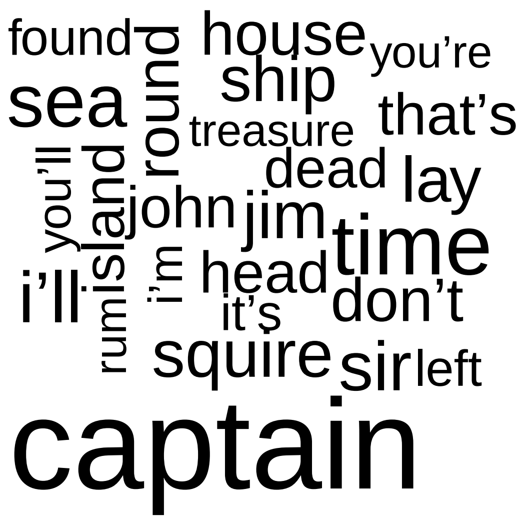
```
    treasure_island_tidy
```

```
## # A tibble: 22,571 x 2
##    gutenberg_id word
##           <int> <chr>
##  1          120 buccaneer
##  2          120 1
##  3          120 sea
##  4          120 dog
##  5          120 admiral
##  6          120 benbow
##  7          120 squire
##  8          120 trelawney
##  9          120 livesey
## 10          120 rest
## # ... with 22,561 more rows
```

```
 #visualizing most frequent words of treasure island after removing custom words
    treasure_island_tidy %>%
      count(word, sort = TRUE) %>%
      filter(n > 105) %>%
      mutate(word = reorder(word, n)) %>%
      ggplot(aes(word, n)) +
      geom_col() +
      xlab(NULL) +
      coord_flip()
```



```
#Exhibiting first 30 words of using Wordclound
treasure_island_tidy %>%
  anti_join(custom_stop_words) %>%
  count(word) %>%
  with(wordcloud(word, n, max.words = 30))
```



```
# removing numbers/digits from my code
treasure_island_tidy <- treasure_island_tidy %>%
  filter(!grepl('[0-9]', word))     #regular expression to remove numbers
treasure_island_tidy
```

```
## # A tibble: 22,526 x 2
##    gutenberg_id word
##           <int> <chr>
##  1          120 buccaneer
##  2          120 sea
##  3          120 dog
##  4          120 admiral
##  5          120 benbow
##  6          120 squire
##  7          120 trelawney
##  8          120 livesey
##  9          120 rest
## 10          120 gentlemen
## # ... with 22,516 more rows
```

```
    treasure_island_tidy %>%count(word, sort = TRUE)  # count most common words after removing numbers
```

```
## # A tibble: 5,480 x 2
##    word         n
##    <chr>    <int>
##  1 captain    210
##  2 silver     205
##  3 doctor     159
##  4 time       131
##  5 sea        110
##  6 i’ll       104
##  7 cried      103
##  8 sir        102
##  9 jim         97
## 10 squire      95
## # ... with 5,470 more rows
```

```
great_expectations  <- gutenberg_download(c(1400))  # Downloading great expectations book
    great_expectations
```

```
## # A tibble: 20,397 x 2
##    gutenberg_id text
##           <int> <chr>
##  1         1400 "[Illustration]"
##  2         1400 ""
##  3         1400 ""
##  4         1400 ""
##  5         1400 ""
##  6         1400 "Great Expectations"
##  7         1400 ""
##  8         1400 "[1867 Edition]"
##  9         1400 ""
## 10         1400 "by Charles Dickens"
## # ... with 20,387 more rows
```

```
    great_expectations_with_Skip_rows <- great_expectations[-c(1:78), ]  # skipping first 78 rows

    great_expectations_tidy <- great_expectations_with_Skip_rows %>%  # unnest i.e. convert to tidy for
      unnest_tokens(word, text) %>%
      anti_join(stop_words) # removal of stop words here
    great_expectations_tidy
```

```
## # A tibble: 57,196 x 2
##    gutenberg_id word
##           <int> <chr>
##  1         1400 chapter
##  2         1400 father's
##  3         1400 family
##  4         1400 pirrip
##  5         1400 christian
##  6         1400 philip
##  7         1400 infant
##  8         1400 tongue
##  9         1400 names
## 10         1400 explicit
## # ... with 57,186 more rows
```

```
great_expectations_tidy <- great_expectations_tidy %>%
  anti_join(custom_stop_words) # removal of custom stop words
great_expectations_tidy %>%count(word, sort = TRUE)  # count most frequent/common words
```

```
## # A tibble: 10,465 x 2
##    word         n
##    <chr>    <int>
##  1 joe        692
##  2 miss       383
##  3 time       373
##  4 pip        326
##  5 looked     325
##  6 herbert    290
##  7 don't      285
##  8 wemmick    256
##  9 havisham   243
## 10 estella    237
## # ... with 10,455 more rows
```

```
great_expectations_tidy <- great_expectations_tidy %>%
  filter(!grepl('[0-9]', word))    # regular expression to remove numbers

# count most common words after converting to tidy format
g_e_most_common <- great_expectations_tidy %>%count(word, sort = TRUE)
g_e_most_common
```

```
## # A tibble: 10,462 x 2
##    word         n
##    <chr>    <int>
##  1 joe        692
##  2 miss       383
##  3 time       373
##  4 pip        326
##  5 looked     325
##  6 herbert    290
##  7 don't      285
##  8 wemmick    256
##  9 havisham   243
## 10 estella    237
## # ... with 10,452 more rows
```

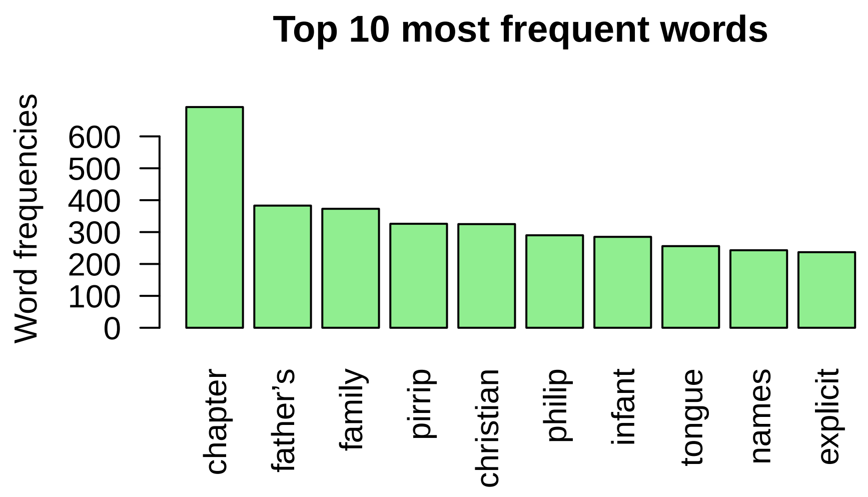
```
remove_reg <- "&amp;|&lt;|&gt;"

custom_regex_cleansing <- great_expectations_with_Skip_rows %>%     # removing special characters
  mutate(text = str_remove_all(text, remove_reg)) %>%
  unnest_tokens(word, text, token = "sentences") %>%
  filter(!word %in% stop_words$word,
         !word %in% str_remove_all(stop_words$word, "'"),
         str_detect(word, "[a-z]"))


 great_expectations_tidy %>%count(word, sort = TRUE)  # count most frequent/common words
```

```
## # A tibble: 10,462 x 2
##    word         n
##    <chr>    <int>
##  1 joe        692
##  2 miss       383
##  3 time       373
##  4 pip        326
##  5 looked     325
##  6 herbert    290
##  7 don't      285
##  8 wemmick    256
##  9 havisham   243
## 10 estella    237
## # ... with 10,452 more rows
```

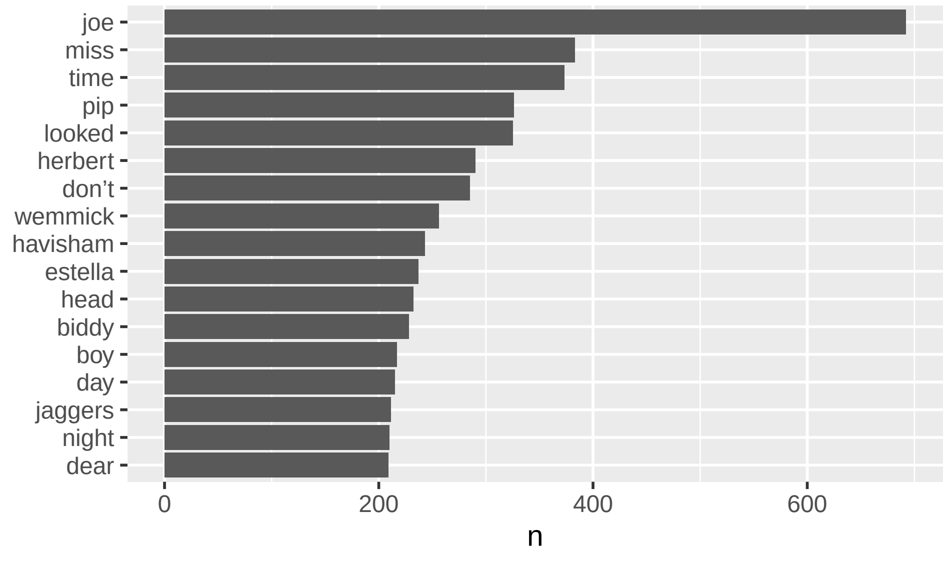
```
 great_expectations_tidy %>%count(word, sort = TRUE)  # count most frequent/common words
```

```
## # A tibble: 10,462 x 2
##    word         n
##    <chr>    <int>
##  1 joe        692
##  2 miss       383
##  3 time       373
##  4 pip        326
##  5 looked     325
##  6 herbert    290
##  7 don't      285
##  8 wemmick    256
##  9 havisham   243
## 10 estella    237
## # ... with 10,452 more rows
```

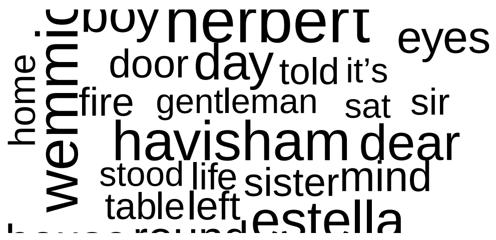
```
 # Ploting the most frequent words
 barplot(g_e_most_common[1:10,]$n, las = 2, names.arg = great_expectations_tidy[1:10,]$word,
         col ="lightgreen", main ="Top 10 most frequent words",
         ylab = "Word frequencies")
```



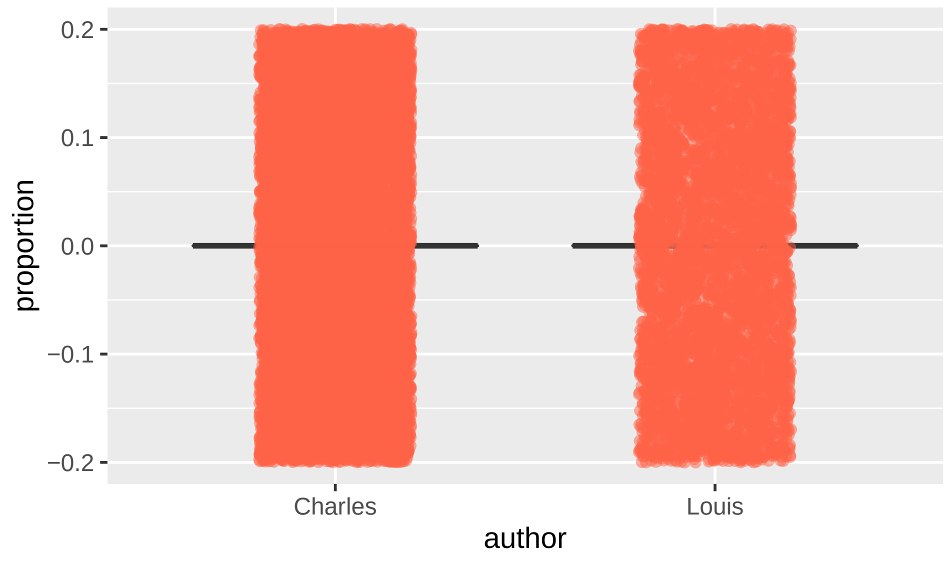
```
#Displaying most frequent words using ggplot library
great_expectations_tidy %>%
  count(word, sort = TRUE) %>%
  filter(n > 200) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n)) +
  geom_col() +
  xlab(NULL) +
  coord_flip()
```



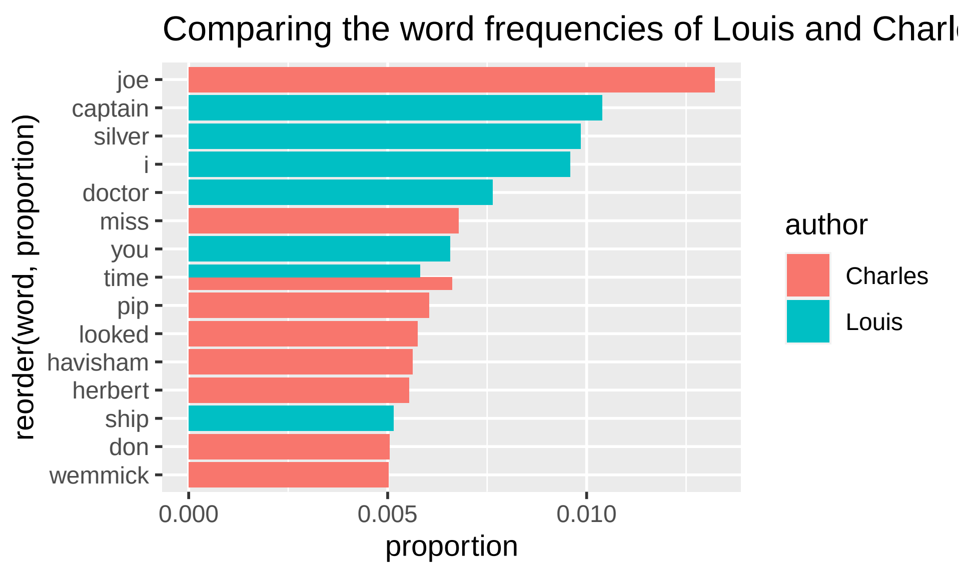
```
#Displaying 40 words using wordclound
great_expectations_tidy %>%
  anti_join(custom_stop_words) %>%
  count(word) %>%
  with(wordcloud(word, n, max.words = 40))
```



```
#Frequency/occurences of words and their counts and grouping words by authors, then finding proportion
    frequency <- bind_rows(mutate(great_expectations_tidy, author = "Charles"),
                           mutate(treasure_island_tidy, author = "Louis")) %>%
      mutate(word = str_extract(word, "[a-z']+")) %>%
      count(author, word) %>%
      group_by(author) %>%
      mutate(proportion = n / sum(n)) %>%
      select(-n) %>%
      spread(author, proportion) %>%
      gather(author, proportion, 'Charles':'Louis')

    frequency %>%        #plotting author and proportion on the basis of frequency
      ggplot(aes(x = author, y = proportion)) +
      geom_boxplot(alpha = 0) +
      geom_jitter(alpha = 0.5,
                  color = "tomato",
                  width = 0.2,
                  height = 0.2)
```



```r
# comparing the word frequencies of two books, Louis and Charles Dickens
# plotting  frequency and proportion where proportion(a subset of words) > 0.0050
frequency$word <- factor(frequency$word,
                         levels=unique(with(frequency,
                                            word[order(proportion, word,
                                                       decreasing = TRUE)])))
frequency <- frequency[complete.cases(frequency), ]

ggplot(aes(x = reorder(word, proportion), y = proportion, fill = author),
  data = subset(frequency, proportion > 0.0050)) +
  geom_bar(stat = 'identity', position = position_dodge())+
  coord_flip() +
  ggtitle('Comparing the word frequencies of Louis and Charles Dickens')
```



```
#Extraction chapters/parts information mostly through regular expressions

   Original_great_expectations <- great_expectations %>%
     mutate(linenumber = row_number(),  # add cols with line and chapter
            chapter = cumsum(str_detect(text,
                                        regex("^chapter [\\divxlc]",
                                              ignore_case = TRUE))))

   tail(Original_great_expectations)  # fetching tail part of the contents/text
```

```
## # A tibble: 6 x 4
##   gutenberg_id text                                            linenumber chapter
##          <int> <chr>                                                <int>   <int>
## 1         1400 ""                                                   20392      59
## 2         1400 "I took her hand in mine, and we went out of ~      20393      59
## 3         1400 "the morning mists had risen long ago when I ~      20394      59
## 4         1400 "the evening mists were rising now, and in al~      20395      59
## 5         1400 "tranquil light they showed to me, I saw no s~      20396      59
## 6         1400 "from her."                                          20397      59
```

```
   # exhibiting chapter numbers for every line of text/words
   # applied head for minimizing the records on pdf file
   head(cumsum(str_detect(Original_great_expectations$text,
                     regex("^chapter [\\divxlc]",
                           ignore_case = TRUE))) )
```

```
## [1] 0 0 0 0 0 0
```

```
   table(Original_great_expectations$chapter) # no of lines per chapter and book
```

```
##
##   0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19
##  79 210 363 224 351 433  72 451 525 316 286 604 220 330  79 476 183 367 589 606
##  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39
## 354 208 555 334 261 317 303 340 255 566 377 226 249 308 236 331 315 285 559 525
##  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59
## 581 251 296 244 340 318 302 262 307 398 189 335 226 521 563 319 224 534 337 182
```

```
   greatExp_tidy <- Original_great_expectations %>%  # unnest i.e. convert to tidy format, converting
     unnest_tokens(word, text) %>%
     anti_join(custom_stop_words) # removal of custom stop words here
   greatExp_tidy %>%count(word, sort = TRUE) # words in desc order after counting them
```

```
## # A tibble: 10,469 x 2
##    word       n
##    <chr>  <int>
##  1 joe      692
##  2 miss     383
##  3 time     373
##  4 pip      326
##  5 looked   325
##  6 herbert    290
##  7 don’t      285
##  8 wemmick    256
##  9 havisham   243
## 10 estella    237
## # ... with 10,459 more rows
```

```
    GExp_Back_to_untidy <- greatExp_tidy %>%          # converting text to untidy format by grouping th
      group_by(chapter, linenumber) %>%               #
      summarize(text = str_c(word, collapse = " ")) %>%   # used str_c function which join multiple str
      ungroup()
    GExp_Back_to_untidy
```

```
## # A tibble: 15,734 x 3
##    chapter linenumber text
##      <int>      <int> <chr>
##  1       0          1 illustration
##  2       0          6 expectations
##  3       0          8 1867 edition
##  4       0         10 charles dickens
##  5       0         13 contents
##  6       0         15 chapter
##  7       0         16 chapter ii
##  8       0         17 chapter iii
##  9       0         18 chapter iv
## 10       0         19 chapter
## # ... with 15,724 more rows
```

```
#working with regular expression to perform different operations.
    # "^chapter [\\divxlc]" regex to locate chapter headings
    head(str_detect(great_expectations$text,regex("^chapter [\\divxlc]", ignore_case = TRUE)))     # ch
```

```
## [1] FALSE FALSE FALSE FALSE FALSE FALSE
```

```
    head(great_expectations$text%>%str_subset(regex("^chapter [\\divxlc]",ignore_case = TRUE)))     # u
```

```
## [1] "Chapter I."   "Chapter II."  "Chapter III." "Chapter IV."  "Chapter V."
## [6] "Chapter VI."
```

```
    sum(str_detect(great_expectations$text,regex("^chapter [\\divxlc]", ignore_case = TRUE))) # sum of
```

```
## [1] 59
```

```
    head(str_subset(great_expectations$text,regex("^chapter [\\divxlc]", ignore_case = TRUE))) #-> usin
```

```
## [1] "Chapter I."   "Chapter II."  "Chapter III." "Chapter IV."  "Chapter V."
## [6] "Chapter VI."
```

```
great_expectations$text%>%str_detect(regex("^chapter [\\divxlc]",ignore_case = TRUE))%>%table
```

```
## .
## FALSE  TRUE
## 20338    59
```

```
 # tables no of lines per chapter and book
head(table(Original_great_expectations$linenumber,Original_great_expectations$chapter))
```

```
##
##      0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27
##    1 1 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##    2 1 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##    3 1 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##    4 1 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##    5 1 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##    6 1 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##
##     28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52
##    1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##    2  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##    3  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##    4  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##    5  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##    6  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##
##     53 54 55 56 57 58 59
##    1  0  0  0  0  0  0  0
##    2  0  0  0  0  0  0  0
##    3  0  0  0  0  0  0  0
##    4  0  0  0  0  0  0  0
##    5  0  0  0  0  0  0  0
##    6  0  0  0  0  0  0  0
```

```
great_expectations_tidy$word%>%str_detect(regex("^chapter",ignore_case = TRUE))%>%table # count of
```

```
## .
## FALSE  TRUE
## 56441    63
```

```
  #Exploratory data analysis using tm package
  #tm packages used for exploratory data analysis performed below operations

  temp <- great_expectations$text

  # using tail to view just tail text as otherwise, it will fill all the pdf file with text
  tail(gsub(' +',' ',temp) )
```

```
## [1] ""
## [2] "I took her hand in mine, and we went out of the ruined place; and, as"
## [3] "the morning mists had risen long ago when I first left the forge, so"
## [4] "the evening mists were rising now, and in all the broad expanse of"
## [5] "tranquil light they showed to me, I saw no shadow of another parting"
## [6] "from her."
```

```
tail(str_trim(temp, side = "both"))  #Removing whitespace
```

```
## [1] ""
## [2] "I took her hand in mine, and we went out of the ruined place; and, as"
## [3] "the morning mists had risen long ago when I first left the forge, so"
## [4] "the evening mists were rising now, and in all the broad expanse of"
## [5] "tranquil light they showed to me, I saw no shadow of another parting"
## [6] "from her."
```

```
text_lower <- tolower(great_expectations$text)    # Convert to lower case
tail(text_lower)
```

```
## [1] ""
## [2] "i took her hand in mine, and we went out of the ruined place; and, as"
## [3] "the morning mists had risen long ago when i first left the forge, so"
## [4] "the evening mists were rising now, and in all the broad expanse of"
## [5] "tranquil light they showed to me, i saw no shadow of another parting"
## [6] "from her."
```

```
tail(gsub('[[:digit:]]+', '', temp))  #Removing numbers
```

```
## [1] ""
## [2] "I took her hand in mine, and we went out of the ruined place; and, as"
## [3] "the morning mists had risen long ago when I first left the forge, so"
## [4] "the evening mists were rising now, and in all the broad expanse of"
## [5] "tranquil light they showed to me, I saw no shadow of another parting"
## [6] "from her."
```

```
tail(gsub('[[:punct:]]', '', temp))  #Removing punctuations
```

```
## [1] ""
## [2] "I took her hand in mine and we went out of the ruined place and as"
## [3] "the morning mists had risen long ago when I first left the forge so"
## [4] "the evening mists were rising now and in all the broad expanse of"
## [5] "tranquil light they showed to me I saw no shadow of another parting"
## [6] "from her"
```

```
st_words <- removeWords(temp, stopwords()) #Removing stop words
tail(st_words)
```

```
## [1] ""
## [2] "I took  hand  mine,   went    ruined place; , "
## [3] " morning mists  risen long ago  I first left  forge, "
## [4] " evening mists  rising now,     broad expanse "
## [5] "tranquil light  showed  , I saw  shadow  another parting"
## [6] " ."
```

```
    tail(wordStem(temp))     #Stemming
```

```
## [1] ""
## [2] "I took her hand in mine, and we went out of the ruined place; and, a"
## [3] "the morning mists had risen long ago when I first left the forge, so"
## [4] "the evening mists were rising now, and in all the broad expanse of"
## [5] "tranquil light they showed to me, I saw no shadow of another part"
## [6] "from her."
```

```
    tail(lemmatize_words(temp))  #Lemmatization
```

```
## [1] ""
## [2] "I took her hand in mine, and we went out of the ruined place; and, as"
## [3] "the morning mists had risen long ago when I first left the forge, so"
## [4] "the evening mists were rising now, and in all the broad expanse of"
## [5] "tranquil light they showed to me, I saw no shadow of another parting"
## [6] "from her."
```

```
# working with sentences and sections
    # splitting text into sentences, lines, using regular expressions

        sentences_from_treasure_island <- tibble(text = treasure_island$text) %>%
          unnest_tokens(sentence, text, token = "sentences")

        tibble(text = treasure_island$text) %>%
          unnest_tokens(line, text, token = "lines")
```

```
## # A tibble: 5,804 x 1
##    line
##    <chr>
##  1 part one--the old buccaneer
##  2 1
##  3 the old sea-dog at the admiral benbow
##  4 squire trelawney, dr. livesey, and the rest of these gentlemen having
##  5 asked me to write down the whole particulars about treasure island, from
##  6 the beginning to the end, keeping nothing back but the bearings of the
##  7 island, and that only because there is still treasure not yet lifted, i
##  8 take up my pen in the year of grace 17__ and go back to the time when
##  9 my father kept the admiral benbow inn and the brown old seaman with the
## 10 sabre cut first took up his lodging under our roof.
## # ... with 5,794 more rows
```

```
        tibble(text = treasure_island$text) %>%
          unnest_tokens(chapter, text, token = "regex", pattern = "^chapter")
```

```
## # A tibble: 5,804 x 1
##    chapter
##    <chr>
##  1 part one--the old buccaneer
##  2 1
##  3 the old sea-dog at the admiral benbow
##  4 squire trelawney, dr. livesey, and the rest of these gentlemen having
##  5 asked me to write down the whole particulars about treasure island, from
##  6 the beginning to the end, keeping nothing back but the bearings of the
##  7 island, and that only because there is still treasure not yet lifted, i
##  8 take up my pen in the year of grace 17__ and go back to the time when
##  9 my father kept the admiral benbow inn and the brown old seaman with the
## 10 sabre cut first took up his lodging under our roof.
## # ... with 5,794 more rows
```

```
          tibble(text = treasure_island$text) %>%
            unnest_tokens(character, text, token = "characters")
```

```
## # A tibble: 274,221 x 1
##    character
##    <chr>
##  1 p
##  2 a
##  3 r
##  4 t
##  5 o
##  6 n
##  7 e
##  8 t
##  9 h
## 10 e
## # ... with 274,211 more rows
```

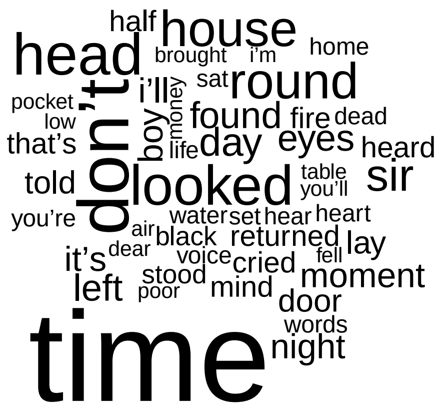
```
          tibble(text = treasure_island$text) %>%
            unnest_tokens(chapter, text, token = "regex",
                          pattern = "PART [\\dIVXLC]") %>%
            ungroup()
```

```
## # A tibble: 5,804 x 1
##    chapter
##    <chr>
##  1 part one--the old buccaneer
##  2 1
##  3 the old sea-dog at the admiral benbow
##  4 squire trelawney, dr. livesey, and the rest of these gentlemen having
##  5 asked me to write down the whole particulars about treasure island, from
##  6 the beginning to the end, keeping nothing back but the bearings of the
##  7 island, and that only because there is still treasure not yet lifted, i
##  8 take up my pen in the year of grace 17__ and go back to the time when
##  9 my father kept the admiral benbow inn and the brown old seaman with the
## 10 sabre cut first took up his lodging under our roof.
## # ... with 5,794 more rows
```

```
    # common/frequent words in two chosen books  using inner join
    common_words <- inner_join(great_expectations_tidy,treasure_island_tidy,by="word")
    common_words%>%count(word, sort = TRUE)  # most common common words
```

```
## # A tibble: 3,450 x 2
##    word       n
##    <chr>  <int>
##  1 time   48863
##  2 don’t  24225
##  3 head   18560
##  4 looked 17225
##  5 house  15540
##  6 round  15023
##  7 sir    13362
##  8 day    10320
##  9 eyes    9720
## 10 found   9555
## # ... with 3,440 more rows
```

```
#Displaying 50 common words of two books using wordclound
common_words %>%
  anti_join(custom_stop_words) %>%
  count(word) %>%
  with(wordcloud(word, n, max.words = 50))
```



```
# using antijoin to find the words exclusive to each book
uncommon_words1 <- anti_join(great_expectations_tidy,treasure_island_tidy,by="word")
uncommon_Great_Expectations <-  uncommon_words1%>%count(word, sort = TRUE)
uncommon_Great_Expectations  # words not in the Treasure Island book
```

```
## # A tibble: 7,012 x 2
##    word          n
##    <chr>     <int>
##  1 joe         692
##  2 pip         326
##  3 herbert     290
##  4 wemmick     256
##  5 havisham    243
##  6 estella     237
##  7 biddy         228
##  8 jaggers       211
##  9 sister        154
## 10 pumblechook   138
## # ... with 7,002 more rows
```

```
uncommon_words2 <- anti_join(treasure_island_tidy,great_expectations_tidy,by="word")
uncommon_Treasure <- uncommon_words2%>%count(word, sort = TRUE)
uncommon_Treasure  # words not in the Great Expectations book
```

```
## # A tibble: 2,030 x 2
##    word           n
##    <chr>      <int>
##  1 jim           97
##  2 squire        95
##  3 livesey       56
##  4 hispaniola    53
##  5 hawkins       51
##  6 ben           49
##  7 cap'n         47
##  8 smollett      45
##  9 stockade      39
## 10 gunn          37
## # ... with 2,020 more rows
```

```
# treasure island words with afinn sentiment dictionary
get_sentiments("afinn")  # sentiment reference
```

```
## # 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
## # ... with 2,467 more rows
```

```
exc_treasure_words <- uncommon_Treasure %>%inner_join(get_sentiments("afinn"),"word")
exc_treasure_words
```

```
## # A tibble: 86 x 3
##    word          n value
##    <chr>     <int> <dbl>
##  1 gray         35    -1
##  2 dick         26    -4
##  3 merry        22     3
##   4 fools         6    -2
##   5 tops          5     2
##   6 blamed        4    -2
##   7 blocks        4    -1
##   8 huge          4     1
##   9 annoyance     3    -2
##  10 glee          3     3
## # ... with 76 more rows
```

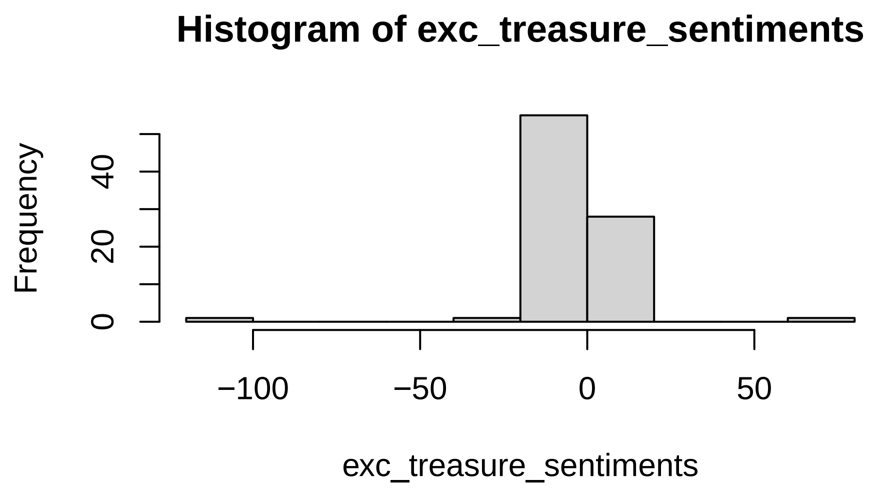
```
exc_treasure_words <- exc_treasure_words%>%mutate(weighted=n*value)
exc_treasure_words  # calc total sentiment contribution of each word
```

```
## # A tibble: 86 x 4
##    word          n value weighted
##    <chr>     <int> <dbl>    <dbl>
##  1 gray         35    -1      -35
##  2 dick         26    -4     -104
##  3 merry        22     3       66
##  4 fools         6    -2      -12
##  5 tops          5     2       10
##  6 blamed        4    -2       -8
##  7 blocks        4    -1       -4
##  8 huge          4     1        4
##  9 annoyance     3    -2       -6
## 10 glee          3     3        9
## # ... with 76 more rows
```

```
str(exc_treasure_words) # note 86 words exc to other book
```

```
## tibble [86 x 4] (S3: tbl_df/tbl/data.frame)
##  $ word    : chr [1:86] "gray" "dick" "merry" "fools" ...
##  $ n       : int [1:86] 35 26 22 6 5 4 4 4 3 3 ...
##  $ value   : num [1:86] -1 -4 3 -2 2 -2 -1 1 -2 3 ...
##  $ weighted: num [1:86] -35 -104 66 -12 10 -8 -4 4 -6 9 ...
```

```
exc_treasure_sentiments <- as.numeric(exc_treasure_words$weighted)

hist(exc_treasure_sentiments)  # distribution of the treasure island exc word sentiments
```



```
# the following does exactly as above for the Great Expectations book
    # first afinn sentiment is joined with uncommon great expectation words then words are assigned wei
    # and distribution can be checked using histogram chart
    get_sentiments("afinn")  # sentiment reference
```

```
## # 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
## # ... with 2,467 more rows
```

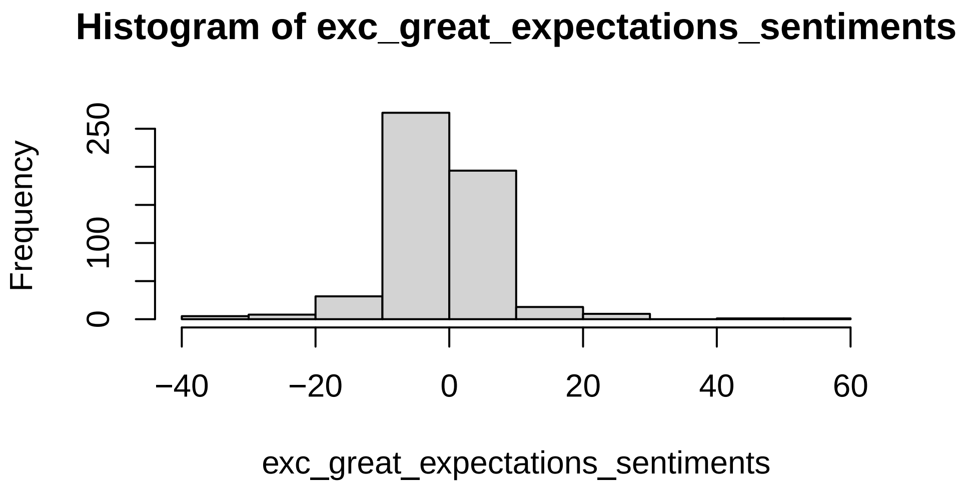
```
    exc_great_expectations_words <- uncommon_Great_Expectations %>%inner_join(get_sentiments("afinn"),"
    exc_great_expectations_words <- exc_great_expectations_words%>%mutate(weighted=n*value)
    exc_great_expectations_words
```

```
## # A tibble: 531 x 4
##    word            n value weighted
##    <chr>       <int> <dbl>    <dbl>
##  1 dismal         19    -2      -38
##  2 dread          19    -2      -38
##  3 loved          18     3       54
##  4 comfortable    15     2       30
##  5 affection      14     3       42
##  6 ha             14     2       28
##  7 suspected      14    -1      -14
##  8 demanded       12    -1      -12
##  9 loss           12    -3      -36
## 10 pray           12     1       12
## # ... with 521 more rows
```

```
str(exc_great_expectations_words)
```

```
## tibble [531 x 4] (S3: tbl_df/tbl/data.frame)
##  $ word    : chr [1:531] "dismal" "dread" "loved" "comfortable" ...
##  $ n       : int [1:531] 19 19 18 15 14 14 14 12 12 12 ...
##  $ value   : num [1:531] -2 -2 3 2 3 2 -1 -1 -3 1 ...
##  $ weighted: num [1:531] -38 -38 54 30 42 28 -14 -12 -36 12 ...
```

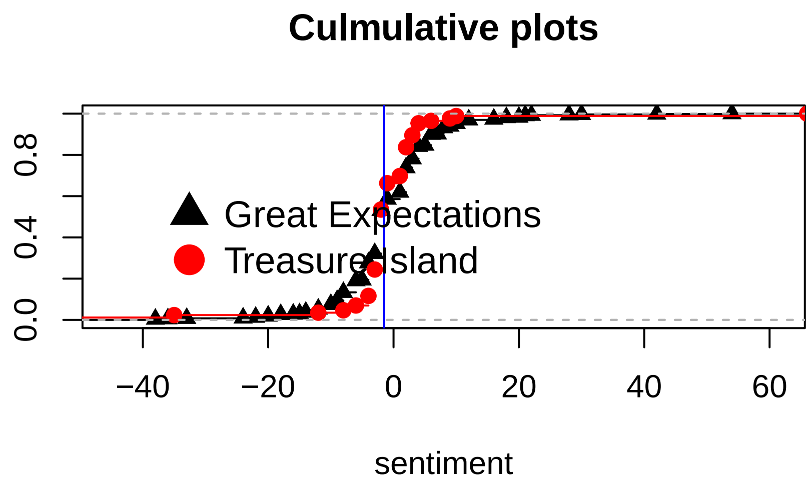
```
exc_great_expectations_sentiments <- as.numeric(exc_great_expectations_words$weighted)
hist(exc_great_expectations_sentiments) # distribution of the Great Expectations exc word sentiment
```



```
# t test difference of means test for the two distributions
t.test(exc_great_expectations_sentiments,exc_treasure_sentiments)
```

```
##
##  Welch Two Sample t-test
##
## data:  exc_great_expectations_sentiments and exc_treasure_sentiments
## t = 0.54242, df = 93.851, p-value = 0.5888
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.266347  3.970022
## sample estimates:
##  mean of x  mean of y
## -0.6365348 -1.4883721
```

```
library("dgof")  # looking at the culmulative distributions
cul_exc_great_exp_sentiments <- ecdf(exc_great_expectations_sentiments)
plot(cul_exc_great_exp_sentiments,main="Culmulative plots",pch = c(17),ylab="",xlab="sentiment")
cul_exc_treasure_sentiments <- ecdf(exc_treasure_sentiments) # CDF for Treasure values
lines(cul_exc_treasure_sentiments,col="red",pch = c(19)) # combine in one plot
abline(v=mean(exc_treasure_sentiments), col="blue") # vertical line  at Treasure mean
legend("bottomleft",
       legend = c("Great Expectations","Treasure Island"),
       col = c("black","red"),
       pch = c(17,19),
       bty = "n",
       pt.cex = 2,
       cex = 1.2,
       text.col = "black",
       horiz = F ,
       inset = c(0.1, 0.1)) # legend added to plot
```



```
# kolmogorov-smirnov test for a difference in distributions
# kolmogorov-smirnov test(kst) compares the commulative distribution of two given sets.
# basically it presents the max difference between the comulative distributions
# it calculates the P value and samples sizes
# Kst has max value 1 and min 0, max value describes the best fit of the data
# while the min value shows the fit is not significant
ks.test(exc_treasure_sentiments,cul_exc_great_exp_sentiments) # p value is sig
```

```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  exc_treasure_sentiments
## D = 0.16056, p-value = 0.02374
## alternative hypothesis: two-sided
```

```r
# very low p value shows that the distributions differ
```

```r
# The below line of code performs the following operations
# sentiments dictionaries afinn, bing and nrc get loaded
# then sentiment related to nrc are tested for joy sentiment words,
# which gives information about the joy sentiment, their frequencies
# similary sentiments of two books have been examined on the basis of different sentiment reference


#different sentiments dictionaries
get_sentiments("afinn")  # sentiment reference
```

```
## # 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
## # ... with 2,467 more rows
```

```r
get_sentiments("bing")
```

```
## # A tibble: 6,786 x 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
## # ... with 6,776 more rows
```

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

```
## # A tibble: 13,875 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
## # ... with 13,865 more rows
```

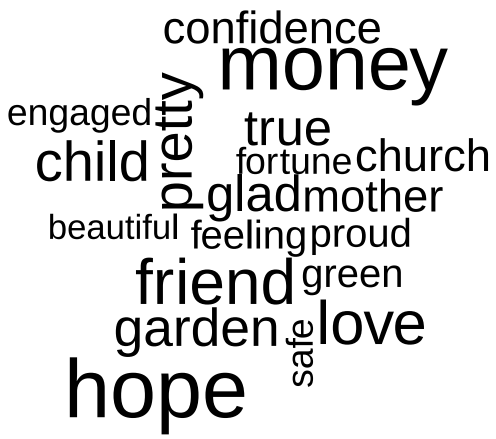
```
    nrc_joy <- get_sentiments("nrc") %>%
      filter(sentiment == "joy")  # select joy sentiment words
    nrc_joy         # result        A tibble: 687 x 2
```

```
## # A tibble: 687 x 2
##    word          sentiment
##    <chr>         <chr>
##  1 absolution    joy
##  2 abundance     joy
##  3 abundant      joy
##  4 accolade      joy
##  5 accompaniment joy
##  6 accomplish    joy
##  7 accomplished  joy
##  8 achieve       joy
##  9 achievement   joy
## 10 acrobat       joy
## # ... with 677 more rows
```

```
    great_expectations_tidy_sentiment_joy <- great_expectations_tidy %>%
      inner_join(nrc_joy)   # joy words in great expectations

    great_expectations_tidy_sentiment_joy
```

```
## # A tibble: 2,546 x 3
##    gutenberg_id word        sentiment
##           <int> <chr>       <chr>
##  1         1400 infant      joy
##  2         1400 mother      joy
##  3         1400 mother      joy
##  4         1400 entertained joy
##  5         1400 vivid       joy
##  6         1400 memorable   joy
##  7         1400 found       joy
##  8         1400 infant      joy
##  9         1400 church      joy
## 10         1400 pray        joy
## # ... with 2,536 more rows
```

```
    #Displaying first 20 words of Great expectations using wordclound
    great_expectations_tidy_sentiment_joy %>%
      anti_join(custom_stop_words) %>%
      count(word) %>%
      with(wordcloud(word, n, max.words = 20))
```



```
  great_expectations_tidy_sentiment_joy %>%
    inner_join(nrc_joy) %>%
    count(word, sort = TRUE)  # most popular joy words
```

```
## # A tibble: 350 x 2
##    word        n
##    <chr>   <int>
##  1 found     147
##  2 hope       86
##  3 money      81
##  4 friend     63
##  5 love       60
##  6 child      54
##  7 pretty     53
##  8 garden     49
##  9 glad       46
## 10 true       44
## # ... with 340 more rows
```

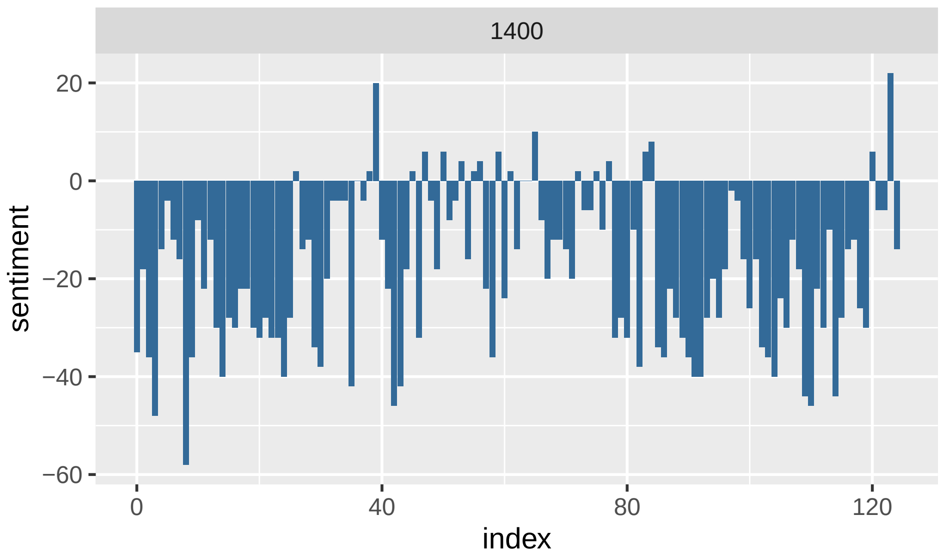
```
  ge_bing_sentiment <- great_expectations_tidy %>%
    inner_join(get_sentiments("bing"),"word")
  ge_bing_sentiment
```

```
## # A tibble: 8,739 x 3
##    gutenberg_id word          sentiment
##           <int> <chr>         <chr>
##  1         1400 unreasonably negative
##  2         1400 odd          negative
##  3         1400 dark         negative
##  4         1400 childish     negative
##  5          1400 sickly       negative
##  6          1400 neat         positive
##  7          1400 exceedingly  positive
##  8          1400 struggle     negative
##  9          1400 indebted     positive
## 10          1400 wound        negative
## # ... with 8,729 more rows
```

```
    #  sentiment is described with bing dictionary and in the below scenario,
    #  words list have been chosen and sentiment difference is found out
    #  sentiment described below is positive sentiment difference negative sentiment words.
    ge_bing_sentiment <- great_expectations_tidy %>%
      inner_join(get_sentiments("bing")) %>%
      count(gutenberg_id, index = row_number() %/% 70, sentiment)%>%     # count pos/neg over 70 words
      pivot_wider(names_from = sentiment, values_from = n, values_fill = 0) %>%
      mutate(sentiment = positive - negative)
    ge_bing_sentiment
```

```
## # A tibble: 125 x 5
##    gutenberg_id index negative positive sentiment
##           <int> <dbl>    <int>    <int>     <int>
##  1         1400     0       52       17       -35
##  2         1400     1       44       26       -18
##  3         1400     2       53       17       -36
##  4         1400     3       59       11       -48
##  5         1400     4       42       28       -14
##  6         1400     5       37       33        -4
##  7         1400     6       41       29       -12
##  8         1400     7       43       27       -16
##  9         1400     8       64        6       -58
## 10         1400     9       53       17       -36
## # ... with 115 more rows
```

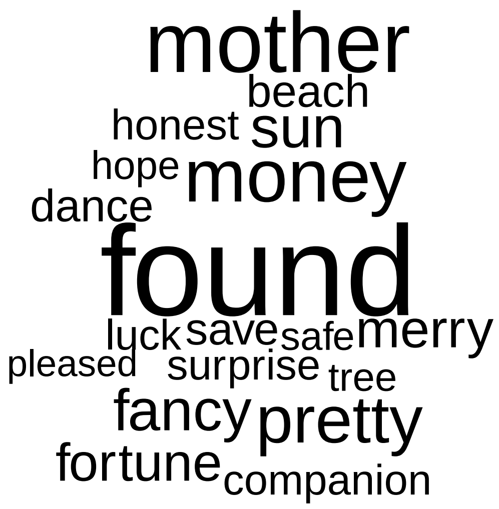
```
# visualization of sentiments difference in ggplot
    # mostly the bing lexican sentiments are negative in great expectations
    ggplot(ge_bing_sentiment, aes(index, sentiment, fill = gutenberg_id )) +
      geom_col(show.legend = FALSE) +
      facet_wrap(~gutenberg_id , ncol = 2, scales = "free_x")
```



```
treasure_island_tidy_sentiment_joy <- treasure_island_tidy %>%
  inner_join(nrc_joy)    # joy words in treasure Island

treasure_island_tidy_sentiment_joy
```

```
## # A tibble: 978 x 3
##    gutenberg_id word        sentiment
##           <int> <chr>       <chr>
##  1          120 treasure    joy
##  2          120 treasure    joy
##  3          120 cove        joy
##  4          120 connoisseur joy
##  5          120 cove        joy
##  6          120 pleasant    joy
##  7          120 cove        joy
##  8          120 cove        joy
##  9          120 pretty      joy
## 10          120 deal        joy
## # ... with 968 more rows
```

```
#Displaying first 20 words of treasure island using wordclound
treasure_island_tidy_sentiment_joy %>%
  anti_join(custom_stop_words) %>%
  count(word) %>%
  with(wordcloud(word, n, max.words = 20))
```



```r
    treasure_island_tidy_sentiment_joy %>%
      inner_join(nrc_joy) %>%
      count(word, sort = TRUE)   # most popular joy words
```

```
## # A tibble: 176 x 2
##    word          n
##    <chr>     <int>
##  1 found        65
##  2 treasure     57
##  3 mother       40
##  4 money        34
##  5 pretty       29
##  6 fancy        25
##  7 sun          25
##  8 merry        22
##  9 fortune      21
## 10 beach        17
## # ... with 166 more rows
```

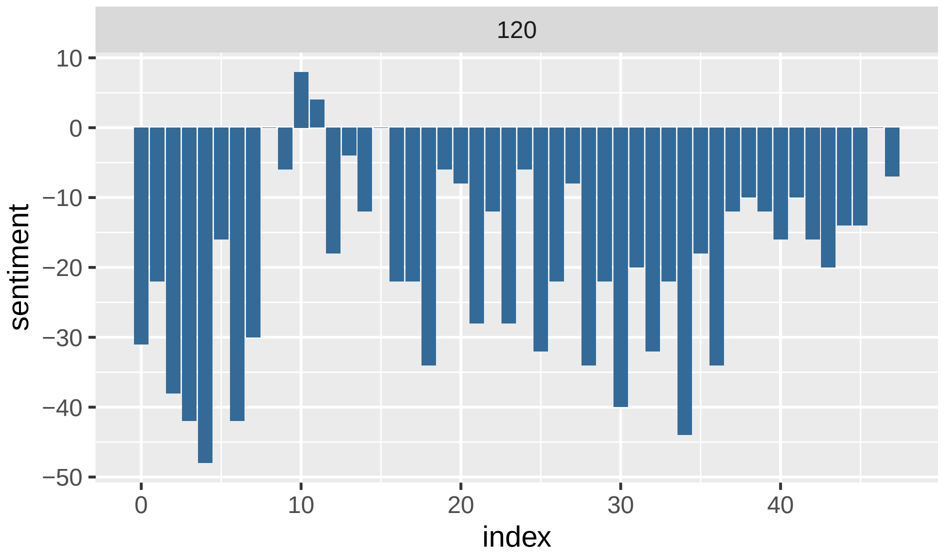
```r
    te_bing_sentiment <- treasure_island_tidy %>%
      inner_join(get_sentiments("bing"),"word")
    te_bing_sentiment
```

```
## # A tibble: 3,346 x 3
##    gutenberg_id word     sentiment
##           <int> <chr>    <chr>
##  1          120 treasure positive
##  2          120 treasure positive
##  3          120 grace    positive
##  4          120 strong   positive
##  5          120 falling  negative
##  6          120 ragged   negative
##  7          120 scarred  negative
##  8          120 broken   negative
## 9           120 dirty    negative
## 10          120 livid    negative
## # ... with 3,336 more rows
```

```
#  sentiment is described with bing dictionary and in the below scenario,
#  words list have been chosen and sentiment difference is found out
#  sentiment described below is positive sentiment difference negative sentiment words.
te_bing_sentiment <- treasure_island_tidy %>%
  inner_join(get_sentiments("bing")) %>%
  count(gutenberg_id, index = row_number() %/% 70, sentiment)%>%     # count pos/neg over 70 words
  pivot_wider(names_from = sentiment, values_from = n, values_fill = 0) %>%
  mutate(sentiment = positive - negative)
te_bing_sentiment
```

```
## # A tibble: 48 x 5
##    gutenberg_id index negative positive sentiment
##           <int> <dbl>    <int>    <int>     <int>
##  1          120     0       50       19       -31
##  2          120     1       46       24       -22
##  3          120     2       54       16       -38
##  4          120     3       56       14       -42
##  5          120     4       59       11       -48
##  6          120     5       43       27       -16
##  7          120     6       56       14       -42
##  8          120     7       50       20       -30
##  9          120     8       35       35         0
## 10          120     9       38       32        -6
## # ... with 38 more rows
```

```
# visualization of sentiments difference in ggplot
ggplot(te_bing_sentiment, aes(index, sentiment, fill = gutenberg_id )) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~gutenberg_id , ncol = 2, scales = "free_x")
```



```
# For both books bing lexicon sentiments are almost negative
```

```
treasure_island_bigrams <- tibble(text = treasure_island$text) %>%
  unnest_tokens(bigram, text, token = "ngrams", n = 2)  # using the bigram option
treasure_island_bigrams
```

```
## # A tibble: 64,631 x 1
##    bigram
##    <chr>
##  1 part one
##  2 one the
##  3 the old
##  4 old buccaneer
##  5 <NA>
##  6 <NA>
##  7 <NA>
##  8 <NA>
##  9 <NA>
## 10 <NA>
## # ... with 64,621 more rows
```

```
treasure_island_bigrams %>%count(bigram, sort = TRUE) # most popular bigrams are stop word pairs
```

```
## # A tibble: 33,821 x 2
##    bigram      n
##    <chr>   <int>
##  1 <NA>     1735
##  2 of the    484
##  3 in the    271
##  4 and the   221
##  5 it was    204
##  6 on the    176
##  7 and i     173
##  8 i was     167
##  9 to the    151
## 10 i had     148
## # ... with 33,811 more rows
```

```
bigrams_separated <- treasure_island_bigrams %>%
  separate(bigram, c("word1", "word2"), sep = " ") # separates the bigram in 2 cols
bigrams_separated
```

```
## # A tibble: 64,631 x 2
##    word1 word2
##    <chr> <chr>
##  1 part  one
##  2 one   the
##  3 the   old
##  4 old   buccaneer
##  5 <NA>  <NA>
##  6 <NA>  <NA>
##  7 <NA>  <NA>
##  8 <NA>  <NA>
##  9 <NA>  <NA>
## 10 <NA>  <NA>
## # ... with 64,621 more rows
```

```
bigrams_filtered <- bigrams_separated %>%
  filter(!word1 %in% stop_words$word) %>%  # removes via single stop words
  filter(!word2 %in% stop_words$word)
bigrams_filtered
```

```
## # A tibble: 6,930 x 2
##    word1    word2
##    <chr>    <chr>
##  1 <NA>     <NA>
##  2 <NA>     <NA>
##  3 <NA>     <NA>
##  4 <NA>     <NA>
##  5 <NA>     <NA>
##  6 <NA>     <NA>
##  7 sea      dog
##  8 admiral benbow
##  9 <NA>     <NA>
## 10 <NA>     <NA>
## # ... with 6,920 more rows
```

```
# new bigram counts:
bigram_counts <- bigrams_filtered %>%count(word1, word2, sort = TRUE)
bigram_counts
```

```
## # A tibble: 4,426 x 3
##    word1    word2        n
##    <chr>    <chr>    <int>
##  1 <NA>     <NA>      1735
##  2 dr       livesey     38
##  3 ben      gunn        31
##  4 captain smollett     29
##  5 spy      glass       23
##  6 black    dog         18
##  7 block    house       17
##  8 cried    silver      15
##  9 john     silver      15
## 10 admiral benbow       14
## # ... with 4,416 more rows
```

```
bigrams_united <- bigrams_filtered %>%
  unite(bigram, word1, word2, sep = " ")
bigrams_united  # can be used to recombine into a bigram
```

```
## # A tibble: 6,930 x 1
##    bigram
##    <chr>
##  1 NA NA
##  2 NA NA
##  3 NA NA
##  4 NA NA
##  5 NA NA
##  6 NA NA
##  7 sea dog
##  8 admiral benbow
##  9 NA NA
## 10 NA NA
## # ... with 6,920 more rows
```

```
    bigrams_separated %>%
      filter(word1 == "not") %>%  # count cases of negation , changing the sentiment
      count(word1, word2, sort = TRUE)
```

```
## # A tibble: 144 x 3
##    word1 word2     n
##    <chr> <chr> <int>
##  1 not   a        35
##  2 not   only     16
##  3 not   i        12
##  4 not   one      10
##  5 not   the      10
##  6 not   been      9
##  7 not   so        9
##  8 not   to        9
##  9 not   know      8
## 10 not   you       6
## # ... with 134 more rows
```

```
    AFINN <- get_sentiments("afinn")
    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
## # ... with 2,467 more rows
```

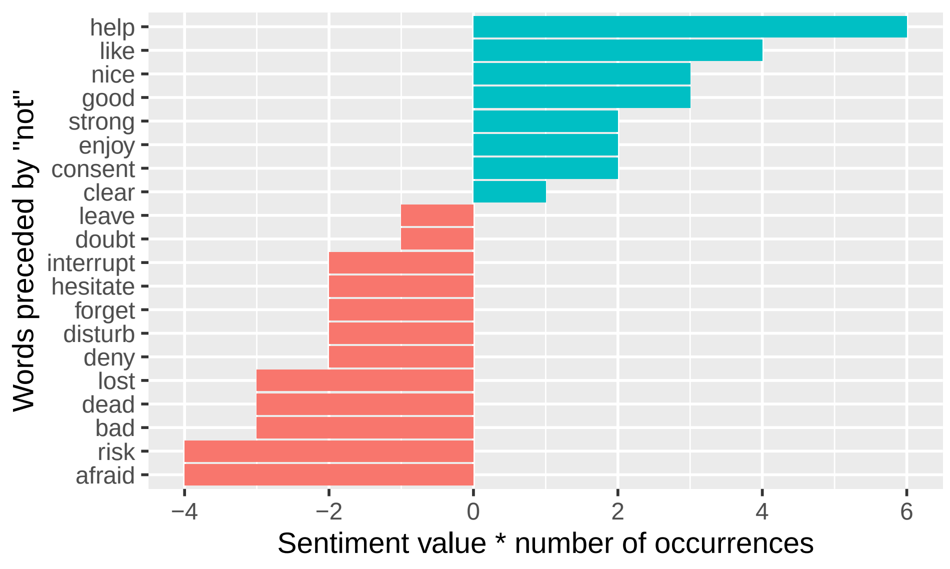
```
    not_words <- bigrams_separated %>%
      filter(word1 == "not") %>%
      inner_join(AFINN, by = c(word2 = "word")) %>%
      count(word2, value, sort = TRUE)
    not_words
```

```
## # A tibble: 24 x 3
##    word2   value     n
##    <chr>   <dbl> <int>
##  1 help        2     3
##  2 afraid     -2     2
##  3 forget     -1     2
##  4 like        2     2
##  5 risk       -2     2
##  6 bad        -3     1
##  7 clear       1     1
##  8 consent     2     1
##  9 dead       -3     1
## 10 deny       -2     1
## # ... with 14 more rows
```

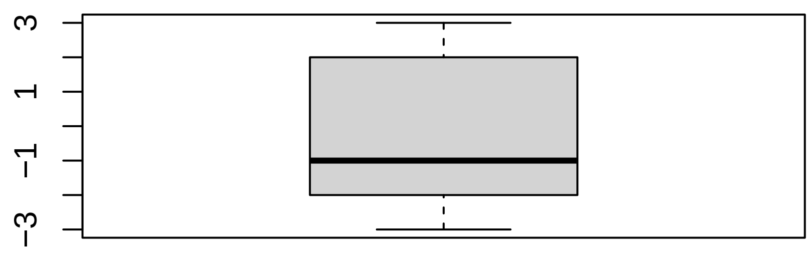
```
    not_words %>%
      mutate(contribution = n * value) %>%  # these sentiments are faulty
      arrange(desc(abs(contribution))) %>%
      head(20)
```

```
## # A tibble: 20 x 4
##    word2     value     n contribution
##    <chr>     <dbl> <int>        <dbl>
##  1 help          2     3            6
##  2 afraid       -2     2           -4
##  3 like          2     2            4
##  4 risk         -2     2           -4
##  5 bad          -3     1           -3
##  6 dead         -3     1           -3
##  7 good          3     1            3
##  8 lost         -3     1           -3
##  9 nice          3     1            3
## 10 forget       -1     2           -2
## 11 consent       2     1            2
## 12 deny         -2     1           -2
## 13 disturb      -2     1           -2
## 14 enjoy         2     1            2
## 15 hesitate     -2     1           -2
## 16 interrupt    -2     1           -2
## 17 strong        2     1            2
## 18 clear         1     1            1
## 19 doubt        -1     1           -1
## 20 leave        -1     1           -1
```

```
not_words %>%
      mutate(contribution = n * value) %>%
      arrange(desc(abs(contribution))) %>%
      head(20) %>%
      mutate(word2 = reorder(word2, contribution)) %>%  # can pipe above to ggplot
      ggplot(aes(n * value, word2, fill = n * value > 0)) +
      geom_col(show.legend = FALSE) +
      labs(x = "Sentiment value * number of occurrences",
           y = "Words preceded by \"not\"")
```



```
negation_words <- c("not", "no", "never", "without") # more negation words

negated_words <- bigrams_separated %>%
  filter(word1 %in% negation_words) %>%  # filter for the set of negation words
  inner_join(AFINN, by = c(word2 = "word")) %>%
  count(word1, word2, value, sort = TRUE)
boxplot(negated_words$value)
```



```
# The box plot is suggesting that most of the bigrams are positive when
# first word is in negation words.


treasure_and_great_expectations <- gutenberg_download(c(120, 1400))

tidy_treasure_and_great_expectations <- treasure_and_great_expectations %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words)
tidy_treasure_and_great_expectations %>%  count(word, sort = TRUE)
```

```
## # A tibble: 12,590 x 2
##    word        n
##    <chr>   <int>
##  1 joe       692
##  2 time      504
##  3 hand      392
##  4 miss      386
##  5 looked    378
##  6 don't     370
##  7 pip       326
##  8 head      312
##  9 hands     290
## 10 herbert   290
## # ... with 12,580 more rows
```

```
    bind_rows(mutate(great_expectations_tidy, author = "A"), # binding by rows produces NAs
              mutate(treasure_island_tidy, author = "B")) %>%
      mutate(word = str_extract(word, "[a-z']+")) %>%
      count(author, word) %>%
      group_by(author) %>%
      mutate(proportion = n / sum(n))%>%  # the values are proportions of word per author
      select(-n)%>%
      pivot_wider(names_from = author, values_from = proportion) # non tidy authors are in the col name
```

```
## # A tibble: 12,183 x 3
##    word                A          B
##    <chr>           <dbl>      <dbl>
##  1 a           0.000106   0.000178
##  2 aback       0.0000177  0.000133
##  3 abandoned   0.0000354  0.0000444
##  4 abased      0.0000177 NA
##  5 abashed     0.0000177 NA
##  6 abbey       0.0000177 NA
##  7 abear       0.0000354 NA
##  8 abel        0.000106  NA
##  9 aberdeen    0.0000177 NA
## 10 aberration 0.0000177 NA
## # ... with 12,173 more rows
```

```
 frequency <- bind_rows(mutate(great_expectations_tidy, author = "A"),
                          mutate(treasure_island_tidy, author = "B")) %>%
      mutate(word = str_extract(word, "[a-z']+")) %>%
      count(author, word) %>%
      group_by(author) %>%
      mutate(proportion = n / sum(n)) %>%
      select(-n) %>%
      pivot_wider(names_from = author, values_from = proportion) %>%
      pivot_longer('A':'B', # addition to the pipe selects two authors
                   names_to = "author", values_to = "proportion")

frequency
```

```
## # A tibble: 24,366 x 3
##    word      author proportion
##    <chr>     <chr>       <dbl>
##  1 a         A       0.000106
##  2 a         B       0.000178
##  3 aback     A       0.0000177
##  4 aback     B       0.000133
##  5 abandoned A       0.0000354
##  6 abandoned B       0.0000444
##  7 abased    A       0.0000177
##  8 abased    B      NA
##  9 abashed   A       0.0000177
## 10 abashed   B      NA
## # ... with 24,356 more rows
```

```
bingnegative <- get_sentiments("bing") %>%   # list of negative words from the Bing lexicon.
  filter(sentiment == "negative")
head(bingnegative)
```

```
## # A tibble: 6 x 2
##   word       sentiment
##   <chr>      <chr>
## 1 2-faces    negative
## 2 abnormal   negative
## 3 abolish    negative
## 4 abominable negative
## 5 abominably negative
## 6 abominate  negative
```

```
table(bingnegative$sentiment) # 4781  negative words
```

```
##
## negative
##     4781
```

```
great_expectations_tidy
```

```
## # A tibble: 56,504 x 2
##    gutenberg_id word
##           <int> <chr>
##  1         1400 chapter
##  2         1400 father's
##  3         1400 family
##  4         1400 pirrip
##  5         1400 christian
##  6         1400 philip
##  7         1400 infant
## 8           1400 tongue
## 9           1400 names
## 10          1400 explicit
## # ... with 56,494 more rows
```

```
bingnegative
```

```
## # A tibble: 4,781 x 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
## # ... with 4,771 more rows
```

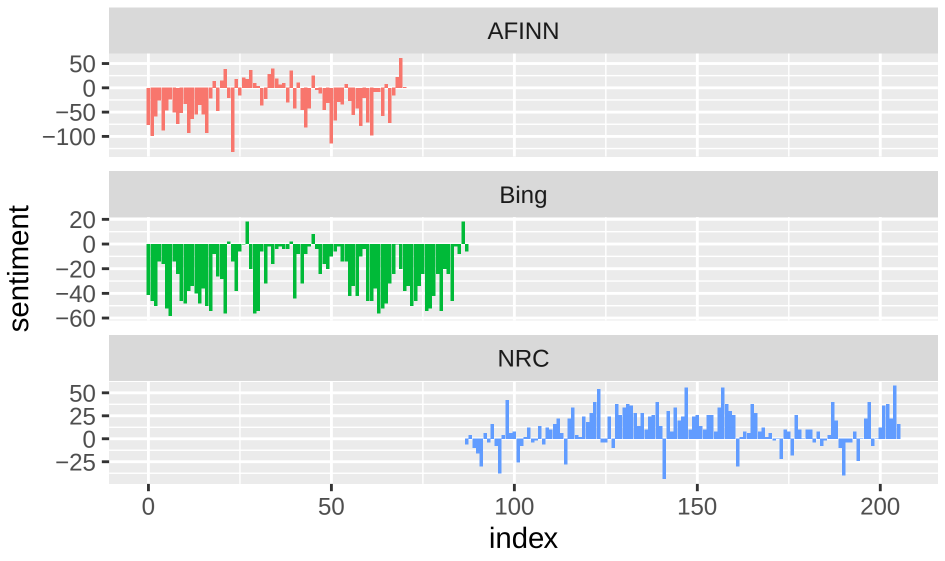
```
great_expectations_tidy %>%semi_join(bingnegative) # negative words in the book
```

```
## # A tibble: 5,515 x 2
##    gutenberg_id word
##           <int> <chr>
##  1         1400 unreasonably
##  2         1400 odd
##  3         1400 dark
##  4         1400 childish
##  5         1400 sickly
##  6         1400 struggle
##  7         1400 wound
##  8         1400 bleak
##  9         1400 dead
## 10         1400 dead
## # ... with 5,505 more rows
```

```
great_expectations_tidy %>%
  semi_join(bingnegative) %>%  #
  group_by(gutenberg_id, word) %>%
  summarize(negativewords = n())  # count neg words by chapt  and book
```

```
## # A tibble: 1,241 x 3
## # Groups:   gutenberg_id [1]
##    gutenberg_id word       negativewords
##           <int> <chr>              <int>
##  1         1400 abominate              1
##  2         1400 abrupt                 1
##  3         1400 absence                6
##  4         1400 absurd                 5
##  5         1400 absurdly              1
##  6         1400 abyss                 1
##  7         1400 accidental            3
##  8         1400 accuse                1
##  9         1400 accuses               1
## 10         1400 accusing              1
## # ... with 1,231 more rows
```

```
#analysis of bing, nrc and afinn lexicals
# It can be observed from the graphs that Afinn and Bing words are more negative
# NRC words are more positive
# The above results are for the book great expectations.
 afinn <- great_expectations_tidy %>%
   inner_join(get_sentiments("afinn")) %>%
   group_by(index = row_number() %/% 100) %>%
   summarise(sentiment = sum(value)) %>%
   mutate(method = "AFINN")

 bing_and_nrc <- bind_rows(
   great_expectations_tidy %>%
     inner_join(get_sentiments("bing")) %>%
     mutate(method = "Bing"),
   great_expectations_tidy %>%
     inner_join(get_sentiments("nrc") %>%
                  filter(sentiment %in% c("positive",
                                          "negative"))
     ) %>%
     mutate(method = "NRC")) %>%
   count(method, index = row_number() %/% 100, sentiment) %>%
   pivot_wider(names_from = sentiment,
               values_from = n,
               values_fill = 0) %>%
   mutate(sentiment = positive - negative)

 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")
```



```
# Calculating positive and negative words for understanding the difference between 3 dictionaries

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

```
## # A tibble: 2 x 2
##   sentiment     n
##   <chr>     <int>
## 1 negative   3318
## 2 positive   2308
```

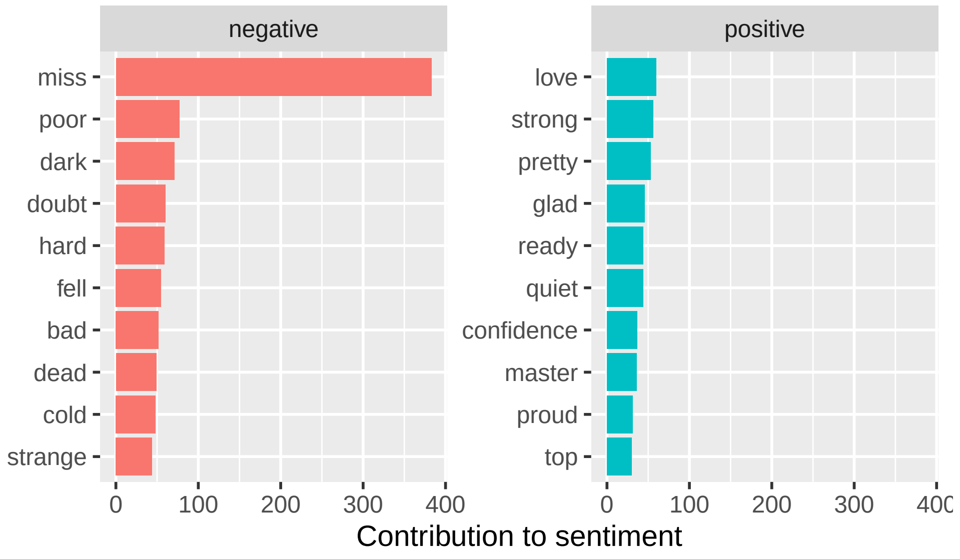
```
           #  negative   3318
           #  positive   2308


     get_sentiments("bing") %>%
       count(sentiment) # negative lexicons have higher values in bing as compared to NRC
```

```
## # A tibble: 2 x 2
##   sentiment     n
##   <chr>     <int>
## 1 negative   4781
## 2 positive   2005
```

```
           #  negative   4781
           #  positive   2005

     #counting bing words

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

     bing_word_counts
```

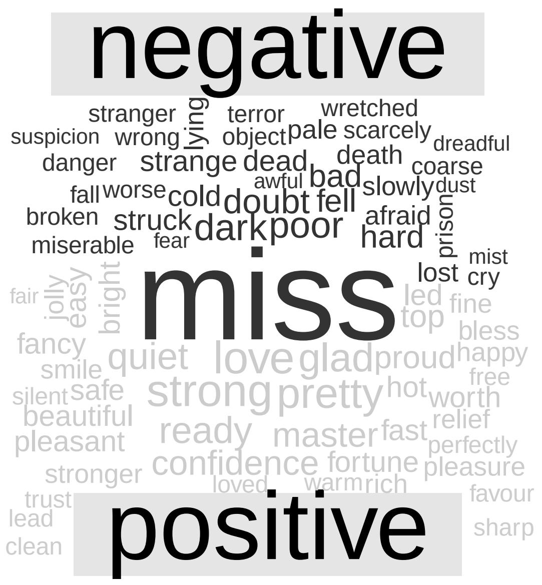
```
## # A tibble: 1,934 x 3
##    word   sentiment     n
##    <chr>  <chr>     <int>
##  1 miss   negative    383
##  2 poor   negative     77
##  3 dark   negative     71
##  4 doubt  negative     60
##  5 love   positive     60
##  6 hard   negative     59
##  7 strong positive     56
##  8 fell   negative     55
##  9 pretty positive     53
## 10 bad    negative     52
## # ... with 1,924 more rows
```

```
  #Results
  # It is observed that bing has more negative lexicons than nrc
  # Hence we can get more clear picture when using bing lexicon

  #contribution to sentiment graph plotting
  # It can be observed that words are negatively skewed
  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)
```

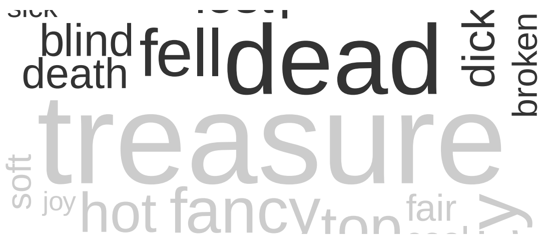


```
# it can be seen clearly that bing has comparitively more negative words than positve
```

```
#Positive and negative words of both books using bing dictionary

great_expectations_tidy %>%   #positive and negative words of great expectations
  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)
```



```
treasure_island_tidy %>%  #positive and negative words of treasure Island
  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)
```



```
#---------------------The End--------------------------------------#
```