Final project

Alejandro Aísa

2023-03-16

Analysing differences in literary style in XIX using text mining.

Introduccion

The story

Science Fiction is arguably one of the most profitable topics in both cinema and contemporary literature. Plots about super heroes and wizards surround nowadays' popular culture. However, as any other story, Science Fiction has a beginning. While there have *magical* and *fantastic* novels across history, it can be said that the race to stardom of Science Fiction took place in the XIX century. The industrial revolution, and the technological advances related to it, gave rise to the imagination of contemporary authors during that time. In other words, the inventions created a world of possibilities, not only in the real globe, but in the literature at the time.

However, these possibilities and consequences of technological advance were not necessarily described from a positive perspective. Authors like Mary Shelley or H.G. Wells wrote about societies in which *advances* had lead to dystopia societies. Considering this scenario, the main objective of this work is to perform an comparative analysis (using text mining tools) between those novels that view technology as a very positive leap forward, and those that were worried about the future that these changes could bring.

The chosen novels and hypothesis

In order to perform this study, I will select six novels from 6 different authors. They would be selected (and divided) according to the way in which they describe the technological advances and the future. The first three of them correspond to novels that would potentially characterise the upcoming times as positive:

- Twenty Thousand Leagues Under the Sea (TTLUS); Jules Verne (1870).
- The Mummy!; Jane Loudon (1827).
- A Princess of Mars; Edgar R. Burroughs (1912).

On the other hand, I have selected other three novels that focus on the potential negative externalities that technology might bring about. These novels highlight either describe dystopia societies or were written as a satire of the previous ones.

- The Time Machine; H. G. Wells (1895).
- The Last Man; Mary Shelley (1826).
- Erewhom; Samuel Butler (1872).

All of them were written in a similar literary context; XIX century and first years of the XX, so we can assume that the writing style and language used of the authors is comparable among the different novels. Finally, to make the analysis even, there are the same number of novels in each side of the spectrum. The six novels would be therefore the corpus.

As a main hypothesis, I expect that those novels that are confident about the future emphasise words and grammatical constructions that represent positive emotions and sentiments (not the same thing). Similarly, I expect that those novels rely on scientific and technological vocabulary to demonstrate the revolutionary spirit. On the other hand, those novels that are pessimistic about the future would built more on descriptive lexicons; They would focus more on the negative emotions and sentiments to accentuate the fatalistic feelings towards the forthcoming times.

The comparative analysis

Preparing the corpus

Downloading the books. For analysing these books, I resorted to the *Gutenberg project*. It has a R Library that would enable to load the selected books into the environment.

```
library(gutenbergr)
library(tidyverse)
```

```
science_fiction <- gutenberg_download(c(1906, 56426, 62, 164, 18247))


# meta_fields = c("title", "author")). I could have use directly this method to add the author and book

science_fiction = science_fiction %>%
  mutate(
         author = case_when(
                              gutenberg_id == 56426 ~ "Jane Loudon",
                              gutenberg_id == 62 ~ "Edgar R. Burroughs",
                              gutenberg_id == 164 ~ "Jules Verne",
                              gutenberg_id == 18247 ~ "Mary Shelley",
                              gutenberg_id == 1906 ~ "Samuel Butler",

                              ),
         book = case_when(
                              gutenberg_id == 56426 ~ "The Mummy!",
                              gutenberg_id == 62 ~ "A Princess of Mars",
                              gutenberg_id == 164 ~ "Twenty Thousand Leagues Under the Sea",
                              gutenberg_id == 18247 ~ "The Last Man",
                              gutenberg_id == 1906 ~ "Erewhom")) %>%
  group_by(book) %>%
  mutate(
    linenumber = row_number(),
     chapter = cumsum(str_detect(text,
                                      regex("^chapter [\\divxlc]",
                                            ignore_case = TRUE)))) %>%
  ungroup() %>%
  filter(chapter > 0)
```

```
wells = gutenberg_download(35) %>%
  mutate(
         author = case_when(gutenberg_id == 35 ~ "HG Wells"),
         book = case_when(gutenberg_id == 35 ~ "The Time Machine"),
         linenumber = row_number(),
         chapter = cumsum(str_detect(text,
                                     regex("[A-Z\\.]", # Does not work
                                           ignore_case = T))))  %>%
  filter(linenumber > 30)


books = rbind(science_fiction, wells)
```

At this point, using *regex* I added the line number of each piece of text, as well as the chapter. This information will be useful as I will only keep the proper novel in the analysis. Introductory notes will be removed to eliminate possible biases in the upcoming techniques. Hence why I filtered by chapter. Also, it should be noted that the chapter regex do not work in the case of Wells' "The Time Machine". Thus, the different chunk for his novel. In this case, I would filter by line instead of by character. However, for *rbind* to work, I would still need a chapter column.

Tokenizing Next step in the analysis is to tokenize the corpus, as we need to have the data structure. This means that each individual word has its own line. In other words, each word is an observation. Similarly, as the analysis will include semantic techniques, we need to filter those words that do not possess a lexical meaning by themselves: the stop words. Articles or prepositions will be removed.

```
library(tidytext) # For tokenizing
```

```
tidy_fiction = books  %>%
  unnest_tokens(word, text) %>% # Tokenizing
   anti_join(stop_words) # Removing stop words.

head(tidy_fiction, 10)
```

```
## # A tibble: 10 x 6
##    gutenberg_id author             book                linenumber chapter word
##           <int> <chr>              <chr>                    <int>   <int> <chr>
##  1           62 Edgar R. Burroughs A Princess of Mars         191       1 chapter
##  2           62 Edgar R. Burroughs A Princess of Mars         192       1 arizona
##  3           62 Edgar R. Burroughs A Princess of Mars         192       1 hills
##  4           62 Edgar R. Burroughs A Princess of Mars         195       1 possib~
##  5           62 Edgar R. Burroughs A Princess of Mars         195       1 hundred
##  6           62 Edgar R. Burroughs A Princess of Mars         196       1 possib~
##  7           62 Edgar R. Burroughs A Princess of Mars         196       1 aged
##  8           62 Edgar R. Burroughs A Princess of Mars         197       1 rememb~
##  9           62 Edgar R. Burroughs A Princess of Mars         197       1 childh~
## 10           62 Edgar R. Burroughs A Princess of Mars         197       1 recoll~
```

Reshaping the corpus At this step, I will create two different datasets for each type of novels. This will be later useful when performing sentiment analysis.

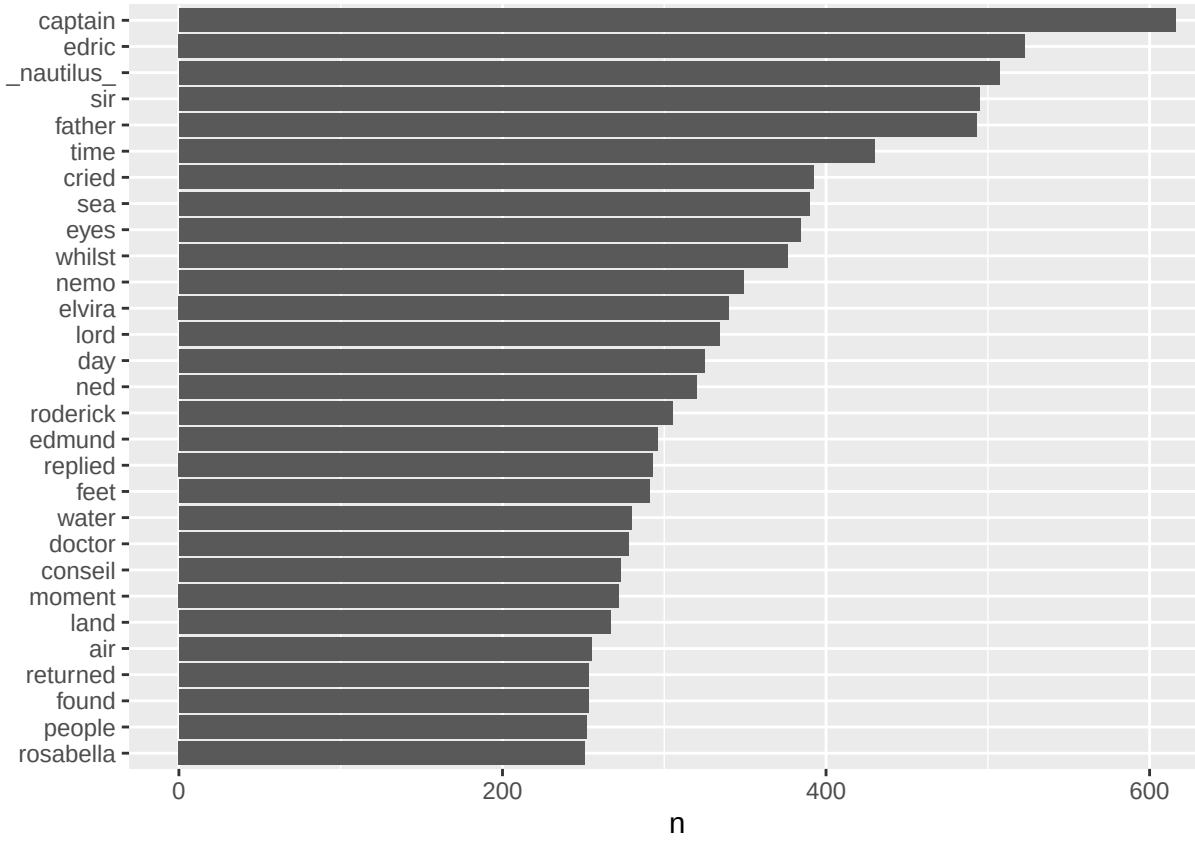
```
tidy_positive = tidy_fiction %>%
  filter(author %in% c("Edgar R. Burroughs", "Jane Loudon", "Jules Verne"))

tidy_negative = tidy_fiction %>%
  filter(author %in% c( "Mary Shelley", "Samuel Butler", "HG Wells"))
```

Analysing word term frequencies

The first technique that enable us to compare the different text is the frequency of the words used in the novels. Word frequency accounts for the number of times a word appears in a work (n). Term frequency is calculated as n divided by the lenght of the document.

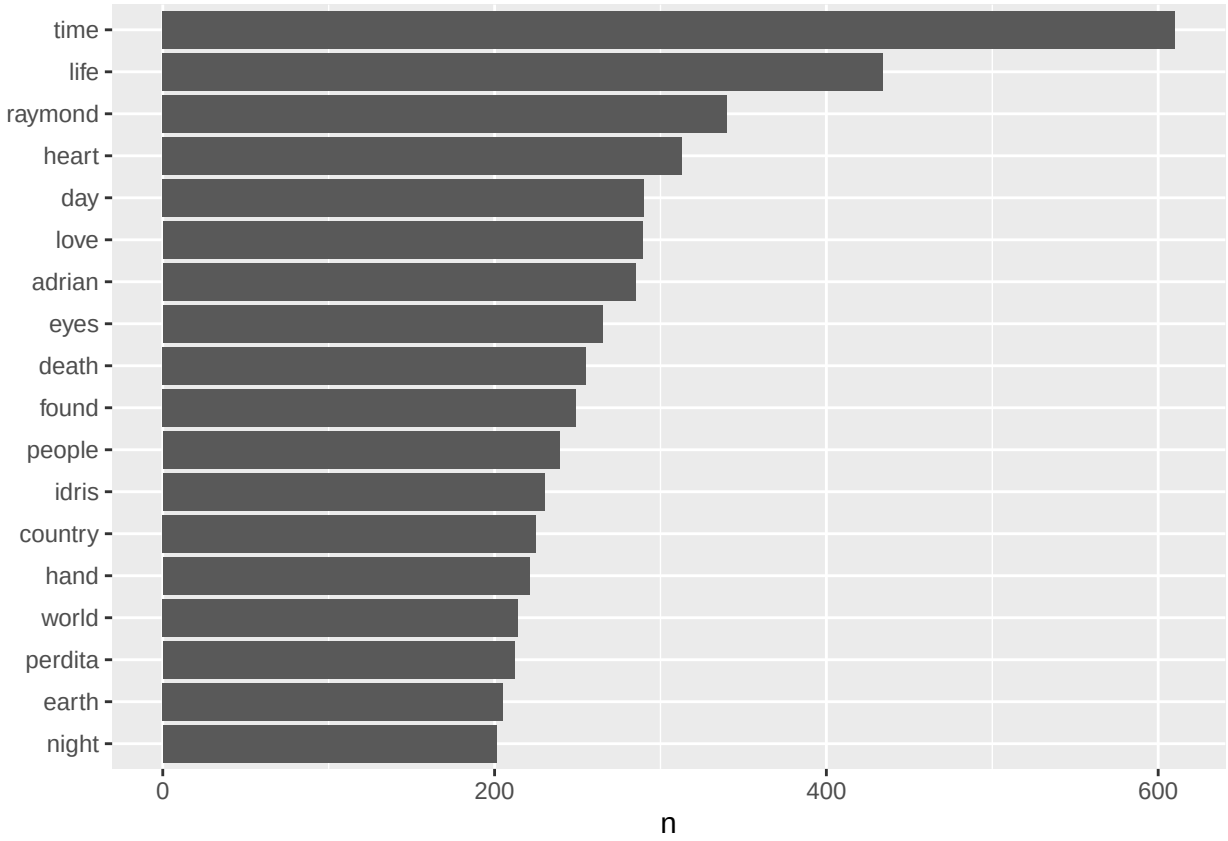
```
library(ggplot2) # To make visualizations during the study
```

```
tidy_positive %>%
  count(word, sort = TRUE) %>%
  filter(n > 250) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL)
```



Word frequency

```
tidy_negative %>%
  count(word, sort = TRUE) %>%
  filter(n > 200) %>% # Lower frequency estipulated as the novels are shorter
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word)) +
  geom_col() +
  labs(y = NULL)
```



Interesting insights may be obtained when plotting the word frequency for both set of novels. On the one hand, in the positive corpus there is a predominance of proper nouns, such as Nemo, Ned or Rosabella. (with the exception of 'captain'). Interestingly, there is a much lower amount of proper nouns in the pessimistic corpus. Hypothetically speaking, this could be the result of a deliberate impersonal style of writing. The novels may reflect that fear that
machines could replace humans in the long run. Similarly, words like 'human' 'machines' or 'death' are among the most used ones in the dystopia novels. Nonetheless, 'time' is the most frequent one. However, as the main theme of one them is time travel, we may not extract useful conclusion about it. However, there is an important limitation to this plot: it accounts for absolute values. Therefore, the over-representation of longer novels may distort the analysis.

Term frequencies

```
frequency_positive = tidy_positive  %>%
 mutate(word = str_extract(word, "[a-z']+")) %>%
 count(author, word) %>%
 group_by(author) %>%
 mutate(proportion = n / sum(n)) %>% # term frequency
 select(-n) %>%
 pivot_wider(names_from = author, values_from = proportion) %>%
 pivot_longer(`Jane Loudon`:`Edgar R. Burroughs`,
              names_to = "author", values_to = "proportion") %>%
 arrange(desc(`Jules Verne`))
```

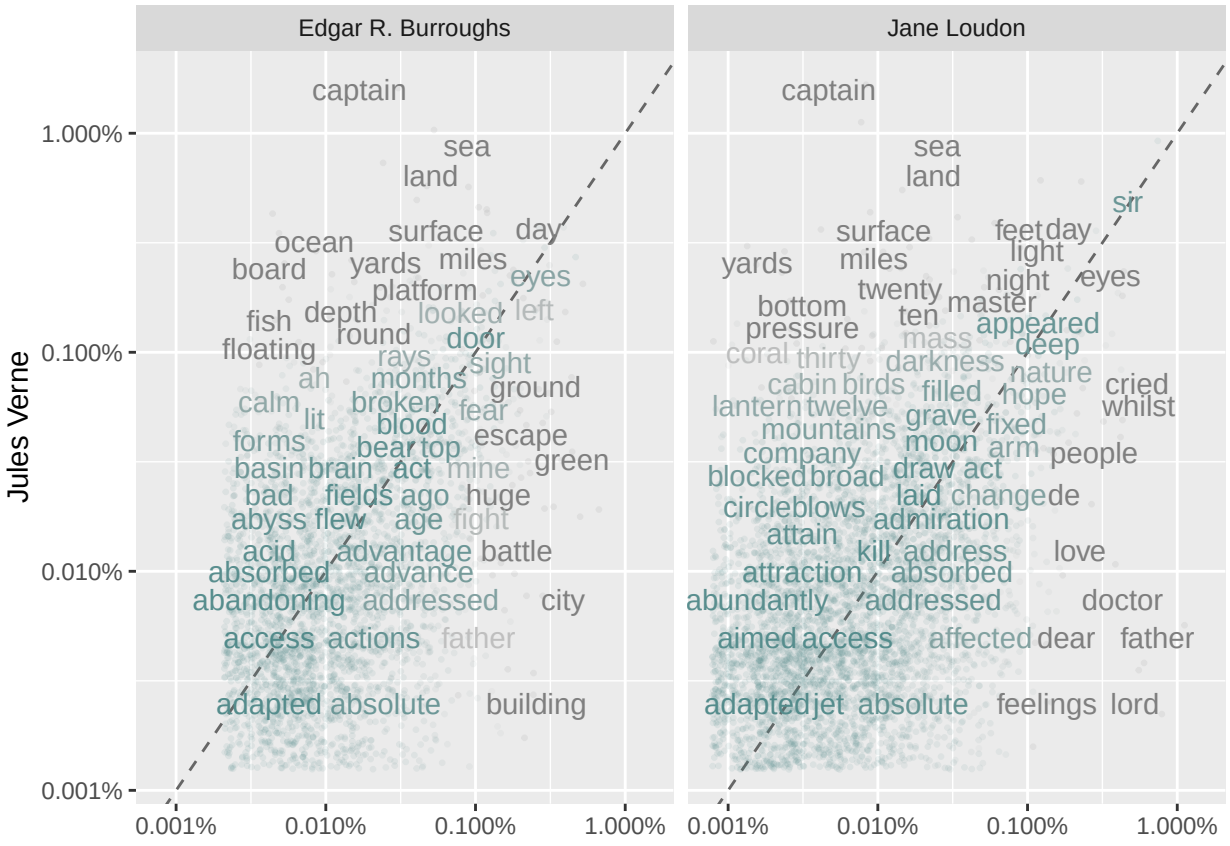
```
frequency_positive
```

Positive corpus

```
## # A tibble: 31,288 x 4
##    word     `Jules Verne` author             proportion
##    <chr>            <dbl> <chr>                   <dbl>
##  1 captain        0.0159  Jane Loudon         0.0000470
##  2 captain        0.0159  Edgar R. Burroughs  0.000168
##  3 nautilus       0.0130  Jane Loudon         NA
##  4 nautilus       0.0130  Edgar R. Burroughs  NA
##  5 nemo           0.00951 Jane Loudon         NA
##  6 nemo           0.00951 Edgar R. Burroughs  NA
##  7 sea            0.00877 Jane Loudon         0.000251
##  8 sea            0.00877 Edgar R. Burroughs  0.000880
##  9 <NA>           0.00800 Jane Loudon         0.0000783
## 10 <NA>           0.00800 Edgar R. Burroughs  0.000210
## # ... with 31,278 more rows
```

The figures above show the term frequency that each word appears within the different novels. Using Verne's text as reference we can obtain some hints about the different styles. At first sight, the vocabulary used by Jules Verne differs from the other two; Apart from the proper nouns, TTLUS is full of words related to sea and voyages such as 'depth', 'coast', 'horizon'. Also, some words regarding technology are much used by Verne: 'light', 'electric' or pressure'. Therefore, using only word frequency we may establish that TTLUS seem to have the advances at the core of the narrative. However, for the other two we can not affirm that.

```
library(scales)

ggplot(frequency_positive, aes(x = proportion, y = `Jules Verne`,
                      color = abs(`Jules Verne` - proportion))) +
  geom_abline(color = "gray40", lty = 2) +
  geom_jitter(alpha = 0.1, size = 0.5, width = 0.3, height = 0.3) +
  geom_text(aes(label = word), check_overlap = TRUE, vjust = 0.5) +
  scale_x_log10(labels = percent_format()) +
  scale_y_log10(labels = percent_format()) +
  scale_color_gradient(limits = c(0, 0.001),
                       low = "darkslategray4", high = "gray75") +
  facet_wrap(~author, ncol = 2) +
  theme(legend.position="none") +
  labs(y = "Jules Verne", x = NULL)
```



Nonetheless, in the plot above we can visualize the relative frequencies; the left bottom part correspond to low frequency area, while the top right corner is full of terms that appears a lot. On the other hand, the black dashed exemplify the words that have a similar frequency in the reference book and the other. In the two cases, verbs like 'added', 'addressed' are shared. These are terms that were usually used in the vocabulary of the XIXth century. However, we can still found some words that are more specific to the genre: 'arms' and 'fear', which can be linked to the action moments in the novels.

```
frequency_negative = tidy_negative %>%
 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(`Mary Shelley`:`Samuel Butler`,
              names_to = "author", values_to = "proportion")%>%
 arrange(desc(`HG Wells`))


frequency_negative
```

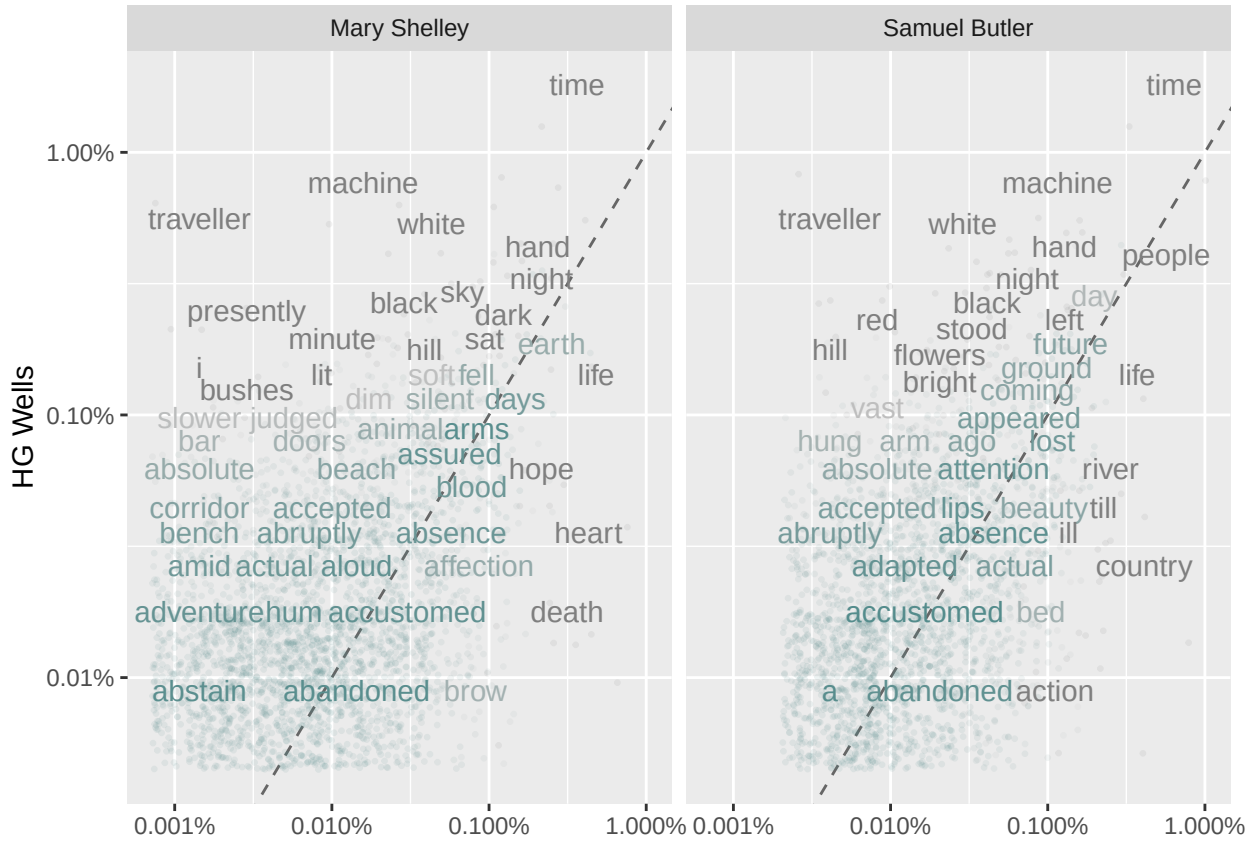
Negative Corpus

```
## # A tibble: 31,134 x 4
```

```
##    word      `HG Wells` author        proportion
##    <chr>          <dbl> <chr>              <dbl>
##  1 time         0.0181  Mary Shelley   0.00364
##  2 time         0.0181  Samuel Butler  0.00638
##  3 machine      0.00766 Mary Shelley   0.000158
##  4 machine      0.00766 Samuel Butler  0.00115
##  5 traveller    0.00561 Mary Shelley   0.0000144
##  6 traveller    0.00561 Samuel Butler  0.0000412
##  7 white        0.00535 Mary Shelley   0.000431
##  8 white        0.00535 Samuel Butler  0.000288
##  9 weena        0.00481 Mary Shelley  NA
## 10 weena        0.00481 Samuel Butler NA
## # ... with 31,124 more rows
```

In the negative corpus of novel, HG Wells was selected the be the reference. Again, we see a lack of proper nouns within the most frequent words. Interestingly, some words with negative connotations do appear in the first pages of the table; terms like 'strange', 'darkness' or 'fear' are present in the work of HG Wells, with lower frequency in the other two. Still, we can start to confirm the initial hypothesis.

```
ggplot(frequency_negative, aes(x = proportion, y = `HG Wells`,
                     color = abs(`HG Wells` - proportion))) +
  geom_abline(color = "gray40", lty = 2) +
  #you can use geom_jitter to adjust the points location and gain visibility
  geom_jitter(alpha = 0.1, size = 0.5, width = 0.3, height = 0.3) +
  geom_text(aes(label = word), check_overlap = TRUE, vjust = 0.5) +
  scale_x_log10(labels = percent_format()) +
  scale_y_log10(labels = percent_format()) +
  scale_color_gradient(limits = c(0, 0.001),
                       low = "darkslategray4", high = "gray75") +
  facet_wrap(~author, ncol = 2) +
  theme(legend.position="none") +
  labs(y = "HG Wells", x = NULL)
```



Using the same plot as before, we can observe in visual terms the differences in term frequency for the three novels. Words like 'abandon', 'afraid' or 'absence' are shared between the reference work and the other two. These terms are usually associated to a negative context. Similarly, the word 'feel' do appear a lot in the three novels! This could hint that these novels are focus on sentiment expressions. Thus, we may acknowledge that the potential negative corpus is behaving as expected. However, frequency alone is not enough to analyse the different vocabulary used.

TF-IDF

In order to analyse the importance the importance of each word, we may resort to the technique known as TF-IDF. It is a weighting method that uses the term frequency of a word (TF) and the inverse document frequency (IDF) to measure the importance of a word, considering the whole set of documents. IDF is calculated as: number of documents / number of documents that posses that term. The main advantage of this method is that we can mathematically assess the distinctiveness of each word. As an example, if the word 'time' appears a lot in the six novels, we can assert that is not distinctive of any work. However, if the word 'nautilus' appears a lot only in one novel, we may say that this term is unique for that book. Hence, distinctive. Therefore, the highest TF-IDF, the more representative is that word in a given text.

To calculate TF-IDF, we would need to tokenize the set of novels, calculate TF and IDF and finally to multiply one for the other. However, this time, due to the nature of the method, we won't need to filter stop words. As they would be present a lot in all the book, they would not be highlighted. Hence:

```
tidy_fiction2 = books  %>%
  unnest_tokens(word, text) %>% # Tokenizing
  count(book, word, sort = TRUE)
```

```
total_fiction = tidy_fiction2 %>%
  group_by(book) %>%
  summarize(total = sum(n)) # Calculating the total words of each book.

books_fiction = left_join(tidy_fiction2, total_fiction) %>% mutate(term_frequency = n/total)    # Calcu

books_fiction
```

```
## # A tibble: 50,931 x 5
##    book                                word      n  total term_frequency
##    <chr>                               <chr> <int>  <int>          <dbl>
##  1 The Last Man                        the   11375 174826         0.0651
##  2 The Mummy!                          the    9969 166812         0.0598
##  3 Twenty Thousand Leagues Under the Sea the    8552 105150         0.0813
##  4 The Last Man                        of     7081 174826         0.0405
##  5 The Last Man                        and    6215 174826         0.0355
##  6 The Mummy!                          and    5437 166812         0.0326
##  7 The Mummy!                          of     5393 166812         0.0323
##  8 The Last Man                        to     5138 174826         0.0294
##  9 The Mummy!                          to     4882 166812         0.0293
## 10 A Princess of Mars                  the    4547  66214         0.0687
## # ... with 50,921 more rows
```

```
book_tf_idf <- books_fiction %>%
  bind_tf_idf(word, book, n) # Function that calculates TF-IDF

book_tf_idf
```

```
## # A tibble: 50,931 x 8
##    book                           word      n  total term_~1     tf   idf tf_idf
##    <chr>                          <chr> <int>  <int>   <dbl>  <dbl> <dbl>  <dbl>
##  1 The Last Man                   the   11375 174826  0.0651 0.0651     0      0
##  2 The Mummy!                     the    9969 166812  0.0598 0.0598     0      0
##  3 Twenty Thousand Leagues Under~ the    8552 105150  0.0813 0.0813     0      0
##  4 The Last Man                   of     7081 174826  0.0405 0.0405     0      0
##  5 The Last Man                   and    6215 174826  0.0355 0.0355     0      0
##  6 The Mummy!                     and    5437 166812  0.0326 0.0326     0      0
##  7 The Mummy!                     of     5393 166812  0.0323 0.0323     0      0
##  8 The Last Man                   to     5138 174826  0.0294 0.0294     0      0
##  9 The Mummy!                     to     4882 166812  0.0293 0.0293     0      0
## 10 A Princess of Mars             the    4547  66214  0.0687 0.0687     0      0
## # ... with 50,921 more rows, and abbreviated variable name 1: term_frequency
```

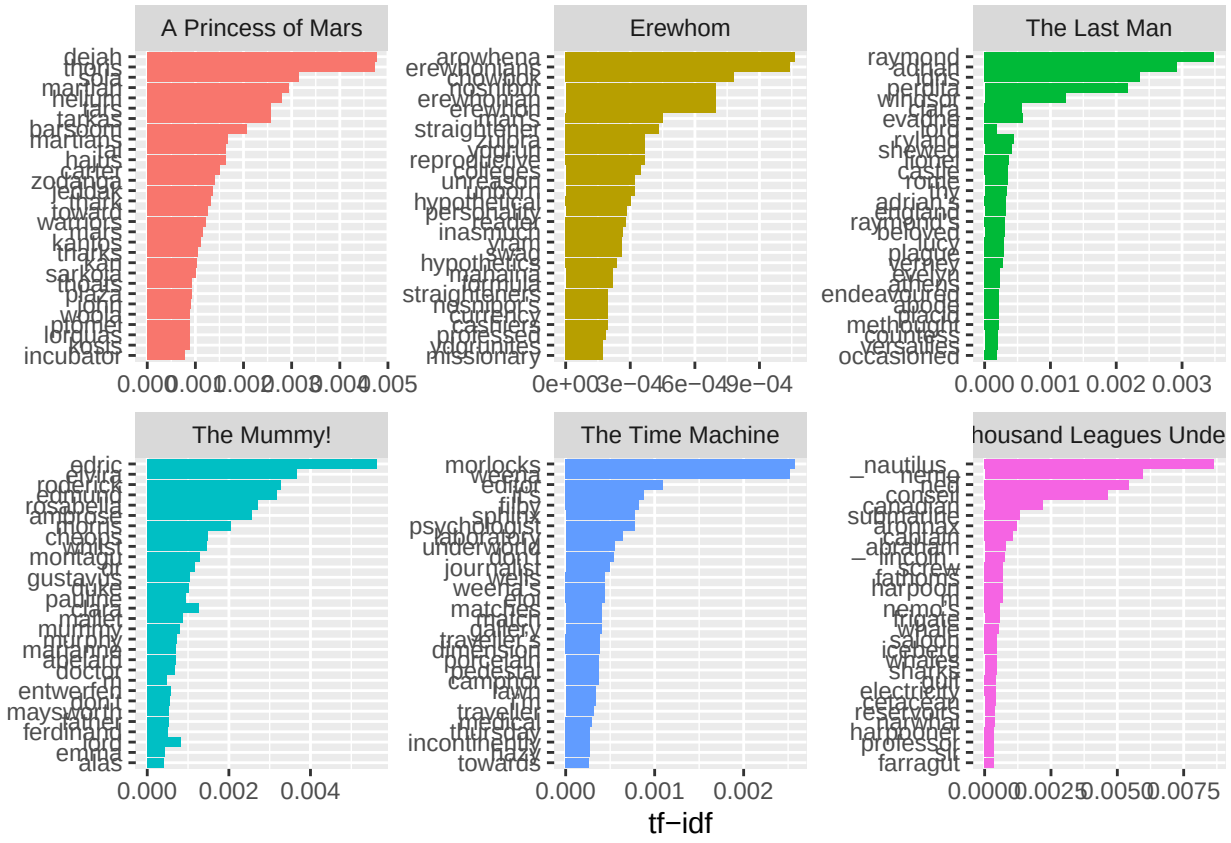
```
book_tf_idf %>%
  select(-total) %>%
  arrange(desc(tf_idf))
```

```
## # A tibble: 50,931 x 7
##    book                                word      n term_~1      tf   idf  tf_idf
##    <chr>                               <chr> <int>   <dbl>   <dbl> <dbl>   <dbl>
##  1 Twenty Thousand Leagues Under the ~ _nau~   507 0.00482 0.00482  1.79 0.00864
##  2 Twenty Thousand Leagues Under the ~ nemo    349 0.00332 0.00332  1.79 0.00595
```

```
##  3 The Mummy!                              edric    523 0.00314 0.00314  1.79 0.00562
##  4 Twenty Thousand Leagues Under the ~ ned      320 0.00304 0.00304  1.79 0.00545
##  5 A Princess of Mars                      dejah    176 0.00266 0.00266  1.79 0.00476
##  6 A Princess of Mars                      thor~    175 0.00264 0.00264  1.79 0.00474
##  7 Twenty Thousand Leagues Under the ~ cons~    273 0.00260 0.00260  1.79 0.00465
##  8 The Mummy!                              elvi~    340 0.00204 0.00204  1.79 0.00365
##  9 The Last Man                            raym~    340 0.00194 0.00194  1.79 0.00348
## 10 The Mummy!                              rode~    305 0.00183 0.00183  1.79 0.00328
## # ... with 50,921 more rows, and abbreviated variable name 1: term_frequency
```

The table below show the TF-IDF value for all the words in the set of novels, arranged in descending order. The first thing to notice is the absence of pessimistic novels in the first pages. Again, the most distinctive words are proper nouns. Again, this is a significative point towards the impersonal tone mentioned before. Similarly, following proper nouns, we can see the importance of word related to technology in the optimistic novels and sometimes in the pessimistic: 'incubator', 'laboratory', 'frigate' or 'electricity' are distinctive in the novels, specially in Twenty Thousand Leagues under the sea.

```
book_tf_idf %>%
  group_by(book) %>%
  #choose maximum number of words
  slice_max(tf_idf, n = 30) %>%
  ungroup() %>%
  ggplot(aes(tf_idf, fct_reorder(word, tf_idf), fill = book)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~book, ncol = 3, scales = "free") +
  labs(x = "tf-idf", y = NULL)
```



The plot above shows the 30 words more distinctive of each novels. Both Erewhom and A Princess of Mars are full of invented terms as they are based in imagined societies. Still there is one feature that is significant: TF-IDF values in Erewhom are very close to 0. This could mean that this novel does not have a particular style. Acknowledging that it is usually labelled as a satire, we might speculate that it uses words and terms repeated in every sci-fi novel, as a way to emphasise the irony. On the other hand, The Last Man accounts for many words related to high classes such as 'thy', 'lord', 'castle' or 'countess'. Given the dystopia nature of the work (the word 'plague' is also distinctive), we may assess that the novel may focus on elites from an ironic point of view.

In this line of reasoning, the distinctive use of the words 'morlocks', 'weena', 'eloi' and 'traveller' in HG Wells' The Time Machine is highly significant. The author never provide names to the humanoid creatures of the future, apart from the feminine protagonist, nor to the human (who is called as *time traveler*. Again, this could be interpreted as a pessimistic feature of the novel: HG Wells is describing a future in which "humans" have lost their personalities and identities. With respect to TTLUS, we can observe the distinctiveness of scientific words. The novel can be easily identified for the importance that biological and technical related terms have. Lastly, The Mummy! have a lot of proper nouns as distinctive words. Thus, it is difficult to extract meaningful information.

Sentiment analysis

The second technique that I will use to analize the differences between optimistic and pessimistic novels is sentiment analysis. At this point, I will rely on different algorithms to calculate the *amount* of positive and negative words in each novel. These algorithms are based on a lexicon, which could either asign each word a value depending on meaning or classify them as negative or positive.

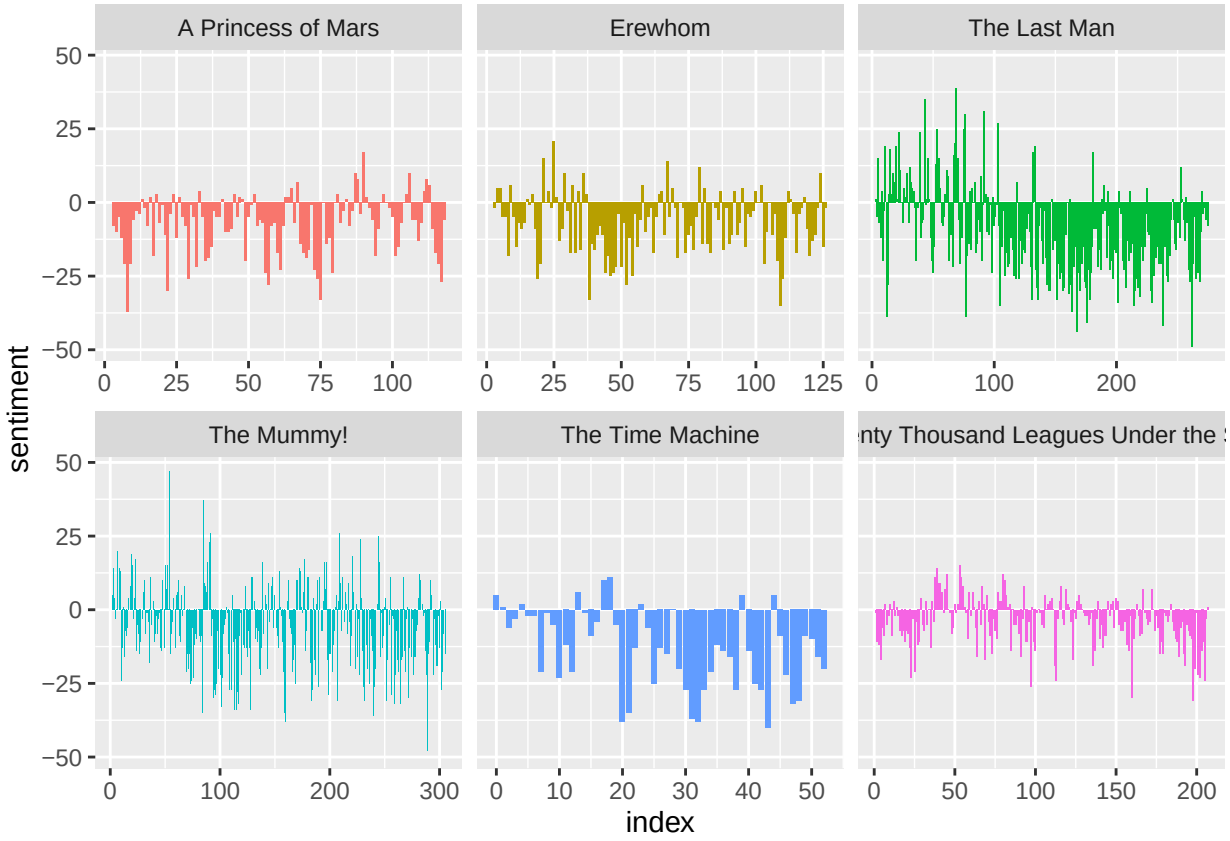
Bing lexicon The first type of algorithm that I will use is the one labelled as 'Bing Lexicon'. It is a multiclass algorithm, as it provides a label to each word (positive and negative). In the next step, I will divide each novel in groups of 60 lines each, and classify the words within. Then I will obtain the net sentiment by subtracting the total number of positive words to the total number of negative.

```
fiction_sentiment <- tidy_fiction %>%
  inner_join(get_sentiments("bing")) %>%
  count(book, index = linenumber %/% 60, sentiment) %>%
  pivot_wider(names_from = sentiment, values_from = n, values_fill = 0)  %>%
  mutate(sentiment = positive - negative)

fiction_sentiment
```

```
## # A tibble: 1,077 x 5
##    book               index negative positive sentiment
##    <chr>              <dbl>    <int>    <int>     <int>
##  1 A Princess of Mars     3       15        7        -8
##  2 A Princess of Mars     4       17        7       -10
##  3 A Princess of Mars     5       17       12        -5
##  4 A Princess of Mars     6       22       10       -12
##  5 A Princess of Mars     7       35       14       -21
##  6 A Princess of Mars     8       42        5       -37
##  7 A Princess of Mars     9       39       18       -21
##  8 A Princess of Mars    10       18       12        -6
##  9 A Princess of Mars    11       14       11        -3
## 10 A Princess of Mars    12       18       14        -4
## # ... with 1,067 more rows
```

The table above shows this process. As an example, the first chunk of 60 lines for A Princes of Mars contains 15 negative terms and 7 positive. Thus, a net sentiment of -8. However, as it is not feasible to go all over the table to see the overall sentiment of the novels, we can make a graph to visualize this information.

```
ggplot(fiction_sentiment, aes(index, sentiment, fill = book)) +
  geom_col(show.legend = TRUE) +
  facet_wrap(~book, ncol = 3, scales = "free_x")+
  guides(fill = "none")
```



Surprisingly, there is not a pattern that differentiates those novels that are optimistic about the future and those that are negative. Among the first group, only The Mummy! have some stable moments of positive moments across all the novel. TTLUS only have minor positive moments at the beginning of the book, and A Princess of Mars do not have at all. Within the negative corpus, The Time Machine and Erewhom are constantly using negative vocabulary. Surprisingly, The Last Man do have a lot of positive moments at the beginning of the novel. Before conducting any further analysis, we could still resort to other lexicons to cross validate the results.

```
library(textdata)

afinn <- tidy_fiction %>%
  inner_join(get_sentiments("afinn")) %>%
  group_by(book, index = linenumber %/% 60) %>%
  summarise(sentiment = sum(value)) %>%
  mutate(method = "AFINN")
```

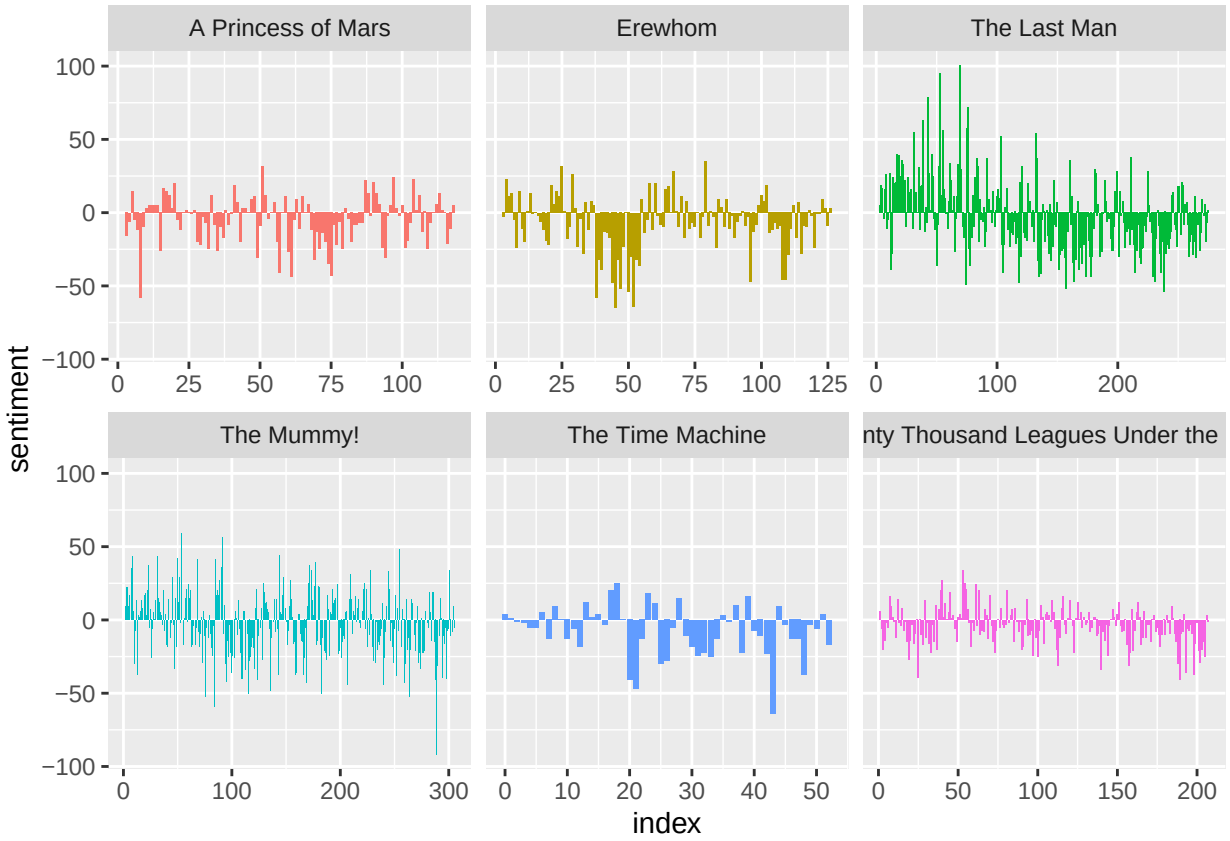
```
afinn
```

AFINN

```
## # A tibble: 1,077 x 4
## # Groups:   book [6]
##    book                index sentiment method
##    <chr>               <dbl>     <dbl> <chr>
##  1 A Princess of Mars      3       -16 AFINN
##  2 A Princess of Mars      4        -6 AFINN
##  3 A Princess of Mars      5        15 AFINN
##  4 A Princess of Mars      6        -5 AFINN
##  5 A Princess of Mars      7       -12 AFINN
##  6 A Princess of Mars      8       -58 AFINN
##  7 A Princess of Mars      9       -10 AFINN
##  8 A Princess of Mars     10         3 AFINN
##  9 A Princess of Mars     11         5 AFINN
## 10 A Princess of Mars     12         5 AFINN
## # ... with 1,067 more rows
```

The AFINN method is considered a regression, as it provides each word a value between -3 (the most negative) to 3 (the most positive). The very first line serves to dmeonstrate the differences between both lexicons; in this case, it is assigned a value of -16, contrary to the -8 posed with the 'Bing' classifier. However, to fully compare both lexicons we can visualize the AFINN method:

```
ggplot(afinn, aes(index, sentiment, fill = book)) +
  geom_col(show.legend = TRUE) +
  facet_wrap(~book, ncol = 3, scales = "free_x")+
  guides(fill = "none")
```



The image provide by the AFINN lexicon is slightly different to the one showed by 'Bing'. On the one hand, The Mummy! have parity within the two types of moments; there positive and negative occasions across all the novel. Similarly,in A Princess of Mars and TTLUS while still overall full of negative moments, there are many spikes in which we could observe chunks with much more positive words. The negative corpus have more or less the same patterns of sentiment. However, the positive moments are enlarged. This is fully acknowledgeable in The Last Man, where the beginning is much more positive with the AFINN algorithm. Overall, these results may hint that the second lexicon tends to be more positive.

```
nrc_sentiment = tidy_fiction %>%
  group_by(book) %>%
  inner_join(get_sentiments("nrc") %>%
  filter(sentiment %in% c("negative", "positive")) %>%
  mutate(method = "NRC")) %>%
  count(method, index = linenumber %/% 80, sentiment) %>%
  pivot_wider(names_from = sentiment,
              values_from = n,
              values_fill = 0) %>%
  mutate(sentiment = positive - negative)

nrc_sentiment
```

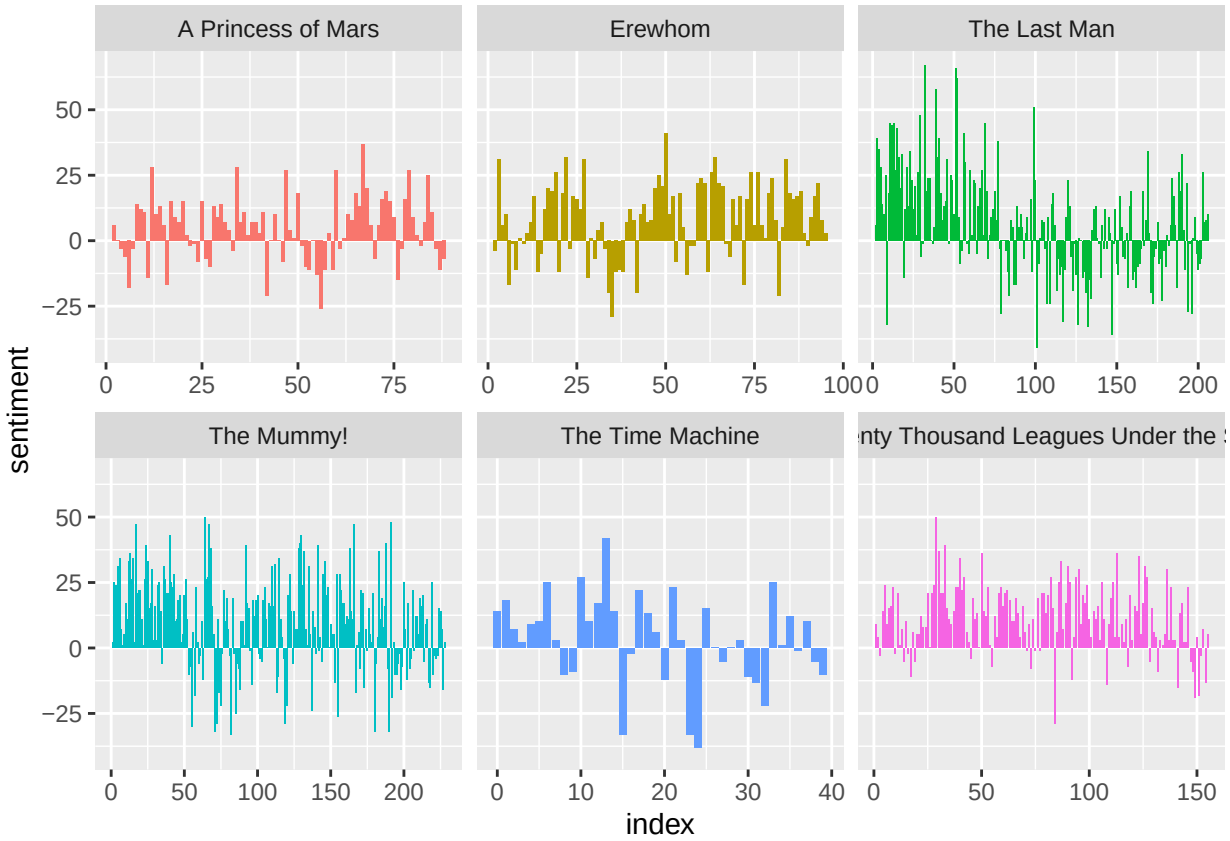
NRC

```
## # A tibble: 809 x 6
```

```
## # Groups:   book [6]
##    book                method index negative positive sentiment
##    <chr>               <chr>  <dbl>    <int>    <int>     <int>
##  1 A Princess of Mars NRC        2       22       28         6
##  2 A Princess of Mars NRC        3       27       27         0
##  3 A Princess of Mars NRC        4       32       29        -3
##  4 A Princess of Mars NRC        5       28       22        -6
##  5 A Princess of Mars NRC        6       49       31       -18
##  6 A Princess of Mars NRC        7       36       33        -3
##  7 A Princess of Mars NRC        8       18       32        14
##  8 A Princess of Mars NRC        9       26       38        12
##  9 A Princess of Mars NRC       10       28       39        11
## 10 A Princess of Mars NRC       11       27       13       -14
## # ... with 799 more rows
```

When applying the third and final lexicon, we observe a complete different visual analysis. The NRC lexicon is a multilabel algorithm that asign either a sentiment or an emotion to each word. Among them, it may assign a 'positive' or a 'negative' label. Therefore, we could apply the same analysis as before. However, as the plot below show, the results are different.

```
ggplot(nrc_sentiment, aes(index, sentiment, fill = book)) +
  geom_col(show.legend = TRUE) +
  facet_wrap(~book, ncol = 3, scales = "free_x")+
  guides(fill = "none")
```



In this case, we can observe that most of the novels are positive sentiment orientated. The difference with the previous two lexicons is significative. Therefore, we may be not sure of the suitability of this algorithm

for the analysis. Nonetheless, there is one aspect to highlight: within this positive *inflation* we can still detect which novel belong to the optimistic corpus. TTLUS and The Mummy! do not possess major or continued chunks of negative words; A Princess of Mars have one big negative chunk. Even if this lexicon is positive biased, these two novels have no clear negative moments. On the other hand, The Time Machine and Erewhom do have some negative *stints*. Finally, The Last Man has the same pattern as before, only this time the positivity is exacerbated. At the beginning, there is a predominance of optimistic terms. Then, this positive words allow for the dominance of negative words.

General conclusion for sentiment analysis The visualization of the lexicon algorithms allow to extract various insights. On the one hand, each method possess a particular way to label/classify each other. This has the consequence that the novel may be biased towards a side of the spectrum. The first two cases seem to be negative oriented, while the third one seem to be positive oriented. Also, it could also be the case that due to the very own literary genre, there is a tendency towards negative words (action moments, conflict, etc). However, in relative terms, we could indeed observe that those novels that were presumably negative, tended to demonstrate it in the plots. Within the three lexicons, The Time Machine and Erewhom stood out for having the most negative valorations. In this line of reasoning, it is worth noting the argument plot that The Last Man seem to have. The first act of the novel seem to be positive (in relative terms), but as soon as the novels advances, most of the chunks of lines are dominated by negative words.

With respect to the potential optimistic novels, it is clear that in relative terms, they include much more positive terms. With the exception of A Princess of Mars, these novels possess much more positive moments than their counterparts in the negative-biases lexicons. Similarly, we could notice that in the positive-oriented lexicon TTLUS and The Mummy! did not posses major negative moments, contrary to the negative corpus. Summing up, if we focus on the relative differences between corpuses, we can clearly see a differences in style between the Positive and Negative corpuses. Thus, a clear hint that the initial hypothesis was right.

N-Grams

The fourth and final technique that I will use to compare both set of corpuses is N-Grams. At this step of the analysis the scope of the work is going to be modified. More specifically, instead of tokenizing the texts into individual words, I will separate the novels into units of two words. Therefore, I will use the unnest_tokens function specifying the bigram.

```
fiction_bigrams <- books %>%
  select(text, book, author, linenumber) %>% # We don't need the chapter anymore
  unnest_tokens(bigram, text, token = "ngrams", n = 2) %>% # We choose pairs of words
  filter(!is.na(bigram)) # Remove those pairs that include an NA

fiction_bigrams
```

```
## # A tibble: 573,190 x 4
##    book                author              linenumber bigram
##    <chr>               <chr>                    <int> <chr>
##  1 A Princess of Mars Edgar R. Burroughs         191 chapter i
##  2 A Princess of Mars Edgar R. Burroughs         192 on the
##  3 A Princess of Mars Edgar R. Burroughs         192 the arizona
##  4 A Princess of Mars Edgar R. Burroughs         192 arizona hills
##  5 A Princess of Mars Edgar R. Burroughs         195 i am
##  6 A Princess of Mars Edgar R. Burroughs         195 am a
##  7 A Princess of Mars Edgar R. Burroughs         195 a very
##  8 A Princess of Mars Edgar R. Burroughs         195 very old
##  9 A Princess of Mars Edgar R. Burroughs         195 old man
## 10 A Princess of Mars Edgar R. Burroughs         195 man how
## # ... with 573,180 more rows
```

Bigrams TF-IDF With the dataset divided into pairs of words, we can perform the same techniques as before. However there are some advantages; as the words are grouped, some context is added to the analysis. Thus, more information can be extracted from the analysis. As an example, we can obtain the most common pairs of words:

```
fiction_bigrams %>%
  count(bigram, sort = TRUE)
```

```
## # A tibble: 229,707 x 2
##    bigram       n
##    <chr>    <int>
##  1 of the    5577
##  2 in the    2485
##  3 to the    2219
##  4 and the   1468
##  5 it was    1105
##  6 on the    1084
##  7 from the   997
##  8 by the     920
##  9 to be      901
## 10 of a       855
## # ... with 229,697 more rows
```

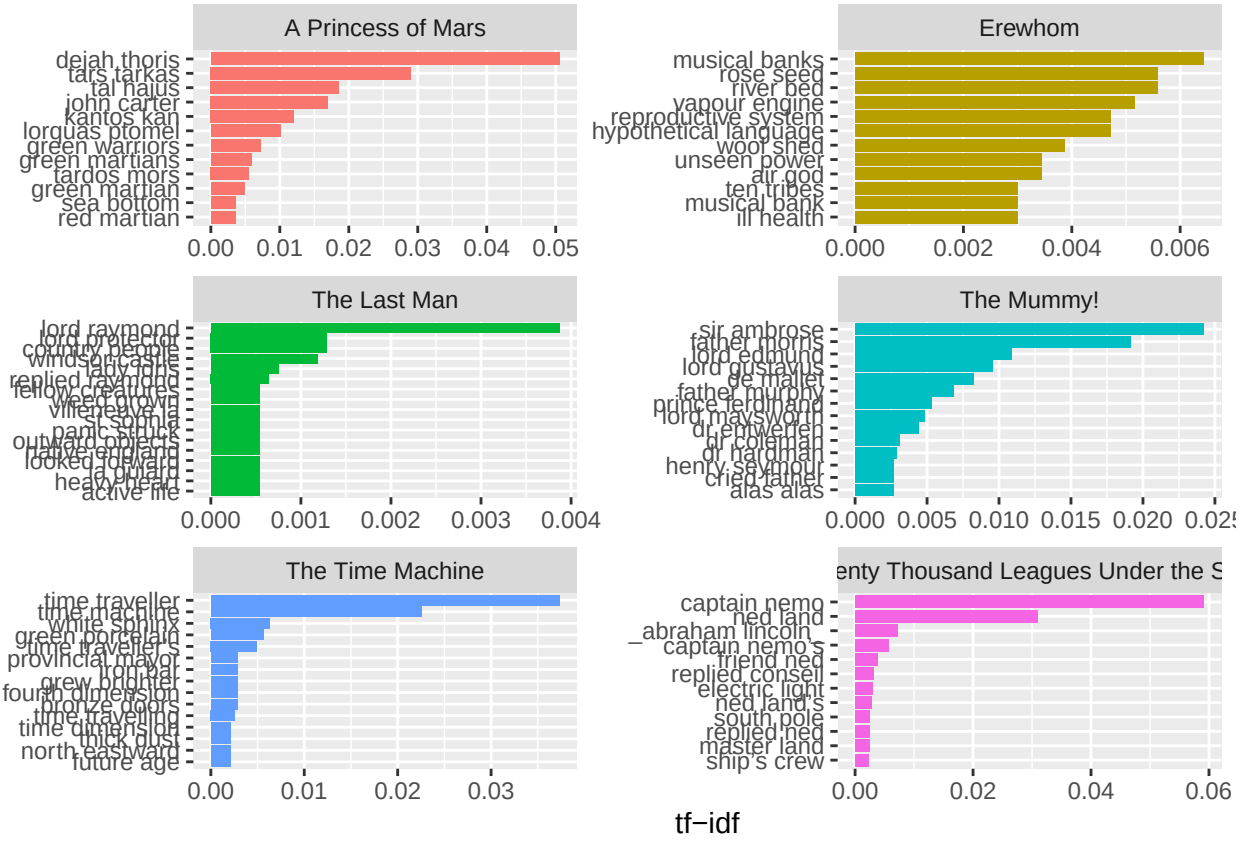
Not surprisingly, the most frequent pair of words contain terms that may be considered as stopwords. Thus, for future analysis, we should filter them from the analysis. To do that, first we need to separate the bigrams in separate columns, so we can easily eliminate those observations that are formed by stop words in the two columns. Then, we can apply the TF-IDF function to the filtered pairs of words.

```
fbigrams <- fiction_bigrams %>%
  #we separate each bigram in two columns, word1 and word2
  separate(bigram, c("word1", "word2"), sep = " ")%>%
  filter(!word1 %in% stop_words$word) %>%
  filter(!word2 %in% stop_words$word) %>%
  unite(bigram, word1, word2, sep = " ") %>% # doing TF-IDF
  count(book, bigram) %>%
  bind_tf_idf(bigram, book, n) %>%
  arrange(desc(tf_idf))
fbigrams
```

```
## # A tibble: 47,028 x 6
##    book                                bigram           n      tf   idf tf_idf
##    <chr>                               <chr>        <int>   <dbl> <dbl>  <dbl>
##  1 Twenty Thousand Leagues Under the Sea captain nemo   326 0.0330   1.79 0.0591
##  2 A Princess of Mars                  dejah thoris   155 0.0282   1.79 0.0506
##  3 The Time Machine                    time travel~    53 0.0208   1.79 0.0373
##  4 Twenty Thousand Leagues Under the Sea ned land       171 0.0173   1.79 0.0310
##  5 A Princess of Mars                  tars tarkas     89 0.0162   1.79 0.0290
##  6 The Mummy!                          sir ambrose    220 0.0135   1.79 0.0242
##  7 The Time Machine                    time machine    32 0.0126   1.79 0.0225
##  8 The Mummy!                          father morr~   174 0.0107   1.79 0.0192
##  9 A Princess of Mars                  tal hajus       57 0.0104   1.79 0.0186
## 10 A Princess of Mars                  john carter     52 0.00947  1.79 0.0170
## # ... with 47,018 more rows
```

According to the tables above, the most distinctive pairs of words are those composed by name and surname/title: Captain Nemo, John Carter or even time traveller. To make the study easier, we can plot the most common bigrams.

```
fbigrams%>%
  group_by(book) %>%
  slice_max(tf_idf, n = 12) %>% # keep only words that appear at least 12 times
  ungroup() %>%
  ggplot(aes(tf_idf, fct_reorder(bigram, tf_idf), fill = book)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~book, ncol = 2, scales = "free") +
  labs(x = "tf-idf", y = NULL)
```



Following the same *rules* of the TF-IDF plot from token-words, we can visualize the most distinctive pairs of words from each novel. Some trends observed before do also appear in this case: there is a significant absence of proper nouns in the negative corpus. It is important here to take into consideration that the male protagonist of The Time Machine does not have a name. Not surprisingly, the bigrams that are unique to this novel are all related to time and/or physics and time travel such as 'time machine', 'time travel' or even 'fourth dimension'. Even more surprisingly, the novel Erewhom differentiates itself by using a lot of technological terms, such as 'vapour engine', 'reproductive system' or 'unseen power'. With respect to The Last Man, apart from Lord Raymond, there is no specific group of words that stand out.

Respecting the positive corpus, we can also observe the same tendency of unitary tokens: many proper nouns. However, there is less frequency of technological terms. There is however a novelty in TTLUS: the bigrams 'replied ned' and 'replied conseil' are distinctive of the work. This may yield and interesting insight. There seem to be a lot of conversations between the protagonist. Apart from a similar single observation in The Last Man, no other novel have this grammatical construction as distinctive.

Summing up, providing some context to the tokenization, it is more clear the unpersonalistic style of the negative corpus, which in the end may be a clear sign of pessimistic style. However, there are a lot of technological words in a presupposed negative novel. This *discovery* goes against our initial hypothesis. For the moment, one can argue that the (distinctive) use of technological vocabulary is not particular from optimistic novels, but maybe in general for these contemporary novels.

Bigrams Sentiment Analysis Another big advantage of bigrams is the possibility to contemplate sentiment analysis in a more comprehensive way; from a grammatical point of view, we usually apply negative complements to words: 'I'm **not** happy' or 'you are **never** laughing'. Unitary tokens do not observe this distinction when performing sentiment analysis. The words 'happy' or 'laughing' would contribute positively to the metrics when in reality they are not adding positivity to the narration.

To exemplify this distinction, we can gather in a single table all the bigrams that start with the word 'not':

```
AFINN <- get_sentiments("afinn")

negative_fiction = fiction_bigrams %>%
  separate(bigram, c("word1", "word2"), sep = " ")%>%
  filter(word1 == "not") %>%
  inner_join(AFINN, by = c(word2 = "word")) %>%
  count(word1, word2, value, sort = TRUE)

negative_fiction
```

```
## # A tibble: 196 x 4
##    word1 word2  value     n
##    <chr> <chr>  <dbl> <int>
##  1 not   wish       1    27
##  2 not   help       2    20
##  3 not   love       3    16
##  4 not   like       2    15
##  5 not   fear      -2    14
##  6 not   doubt     -1    13
##  7 not   forget    -1    13
##  8 not   leave     -1    13
##  9 not   die       -3    12
## 10 not   want       1    12
## # ... with 186 more rows
```

In the table above, the column label as value account for the amount that the second word would contribute to the sentiment analysis if it were analyse as a token (using the AFINN lexicon). The second column how many times each pair formed by 'not' appears. Considering this table, we may calculate the exact contribution to the sentiment analysis.

```
frequent_not_words <- negative_fiction %>%
  mutate(contribution = n * value) %>% # measuring the contribution to sentiment analysis.
  arrange(desc(abs(contribution))) %>%
  mutate(word2 = reorder(word2, contribution)) %>%
  head(20)

frequent_not_words
```

```
## # A tibble: 20 x 5
```

```
##    word1 word2     value     n contribution
##    <chr> <fct>     <dbl> <int>        <dbl>
##  1 not   love          3    16           48
##  2 not   help          2    20           40
##  3 not   die          -3    12          -36
##  4 not   like          2    15           30
##  5 not   fear         -2    14          -28
##  6 not   wish          1    27           27
##  7 not   despair      -3     8          -24
##  8 not   hesitate     -2    10          -20
##  9 not   hope          2    10           20
## 10 not   satisfied     2     8           16
## 11 not   suffer       -2     8          -16
## 12 not   daring        2     7           14
## 13 not   doubt        -1    13          -13
## 14 not   forget       -1    13          -13
## 15 not   leave        -1    13          -13
## 16 not   want          1    12           12
## 17 not   care          2     6           12
## 18 not   mistaken     -2     6          -12
## 19 not   refuse       -2     6          -12
## 20 not   dead         -3     4          -12
```

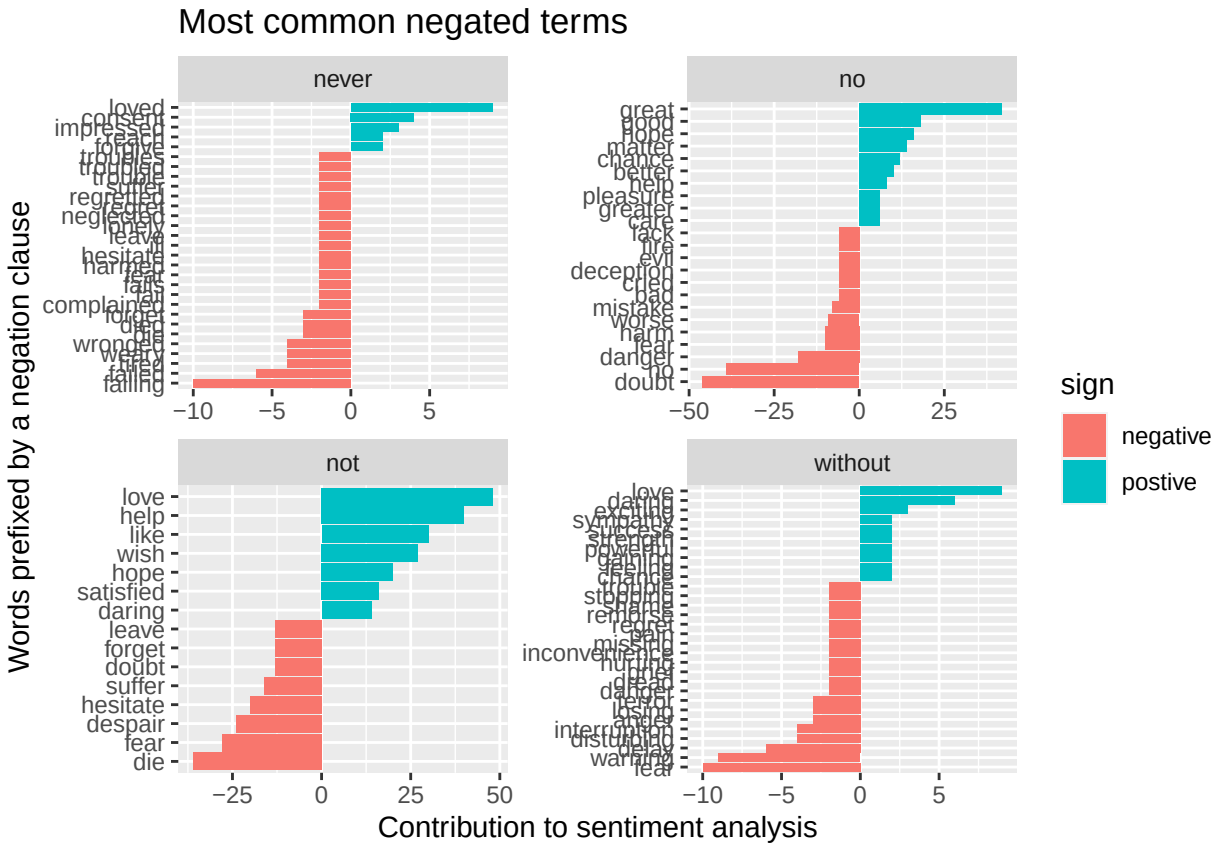
As an example, the word love (which sum up to 3 in the AFINN scale), appear 16 times preceded by 'not', adding a total of 48 to the positive side. This would be the noise that the absence of context would add to the sentiment analysis. Nonetheless, we discover the fact that negative words may also be neglected. Thus, the bias toward positivity could be compensated with these double negations. In this line of reasoning, 'not' is not the only word that serves to neglect words. Other terms such as 'no', 'never' or 'without'. Thus, we should expand the previous analysis adding these set of words:

```
negation_words <- c("not", "no", "never", "without") # vector with the negating words

negated_words <- fiction_bigrams %>%
  separate(bigram, c("word1", "word2"), sep = " ") %>%
  filter(word1 %in% negation_words) %>%
  inner_join(AFINN, by = c(word2 = "word")) %>%
  count(word1, word2, value, sort = TRUE)

negated_words
```

```
## # A tibble: 364 x 4
##    word1 word2  value     n
##    <chr> <chr>  <dbl> <int>
##  1 no    doubt     -1    46
##  2 no    no        -1    39
##  3 not   wish       1    27
##  4 not   help       2    20
##  5 not   love       3    16
##  6 not   like       2    15
##  7 no    great      3    14
##  8 no    matter     1    14
##  9 not   fear      -2    14
## 10 not   doubt     -1    13
## # ... with 354 more rows
```

The table above demonstrates that many negative words are often negated. For example, the word 'delay' is accompanied by 'without' six times; failing is anticipated 5 by never. In order to effectively gather information from the table below, we may visualize which are the words more usually prefixed with a negation clause, in order to evaluate their impact in sentiment analysis.

```
negated_words %>%
  mutate(contribution = n * value,
         sign = if_else(value > 0, "postive", "negative")) %>% # creating the categories according thei
  group_by(word1) %>%
  top_n(15, abs(contribution)) %>% # plotting the actual contribution to sentiment analysis
  ungroup() %>%
  ggplot(aes(y = reorder_within(word2, contribution, word1),
             x = contribution,
             fill = sign)) +
  geom_col() +
  scale_y_reordered() +
  facet_wrap(~ word1, scales = "free") +
  labs(y = 'Words prefixed by a negation clause',
       x = "Contribution to sentiment analysis",
       title = "Most common negated terms")
```



The first interesting insight that may be obtained with the plot showed above refers to the uneven status of words negated; apart from 'no' and 'not' there seem to be more incorrectly assessed contribution to the negative side. The words 'never' and 'without' seem to be more usually followed by negative words such as 'never troubled' and 'never failed' or 'without fear' or 'without delay'. These two words may serve to actually emphasise positive features of the protagonist. In other words, they could be representing the absence of

negative traits of the characters.

For the shake of our analysis, we may asses that this is another key bias towards negativity. While in absolute terms the novels should be less pessimistic, if these negative-inflations is distributed around the six novels, we could still contemplate that in relative circumstances, there would be a difference between both corpuses.

Networks The last technique that I will perform is the network visualization of the bigrams. Using the *ggraph* and the *igraph* libraries, we may represent the most important relationships between the words, to finally form bigrams. In other words, which set of terms usually relate to each other.

As a previous step, we would need to re-do the table of tokenize bigrams without stopwords (I did this but applying TF-IDF at the same time) and couting again the most frequent ones.

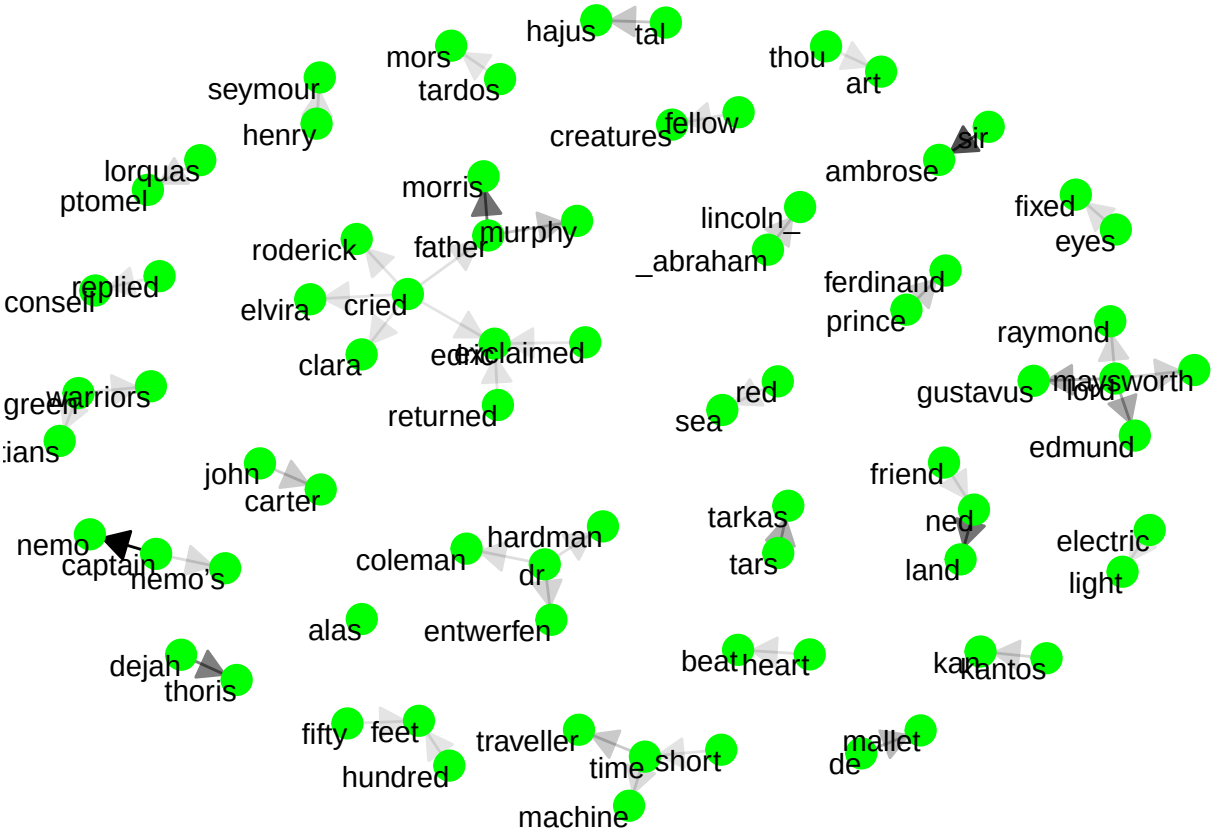
```
library(ggraph)
library(igraph)


fiction_count = fiction_bigrams %>%
  separate(bigram, c("word1", "word2"), sep = " ")%>%
  filter(!word1 %in% stop_words$word) %>% # Eliminate bigrams with stop words in both
  filter(!word2 %in% stop_words$word) %>%
  count(word1, word2, sort = TRUE)

fbigram_for_graph <- fiction_count %>%
  filter(n > 15) %>%
  graph_from_data_frame() # prepare the inputs for the graphical representation
```

After setting the inputs for the graph, we may create and actual arrow to represent the direction of the relationship.

```
a <- grid::arrow(type = "closed", length = unit(.15, "inches"))
```

Finally, using *ggraph* we can plot the network of the bigrams

```
ggraph(fbigram_for_graph, layout = "fr") +
  geom_edge_link(aes(edge_alpha = n), show.legend = FALSE,
                 arrow = a, end_cap = circle(.07, 'inches')) +
  geom_node_point(color = "green", size = 5) +
  geom_node_text(aes(label = name), vjust = 1, hjust = 1) +
  theme_void()
```



This cloud of words interacting to each other represent all the connections established during the novels at least 15 times. Not surprinsigly, words like 'lord', 'father' or 'dr' appear to have numerous relationships. As the novels belong to the XIXth century, this could be anticipated. These words are typical epithet and/or motives from this period of time. Similarly, names and surnames are usually related. Lastly, some scientific annotations usually links to each other such as 'feet' with a particular number or 'electric' and 'light'.

General conclusion

The main objective of this work was to evaluate the differences in styles between science fiction novels of the XIXth century. More specifically, I hypothesised that there were different patterns between those novels that are presumably optimistic and pessimistic. On the one hand, frequency analysis and TF-IDF do hint that negative novels tend to be more deliberative impersonal, and more abundant on words that emphasise negative aspects as 'fear' or 'arms'. Sentiment analysis also demonstrated that in relative terms, the pessimistic set of novels possess more sections of the novels with an abundance of negative words. In absolute terms, sentiment analysis is not as clear as the lexicons do differ between them in assigning sentiment values, and bigrams showed that there is a tendency to overestimate negative words.

Finally, I also hypothesised that scientific terms could be more disticntive of optimistic novels, as a mean to represent the hope for the future. However, both term frequency and bigrams, clue that these type of notations are somehow spread around all the novels.