Taming the Tiger: Sentiment Analysis of Clemenceau

Claire

2024-05-07

An application of Text Mining with R: a Tidy Approach by Julia Silge and Davig Robinson, 2017 to Clemenceau's writings.

R Data Preparation

The first step is to install and load the necessary packages. Data for this analysis will come from Project Gutenberg.

```
#install.packages("gutenbergr")
#devtools::install_github("ropensci/gutenbergr")


library(tidytext)
library(tidyr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(stringi)
library(stringr)
library(rJava)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --

## v ggplot2 3.4.0     v purrr   0.3.5
## v tibble  3.1.8     v forcats 0.5.2
## v readr   2.1.3
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(tm)
```

```
## Loading required package: NLP
##
## Attaching package: 'NLP'
##
## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```r
library(gutenbergr)
```

The next step is to load in the data. Project Gutenberg only has two works written by former Prime Minister of France, George Clemenceau currently, so I will use both of them.

```r
#The Surprises of Life, translated by Grace Hall, 2012
#https://www.gutenberg.org/cache/epub/40618/pg40618-images.html

surprises <- gutenberg_download(40618, mirror =
                                "http://mirrors.xmission.com/gutenberg/")

#South America To-day, 2014
#https://www.gutenberg.org/cache/epub/45621/pg45621-images.html

south <- gutenberg_download(45621, mirror =
                                "http://mirrors.xmission.com/gutenberg/")
```

With the data loaded, now it can be cleaned. I want to remove the introduction in Surprises of Life, the end notes, and any blank rows. For South America To-Day, we can leave the introduction since it was written by Clemenceau himself. But, we'll want to get rid of the footnotes. The idea here is to get rid of as much of the text that Clemenceau didn't write as possible.

```r
#Removing intro and addenda
surprises_two <- surprises[-c(6523:6538),]
surprises_clean <- surprises_two[-c(1:88),]

south_two <- south[-c(8075:9090, 8076:8083, 7999:8017, 7389:7393, 6683:6733,
                      6035:6079, 5556:5574, 4043:4067, 3494:3525, 2879:2918,
                      2247:2270, 1636:1733, 1083:1089),]
south_clean <- south_two[-c(1:43),]

#Removing blank rows
surprises_clean <- surprises_clean[!surprises_clean$text == "", ]
south_clean <- south_clean[!south_clean$text == "", ]

#Add work column
surprises_clean$Work <- "Surprise"
south_clean$Work <- "South"

#Remove the Gutenberg ID column
surprises_clean <- surprises_clean %>% select(-c(gutenberg_id))
south_clean <- south_clean %>% select(-c(gutenberg_id))
```

That's better! Now that we've removed all of the extra text, we can begin to set up the data set so that's it's useful for analysis.

I think it would be nice to have a version of each book separately, and then one combined.

```
clemenceau_corpus <- rbind(south_clean, surprises_clean)
```

We also have to remove all the stop words. These words don't add anything in terms of sentiment. We can't tell how Clemenceau feels from his use of the word "the". So let's get rid of it! We'll set up a custom stop-word dictionary that can remove other words we don't want, like the footnote indicators.

```
#for getting rid of footnote indicators
word <- c("1","2","3","4","5","6","7","8","9","0","[","]")
lexicon <-  rep("custom.stop", times=length(word))

custom.stopwords <- data.frame(word, lexicon)
names(custom.stopwords) <- c("word", "lexicon")

stop_words <-  dplyr::bind_rows(stop_words, custom.stopwords)
```

And now, for the first text-mining aspect of this analysis, we'll divide the texts into words, and remove all the stop words.

```
#unnest tokens
clemenceau_byword <- clemenceau_corpus %>% unnest_tokens(word, text)
south_byword <- south_clean %>% unnest_tokens(word, text)
surprise_byword <- surprises_clean %>% unnest_tokens(word, text)

#remove stop words
tidy_clemenceau <- clemenceau_byword %>% anti_join(stop_words)
```

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

```
tidy_south <- south_byword %>% anti_join(stop_words)
```

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

```
tidy_surprise <- surprise_byword %>% anti_join(stop_words)
```

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

This leaves us with a combined data set of 53660 total words. South America To-Day has 31295 words, and The Surprises of Life has 22365 words.

Introductory Analysis

Next, let's do some more text mining to determine Clemenceau's most commonly used words.

```r
tidy_clemenceau %>% count(word, sort = TRUE)
```

```
## # A tibble: 12,963 x 2
##    word          n
##    <chr>     <int>
##  1 time        209
##  2 day         204
##  3 argentine   195
##  4 life        176
##  5 french      143
##  6 world       125
##  7 public      123
##  8 country     116
##  9 found       114
## 10 ayres       102
## # i 12,953 more rows
```

```r
tidy_south %>% count(word, sort = TRUE)
```

```
## # A tibble: 9,078 x 2
##    word          n
##    <chr>     <int>
##  1 argentine   195
##  2 french      140
##  3 time        116
##  4 ayres       102
##  5 buenos      102
##  6 public      100
##  7 life         92
##  8 country      90
##  9 land         90
## 10 brazil       73
## # i 9,068 more rows
```

```r
tidy_surprise %>% count(word, sort = TRUE)
```

```
## # A tibble: 7,884 x 2
##    word       n
##    <chr>  <int>
##  1 day      133
##  2 time      93
##  3 life      84
##  4 people    64
##  5 eyes      59
##  6 days      54
##  7 world     54
##  8 love      53
##  9 found     52
## 10 simon     52
## # i 7,874 more rows
```

```r
#counts Clemenceau Corpus - tf

count_clemenceau <- tidy_clemenceau %>%
  count(Work, word,sort = TRUE) %>%
  ungroup()

total_clemenceau <- count_clemenceau %>%
  group_by(Work) %>%
  summarize(total = sum(n))

words_by_book <- left_join(count_clemenceau, total_clemenceau)
```

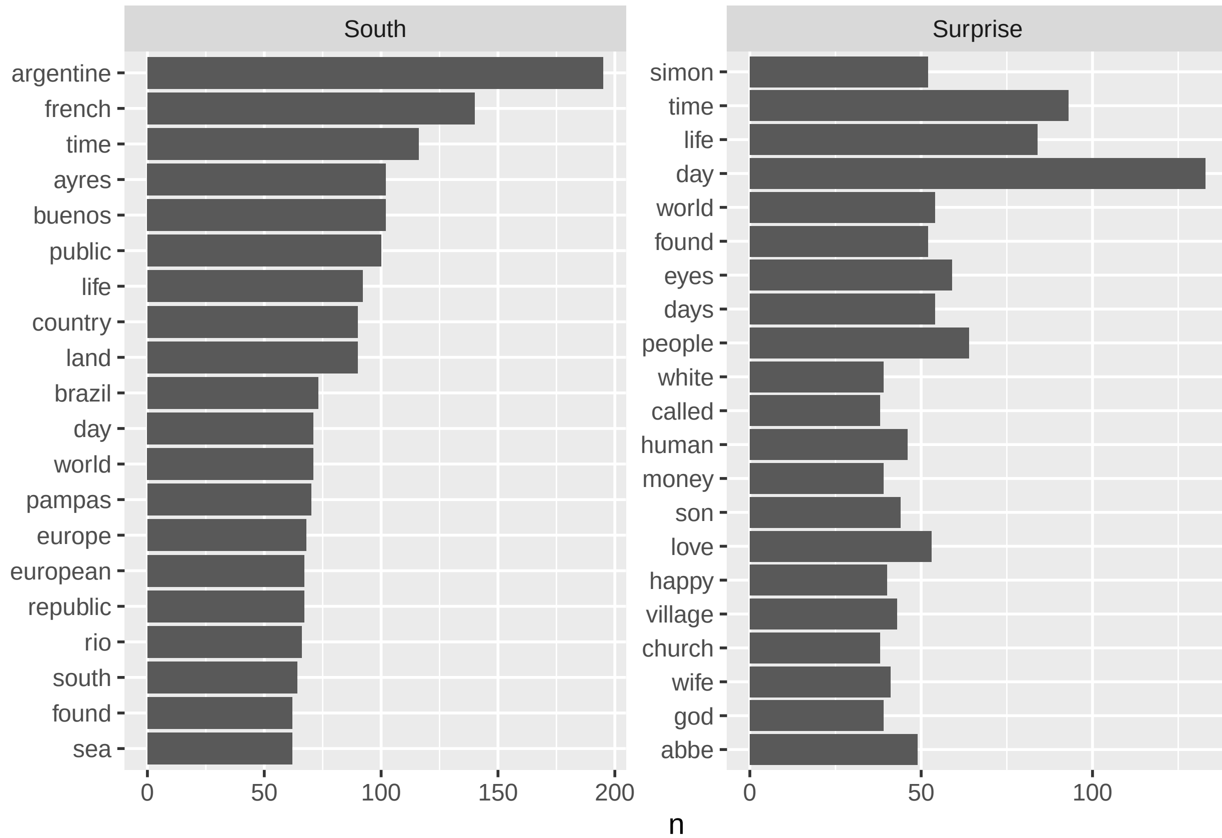
```
## Joining, by = "Work"
```

```r
freq_bybook <- words_by_book %>%
  group_by(Work) %>%
  mutate(rank = row_number(),
         'term frequency' = n/total)
```

From this, we see the most commonly used words in the entire corpus are time and day, then Argentine. Let's graph it to see how it looks.

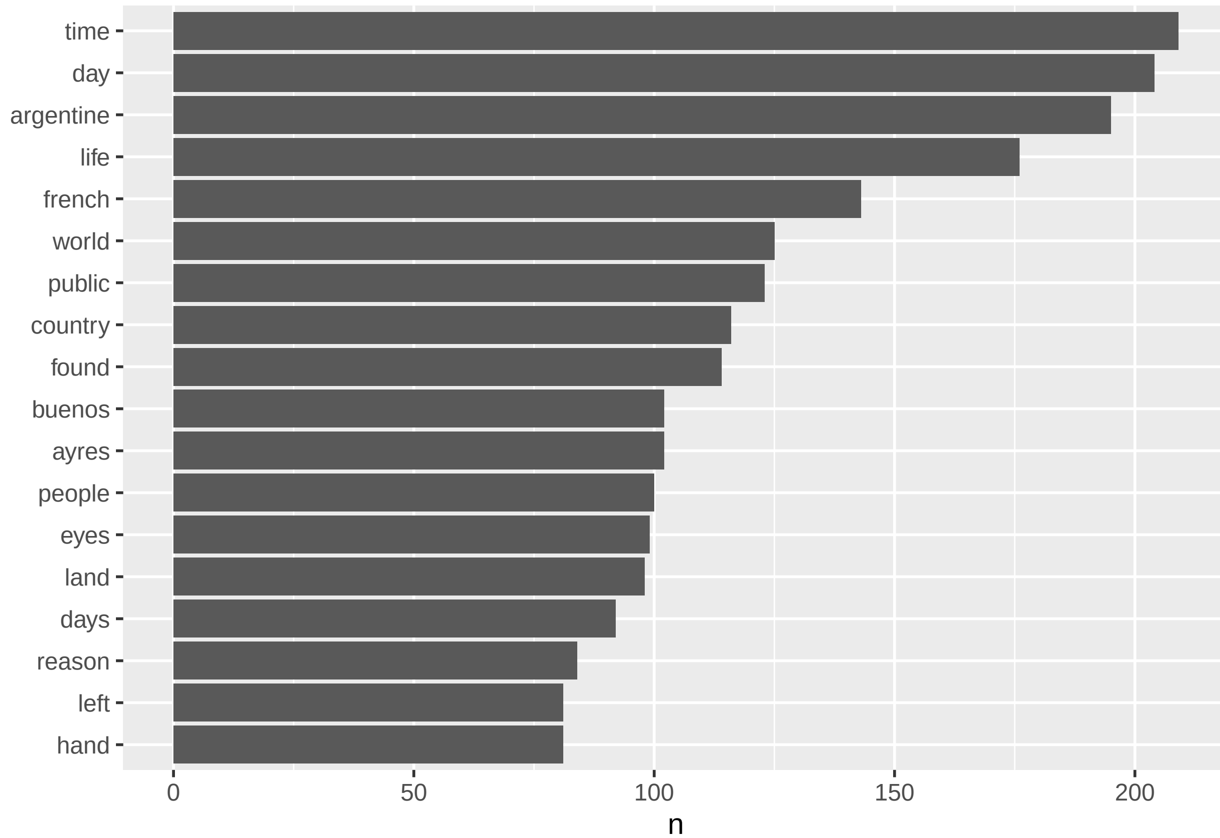
```r
#graphing counts by work - tf

freq_bybook %>%
  arrange(desc(n)) %>%
  mutate(word = factor (word, levels = rev(unique(word)))) %>%
  group_by(Work) %>%
  top_n(20) %>%
  ungroup %>%
  ggplot(aes(word, n, bill = Work)) +
  geom_col(show.legend = FALSE) +
  labs(x = NULL, y = "n") +
  facet_wrap(~Work, ncol = 2, scales = "free") +
  coord_flip()
```

```
## Selecting by term frequency
```



The graph above allows us to compare the two works.But what about the corpus overall?

```
tidy_clemenceau %>%
  count(word, sort = TRUE) %>% filter(n > 75) %>%
  mutate(word=reorder(word, n)) %>%
  ggplot(aes(word,n)) +
  geom_col() +
  xlab(NULL) + coord_flip()
```



And now we'll repeat the process with tf_idf (term frequency–inverse document frequency; aka how important a specific word is in the context of the text). This changes the weight of importance and uniqueness given to the word, and can yield different results than just using tf (term frequency; raw count of instances).

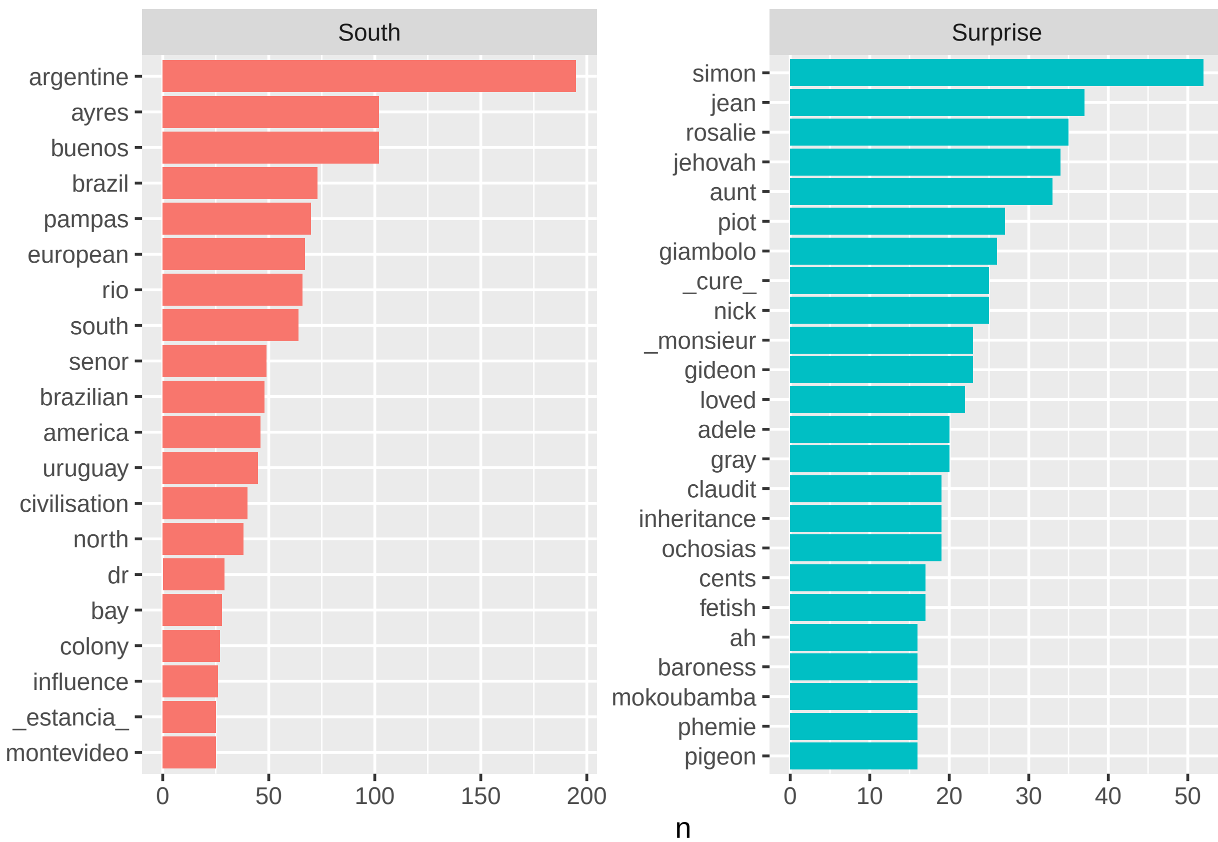
```
tidy_clemenceau %>% count(word, sort = TRUE)
```

```
## # A tibble: 12,963 x 2
##    word          n
##    <chr>     <int>
##  1 time        209
##  2 day         204
##  3 argentine   195
##  4 life        176
##  5 french      143
##  6 world       125
##  7 public      123
##  8 country     116
##  9 found       114
## 10 ayres       102
## # i 12,953 more rows
```

```
count_clemenceau_idf <- tidy_clemenceau %>%
  count(Work, word,sort = TRUE) %>%
  ungroup()
```

```
count_idf <- count_clemenceau_idf %>%
  bind_tf_idf(word, Work, n)

count_idf %>%
  arrange(desc(tf_idf)) %>%
  mutate(word = factor(word, levels = rev(unique(word)))) %>%
  group_by(Work) %>%
  top_n(20) %>%
  ungroup %>%
  ggplot(aes(word, n, fill = Work)) +
  geom_col(show.legend = FALSE) +
  labs(x = NULL, y = "n") +
  facet_wrap(~Work, ncol = 2, scales = "free") +
  coord_flip()
```

```
## Selecting by tf_idf
```



And it does! The rankings in South America To-Day are different, and now more country names are featured. Similarly, in Surprises of Life, there are many more character names that are counted.

#N-grams

The next step of the analysis is to investigate Clemenceau's most common phrases. For this analysis, I'll look at both bigrams (2 word phrases) and trigrams (3 word phrases). Since the Clemenceau work is composed of 2 books, it might be better to use tf_idf as this will take into account the frequency within the work.

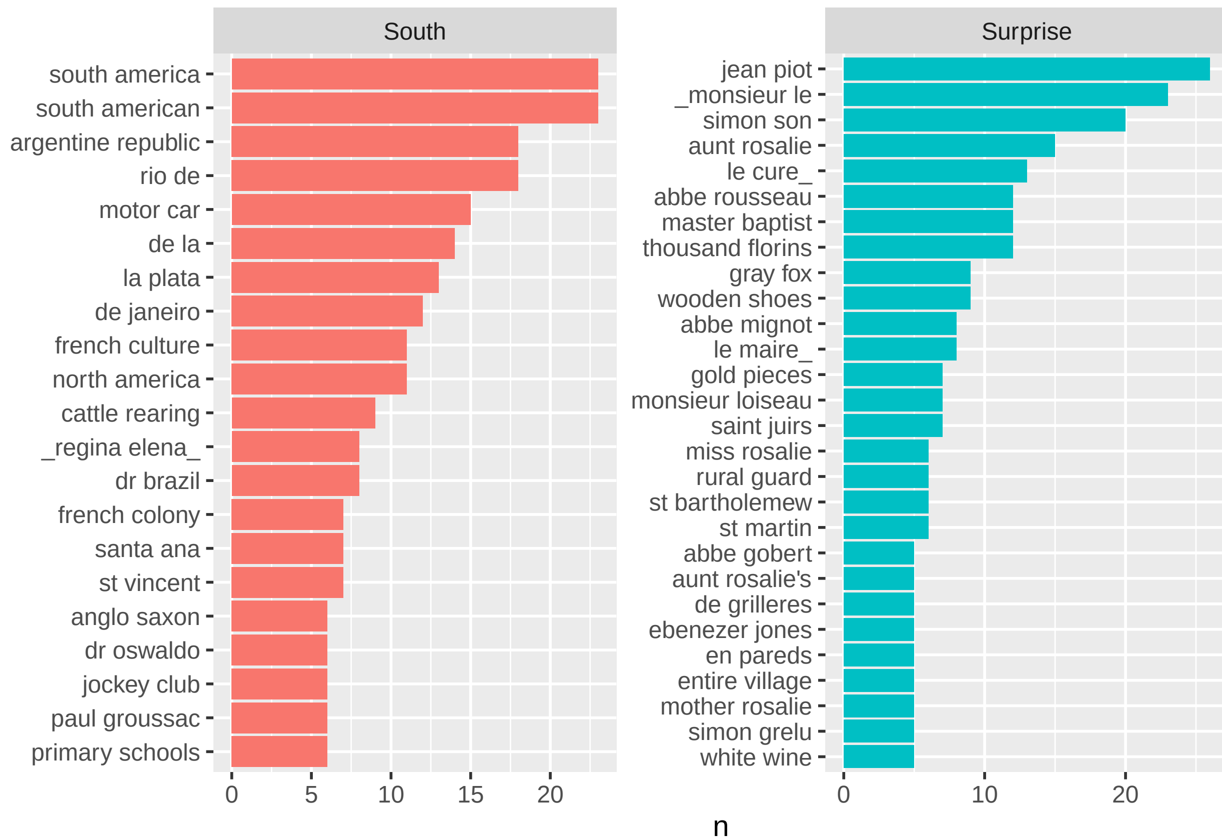
```r
#Bigrams

clem_bigrams <- clemenceau_corpus %>%
  unnest_tokens(bigram, text, token = "ngrams", n = 2)

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

```
## # A tibble: 73,267 x 2
##    bigram       n
##    <chr>    <int>
##  1 of the    1578
##  2 in the     825
##  3 to the     599
##  4 on the     339
##  5 it is      330
##  6 of a       314
##  7 for the    287
##  8 and the    281
##  9 by the     272
## 10 that the   271
## # i 73,257 more rows
```

```r
bigrams_separated <- clem_bigrams %>%
  separate(bigram, c("word1","word2"), sep = " ")

bigrams_filtered <- bigrams_separated %>%
  filter(!word1 %in% stop_words$word) %>%
  filter(!word2 %in% stop_words$word)

bigram_counts <- bigrams_filtered %>%
  count(Work, word1, word2, sort = TRUE)

bigram_counts = bigram_counts[-1:-4,]

#plotting bigrams

bigram_plot <- bigram_counts

bigram_plot$bigram <- paste(bigram_plot$word1, bigram_plot$word2, sep = " ")

bigram_plot_work <- bigram_plot %>%
  bind_tf_idf(bigram, Work, n)

bigram_plot_work %>%
  arrange(desc(tf_idf)) %>%
  mutate(bigram = factor (bigram, levels = rev(unique(bigram)))) %>%
  group_by(Work) %>%
  top_n(20) %>%
  ungroup %>%
  ggplot(aes(bigram, n, fill = Work)) +
  geom_col(show.legend = FALSE) +
  labs(x = NULL, y = "n") +
```

```
  facet_wrap(~Work, ncol = 2, scales = "free") +
  coord_flip()
```

```
## Selecting by tf_idf
```



The bigrams are a lot of words that go together; place names that are two words, peoples' full names, or terms like "primary school". And let's see what the tri-grams look like.

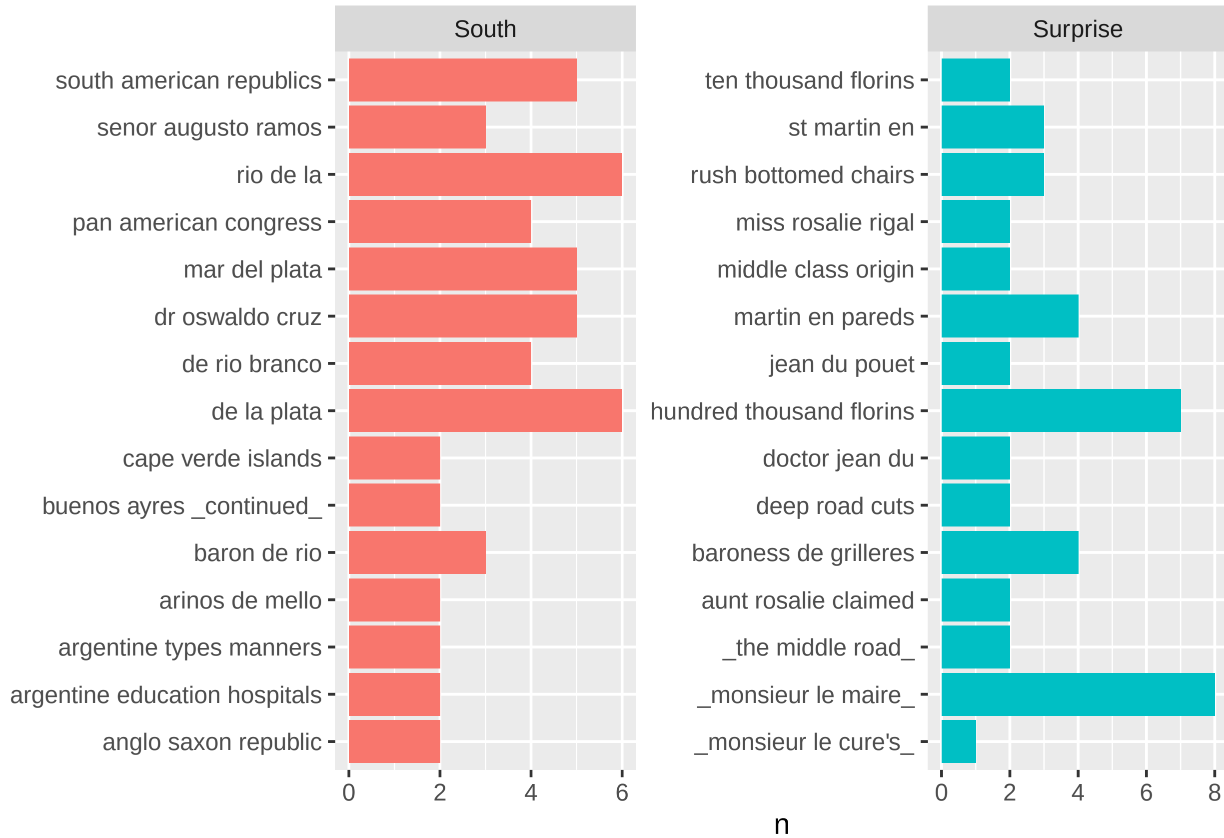
```
#Trigrams

clem_trigrams <- clemenceau_corpus %>%
  unnest_tokens(trigram, text, token = "ngrams", n = 3)

clem_trigrams %>%
  count(trigram, sort = TRUE)
```

```
## # A tibble: 105,617 x 2
##    trigram              n
##    <chr>            <int>
##  1 <NA>               322
##  2 of buenos ayres     40
##  3 one of the          39
##  4 it is not           31
##  5 in the argentine    30
```

```
##   6 there is no         30
##   7 part of the         29
##   8 more or less        27
##   9 of the argentine    27
##  10 i do not            26
## # i 105,607 more rows
```

```
trigrams_separated <- clem_trigrams %>%
  separate(trigram, c("word1","word2","word3"), sep = " ")

trigrams_filtered <- trigrams_separated %>%
  filter(!word1 %in% stop_words$word) %>%
  filter(!word2 %in% stop_words$word) %>%
  filter(!word3 %in% stop_words$word)

trigram_counts <- trigrams_filtered %>%
  count(Work, word1, word2, word3, sort = TRUE)

trigram_counts = trigram_counts[-1:-4,]

#plotting trigrams

trigram_plot <- trigram_counts

trigram_plot$trigram <- paste(trigram_plot$word1, trigram_plot$word2,
                              trigram_plot$word3, sep = " ")

trigram_plot_work <- trigram_plot %>%
  bind_tf_idf(trigram, Work, n)

trigram_plot_work %>%
  arrange(desc(tf_idf)) %>%
  mutate(bigram = factor (trigram, levels = rev(unique(trigram)))) %>%
  group_by(Work) %>%
  top_n(15) %>%
  ungroup %>%
  ggplot(aes(trigram, n, fill = Work)) +
  geom_col(show.legend = FALSE) +
  labs(x = NULL, y = "n") +
  facet_wrap(~Work, ncol = 2, scales = "free") +
  coord_flip()
```

```
## Selecting by bigram
```



There's a lot of names; more than there were before! If we want, we can add some of these names to the stopword dictionary and remove them from the analysis. I'll leave them for now. But now there's some more interesting phrases, like "rush bottomed chairs".

From the bigram and trigram analysis, we can see that Clemenceau talks a lot about people and about countries. He also talks to some lesser extent about large sums of money. This makes sense. Clemenceau was a prime minister, so he would be preoccupied with people and nations, and his dealings with them. South America To-Day is also a book focused on the countries of South America and their relationships, so it is not unthinkable that the names of leaders and countries, political bodies, would be the most frequent in that book. The Surprises of Life is a fiction; the most common words are character names.

should we add the names to the stop words and redo?

Now, we'll compare word usage in the two books to each other.

```
library(ggplot2)
library(scales)
```

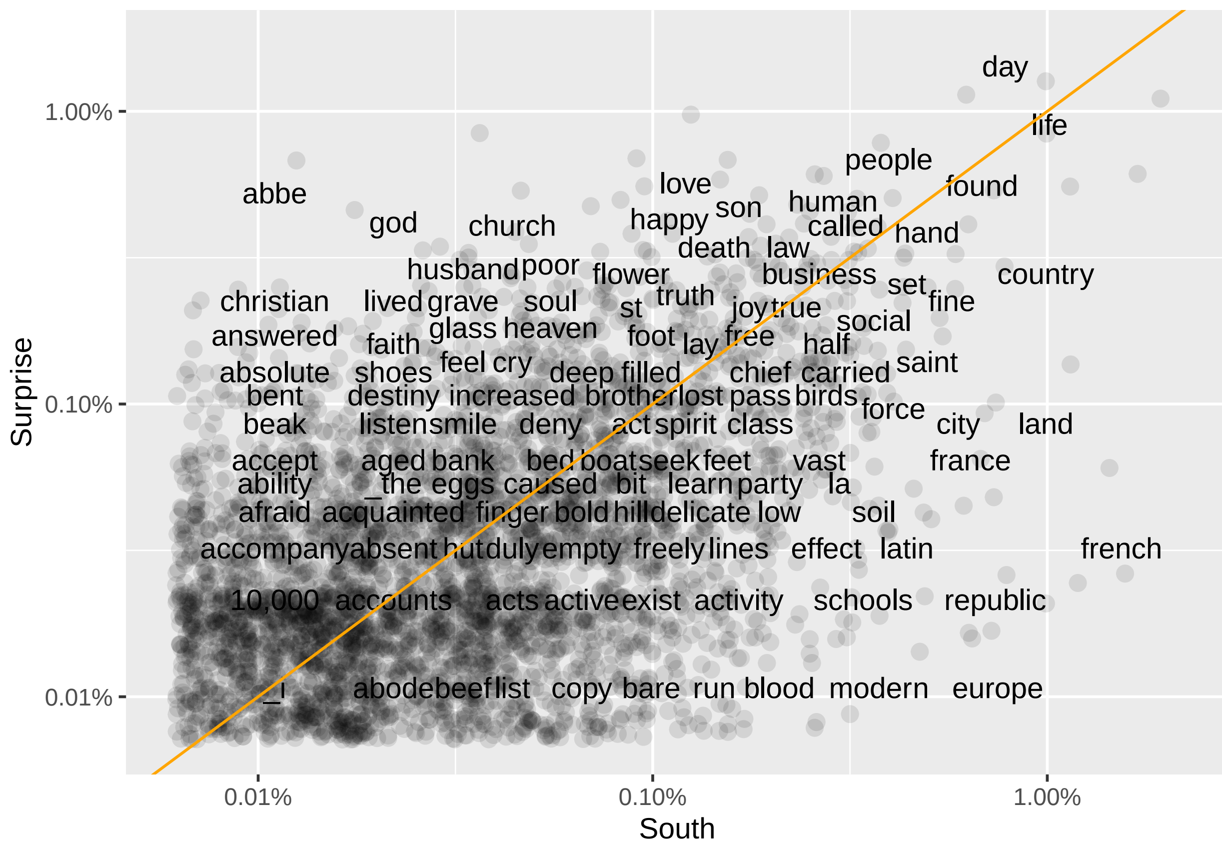
```
##
## Attaching package: 'scales'

## The following object is masked from 'package:purrr':
##
##     discard

## The following object is masked from 'package:readr':
##
##     col_factor
```

```r
tidy_clemenceau2 <- tidy_clemenceau %>%
  count(Work, word, sort = TRUE)

tidy_clemenceau3 <- tidy_clemenceau2 %>%
  bind_tf_idf(Work, word, n) %>%
  arrange(desc(tf_idf))

frequency <- tidy_clemenceau3 %>%
  group_by(Work) %>%
  left_join(tidy_clemenceau3 %>%
              group_by(Work) %>%
              summarise(total = n())) %>%
  mutate(freq = n/total)
```

```
## Joining, by = "Work"
```

```r
frequency <- frequency %>%
  select(Work, word, freq) %>%
  spread(Work, freq) %>%
  arrange(South, Surprise)


ggplot(frequency, aes(South, Surprise)) +
  geom_jitter(alpha = 0.1, size = 2.5, width = 0.25, height = 0.25) +
  geom_text(aes(label = word), check_overlap = TRUE, vjust = 1.5) +
  scale_x_log10(labels = percent_format()) +
  scale_y_log10(labels = percent_format()) +
  geom_abline(color = "Orange")
```

```
## Warning: Removed 8964 rows containing missing values (`geom_point()`).

## Warning: Removed 8964 rows containing missing values (`geom_text()`).
```




```
#https://stackoverflow.com/questions/53240576/ggplot-scaling-with-scalepercent-format-producing-strange
```

From this, we see that words like French, republic, europe, and modern are much more likely to be found in the South America To-Day work, while church, love, death, law, home, and others are more likely to be found in The Surprises of Life. This is in line with the purpose of both works. South America To-Day focuses on geopolitics. While as a fiction, The Surprises of Life is focused mostly on daily life and the quotidian.

```
#Comparing word usage

word_ratios <- tidy_clemenceau3 %>%
  ungroup() %>%
  arrange(desc(tf_idf)) %>%
  spread(Work, n, fill = 0) %>%
  mutate_if(is.numeric, funs((. +1) / sum(. + 1))) %>%
  mutate(logratio = log(South / Surprise)) %>%
  arrange(desc(logratio))
```

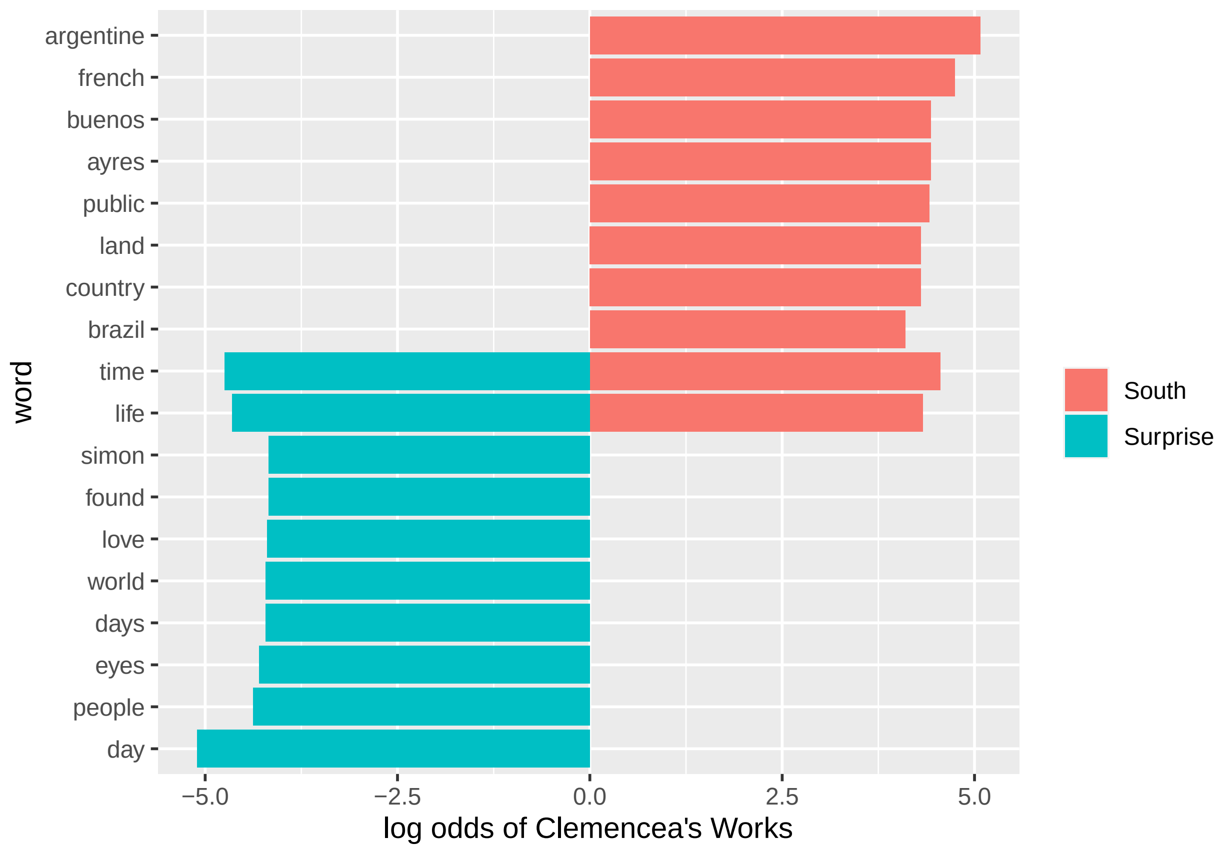
```
## Warning: `funs()` was deprecated in dplyr 0.8.0.
## i Please use a list of either functions or lambdas:
##
## # Simple named list: list(mean = mean, median = median)
##
## # Auto named with `tibble::lst()`: tibble::lst(mean, median)
##
## # Using lambdas list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))
```

```
word_ratios %>% arrange(abs(logratio))
```

```
## # A tibble: 16,962 x 7
##    word                   tf       idf    tf_idf     South  Surprise logratio
##    <chr>               <dbl>     <dbl>     <dbl>     <dbl>     <dbl>    <dbl>
##  1 _a              0.0000668 0.0000562 0.0000606 0.0000414 0.0000254    0.489
##  2 _ad             0.0000668 0.0000562 0.0000606 0.0000414 0.0000254    0.489
##  3 _annexe_        0.0000668 0.0000562 0.0000606 0.0000414 0.0000254    0.489
##  4 _apaches_       0.0000668 0.0000562 0.0000606 0.0000414 0.0000254    0.489
##  5 _argente_       0.0000668 0.0000562 0.0000606 0.0000414 0.0000254    0.489
##  6 _argentina_     0.0000668 0.0000562 0.0000606 0.0000414 0.0000254    0.489
##  7 _battues_       0.0000668 0.0000562 0.0000606 0.0000414 0.0000254    0.489
##  8 _black          0.0000668 0.0000562 0.0000606 0.0000414 0.0000254    0.489
##  9 _boggies_       0.0000668 0.0000562 0.0000606 0.0000414 0.0000254    0.489
## 10 _bongarconnisme_ 0.0000668 0.0000562 0.0000606 0.0000414 0.0000254    0.489
## # i 16,952 more rows
```

```
word_ratios %>% group_by(logratio < 0 ) %>%
  top_n(10, abs(logratio)) %>%
  ungroup() %>%
  mutate(word = reorder(word, logratio)) %>%
  ggplot(aes(word, logratio, fill = logratio < 0 )) +
  geom_col(sho.legend = FALSE) +
  coord_flip() +
  ylab("log odds of Clemencea's Works") +
  scale_fill_discrete(name = "", labels = c("South", "Surprise"))
```

```
## Warning in geom_col(sho.legend = FALSE): Ignoring unknown parameters:
## ‘sho.legend‘
```



Here, we see the relationship between the word used, and the probability that it came from one of the works. So, time and life have approximately equal chances of indicating either work. While day is more closely tied to The Surprises of Life and Argentine is more closely liked with South America To-Day.

#Sentiment Analysis

The next step in this analysis is a sentiment analysis. From this, we might be able to tell how Clemenceau is feeling in his writing, and his opinions. This is based on his use of "happy" vs "sad" words, or "positive" vs "negative" words.

```
#unnest tokens
full_sentiment <- clemenceau_corpus %>% unnest_tokens(word, text)

#remove stop words
tidy_full_s <- full_sentiment %>% anti_join(stop_words)
```

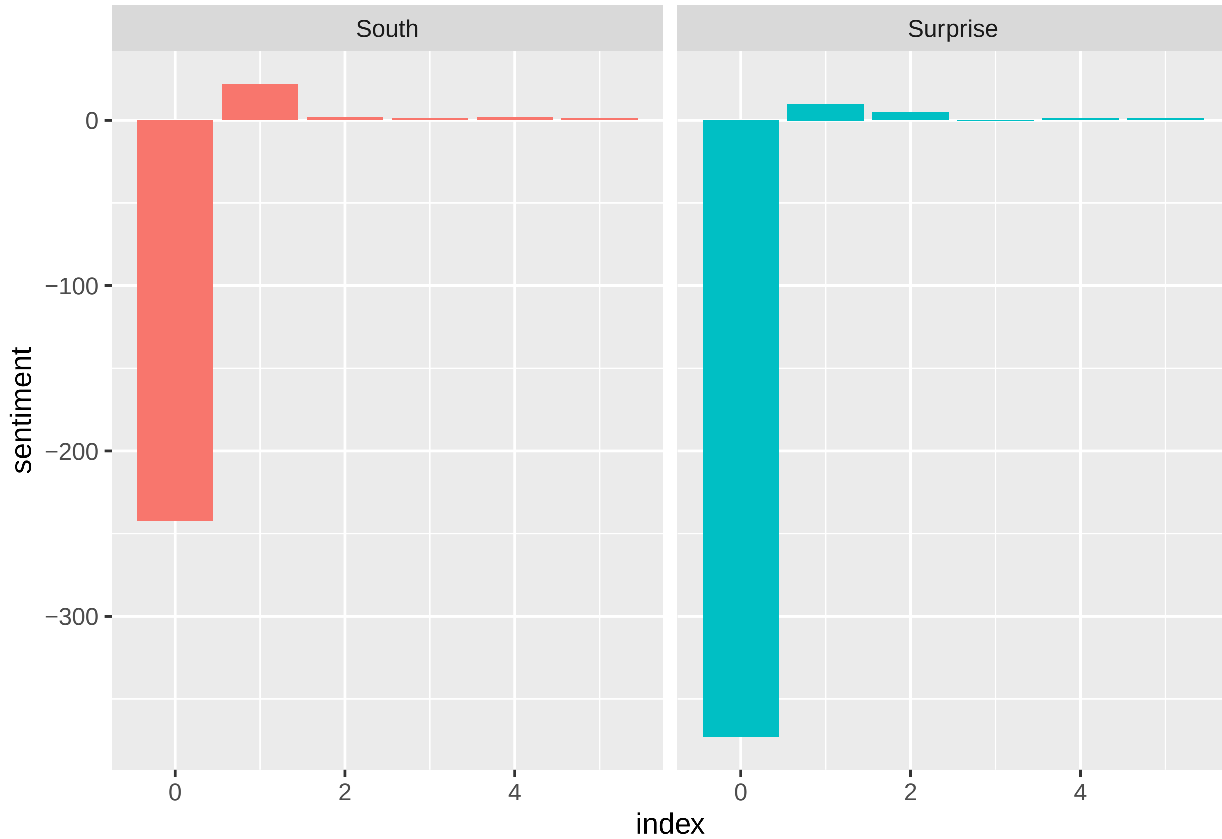
```
## Joining, by = "word"
```

```
#Prepare - By Work
tidy_full_s %>%
  count(word, sort = TRUE)
```

```
## # A tibble: 12,963 x 2
##    word          n
##    <chr>     <int>
##  1 time        209
```

```
##   2 day          204
##   3 argentine    195
##   4 life         176
##   5 french       143
##   6 world        125
##   7 public       123
##   8 country      116
##   9 found        114
##  10 ayres        102
## # i 12,953 more rows
```

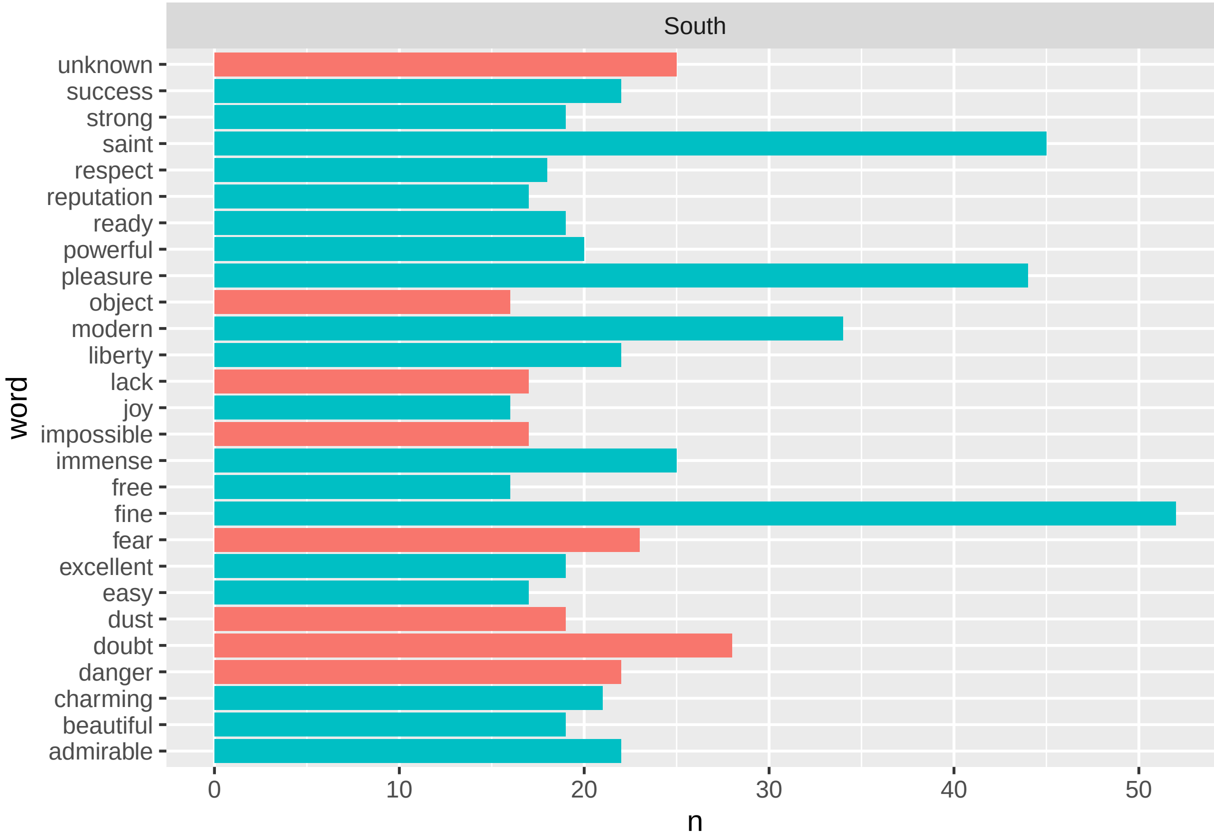
```
word_by_work <- tidy_full_s %>%
  count(Work, word, sort = TRUE) %>%
  ungroup()


#finding tf-idf within Work

tf_idf_work <- word_by_work %>%
  bind_tf_idf(word, Work, n) %>%
  arrange(desc(tf_idf))

#Sentiment Analysis by Work

work_sentiments <- tf_idf_work %>%
  inner_join(get_sentiments("bing"), by = "word") %>%
  count(Work, index = n %/% 10, sentiment) %>%
  spread(sentiment, n, fill = 0) %>%
  mutate(sentiment = positive - negative)

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



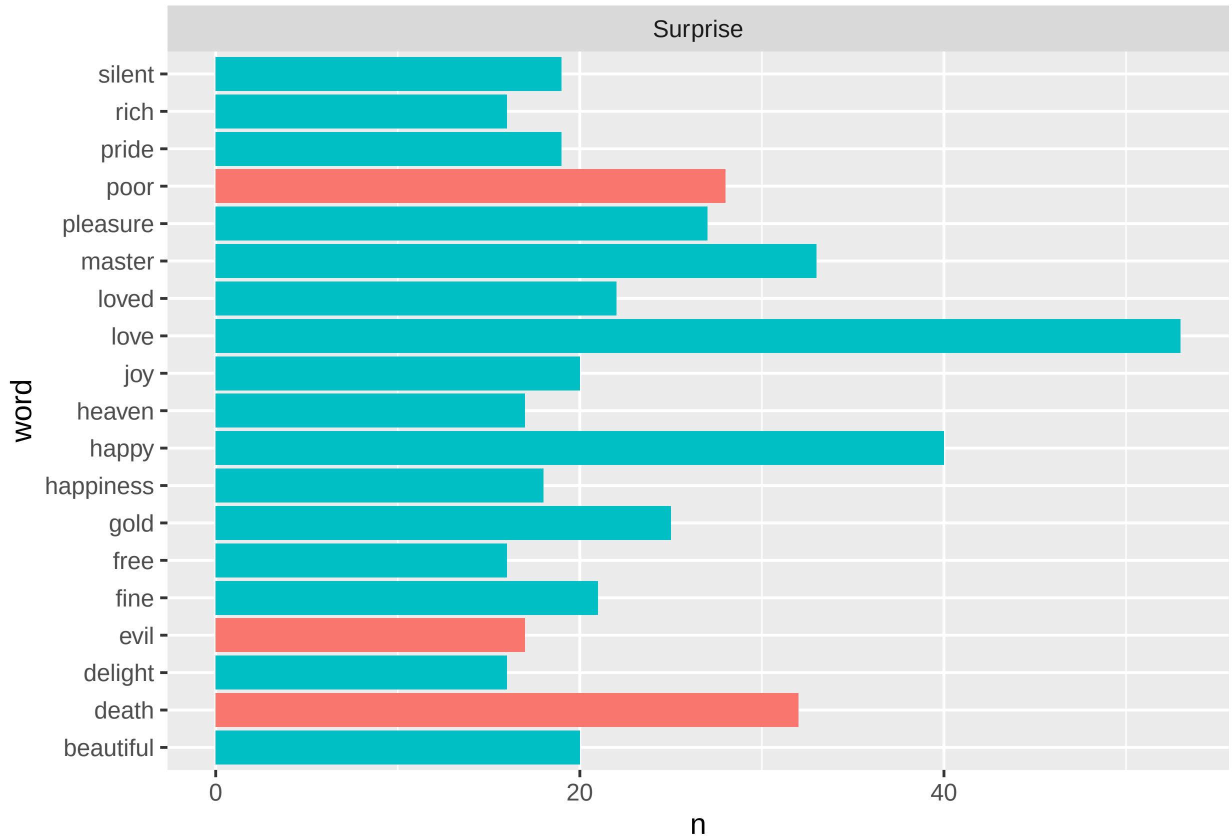

```r
#Sentiment words per work

work_sentiments2 <- word_by_work %>%
  inner_join(get_sentiments("bing"), by = "word") %>%
  count(word, Work, n , sentiment)
```

```
## Storing counts in ‘nn‘, as ‘n‘ already present in input
## i Use ‘name = "new_name"‘ to pick a new name.
```

```r
work_filtered_south <- filter(work_sentiments2, Work == 'South')

work_filtered_south %>%
  group_by(sentiment) %>%
  filter(n > 15) %>%
  ungroup() %>%
  select(-nn) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs( y = "n") +
  coord_flip() +
  facet_wrap(~ Work, ncol = 4, scales = "free_x")
```




```
work_filtered_surprise <- filter(work_sentiments2, Work == 'Surprise')

work_filtered_surprise %>%
  group_by(sentiment) %>%
  filter(n > 15) %>%
  ungroup() %>%
  select(-nn) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs( y = "n") +
  coord_flip() +
  facet_wrap(~ Work, ncol = 4, scales = "free_x")
```

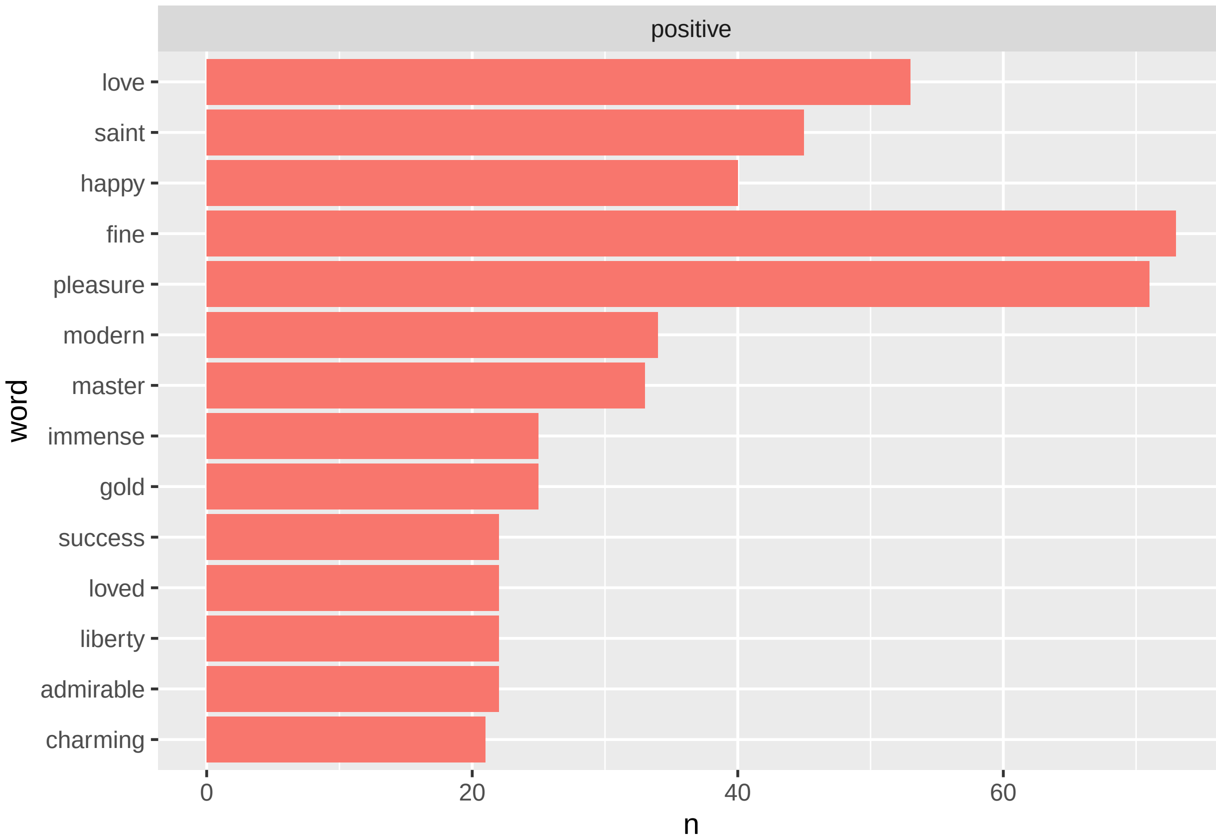



```
#sentiment - no work
#I removed the tf_idf because it made the word counts too small

just_words <- count_clemenceau %>% select(-c(Work))

word_sentiments <- just_words %>%
  inner_join(get_sentiments("bing")) %>%
  ungroup()
```

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

```
word_sentiments <- word_sentiments %>% filter(n > 20)

word_sentiments %>%
  group_by(sentiment) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  top_n(10) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs( y = "n") +
  coord_flip()
```

```
## Selecting by sentiment
```



Both books, have a good number of sentiments words. Likewise, in both works, most of the words are positive in nature, such as "fine", "saint", "pleasure", "love", and "happy". However, there are also some sad words such as "death", "doubt", and "danger". It is also worth noting that South America To-Day has a good number more "sentiment" words in use than The Surprises of Life. This may be unexpected given the nature of the works. Perhaps this indicates that Clemenceau took a more clinical and neutral approach to his description of the surprises, or perhaps that he is particularly optimistic about the situation in South American. The most commonly used words include "fine", "pleasure", and "love". In both works, it would seem Clemenceau looks fondly upon what he is writing.

#Sentiment Analysis with Bigrams - pg 48

Next, we will do another round of sentiment analysis, but this time, focusing on bigrams. By looking at which words are commonly paired with negations, we can see what Clemenceau thinks negatively about. This helps bring more context to the sentiment analysis.

```
#setting up the data
library(textdata)
```

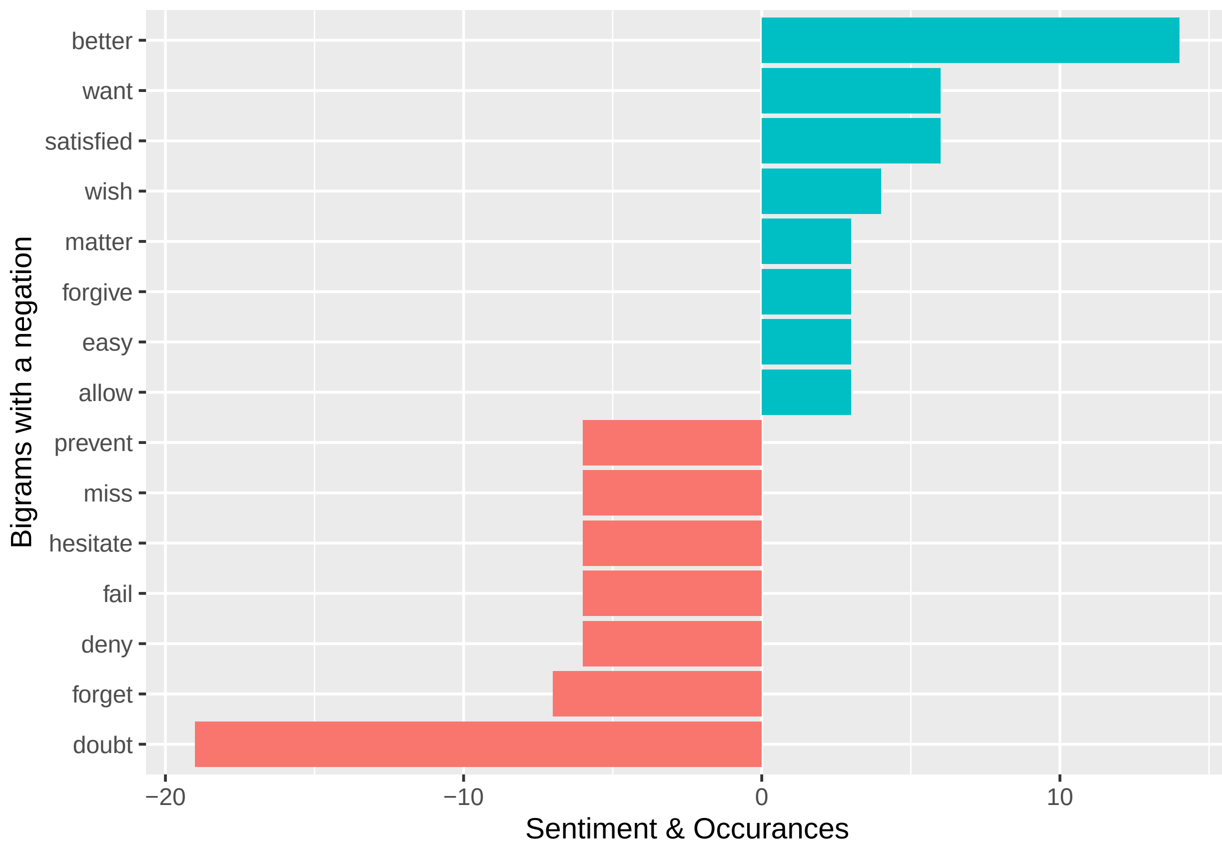
```
## Warning: package 'textdata' was built under R version 4.2.3
```

```
#sentiment dictionary
sent_afinn <- get_sentiments("afinn")
negations <- c("not", "no", "never", "without", "neither",
               "nothing", "nobody", "nowhere", "none")

bigram_sent <- bigrams_separated %>%
  filter(word1 %in% negations) %>%
```

```r
  inner_join(sent_afinn, by = c(word2 = "word")) %>%
  count(word1, word2, value, sort = TRUE) %>%
  ungroup()

bigram_sent
```

```
## # A tibble: 157 x 4
##    word1 word2   value     n
##    <chr> <chr>   <dbl> <int>
##  1 no    doubt      -1    19
##  2 no    better      2     7
##  3 not   forget     -1     7
##  4 not   prevent    -1     6
##  5 not   want        1     6
##  6 not   wish        1     4
##  7 never forgive     1     3
##  8 never miss       -2     3
##  9 no    matter      1     3
## 10 not   allow       1     3
## # i 147 more rows
```

```r
#graphing bigrams
bigram_sent %>%
  mutate(contribution = n * value) %>%
  head(15) %>%
  arrange(desc(abs(contribution))) %>%
  mutate(word2 = reorder(word2, contribution)) %>%
  ggplot(aes(word2, n*value, fill = n * value > 0)) +
  geom_col(show.legend = FALSE) +
  xlab("Bigrams with a negation") +
  ylab("Sentiment & Occurances") +
  coord_flip()
```



The first step was to see which bigrams contain a negation word. In the table, most of these words are metaphysical: doubt, forget, prevent, want, wish, forgive, etc. Bigrams featuring a more concrete second word, such as slavery, poison, or violence occur very infrequently.

Clemenceau does not condemn or lament very much in his writing. Moreso, he is speculating, he doesn't doubt, he doesn't wish, he doesn't think something or someone will prevent or forgive something.

#Topic Modeling? - 89

The next phase in this analysis is topic modeling. This will give us a better indication of what Clemenceau is talking about in each discrete part of his writing. Topic Modeling uses the most frequent words to match topics to different texts (documents) to see how documents differ in their focus. We only have 2 works here. So, instead, we will break the works into chapters. The topics with the most chapters dedicated will indicate Clemenceau's focus.

```
#remove front matter
surprises_topic <- surprises_two[-c(1:85),]

#add chapter
surprises_topic[6093, 2] = "Chapter 25"
surprises_topic[5832, 2] = "Chapter 24"
surprises_topic[5545, 2] = "Chapter 23"
surprises_topic[5302, 2] = "Chapter 22"
surprises_topic[5066, 2] = "Chapter 21"
surprises_topic[4750, 2] = "Chapter 20"
surprises_topic[4432, 2] = "Chapter 19"
surprises_topic[4213, 2] = "Chapter 18"
surprises_topic[4011, 2] = "Chapter 17"
```

```r
surprises_topic[3785, 2] = "Chapter 16"
surprises_topic[3573, 2] = "Chapter 15"
surprises_topic[3351, 2] = "Chapter 14"
surprises_topic[3118, 2] = "Chapter 13"
surprises_topic[2894, 2] = "Chapter 12"
surprises_topic[2650, 2] = "Chapter 11"
surprises_topic[2411, 2] = "Chapter 10"
surprises_topic[2182, 2] = "Chapter 9"
surprises_topic[1950, 2] = "Chapter 8"
surprises_topic[1661, 2] = "Chapter 7"
surprises_topic[1376, 2] = "Chapter 6"
surprises_topic[1135, 2] = "Chapter 5"
surprises_topic[807, 2] = "Chapter 4"
surprises_topic[551, 2] = "Chapter 3"
surprises_topic[312, 2] = "Chapter 2"
surprises_topic[1, 2] = "Chapter 1"


surprises_topic <- surprises_topic[!surprises_topic$text == "", ]
surprises_topic <- surprises_topic %>% select(-c(gutenberg_id))
```

Surprises of life is actually a collection of short stories, so it didn't have chapters. As such, I added the word "chapter" at the beginning of every new story so that the function would be able to divide it into documents in the same way it divides South America To-Day, which does have chapters.

```r
library(topicmodels)
```

```
## Warning: package 'topicmodels' was built under R version 4.2.3
```

```r
set.seed(1234)

surprises_topic$Work <- "Surprise"
clemenceau_corpus2 <- rbind(south_clean, surprises_topic)

#divide into documents - one document per chapter
reg <- regex("^chapter", ignore_case = TRUE)
by_chapter <- clemenceau_corpus2 %>%
  group_by(Work) %>%
  mutate(chapter = cumsum(str_detect(text, reg))) %>%
  ungroup() %>%
  filter(chapter > 0 ) %>%
  unite(document, Work, chapter)

#split into words
word_chapter <- by_chapter %>%
  unnest_tokens(word, text)

#find document-word counts
word_counts <- word_chapter %>%
  anti_join(stop_words) %>%
  count(document, word, sort = TRUE) %>%
  ungroup
```

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

```
word_counts
```

```
## # A tibble: 37,971 x 3
##    document    word          n
##    <chr>       <chr>     <int>
##  1 Surprise_6  simon        45
##  2 South_4     argentine    39
##  3 South_14    coffee       35
##  4 Surprise_19 jean         32
##  5 South_11    uruguay      30
##  6 Surprise_4  aunt         30
##  7 South_4     french       28
##  8 Surprise_4  rosalie      28
##  9 South_13    french       27
## 10 South_13    rio          27
## # i 37,961 more rows
```

Each chapter (document) is assigned a "topic" based on the most frequet words. A few of the most common topics in South America To-Day are Uraguay, Argentina, and Coffee. The topics of Surprises of Life are once again the characters in the short stories.

Now that we have the topics, we can see if it's possible to use those topics for predictive methods. One way to do this, is to LDA (latent Dirichlet allocation). This is a Bayesian model for topics that classifies them as belonging to one text or another.

```
#LDA on Chapters

chapters_dtm <- word_counts %>%
  cast_dtm(document, word, n)

chapters_dtm
```

```
## <<DocumentTermMatrix (documents: 39, terms: 12887)>>
## Non-/sparse entries: 37971/464622
## Sparsity           : 92%
## Maximal term length: 17
## Weighting          : term frequency (tf)
```

```
clem_lda <- LDA(chapters_dtm, k= 2, control = list(seed = 1234))
clem_lda
```

```
## A LDA_VEM topic model with 2 topics.
```

```
chapter_topics <- tidy(clem_lda, matrix = "beta")
chapter_topics
```

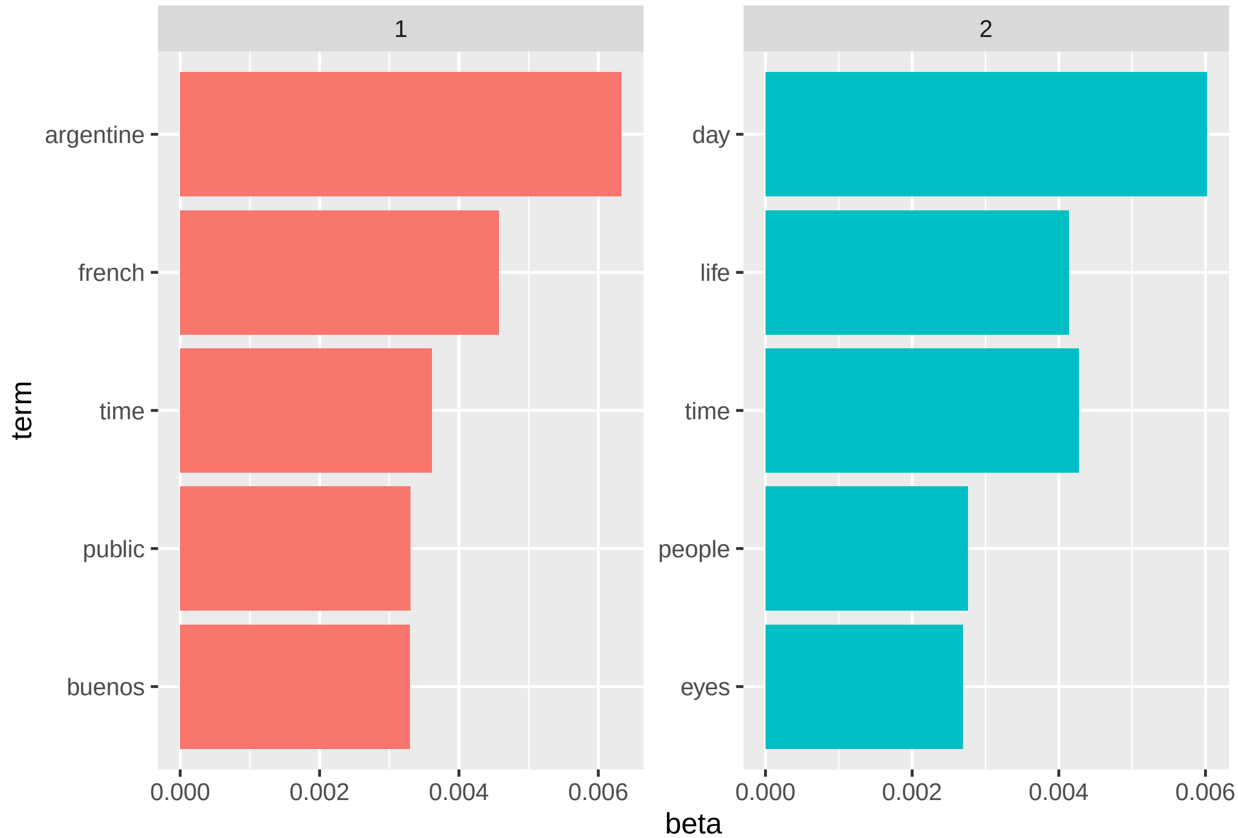
```
## # A tibble: 25,774 x 3
##    topic term          beta
##    <int> <chr>        <dbl>
```

```
## 1        1 simon     1.13e- 3
## 2        2 simon     7.76e- 4
## 3        1 argentine 6.33e- 3
## 4        2 argentine 1.39e- 7
## 5        1 coffee    1.48e- 3
## 6        2 coffee    1.32e- 4
## 7        1 jean      2.19e-17
## 8        2 jean      1.62e- 3
## 9        1 uruguay   1.45e- 3
## 10       2 uruguay   4.47e-69
## # i 25,764 more rows
```

```r
top_terms <- chapter_topics %>%
  group_by(topic) %>%
  top_n(5, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)

top_terms
```

```
## # A tibble: 10 x 3
##    topic term         beta
##    <int> <chr>       <dbl>
## 1      1 argentine 0.00633
## 2      1 french    0.00457
## 3      1 time      0.00361
## 4      1 public    0.00330
## 5      1 buenos    0.00330
## 6      2 day       0.00602
## 7      2 time      0.00427
## 8      2 life      0.00414
## 9      2 people    0.00276
## 10     2 eyes      0.00269
```

```r
top_terms %>%
  mutate(term = reorder(term, beta)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip()
```



Unsurprisingly, argentine, french, time, public and buenos indicate South America Today while day, life, time, people, eyes indicate Surprises of Life. It is interesting to note that Clemenceau talks at great length about Argentina, as opposed to the other countries in South and Central America. And also, that he talks often about time. Time is a common thread in much of Clemenceau's writing, it seems.

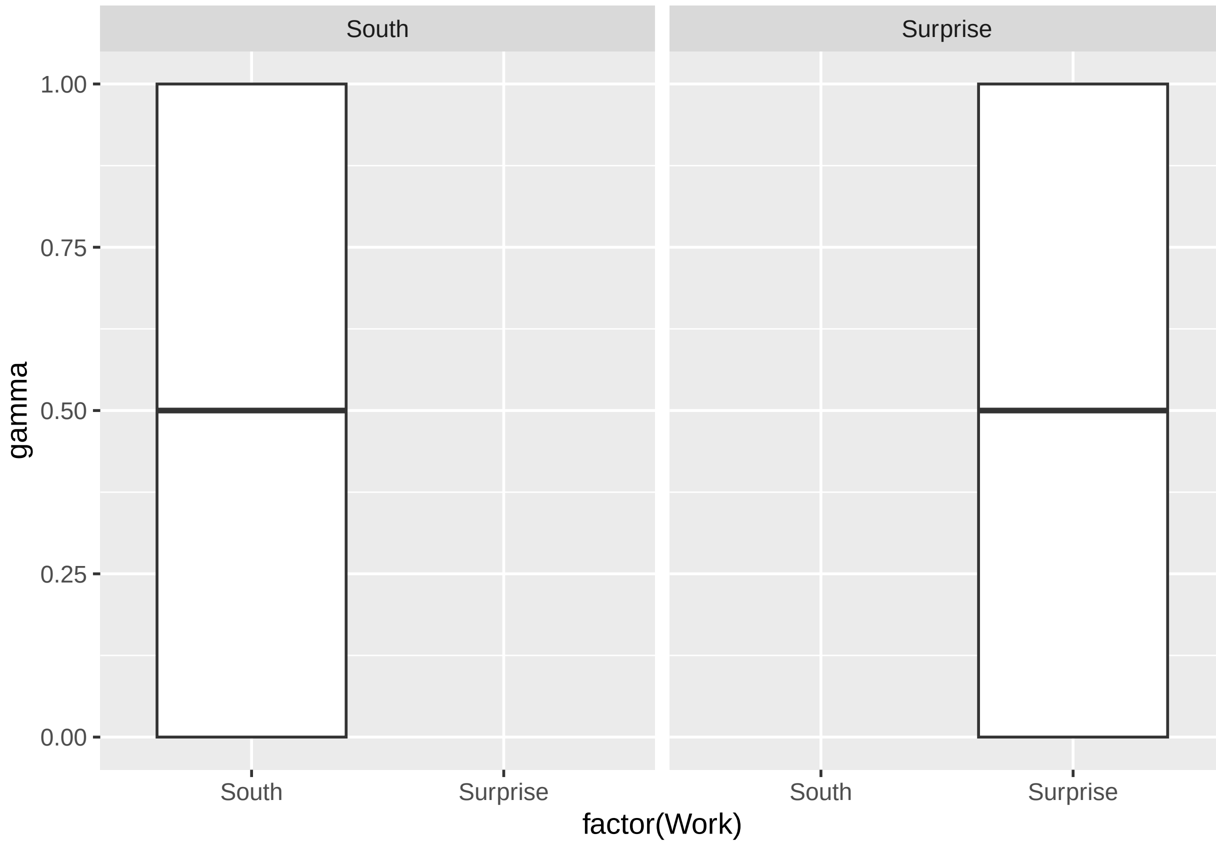
#Per-Document Classification

Next we will see if lda could be used to classify individual documents back into their original works.

```
chapters_gamma <- tidy(clem_lda, matrix = "gamma")
chapters_gamma
```

```
## # A tibble: 78 x 3
##    document    topic     gamma
##    <chr>       <int>     <dbl>
##  1 Surprise_6      1 0.590
##  2 South_4         1 1.00
##  3 South_14        1 1.00
##  4 Surprise_19     1 0.0000367
##  5 South_11        1 1.00
##  6 Surprise_4      1 0.581
##  7 South_13        1 1.00
##  8 South_7         1 1.00
##  9 Surprise_25     1 0.0000354
## 10 South_8         1 0.339
## # i 68 more rows
```

```
chapters_gamma <- chapters_gamma %>%
  separate(document, c("Work", "chapter"), sep = "_", convert = TRUE)

chapters_gamma %>%
  mutate(Work = reorder(Work, gamma * topic)) %>%
  ggplot(aes(factor(Work), gamma)) +
  geom_boxplot() +
  facet_wrap( ~Work)
```



The classifier looks at the topic for each chapter, and then assigns it a probability of being in one work vs the other based on that topic. As we saw previously, topics such as country names were often in South American Today, and topics such as character names were often in Surprises of Life.

Clearly, it is very good at classifiying the documents between South America To-Day and The Surprises of Life. This also shows that both works have multiple topics and cannot be classified based on one topic alone, but that the topics are distinct.

```
chapter_classifications <- chapters_gamma %>%
  group_by(Work, chapter) %>%
  top_n(1, gamma) %>%
  ungroup()

book_topics <- chapter_classifications %>%
  count(Work, topic) %>%
  group_by(Work) %>%
  top_n(1, n) %>%
```

```r
  ungroup() %>%
  transmute(consensus = Work, topic)

chapter_classifications %>%
  inner_join(book_topics, by = "topic") %>%
  filter(Work != consensus)
```

```
## # A tibble: 6 x 5
##   Work     chapter topic gamma consensus
##   <chr>      <int> <int> <dbl> <chr>
## 1 Surprise       6     1 0.590 South
## 2 Surprise       4     1 0.581 South
## 3 Surprise       5     1 0.617 South
## 4 Surprise      12     1 0.540 South
## 5 South          8     2 0.661 Surprise
## 6 South          9     2 0.729 Surprise
```

These are the only chapters that were mis-classified. There were 6 total chapters; four belonged to Surprises of Life, and two to South America To-Day. This was due to some cross-over in the titles assigned. Thinking back to the previous analysis of Clemenceau's most used words and phrases, words like "time" and "life" showed up frequently in both works. A similar situation could be causing this misassignment.

#Bigram network

An interesting aspect of bigrams, and other n-grams, is the frequency with which words are paired together. This can form a sort of network connecting different terms.

```r
library(igraph)
```

```
## Warning: package 'igraph' was built under R version 4.2.3

##
## Attaching package: 'igraph'

## The following objects are masked from 'package:purrr':
##
##     compose, simplify

## The following object is masked from 'package:tibble':
##
##     as_data_frame

## The following objects are masked from 'package:dplyr':
##
##     as_data_frame, groups, union

## The following object is masked from 'package:tidyr':
##
##     crossing

## The following objects are masked from 'package:stats':
##
##     decompose, spectrum
```

```
## The following object is masked from 'package:base':
##
##     union
```

```r
bigram_counts
```

```
## # A tibble: 12,244 x 4
##    Work     word1     word2        n
##    <chr>    <chr>     <chr>    <int>
##  1 Surprise jean      piot        26
##  2 South    south     america     23
##  3 South    south     american    23
##  4 Surprise _monsieur le          23
##  5 Surprise simon     son         20
##  6 South    argentine republic    18
##  7 South    rio       de          18
##  8 South    motor     car         15
##  9 Surprise aunt      rosalie     15
## 10 South    de        la          14
## # i 12,234 more rows
```

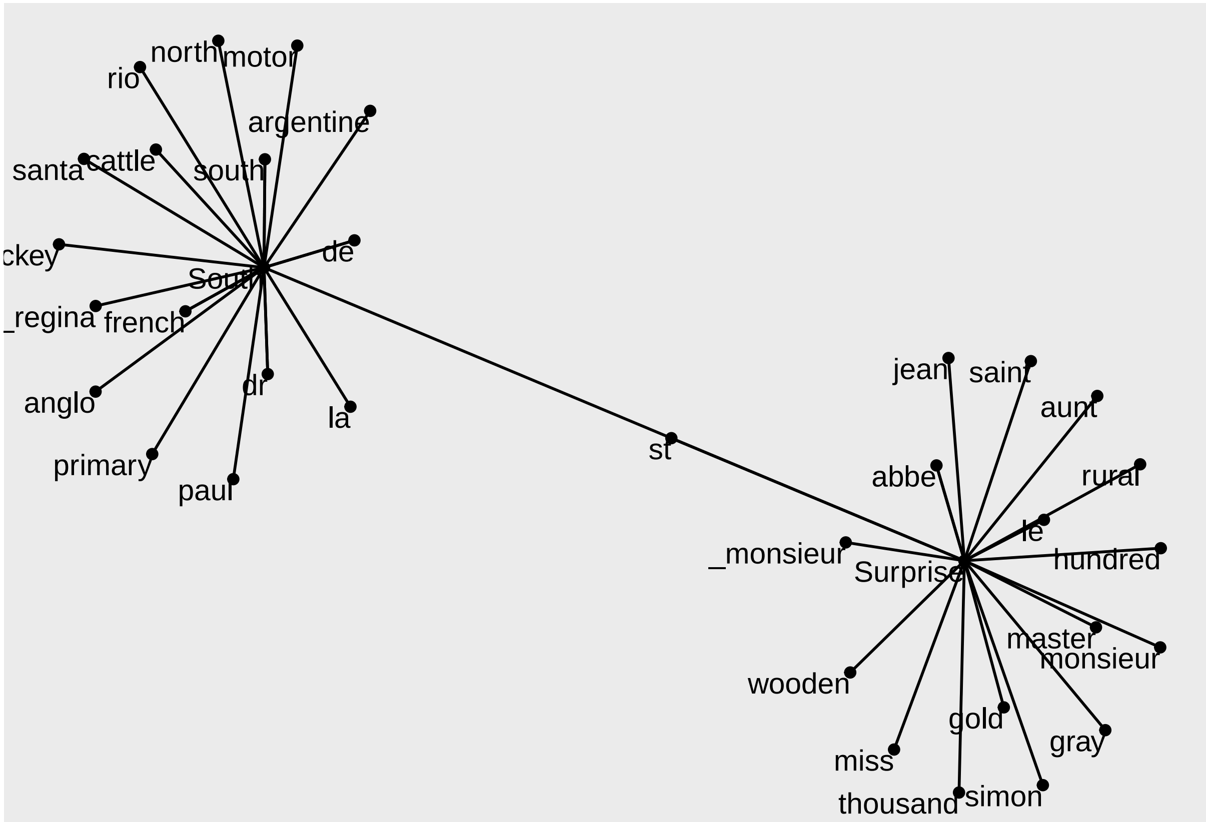
```r
bigram_graph <- bigram_counts %>%
  filter(n > 5) %>%
  graph_from_data_frame()

bigram_graph
```

```
## IGRAPH 55e584a DN-- 35 41 --
## + attr: name (v/c), word2 (e/c), n (e/n)
## + edges from 55e584a (vertex names):
##  [1] Surprise->jean      South   ->south     South   ->south
##  [4] Surprise->_monsieur Surprise->simon     South   ->argentine
##  [7] South   ->rio       South   ->motor     Surprise->aunt
## [10] South   ->de        South   ->la        Surprise->le
## [13] South   ->de        Surprise->abbe      Surprise->master
## [16] Surprise->thousand  South   ->french    South   ->north
## [19] South   ->cattle    Surprise->gray      Surprise->hundred
## [22] Surprise->wooden    South   ->_regina   South   ->dr
## + ... omitted several edges
```

Here we see that phrases such as "motor car" or "south american" can be found in the frequent bigrams where the singular term "motor" often appears next to "car". Similarly, argentine is often paired with republic. We also see that in Surprises of Life, consistent with previous sections of the analysis, aunt is often paired with rosalie, as this is the full character's name.

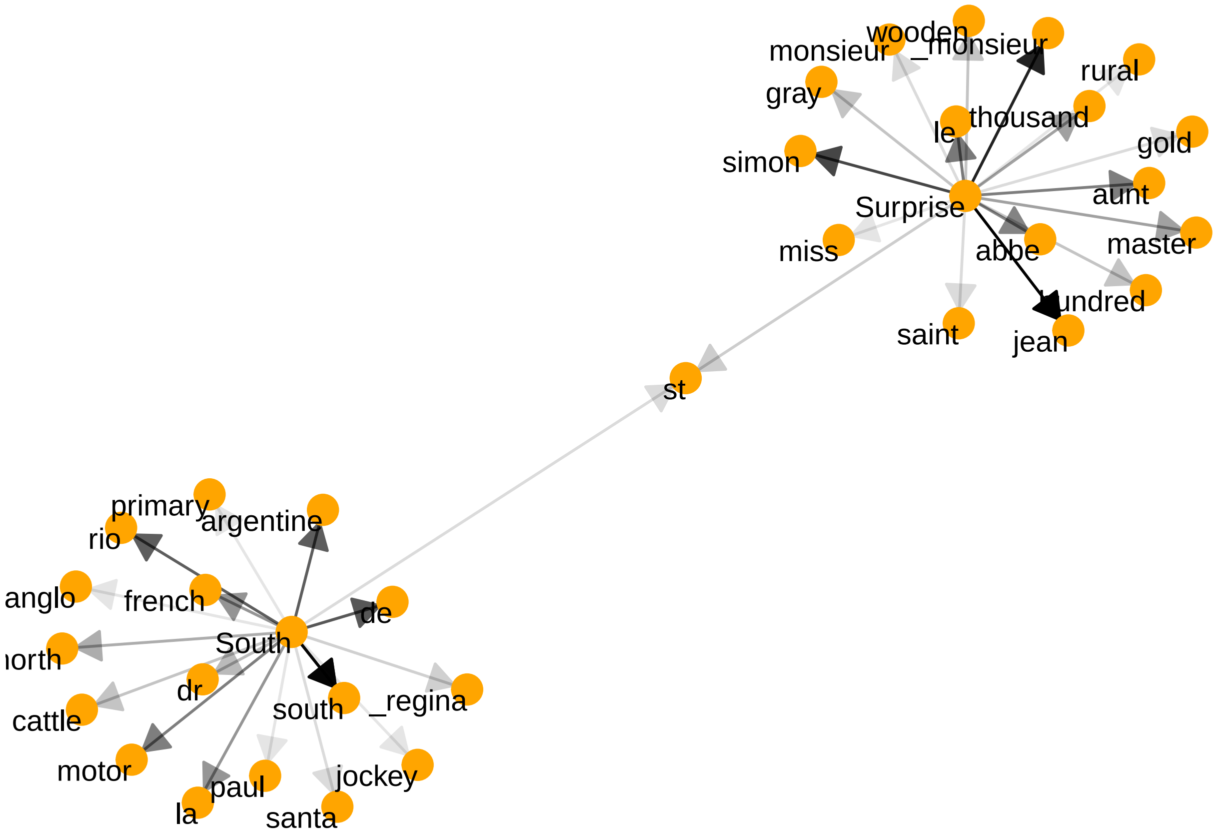
```r
library(ggraph)

ggraph(bigram_graph, layout = "fr") +
  geom_edge_link() +
  geom_node_point() +
  geom_node_text(aes(label = name), vjust = 1, hjust = 1 )
```

```
## Warning: Using the `size` aesthetic in this geom was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` in the `default_aes` field and elsewhere instead.
```



Here we make a web of the words. Surprises of Life is often peppered with nominal terms: miss, master, monsieur, but also descriptive words such as wooden and gray. Meanwhile, South America Today has terms for nationalities, and also trade goods such as cattle. "st" is the word that links both together, often appearing as part of a name, most likely.

```r
a <- grid::arrow(type = "closed", length = unit(.15, "inches"))
ggraph(bigram_graph, layout = "fr") +
  geom_edge_link(aes(edge_alpha = n), show.legend = FALSE, arrow = a,
                 end_cap = circle(.07, "inches")) +
  geom_node_point(color = "orange", size = 5) +
  geom_node_text(aes(label = name), vjust = 1,hjust = 1) +
  theme_void()
```



Here is the word web again. In this graph, darker lines indicate a higher frequency. Similarly to topic modeling, this bigram network can be used for prediction or classification. But here, we are just interested to know what Clemenceau is talking about, and bigrams help give context to the high frequency words. These clusters exist because the words are more likely to appear together in a bigram.

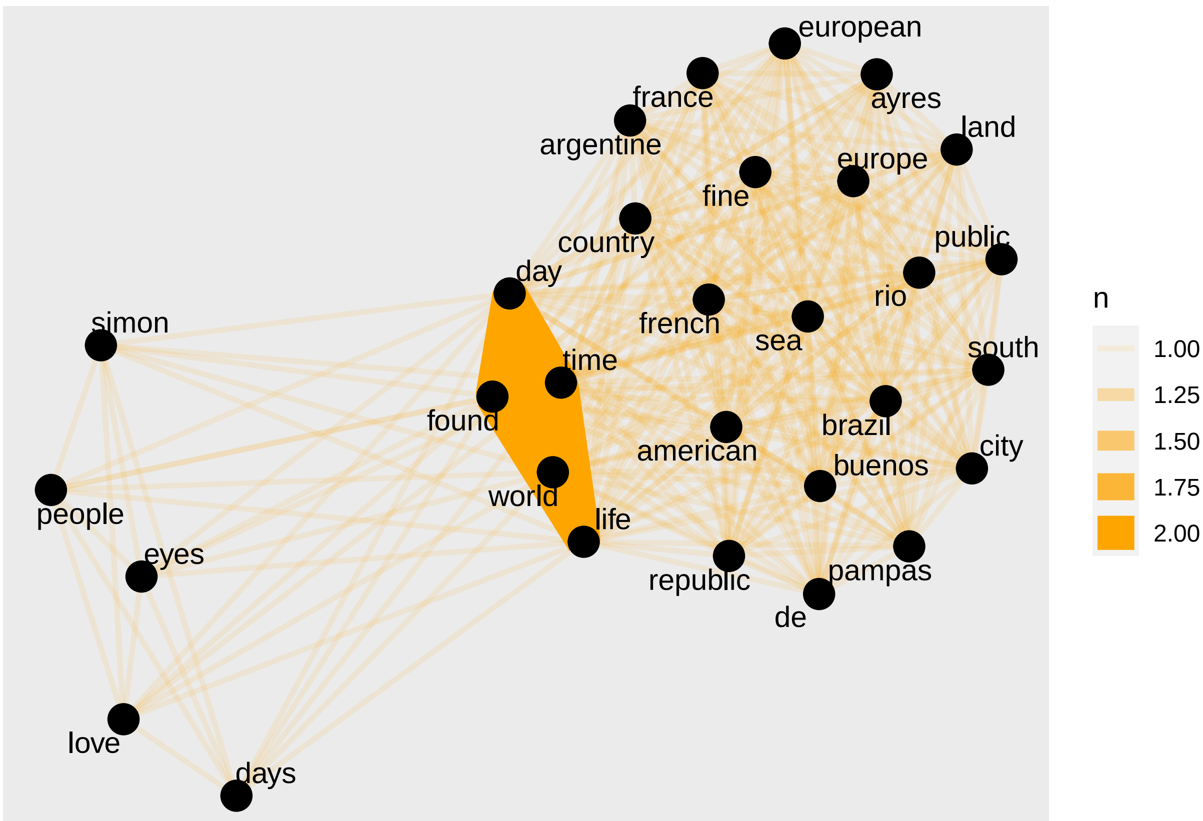
Here again, we see South America To-Day highly associated with words like south and rio, and then more loosely associated with words like anglo or cattle. Meanwhile, Surprises of Life is highly associated with Jean and monsieur and more loosely associated with wooden and gray.

```
library(widyr)

clem_net <- count_clemenceau %>%
  filter(n > 50) %>%
  pairwise_count(word, Work, sort = TRUE, upper = FALSE)

clem_net
```

```
## # A tibble: 335 x 3
##    item1 item2     n
##    <chr> <chr> <dbl>
##  1 day   time      2
##  2 day   life      2
##  3 time  life      2
##  4 day   world     2
##  5 time  world     2
##  6 life  world     2
```

```
## 7 day   found     2
## 8 time  found     2
## 9 life  found     2
## 10 world found     2
## # i 325 more rows
```

```
clem_net %>%
  graph_from_data_frame() %>%
  ggraph(layout = "fr") +
  geom_edge_link(aes(edge_alpha = n, edge_width = n), edge_colour = "orange") +
  geom_node_point(size = 5) +
  geom_node_text(aes(label = name), repel = TRUE,
                 point.padding = unit(0.2, "lines"))
```



Here we see the word network without the book associations. This shows how the words are paired together and the correlation (based on number of times). In other words, if the word is "found" what word will usually be in the bigram. We see that day, time, found, world, and life, are the basis for a lot of the pairings, with the large cluster to the right also being highly paired, and then the cluster on the left: simon, people, eyes, love, days being less paired. It is interesting that days is so frequent, especially compared with days, the plural.

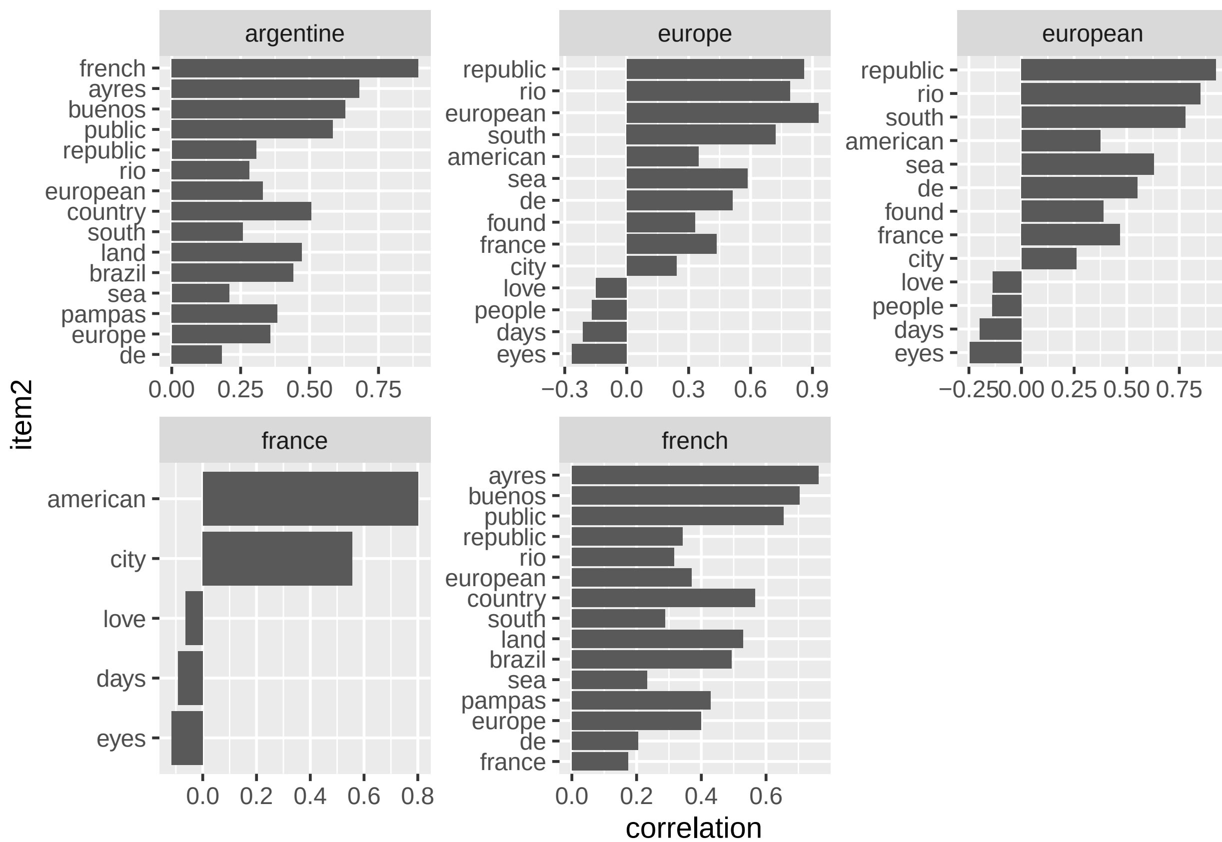
One issue with the analysis is that most of Clemenceau's words are actually fairly low in frequency, and we don't have enough material to find interesting relationships between the words.

#Pairwise correlation

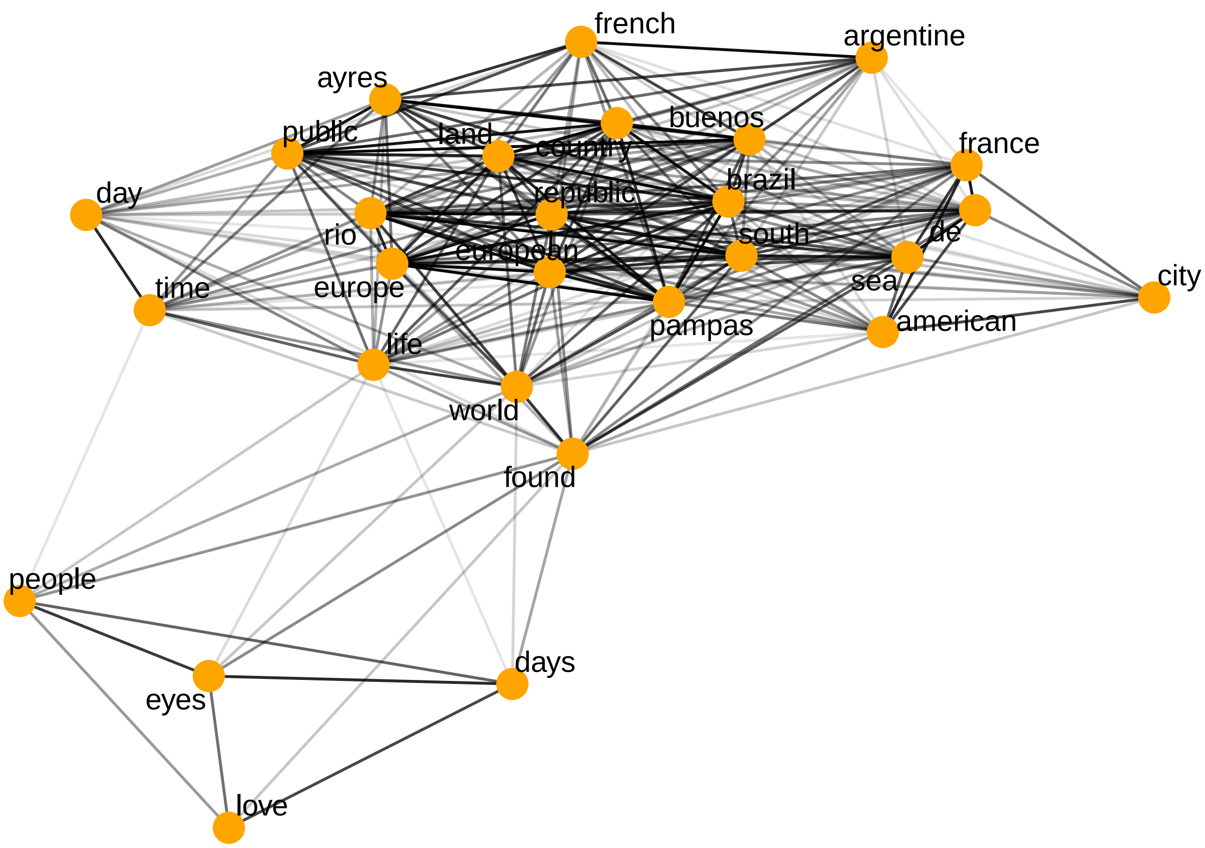
Continuing with the analysis of correlation between words that appear together, we can try to establish some pairwise correlations between specific words of interest. Due to the nature of the two works, I chose a few core words to see what other words are correlated with them: “french”, “france”, “europe”, “european”, “argentine”.

```
clem_net_grouped <- clem_net %>% group_by(item1)

clem_net_cors <- clem_net_grouped %>%
  pairwise_cor(item1, item2, sort = TRUE, upper = FALSE)


#65
clem_net_cors %>%
  filter(item1 %in% c("french", "france", "europe", "european", "argentine")) %>%
  group_by(item1) %>%
  top_n(15) %>%
  ungroup() %>%
  mutate(item2 = reorder(item2, correlation)) %>%
  ggplot(aes(item2, correlation)) +
  geom_bar(stat = "identity") +
  facet_wrap( ~ item1, scales = "free") +
  coord_flip()
```

```
## Selecting by correlation
```



```r
clem_net_cors %>%
  filter(correlation > .15) %>%
  graph_from_data_frame() %>%
  ggraph(layout = "fr") +
  geom_edge_link(aes(edge_alpha = correlation), show.legend = FALSE) +
  geom_node_point(color = "orange", size = 5) +
  geom_node_text(aes(label = name), repel = TRUE) +
  theme_void()
```



For our core words, "french", "france", "europe", "european", "argentine", we see which words are most likely to appear together, vs which may be more likely not to appear together. For the word "france", for example, "american" and "city" are much more likely to appear together, while words like "love", "days", and "eyes" are less likely to appear together. This is also similar for "europe" and "european".

From this, we see that if Clemenceau is talking about a country, he's likely going to be discussing a specific nationality, or maybe a specific place. However, if he is talking about people, their features, or emotions such as "love", he most likely isn't going to be talking about a place or a nationality. This is interesting firstly, because it shows that Clemenceau keeps his topics separate. However, it also indicates that Clemenceau, when speaking about countries, is mostly talking about the machinations of those countries, or their economies, and is not often talking about the people and cultures of those countries, since people is negatively correlated with "french" or "europe".

This sort of analysis the the very rudimentary start of the sort of math that powers Large Language Models.

#A Word Cloud

We conclude this analysis with a word cloud of Clemenceau's words. This shows his most frequent words, and shows the most frequent as larger.

```r
library(wordcloud)
```

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

```r
suppressWarnings({
tidy_clemenceau %>% select(-c(Work)) %>%
  count(word, sort = TRUE) %>%
  bind_tf_idf(word, word, n) %>%
  arrange(desc(tf_idf)) %>%
  with(wordcloud(word, n, max.words = 1000)) })
```

The most common word is French, followed by American, Reason, tree, son, sky, coffee, labor, world, days, wife, and then all the rest. This is interesting; in all the previous analyses, we very rarely saw world, or wife. And yet, of Clemenceau's words here are some of the most prominent.

However, this word cloud is not altogether out of character with what we've already seen. Clemenceau's words are split between words for a political and economic treatise: coffee, world, iron, wealth. And then, the words of daily life: wife, son, soul.

This cloud does highlight some of the sentiment words that may have escaped our previous analysis such as anxious or fair.

We began this analysis eager to see what Clemenceau felt. Have we found it? We learned that when he talks about politics, he does so at the high level of international systems and how each place contributes. And we saw that if he speaks of daily life, he does so by focusing on the people living those lives. We also saw that Clemenceau very rarely rails against anything. His sentiments, especially negative, are often in the abstract.

However, this analysis is somewhat harangued by the lack of texts (only 2), and the fact that both texts are translations. A course for further study would be to gather more of Clemenceau's writings, and specifically in their original French and see what other aspects of Clemenceau can be gleaned.