Some Simple SPOOKY Data Analysis

Chuyuan Zhou

February 5, 2018

Introduction

This files contains text mining analysis of the SPOOKY data. The goal is to remind ourselves of some of our basic tools for working with text data in `R` and also to practice reproducibility. You should be able to put this file in the `doc` folder of your `Project 1` repository and it should just run (provided you have `multiplot.R` in the `libs` folder and `spooky.csv` in the `data` folder). ## Read in the data The following code assumes that the dataset `spooky.csv` lives in a `data` folder (and that we are inside a `docs` folder).

```
spooky <- read.csv('../data/spooky.csv', as.is = TRUE)
```

Setup the libraries

First we want to install and load libraries we need along the way. Note that the following code is completely reproducible – you don't need to add any code on your own to make it run.

```
packages.used <- c("ggplot2", "dplyr", "tibble", "tidyr",  "stringr", "tidytext", "topicmodels", "wordcl

library(caret)
in_train <- createDataPartition(y = spooky$text,
                                p = 3 / 4, list = FALSE)
train <- spooky[in_train, ]
test  <- spooky[-in_train, ]

# check packages that need to be installed.
packages.needed <- setdiff(packages.used, intersect(installed.packages()[,1], packages.used))

# install additional packages
if(length(packages.needed) > 0) {
  install.packages(packages.needed, dependencies = TRUE, repos = 'http://cran.us.r-project.org')
}

library(ggplot2)
library(dplyr)
library(tibble)
library(tidyr)
library(stringr)
library(tidytext)
library(topicmodels)
library(wordcloud)
library(ggridges)

source("../lib/multiplot.R")
```

An overview of the data structure and content

Let's first remind ourselves of the structure of the data.

```
head(spooky)
```

```
##        id
## 1 id26305
## 2 id17569
## 3 id11008
## 4 id27763
## 5 id12958
## 6 id22965
##
## 1
## 2
## 3
## 4
## 5
## 6 A youth passed in solitude, my best years spent under your gentle and feminine fosterage, has so re
##   author
## 1    EAP
## 2    HPL
## 3    EAP
## 4    MWS
## 5    HPL
## 6    MWS
```

```
summary(spooky)
```

```
##       id                text              author
##  Length:19579       Length:19579       Length:19579
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
```

We see from the above that each row of our data contains a unique ID, a single sentence text excerpt, and an abbreviated author name. `HPL` is Lovecraft, `MWS` is Shelly, and `EAP` is Poe. We finally note that there are no missing values, and we change author name to be a factor variable, which will help us later on.

```
sum(is.na(spooky))
```

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

```
spooky$author <- as.factor(spooky$author)
```

Data Cleaning

We first use the `unnest_tokens()` function to drop all punctuation and transform all words into lower case. At least for now, the punctuation isn't really important to our analysis – we want to study the words. In addition, `tidytext` contains a dictionary of stop words, like "and" or "next", that we will get rid of for our analysis, the idea being that the non-common words (. . . maybe the SPOOKY words) that the authors use will be more interesting. If this is new to you, here's a textbook that can help: *Text Mining with R; A Tidy Approach*. It teaches the basic handling of natural language data in `R` using tools from the "tidyverse". The tidy text format is a table with one token per row, where a token is a word.

```
# Make a table with one word per row and remove `stop words` (i.e. the common words).
spooky_wrd <- unnest_tokens(spooky, word, text)
spooky_wrd <- anti_join(spooky_wrd, stop_words, by = "word")
```

Last Time and some More

Word Frequency

Now we study some of the most common words in the entire data set. With the below code we plot the sixty most common words in the entire datset. We see that "time", "life", and "night" all appear frequently.

```
# Words is a list of words, and freqs their frequencies
words <- count(group_by(spooky_wrd, word))$word
freqs <- count(group_by(spooky_wrd, word))$n

head(sort(freqs, decreasing = TRUE))
```

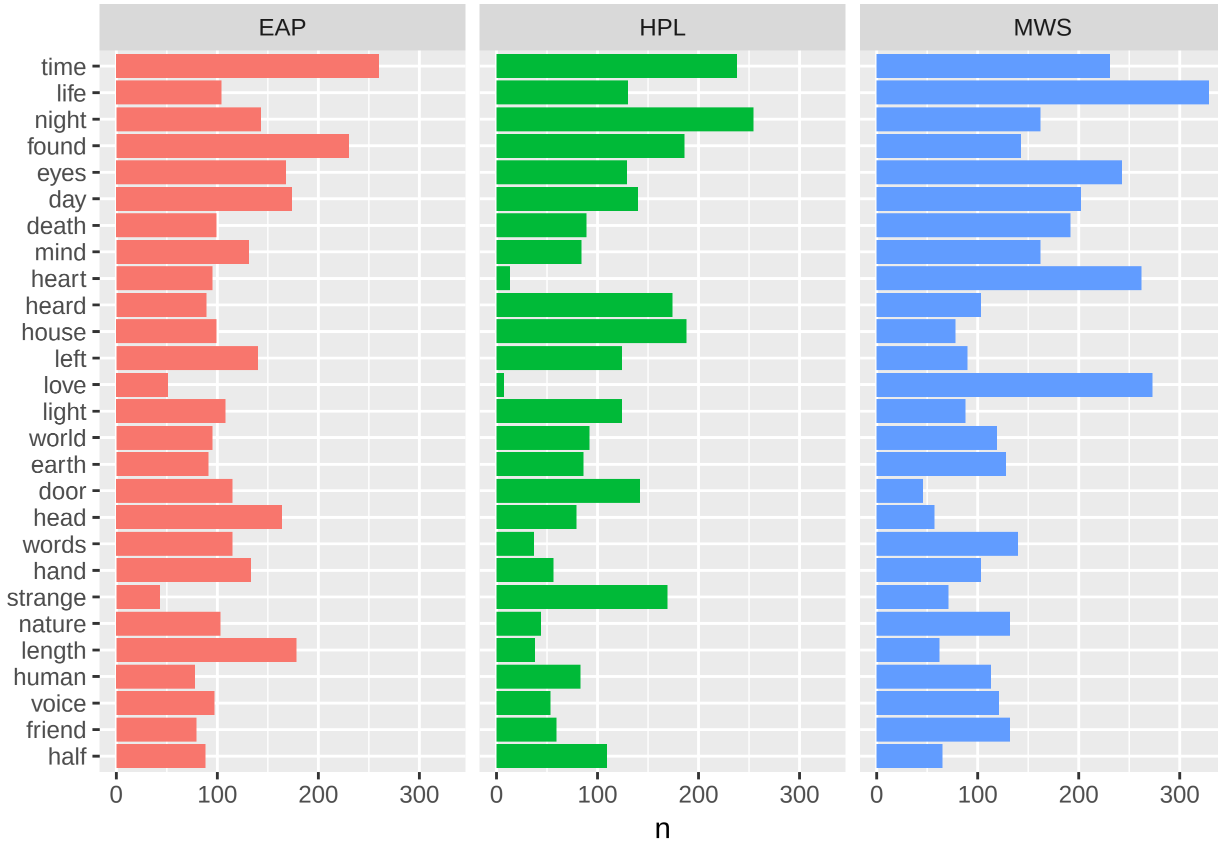
```
## [1] 729 563 559 559 540 516
```

```
png("../figs/Wordcloud_all.png")
wordcloud(words, freqs, max.words = 60 , color = c("purple4", "red4", "black"))
dev.off()
```

```
## pdf
##   2
```

We can compare the way the authors use the most frequent words too.

```
# Counts number of times each author used each word.
author_words <- count(group_by(spooky_wrd, word, author))

# Counts number of times each word was used.
all_words    <- rename(count(group_by(spooky_wrd, word)), all = n)

author_words <- left_join(author_words, all_words, by = "word")
author_words <- arrange(author_words, desc(all))
author_words <- ungroup(head(author_words, 81))

ggplot(author_words) +
  geom_col(aes(reorder(word, all, FUN = min), n, fill = author)) +
  xlab(NULL) +
  coord_flip() +
  facet_wrap(~ author) +
  theme(legend.position = "none")
```



Data Visualization

We'll do some simple numerical summaries of the data to provide some nice visualizations.

```
p1 <- ggplot(spooky) +
      geom_bar(aes(author, fill = author)) +
      theme(legend.position = "none")


spooky$sen_length <- str_length(spooky$text)
head(spooky$sen_length)
```

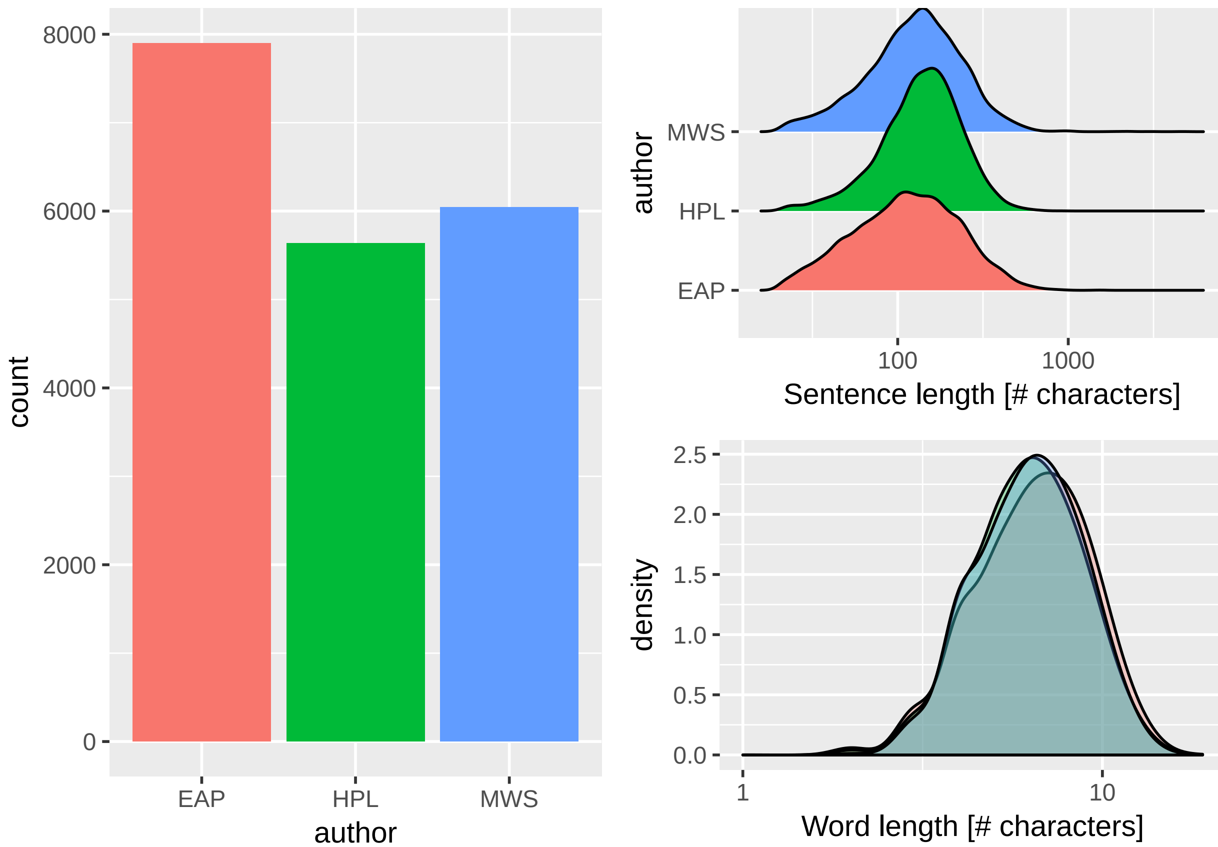
```
## [1] 231  71 200 206 174 468
```

```
p2 <- ggplot(spooky) +
      geom_density_ridges(aes(sen_length, author, fill = author)) +
      scale_x_log10() +
      theme(legend.position = "none") +
      labs(x = "Sentence length [# characters]")


spooky_wrd$word_length <- str_length(spooky_wrd$word)
head(spooky_wrd$word_length)
```

```
## [1]  7  8  5 12 10  7
```

```
p3 <- ggplot(spooky_wrd) +
      geom_density(aes(word_length, fill = author), bw = 0.05, alpha = 0.3) +
```

```
    scale_x_log10() +
    theme(legend.position = "none") +
    labs(x = "Word length [# characters]")

layout <- matrix(c(1, 2, 1, 3), 2, 2, byrow = TRUE)
multiplot(p1, p2, p3, layout = layout)
```



From the above plots we find:

- EAP is featured most frequently.
- Sentence length for EAP is more variable.
-

TF-IDF

TF stands for term frequency or how often a word appears in a text and it is what is studied above in the word cloud. IDF stands for inverse document frequncy, and it is a way to pay more attention to words that are rare within the entire set of text data that is more sophisticated than simply removing stop words. Multiplying these two values together calculates a term's tf-idf, which is the frequency of a term adjusted for how rarely it is used.

We'll use tf-idf as a heuristic index to indicate how frequently a certain author uses a word relative to the frequency that all the authors use the word. Therefore we will find words that are characteristic for a specific author, a good thing to have if we are interested in solving the author identification problem.

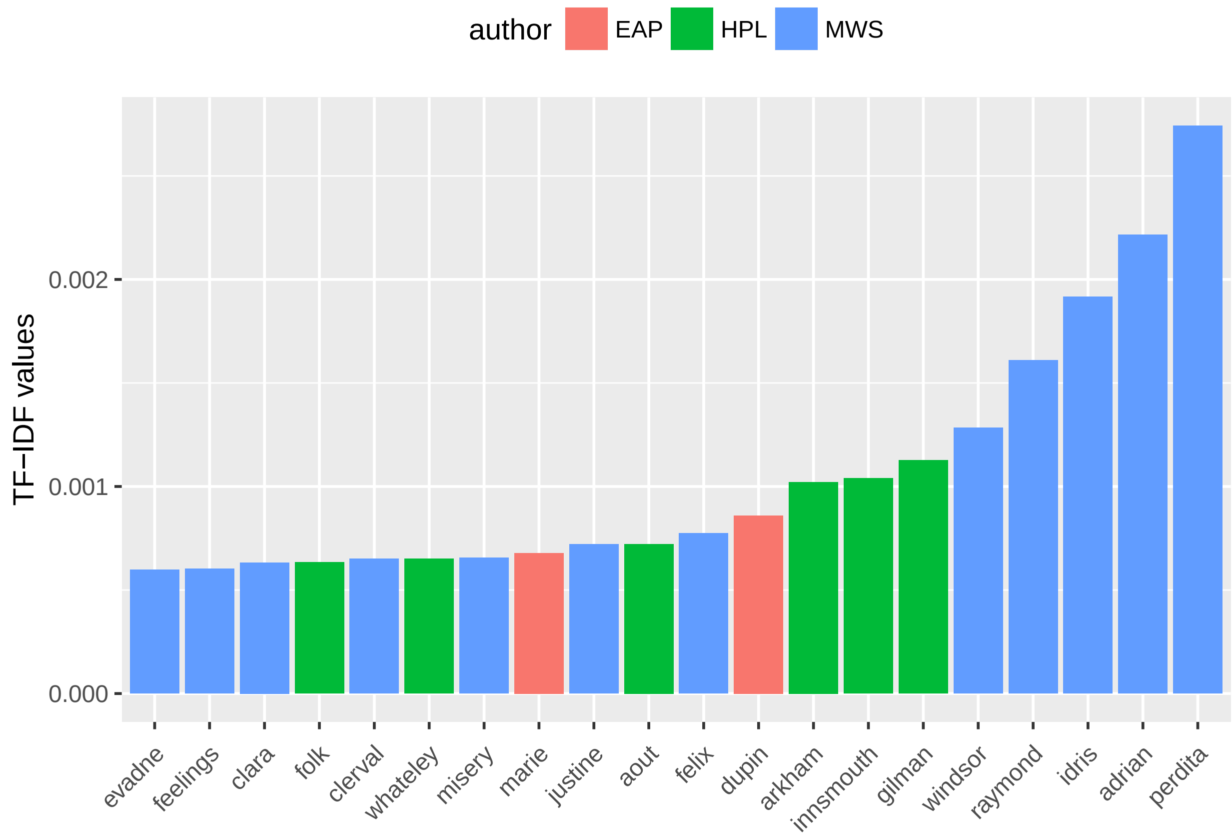
```r
frequency <- count(spooky_wrd, author, word)
tf_idf    <- bind_tf_idf(frequency, word, author, n)
head(tf_idf)
```

```
## # A tibble: 6 x 6
##   author word                    n        tf   idf    tf_idf
##   <fct>  <chr>               <int>     <dbl> <dbl>     <dbl>
## 1 EAP    "\u00e0"                9 0.000124   1.10 0.000136
## 2 EAP    "\u00e6rial"            1 0.0000137  1.10 0.0000151
## 3 EAP    "\u00e6ronaut"          2 0.0000275  1.10 0.0000302
## 4 EAP    "\u00e6ronauts"         1 0.0000137  1.10 0.0000151
## 5 EAP    "\u00e6rostation"       1 0.0000137  1.10 0.0000151
## 6 EAP    "\u00e6schylus"         1 0.0000137  1.10 0.0000151
```

```r
tail(tf_idf)
```

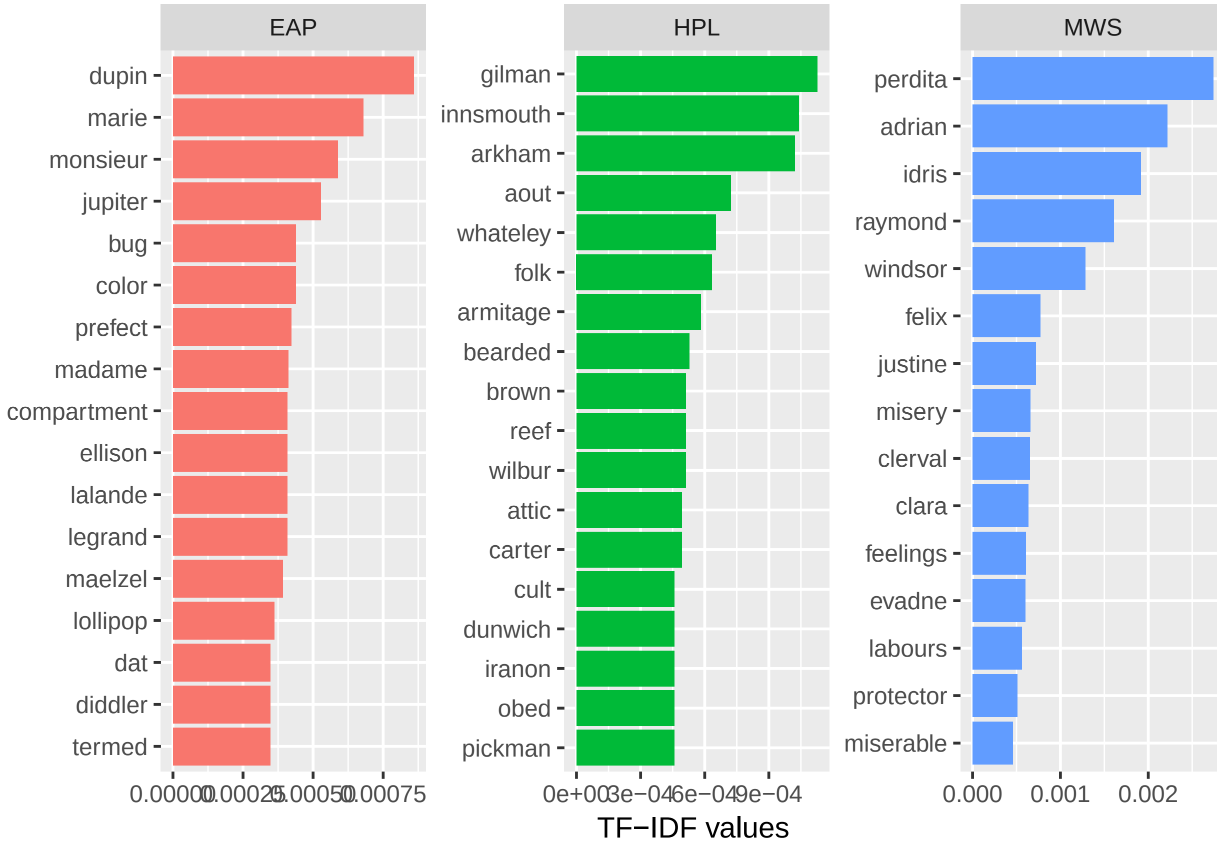
```
## # A tibble: 6 x 6
##   author word         n        tf   idf     tf_idf
##   <fct>  <chr>    <int>     <dbl> <dbl>      <dbl>
## 1 MWS    youth's      1 0.0000160 0.405 0.00000649
## 2 MWS    youthful    10 0.000160  0     0
## 3 MWS    youths       2 0.0000320 0.405 0.0000130
## 4 MWS    zaimi        2 0.0000320 1.10  0.0000352
## 5 MWS    zeal         7 0.000112  0     0
## 6 MWS    zest         3 0.0000480 0     0
```

```r
tf_idf    <- arrange(tf_idf, desc(tf_idf))
tf_idf    <- mutate(tf_idf, word = factor(word, levels = rev(unique(word))))

# Grab the top twenty tf_idf scores in all the words
tf_idf_20 <- top_n(tf_idf, 20, tf_idf)

ggplot(tf_idf_20) +
  geom_col(aes(word, tf_idf, fill = author)) +
  labs(x = NULL, y = "TF-IDF values") +
  theme(legend.position = "top", axis.text.x  = element_text(angle=45, hjust=1, vjust=0.9))
```



Note that in the above, many of the words recognized by their tf-idf scores are names. This makes sense – if we see text referencing Raymond, Idris, or Perdita, we know almost for sure that MWS is the author. But some non-names stand out. EAP often uses "monsieur" and "jupiter" while HPL uses the words "bearded" and "attic" more frequently than the others. We can also look at the most characteristic terms per author.

```
# Grab the top fifteen tf_idf scores in all the words for each author
tf_idf <- ungroup(top_n(group_by(tf_idf, author), 15, tf_idf))

ggplot(tf_idf) +
  geom_col(aes(word, tf_idf, fill = author)) +
  labs(x = NULL, y = "tf-idf") +
  theme(legend.position = "none") +
  facet_wrap(~ author, ncol = 4, scales = "free") +
  coord_flip() +
  labs(y = "TF-IDF values")
```



Sentiment Analysis

We will use sentences as units of analysis for this part of the tutorial, as sentences are natural languge units for organizing thoughts and ideas. For each sentence, we apply sentiment analysis using NRC sentiment lexion. "The NRC Emotion Lexicon is a list of English words and their associations with eight basic emotions (anger, fear, anticipation, trust, surprise, sadness, joy, and disgust) and two sentiments (negative and positive). The annotations were manually done by crowdsourcing."

From *Text Mining with R; A Tidy Approach*, "When human readers approach text, we use our understanding of the emotional intent of words to infer whether a section of text is positive or negative, or perhaps characterized by some other more nuanced emotion like surprise or disgust. We can also use the tools of text mining to approach the emotional content of text programmatically." This is the goal of sentiment analysis.

```
# Keep words that have been classified within the NRC lexicon.
get_sentiments('nrc')
```

```
## # A tibble: 13,901 x 2
##    word        sentiment
##    <chr>       <chr>
##  1 abacus      trust
##  2 abandon     fear
##  3 abandon     negative
##  4 abandon     sadness
##  5 abandoned   anger
##  6 abandoned   fear
##  7 abandoned   negative
```

```
##  8 abandoned   sadness
##  9 abandonment anger
## 10 abandonment fear
## # ... with 13,891 more rows
```

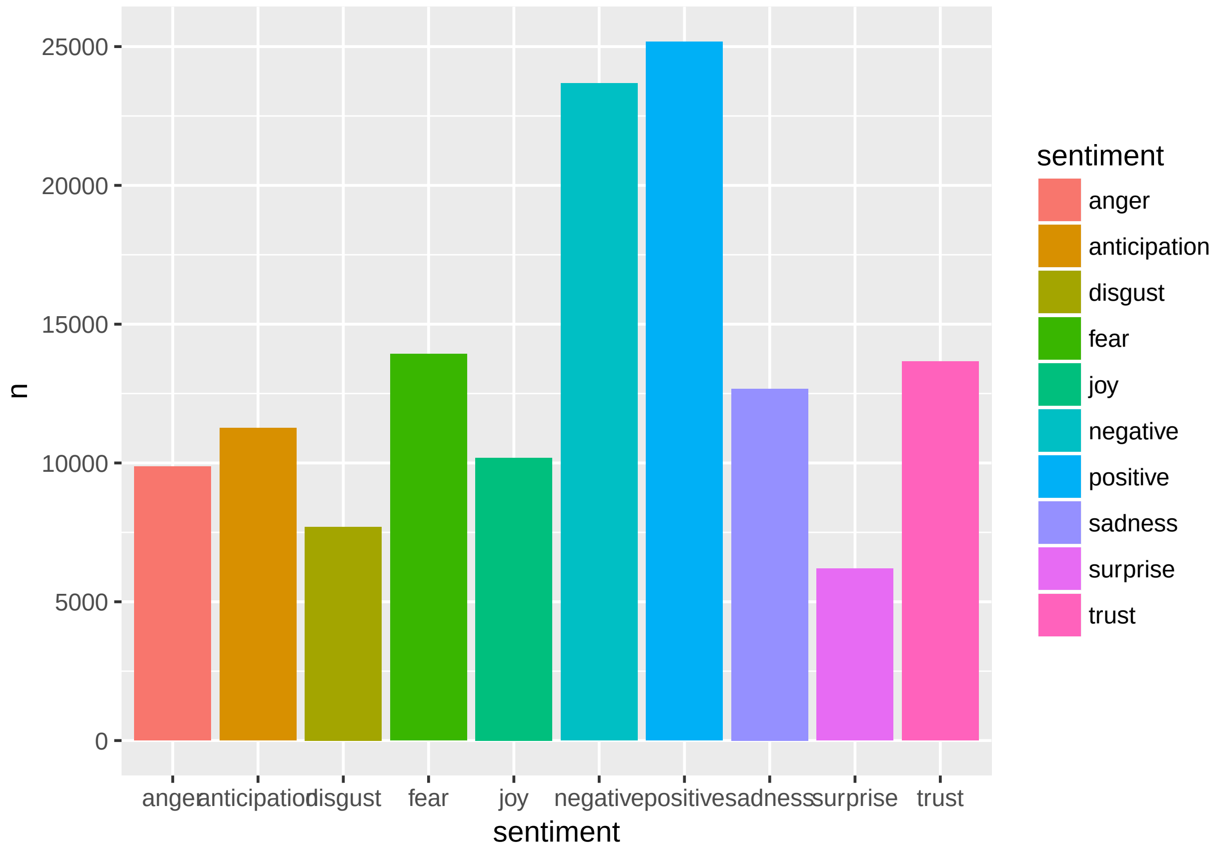
```r
sentiments <- inner_join(spooky_wrd, get_sentiments('nrc'), by = "word")

count(sentiments, sentiment)
```

```
## # A tibble: 10 x 2
##    sentiment        n
##    <chr>        <int>
##  1 anger         9869
##  2 anticipation 11258
##  3 disgust       7697
##  4 fear         13927
##  5 joy          10190
##  6 negative     23674
##  7 positive     25175
##  8 sadness      12674
##  9 surprise      6199
## 10 trust        13655
```

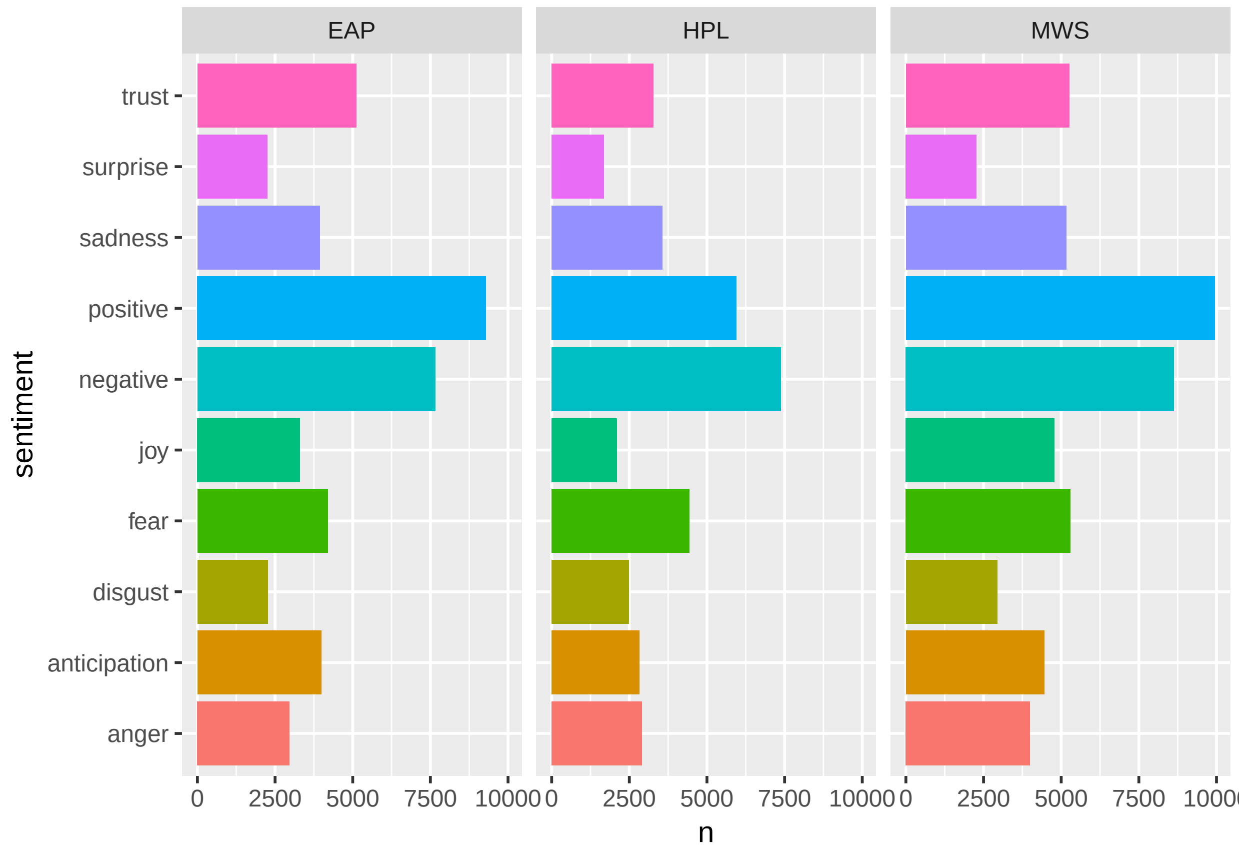
```r
count(sentiments, author, sentiment)
```

```
## # A tibble: 30 x 3
##    author sentiment        n
##    <fct>  <chr>        <int>
##  1 EAP    anger         2962
##  2 EAP    anticipation  3982
##  3 EAP    disgust       2261
##  4 EAP    fear          4194
##  5 EAP    joy           3302
##  6 EAP    negative      7659
##  7 EAP    positive      9291
##  8 EAP    sadness       3938
##  9 EAP    surprise      2244
## 10 EAP    trust         5116
## # ... with 20 more rows
```

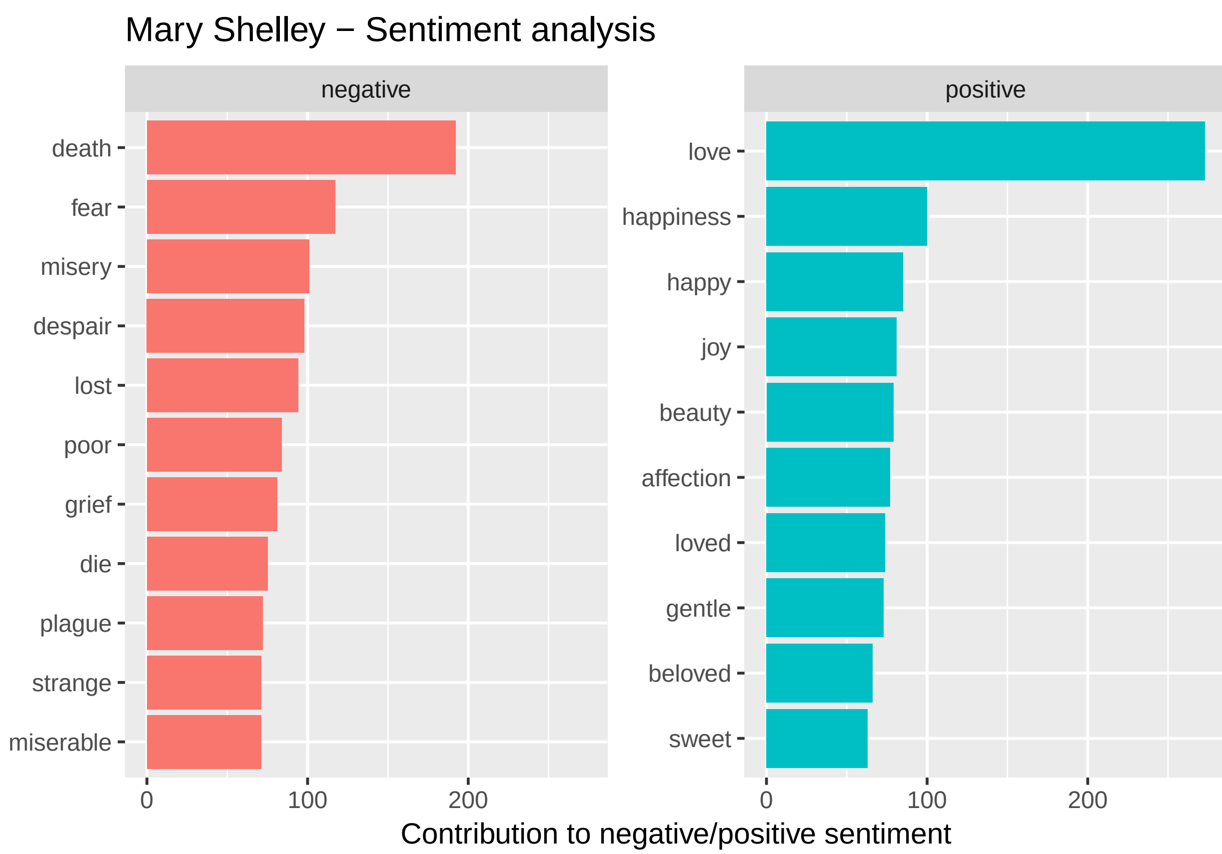
```r
ggplot(count(sentiments, sentiment)) +
  geom_col(aes(sentiment, n, fill = sentiment))
```



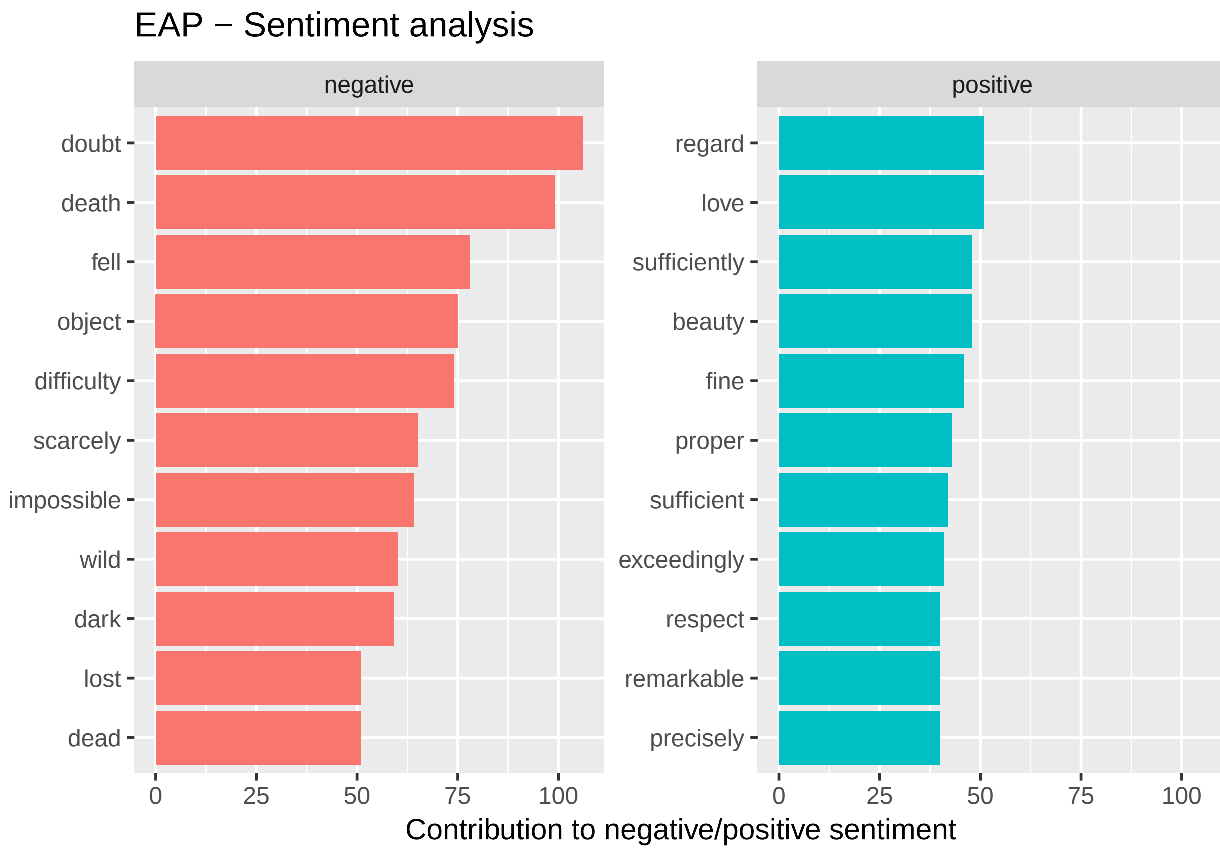

```
ggplot(count(sentiments, author, sentiment)) +
  geom_col(aes(sentiment, n, fill = sentiment)) +
  facet_wrap(~ author) +
  coord_flip() +
  theme(legend.position = "none")
```



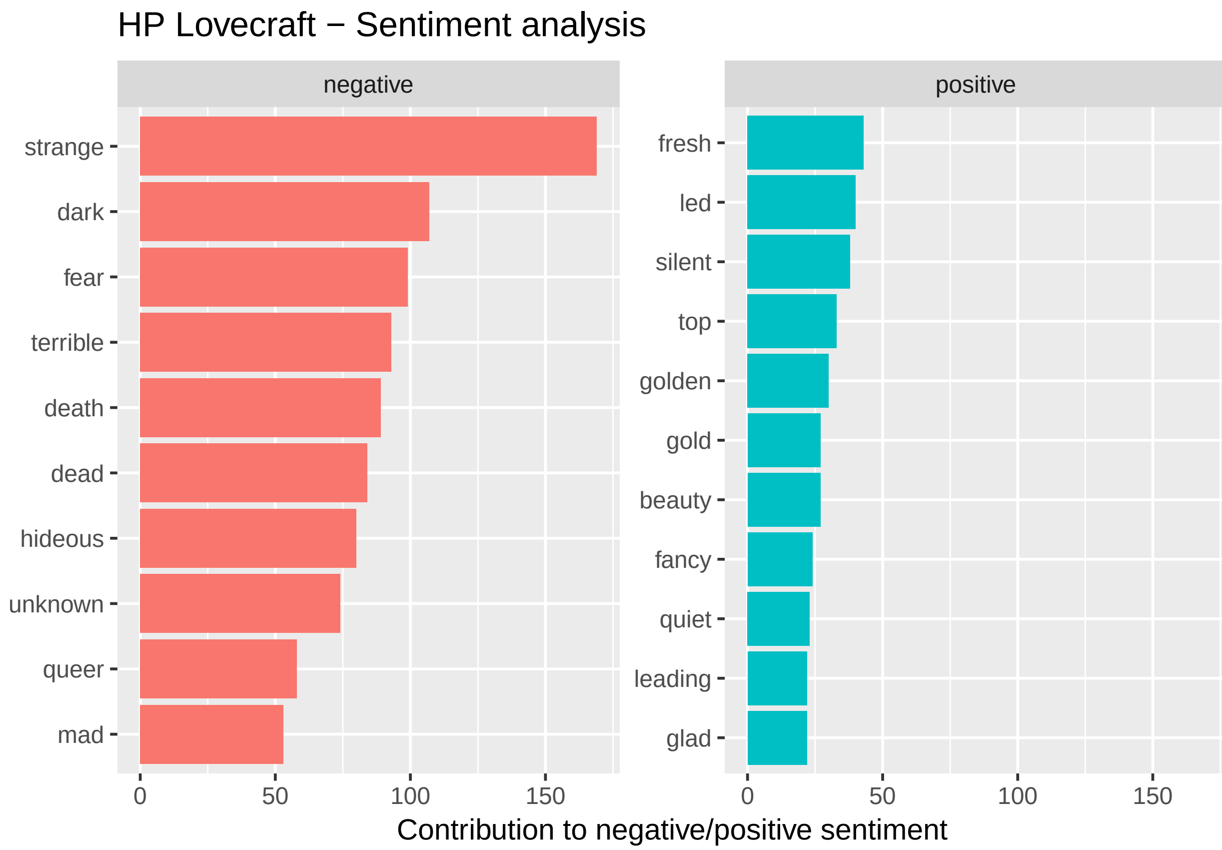

```
spooky_wrd%>%
  filter(author == "MWS") %>%
  inner_join(get_sentiments("bing"), by = "word") %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup() %>%
  group_by(sentiment) %>%
  top_n(10, n) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(y = "Contribution to negative/positive sentiment", x = NULL) +
  coord_flip() +
  ggtitle("Mary Shelley - Sentiment analysis")
```



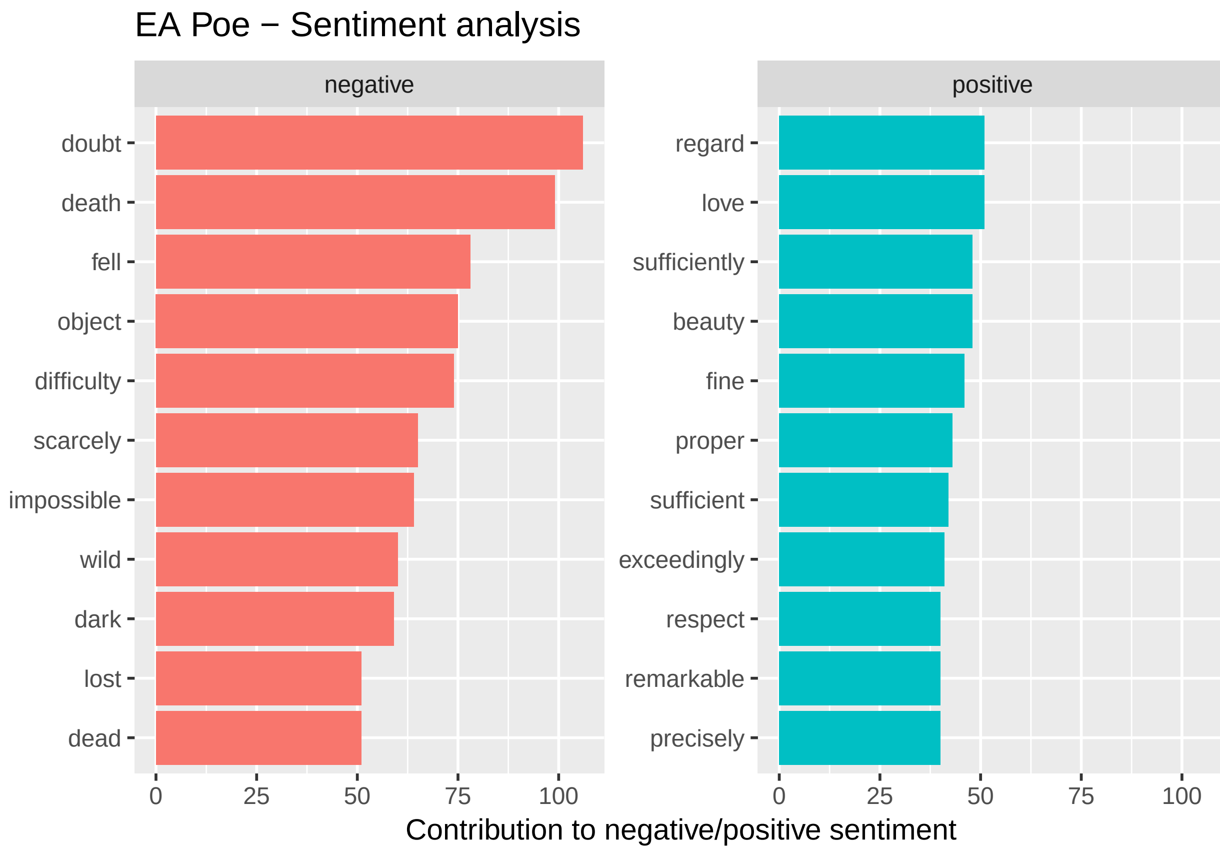

```r
spooky_wrd%>%
  filter(author == "EAP") %>%
  inner_join(get_sentiments("bing"), by = "word") %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup() %>%
  group_by(sentiment) %>%
  top_n(10, n) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(y = "Contribution to negative/positive sentiment", x = NULL) +
  coord_flip() +
  ggtitle("EAP - Sentiment analysis")
```




```r
spooky_wrd%>%
  filter(author == "HPL") %>%
  inner_join(get_sentiments("bing"), by = "word") %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup() %>%
  group_by(sentiment) %>%
  top_n(10, n) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(y = "Contribution to negative/positive sentiment", x = NULL) +
  coord_flip() +
  ggtitle("HP Lovecraft - Sentiment analysis")
```

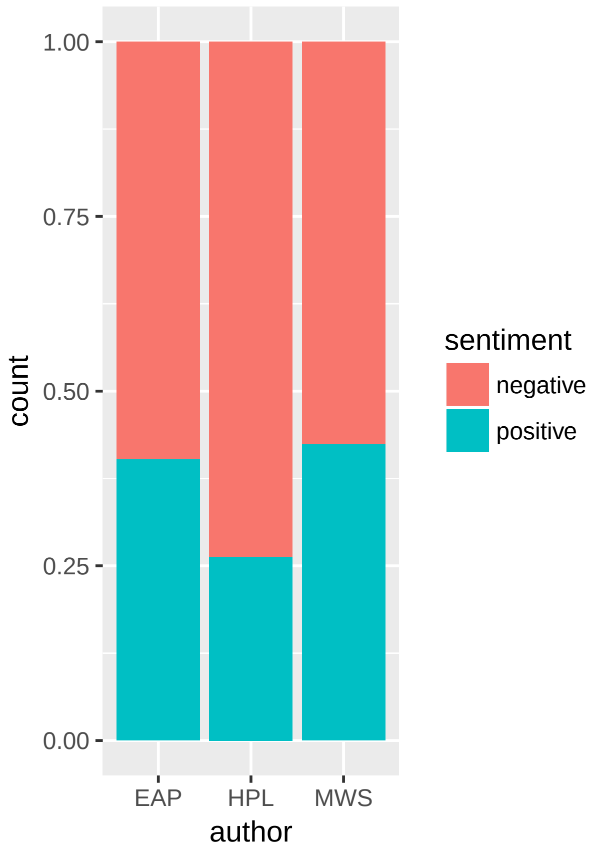


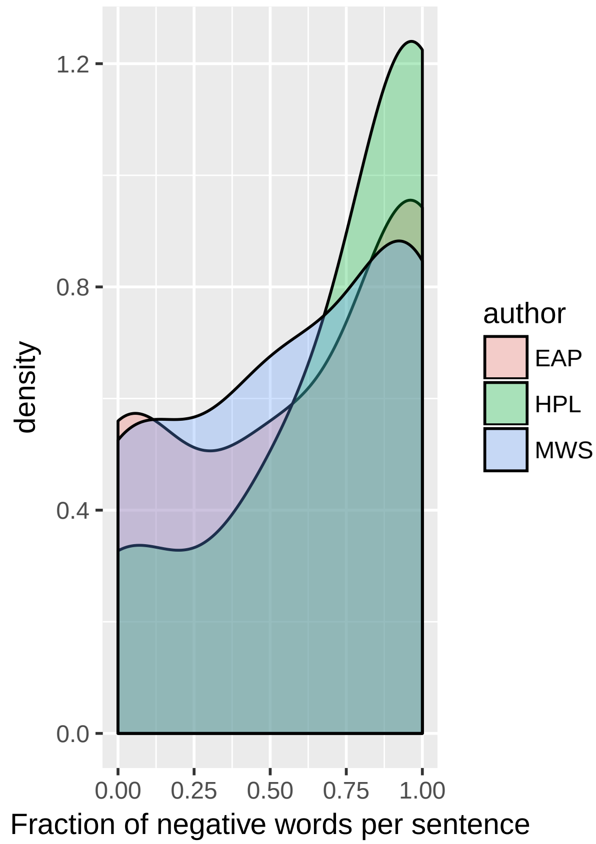

```
spooky_wrd%>%
  filter(author == "EAP") %>%
  inner_join(get_sentiments("bing"), by = "word") %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup() %>%
  group_by(sentiment) %>%
  top_n(10, n) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(y = "Contribution to negative/positive sentiment", x = NULL) +
  coord_flip() +
  ggtitle("EA Poe - Sentiment analysis")
```



The negative Index

```
p1 <- spooky_wrd %>%
  inner_join(get_sentiments("bing"), by = "word") %>%
  ggplot(aes(author, fill = sentiment)) +
  geom_bar(position = "fill")

p2 <- spooky_wrd %>%
  inner_join(get_sentiments("bing"), by = "word") %>%
  group_by(author, id, sentiment) %>%
  count() %>%
  spread(sentiment, n, fill = 0) %>%
  group_by(author, id) %>%
  summarise(neg = sum(negative),
            pos = sum(positive)) %>%
  arrange(id) %>%
  mutate(frac_neg = neg/(neg + pos)) %>%
  ggplot(aes(frac_neg, fill = author)) +
  geom_density(bw = .2, alpha = 0.3) +
  theme(legend.position = "right") +
  labs(x = "Fraction of negative words per sentence")

layout <- matrix(c(1,2),1,2,byrow=TRUE)
multiplot(p1, p2, layout=layout)
```





We find:

H P Lovecraft???s texts are on average notably more negative than the author???s works: Only about 25% positive words vs around 40% for Poe and Shelley.

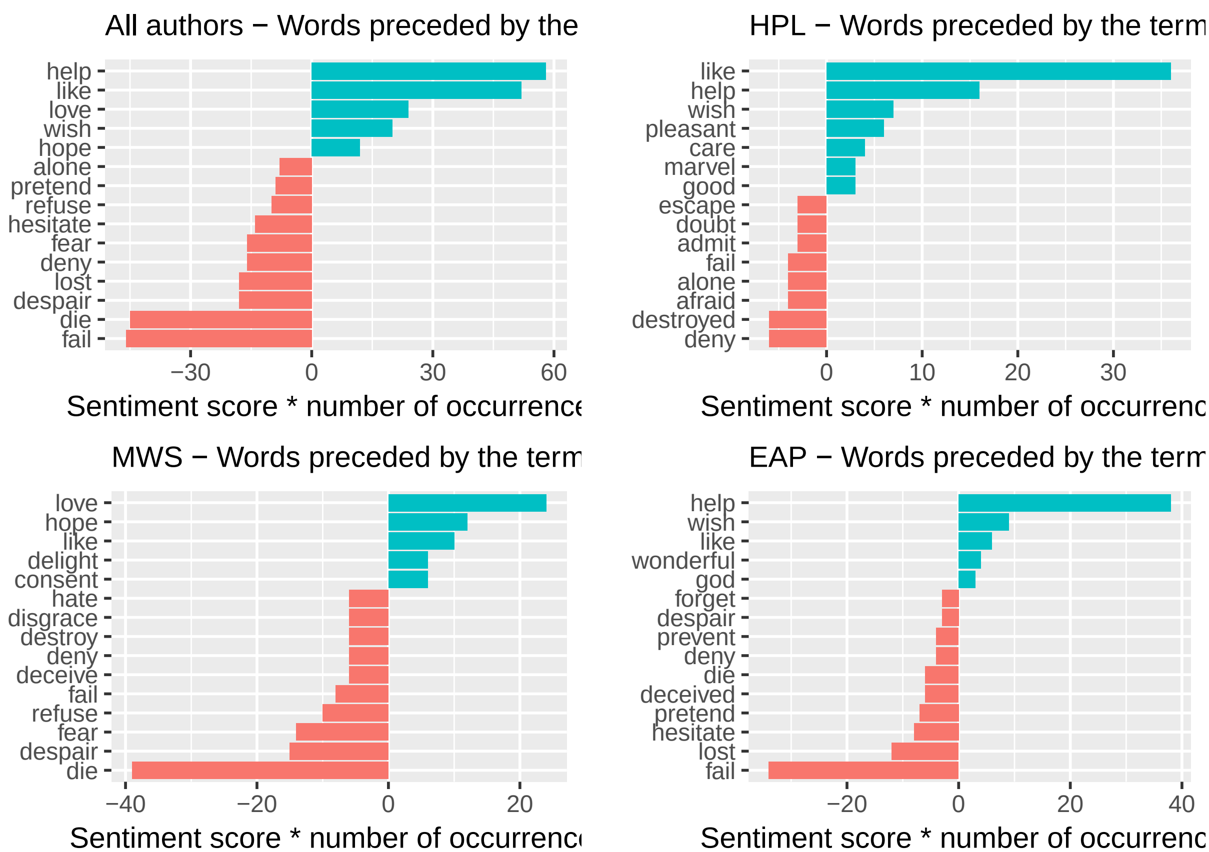
And also the distribution of the fraction of negative words per sentence is clearly skewed towards larger values for HPL (green) than in the case of MWS and EAP. The difference between Shelley and Poe is more subtle. The fraction of negative words in Mary Shelley???s work rises gradually toward larger values, whereas for Edgar Allan Poe is goes more into a direction of ???all or nothing???: his texts show an almost bimodal distribution with peak at low and high negativity, respectively.

Negated negativity

```
bi_sep <- train %>%
  select(author, text) %>%
  unnest_tokens(bigram, text, token = "ngrams", n = 2) %>%
  separate(bigram, c("word1", "word2"), sep = " ")

p1 <- bi_sep %>%
  filter(word1 == "not") %>%
  inner_join(get_sentiments("afinn"), by = c(word2 = "word")) %>%
  count(word1, word2, score, sort = TRUE) %>%
  ungroup() %>%
  mutate(contribution = n * score) %>%
  arrange(desc(abs(contribution))) %>%
  head(15) %>%
  mutate(word2 = reorder(word2, contribution)) %>%
```

```
  ggplot(aes(word2, n * score, fill = n * score > 0)) +
  geom_col(show.legend = FALSE) +
  xlab("") +
  ylab("Sentiment score * number of occurrences") +
  coord_flip() +
  theme(plot.title = element_text(size=11)) +
  ggtitle("All authors - Words preceded by the term 'not'")

p2 <- bi_sep %>%
  filter(author == "HPL") %>%
  filter(word1 == "not") %>%
  inner_join(get_sentiments("afinn"), by = c(word2 = "word")) %>%
  count(word1, word2, score, sort = TRUE) %>%
  ungroup() %>%
  mutate(contribution = n * score) %>%
  arrange(desc(abs(contribution))) %>%
  head(15) %>%
  mutate(word2 = reorder(word2, contribution)) %>%
  ggplot(aes(word2, n * score, fill = n * score > 0)) +
  geom_col(show.legend = FALSE) +
  xlab("") +
  ylab("Sentiment score * number of occurrences") +
  coord_flip() +
  theme(plot.title = element_text(size=11)) +
  ggtitle("HPL - Words preceded by the term 'not'")

p3 <- bi_sep %>%
  filter(author == "MWS") %>%
  filter(word1 == "not") %>%
  inner_join(get_sentiments("afinn"), by = c(word2 = "word")) %>%
  count(word1, word2, score, sort = TRUE) %>%
  ungroup() %>%
  mutate(contribution = n * score) %>%
  arrange(desc(abs(contribution))) %>%
  head(15) %>%
  mutate(word2 = reorder(word2, contribution)) %>%
  ggplot(aes(word2, n * score, fill = n * score > 0)) +
  geom_col(show.legend = FALSE) +
  xlab("") +
  ylab("Sentiment score * number of occurrences") +
  coord_flip() +
  theme(plot.title = element_text(size=11)) +
  ggtitle("MWS - Words preceded by the term 'not'")

p4 <- bi_sep %>%
  filter(author == "EAP") %>%
  filter(word1 == "not") %>%
  inner_join(get_sentiments("afinn"), by = c(word2 = "word")) %>%
  count(word1, word2, score, sort = TRUE) %>%
  ungroup() %>%
  mutate(contribution = n * score) %>%
  arrange(desc(abs(contribution))) %>%
  head(15) %>%
```

```r
  mutate(word2 = reorder(word2, contribution)) %>%
  ggplot(aes(word2, n * score, fill = n * score > 0)) +
  geom_col(show.legend = FALSE) +
  xlab("") +
  ylab("Sentiment score * number of occurrences") +
  coord_flip() +
  theme(plot.title = element_text(size=11)) +
  ggtitle("EAP - Words preceded by the term 'not'")

layout <- matrix(c(1,2,3,4),2,2,byrow=TRUE)
multiplot(p1, p2, p3, p4, layout=layout)
```



Comparing Positivity

Let's only study the "positive" words. Note that the amount of "postive" words attributed to each author varies greatly, and the relative frequency of "positive" words to the other sentiments also varies between authors.

```r
nrc_pos <- filter(get_sentiments('nrc'), sentiment == "positive")
nrc_pos
```

```
## # A tibble: 2,312 x 2
##    word          sentiment
##    <chr>         <chr>
##  1 abba          positive
##  2 ability       positive
```

```
##  3 abovementioned positive
##  4 absolute       positive
##  5 absolution     positive
##  6 absorbed       positive
##  7 abundance      positive
##  8 abundant       positive
##  9 academic       positive
## 10 academy        positive
## # ... with 2,302 more rows
```

```
positive <- inner_join(spooky_wrd, nrc_pos, by = "word")
head(positive)
```

```
##        id author     word word_length sentiment
## 1 id11008    EAP     gold           4  positive
## 2 id27763    MWS   lovely           6  positive
## 3 id27763    MWS  fertile           7  positive
## 4 id27763    MWS    happy           5  positive
## 5 id27763    MWS cheering           8  positive
## 6 id27763    MWS     fair           4  positive
```

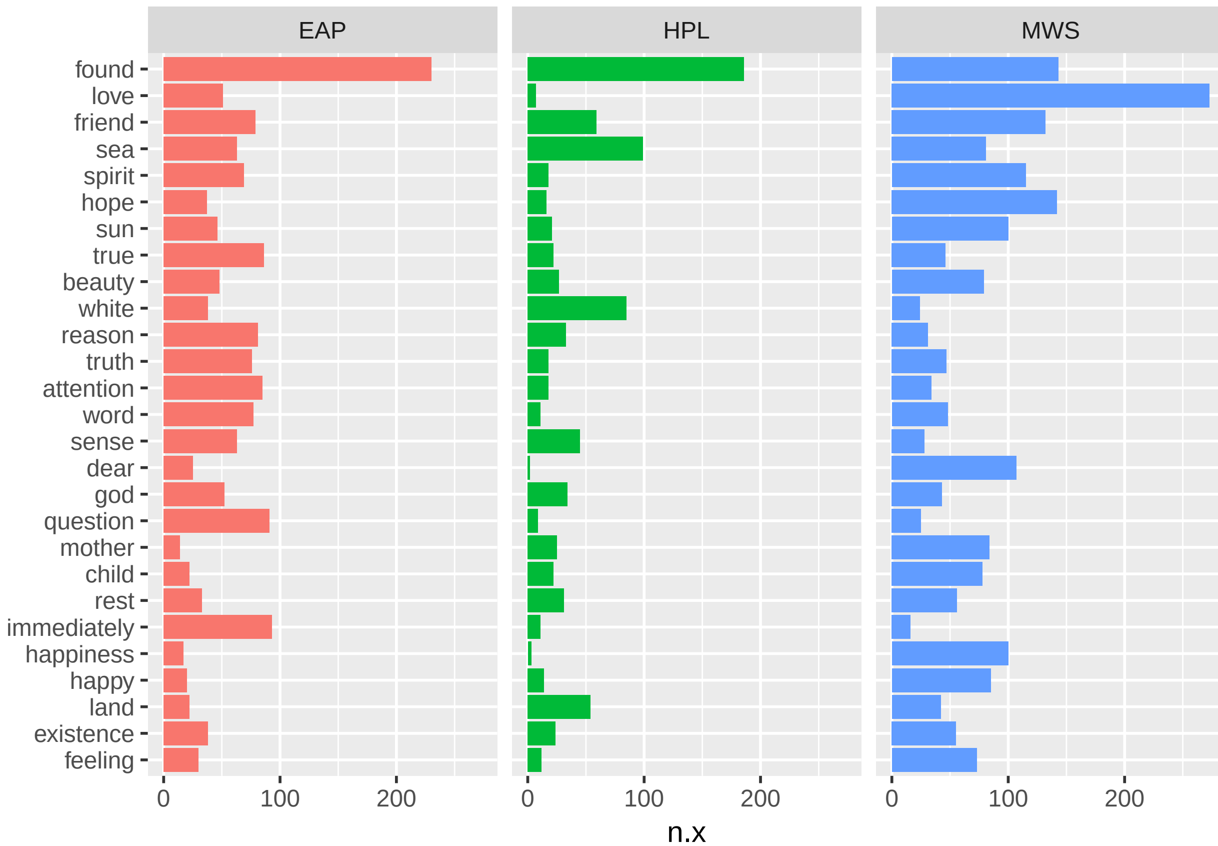
```
count(positive, word, sort = TRUE)
```

```
## # A tibble: 1,690 x 2
##    word       n
##    <chr>  <int>
##  1 found    559
##  2 love     331
##  3 friend   270
##  4 sea      243
##  5 spirit   202
##  6 hope     195
##  7 sun      167
##  8 beauty   154
##  9 true     154
## 10 white    147
## # ... with 1,680 more rows
```

Now we plot a frequency comparison of these "positive" words. Namely, we show the frequencies of the overall most frequently-used positive words split between the three authors. Now we plot a frequency comparison of these "positive" words. Namely, we show the frequencies of the overall most frequently-used positive words split between the three authors.

```
pos_words     <- count(group_by(positive, word, author))
pos_words_all <- count(group_by(positive, word))

pos_words <- left_join(pos_words, pos_words_all, by = "word")
pos_words <- arrange(pos_words, desc(n.y))
pos_words <- ungroup(head(pos_words, 81))

# Note the above is the same as
# pos_words <- pos_words  %>%
#                 left_join(pos_words_all, by = "word") %>%
#                 arrange(desc(n.y)) %>%
#                 head(81) %>%
#                 ungroup()
```

```r
ggplot(pos_words) +
  geom_col(aes(reorder(word, n.y, FUN = min), n.x, fill = author)) +
  xlab(NULL) +
  coord_flip() +
  facet_wrap(~ author) +
  theme(legend.position = "none")
```



The matrix `spooky_wrd_tm` is a sparse matrix with 19467 rows, corresponding to the 19467 ids (or originally, sentences) in the `spooky_wrd` dataframe, and 24941 columns corresponding to the total number of unique words in the `spooky_wrd` dataframe. So each row of `spooky_wrd_tm` corresponds to one of the original sentences. The value of the matrix at a certain position is then the number of occurences of that word (determined by the column) in this specific sentence (determined by the row). Since most sentence/word pairings don't occur, the matrix is sparse meaning there are many zeros.

For LDA we must pick the number of possible topics. Let's try 10, though this selection is admittedly arbitrary. ## Topic models

```r
# Counts how many times each word appears in each sentence
sent_wrd_freqs <- count(spooky_wrd, id, word)
head(sent_wrd_freqs)
```

```
## # A tibble: 6 x 3
##   id      word        n
##   <chr>   <chr>   <int>
## 1 id00001 content     1
## 2 id00001 idris       1
## 3 id00001 mine        1
## 4 id00001 resolve     1
```

```
## 5 id00002 accursed     1
## 6 id00002 city         1
```

```
# Creates a DTM matrix
spooky_wrd_tm <- cast_dtm(sent_wrd_freqs, id, word, n)
spooky_wrd_tm
```

```
## <<DocumentTermMatrix (documents: 19467, terms: 24941)>>
## Non-/sparse entries: 193944/485332503
## Sparsity           : 100%
## Maximal term length: 19
## Weighting          : term frequency (tf)
```

```
length(unique(spooky_wrd$id))
```

```
## [1] 19467
```

```
length(unique(spooky_wrd$word))
```

```
## [1] 24941
```

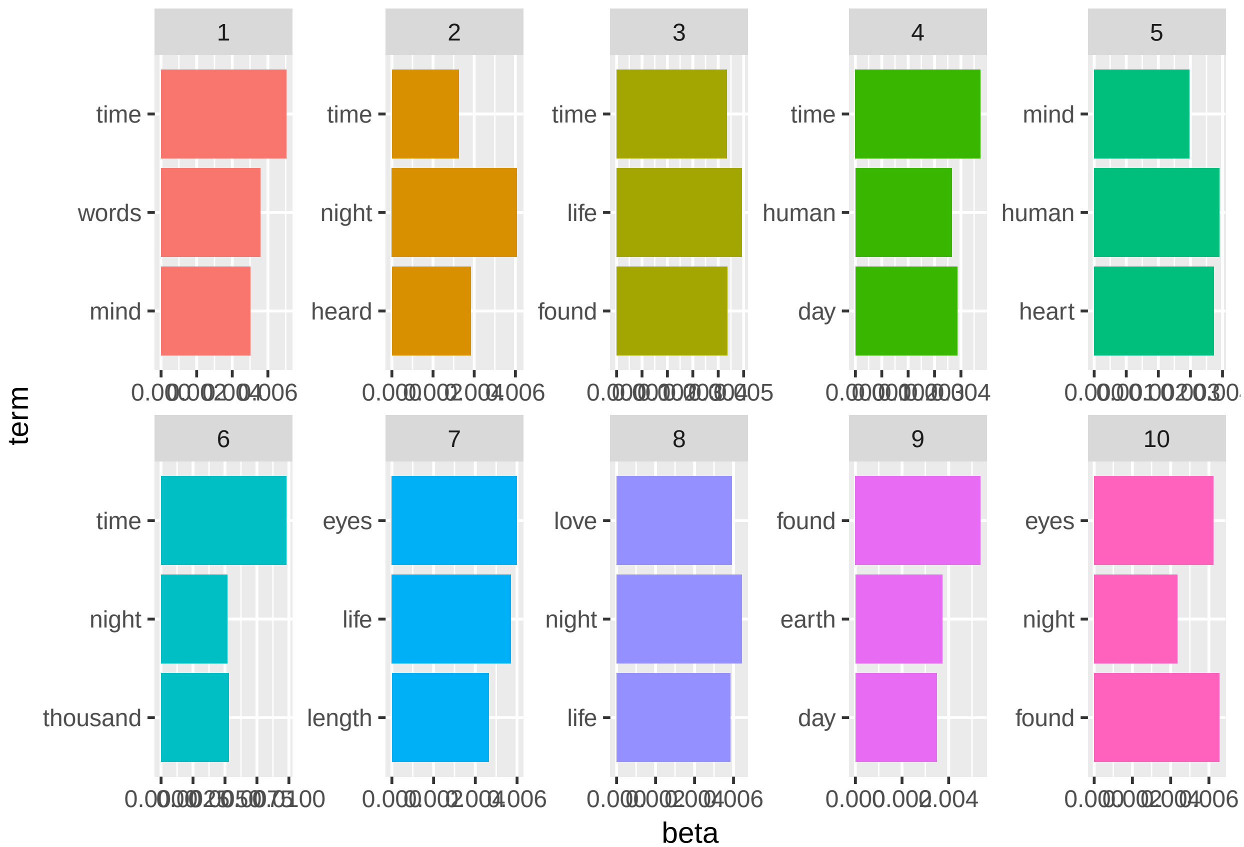
```
spooky_wrd_lda    <- LDA(spooky_wrd_tm, k = 10, control = list(seed = 1989))
spooky_wrd_topics <- tidy(spooky_wrd_lda, matrix = "beta")
spooky_wrd_topics
```

```
## # A tibble: 249,410 x 3
##    topic term         beta
##    <int> <chr>       <dbl>
##  1     1 content 0.0000278
##  2     2 content 0.000108
##  3     3 content 0.0000774
##  4     4 content 0.000112
##  5     5 content 0.000204
##  6     6 content 0.000133
##  7     7 content 0.0000579
##  8     8 content 0.000635
##  9     9 content 0.000144
## 10    10 content 0.000219
## # ... with 249,400 more rows
```

Topics Terms

We note that in the above we use the `tidy` function to extract the per-topic-per-word probabilities, called "beta" or β, for the model. The final output has a one-topic-per-term-per-row format. For each combination, the model computes the probability of that term being generated from that topic. For example, the term ???content??? has a 1.619628×10^{-5} probability of being generated from topic 4. We visualize the top terms (meaning the most likely terms associated with each topic) in the following.

```
# Grab the top three words for each topic.
spooky_wrd_topics_3 <- ungroup(top_n(group_by(spooky_wrd_topics, topic), 3, beta))
spooky_wrd_topics_3 <- arrange(spooky_wrd_topics_3, topic, -beta)
spooky_wrd_topics_3 <- mutate(spooky_wrd_topics_3, term = reorder(term, beta))

ggplot(spooky_wrd_topics_3) +
  geom_col(aes(term, beta, fill = factor(topic)), show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free", ncol = 5) +
  coord_flip()
```



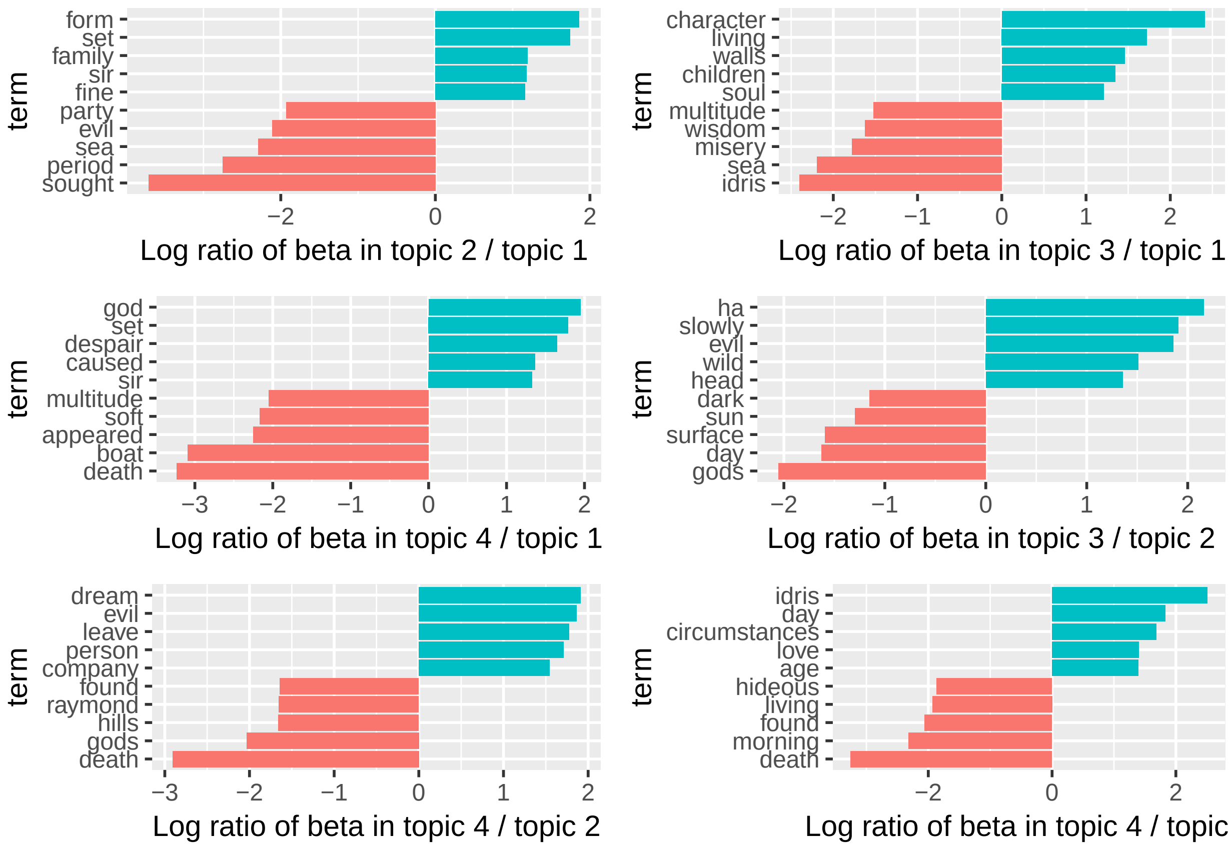
In the above, we see that the first topic is characterized by words like "love", "earth", and "words" while the third topic includes the word "thousand", and the fifth topic the word "beauty". Note that the words "eyes" and "time" appear in many topics. This is the advantage to topic modelling as opposed to clustering when using natural language – often a word may be likely to appear in documents characterized by multiple topics.

We can also study terms that have the greatest difference in probabilities between the topics, ignoring the words that are shared with similar frequency between topics. We choose only the first 3 topics as example and visualise the differences by plotting log ratios: $log_{10}(\beta$ of topic x $/\beta$ of topic y). So if a word is 10 times more frequent in topic x the log ratio will be 1, whereas it will be -1 if the word is 10 times more frequent in topic y.

```
spooky_wrd_topics <- mutate(spooky_wrd_topics, topic = paste0("topic", topic))
spooky_wrd_topics <- spread(spooky_wrd_topics, topic, beta)

spooky_wrd_topics_12 <- filter(spooky_wrd_topics, topic2 > .001 | topic1 > .001)
spooky_wrd_topics_12 <- mutate(spooky_wrd_topics_12, log_ratio = log10(topic2 / topic1))
spooky_wrd_topics_12 <- group_by(spooky_wrd_topics_12, direction = log_ratio > 0)
spooky_wrd_topics_12 <- ungroup(top_n(spooky_wrd_topics_12, 5, abs(log_ratio)))
spooky_wrd_topics_12 <- mutate(spooky_wrd_topics_12, term = reorder(term, log_ratio))

p1 <- ggplot(spooky_wrd_topics_12) +
      geom_col(aes(term, log_ratio, fill = log_ratio > 0)) +
      theme(legend.position = "none") +
      labs(y = "Log ratio of beta in topic 2 / topic 1") +
      coord_flip()
```

```r
spooky_wrd_topics_13 <- filter(spooky_wrd_topics, topic3 > .001 | topic1 > .001)
spooky_wrd_topics_13 <- mutate(spooky_wrd_topics_13, log_ratio = log10(topic3 / topic1))
spooky_wrd_topics_13 <- group_by(spooky_wrd_topics_13, direction = log_ratio > 0)
spooky_wrd_topics_13 <- ungroup(top_n(spooky_wrd_topics_13, 5, abs(log_ratio)))
spooky_wrd_topics_13 <- mutate(spooky_wrd_topics_13, term = reorder(term, log_ratio))

p2 <- ggplot(spooky_wrd_topics_13) +
      geom_col(aes(term, log_ratio, fill = log_ratio > 0)) +
      theme(legend.position = "none") +
      labs(y = "Log ratio of beta in topic 3 / topic 1") +
      coord_flip()

spooky_wrd_topics_14 <- filter(spooky_wrd_topics, topic1 > .001 | topic4 > .001)
spooky_wrd_topics_14 <- mutate(spooky_wrd_topics_14, log_ratio = log10(topic4 / topic1))
spooky_wrd_topics_14 <- group_by(spooky_wrd_topics_14, direction = log_ratio > 0)
spooky_wrd_topics_14 <- ungroup(top_n(spooky_wrd_topics_14, 5, abs(log_ratio)))
spooky_wrd_topics_14 <- mutate(spooky_wrd_topics_14, term = reorder(term, log_ratio))

p3 <- ggplot(spooky_wrd_topics_14) +
      geom_col(aes(term, log_ratio, fill = log_ratio > 0)) +
      theme(legend.position = "none") +
      labs(y = "Log ratio of beta in topic 4 / topic 1") +
      coord_flip()

spooky_wrd_topics_23 <- filter(spooky_wrd_topics, topic2 > .001 | topic3 > .001)
spooky_wrd_topics_23 <- mutate(spooky_wrd_topics_23, log_ratio = log10(topic3 / topic2))
spooky_wrd_topics_23 <- group_by(spooky_wrd_topics_23, direction = log_ratio > 0)
spooky_wrd_topics_23 <- ungroup(top_n(spooky_wrd_topics_23, 5, abs(log_ratio)))
spooky_wrd_topics_23 <- mutate(spooky_wrd_topics_23, term = reorder(term, log_ratio))

p4 <- ggplot(spooky_wrd_topics_23) +
      geom_col(aes(term, log_ratio, fill = log_ratio > 0)) +
      theme(legend.position = "none") +
      labs(y = "Log ratio of beta in topic 3 / topic 2") +
      coord_flip()


spooky_wrd_topics_24 <- filter(spooky_wrd_topics, topic2 > .001 | topic4 > .001)
spooky_wrd_topics_24 <- mutate(spooky_wrd_topics_24, log_ratio = log10(topic4 / topic2))
spooky_wrd_topics_24 <- group_by(spooky_wrd_topics_24, direction = log_ratio > 0)
spooky_wrd_topics_24 <- ungroup(top_n(spooky_wrd_topics_24, 5, abs(log_ratio)))
spooky_wrd_topics_24 <- mutate(spooky_wrd_topics_24, term = reorder(term, log_ratio))

p5 <- ggplot(spooky_wrd_topics_24) +
      geom_col(aes(term, log_ratio, fill = log_ratio > 0)) +
      theme(legend.position = "none") +
      labs(y = "Log ratio of beta in topic 4 / topic 2") +
      coord_flip()

spooky_wrd_topics_34 <- filter(spooky_wrd_topics, topic3 > .001 | topic4 > .001)
spooky_wrd_topics_34 <- mutate(spooky_wrd_topics_34, log_ratio = log10(topic4 / topic3))
spooky_wrd_topics_34 <- group_by(spooky_wrd_topics_34, direction = log_ratio > 0)
spooky_wrd_topics_34 <- ungroup(top_n(spooky_wrd_topics_34, 5, abs(log_ratio)))
```

```r
spooky_wrd_topics_34 <- mutate(spooky_wrd_topics_34, term = reorder(term, log_ratio))

p6 <- ggplot(spooky_wrd_topics_34) +
      geom_col(aes(term, log_ratio, fill = log_ratio > 0)) +
      theme(legend.position = "none") +
      labs(y = "Log ratio of beta in topic 4 / topic 3") +
      coord_flip()

layout <- matrix(c(1,2,3,4,5,6),3,2, byrow = TRUE)
multiplot(p1, p2, p3,p4, p5,p6, layout = layout)
```



In the above, the words more common to topic 2 than topic 1 are "moon", "air", and "window" while the words more common to topic 1 are "moment", "marie", and "held".

Sentence Topics

```r
spooky_wrd_docs <- tidy(spooky_wrd_lda, matrix = "gamma")
spooky_wrd_docs
```

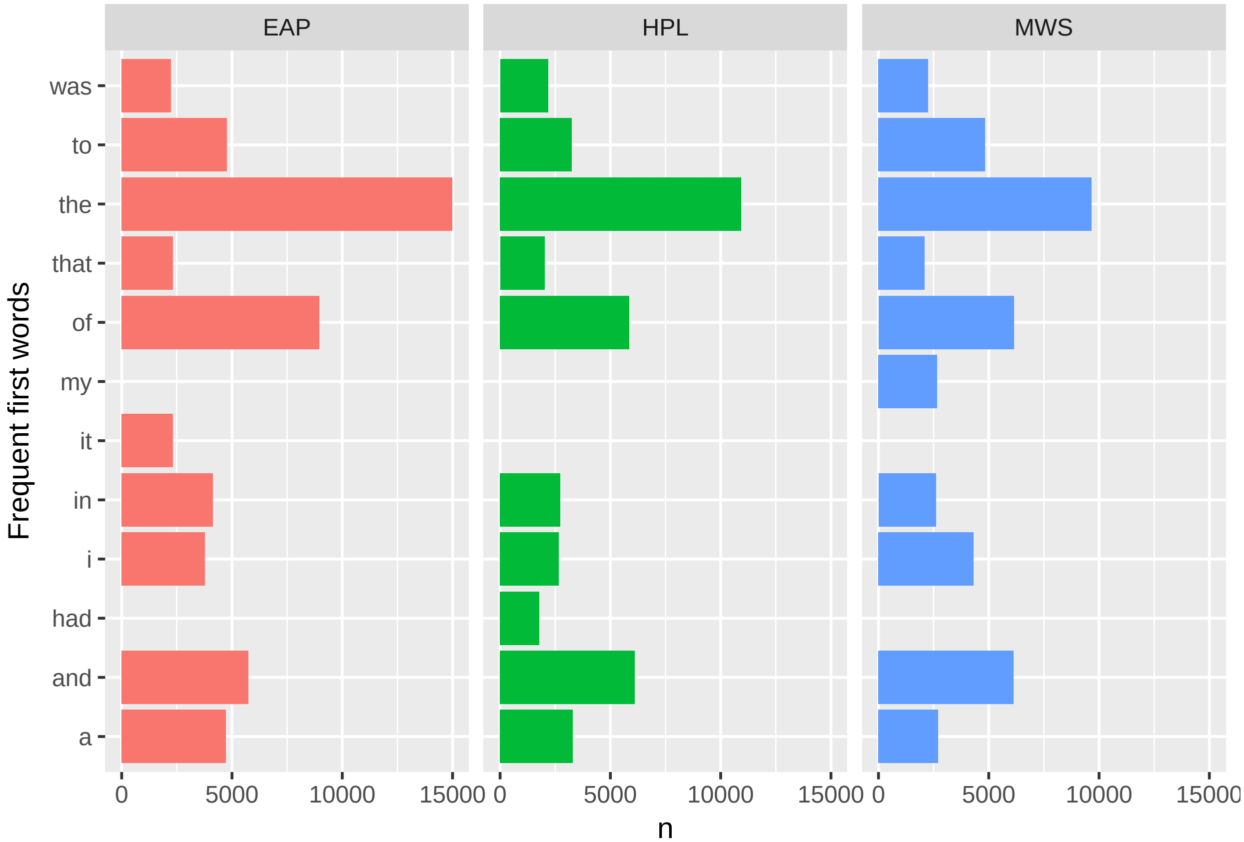
```
## # A tibble: 194,670 x 3
##    document topic  gamma
##    <chr>    <int>  <dbl>
##  1 id00001      1 0.1000
##  2 id00002      1 0.100
##  3 id00003      1 0.0997
##  4 id00004      1 0.0992
```

```
##   5 id00005       1 0.101
##   6 id00006       1 0.101
##   7 id00007       1 0.100
##   8 id00009       1 0.0955
##   9 id00010       1 0.0996
## 10 id00012       1 0.0993
## # ... with 194,660 more rows
```

first and last words

we now plot the top first words for the different authors. Here the barplot scales are fixed between the facets so that the frequencies of each word can be compared for the three authors. If a bar is missing in one facet then that means that its frequency was lower than the shortest bar in this facet:

```r
first <- train %>%
  unnest_tokens(word, text) %>%
  rownames_to_column() %>%
  mutate(rowname = as.integer(rowname)) %>%
  group_by(id) %>%
  top_n(-1, rowname) %>%
  ungroup()

last <- train %>%
  unnest_tokens(word, text) %>%
  rownames_to_column() %>%
  mutate(rowname = as.integer(rowname)) %>%
  group_by(id) %>%
  top_n(1, rowname) %>%
  ungroup()
first %>%
  group_by(author, word) %>%
  count() %>%
  ungroup() %>%
  group_by(author) %>%
  top_n(10,n) %>%
  ggplot(aes(word, n, fill = author)) +
  geom_col() +
  theme(legend.position = "none") +
  labs(x = "Frequent first words") +
  coord_flip() +
  facet_wrap(~ author)
```



We find:

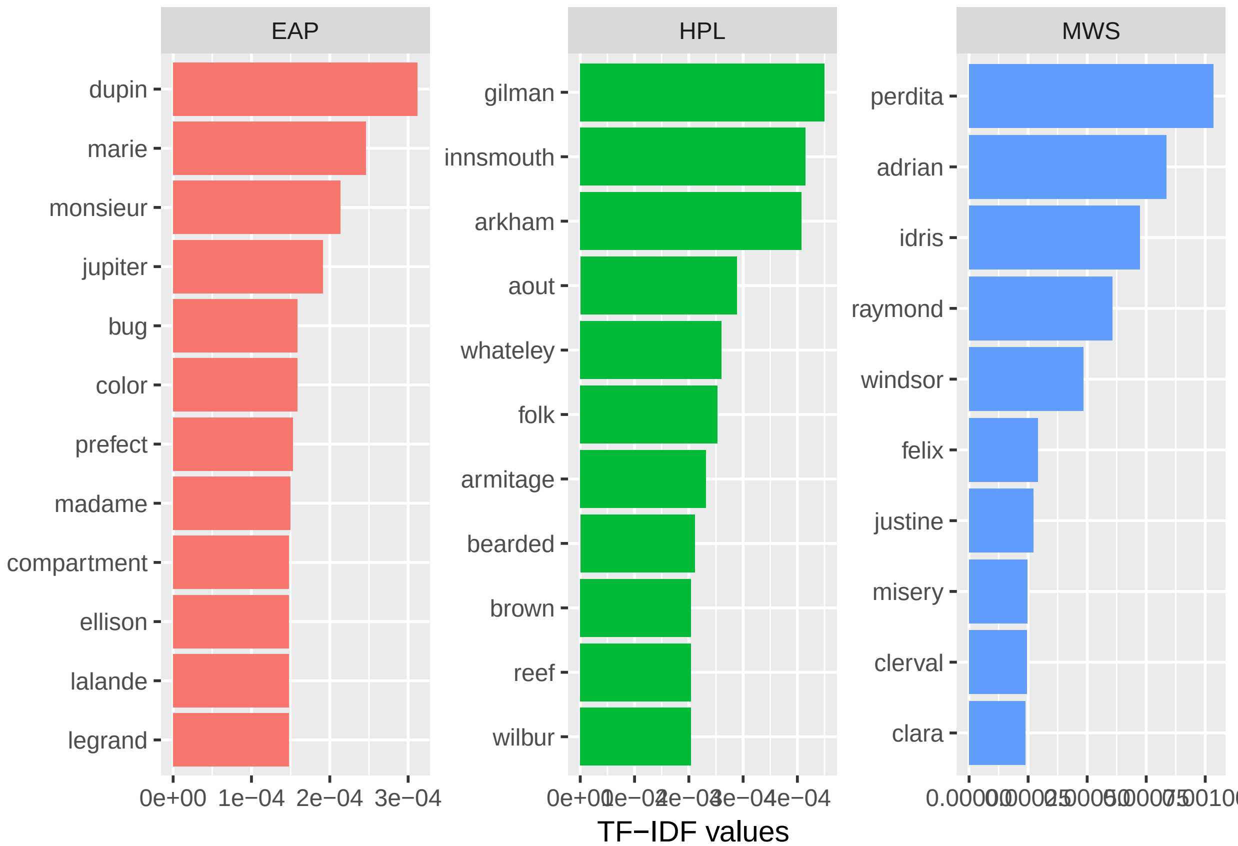
The fact that Edgar Allan Poe really likes to start sentences with ???the???. I also think that he and Mary Shelley like to start out with ???I???. Those are the two most frequent words for HP Lovecraft, too; but he doesn???t use them nearly as often as the other two authors.

Lovecraft does not like using ???this??? and ???we??? at the beginning of a sentence quite as much as the others do. In general, his distribution is flatter, suggesting a higher diversity in opening words.

We could then extract the TF-IDF informatin for the first and last word data:

```
tf_idf_first <- first %>%
  count(author, word) %>%
  bind_tf_idf(word, author, n)

tf_idf_last <- last %>%
  count(author, word) %>%
  bind_tf_idf(word, author, n)
tf_idf_first %>%
  arrange(desc(tf_idf)) %>%
  mutate(word = factor(word, levels = rev(unique(word)))) %>%
  group_by(author) %>%
  top_n(10, tf_idf) %>%
  ungroup() %>%
  ggplot(aes(word, tf_idf, fill = author)) +
  geom_col() +
  labs(x = NULL, y = "tf-idf") +
  theme(legend.position = "none") +
  facet_wrap(~ author, ncol = 3, scales = "free") +
```

```r
  coord_flip() +
  labs(y = "TF-IDF values")
```



We find:

The openings ???later???, ???old???, and ???instead??? are tell-tale signs of Lovecraft???s works. Poe likes to use ???meantime??? and ???hereupon???, whereas Shelley prefers ???alas???. Specific character names also make an appearance here.

The presence of the word ???chapter??? in Shelley???s sentences might indicate that those still contain some structuring work that we don???t have in the other author???s samples.

```r
train %>%
  filter(str_sub(text, start = 1, end = 7) == "Chapter") %>%
  mutate(sample = str_sub(text, start = 1, end = 60)) %>%
  select(-text, -id) %>%
  head(5)
```

```
##   author                                                       sample
## 1    MWS Chapter Day after day, week after week, passed away on my re
## 2    MWS Chapter I sat one evening in my laboratory; the sun had set,
## 3    MWS Chapter My present situation was one in which all voluntary
## 4    MWS        Chapter "Such was the history of my beloved cottagers.
## 5    MWS Chapter On my return, I found the following letter from my f
```

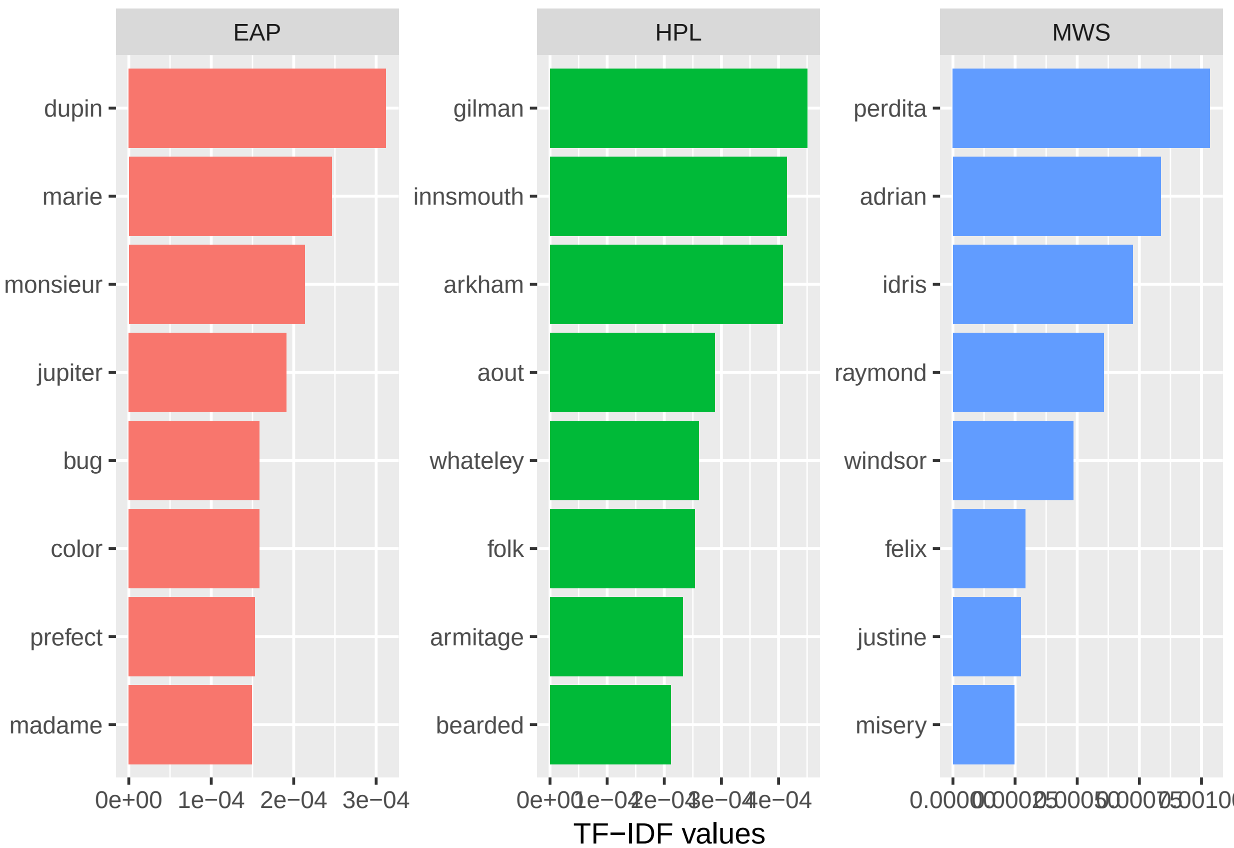
contain some structuring work that we don???t have in the other author???s samples.

```r
train %>%
  filter(str_sub(text, start = 1, end = 2) == "P.") %>%
  mutate(sample = str_sub(text, start = 1, end = 60)) %>%
```

```
  select(-text, -id) %>%
  head(5)
```

```
##   author                                                       sample
## 1    EAP                                       P. I do not comprehend.
## 2    EAP P. Can you give me no more precise idea of what you term the
## 3    EAP                                     P. Is not God immaterial?
## 4    EAP                                     P. What then shall I ask?
## 5    EAP             P. I wish you would explain yourself, Mr. Vankirk.
```

```
tf_idf_last %>%
  arrange(desc(tf_idf)) %>%
  mutate(word = factor(word, levels = rev(unique(word)))) %>%
  group_by(author) %>%
  top_n(8, tf_idf) %>%
  ungroup() %>%
  ggplot(aes(word, tf_idf, fill = author)) +
  geom_col() +
  labs(x = NULL, y = "tf-idf") +
  theme(legend.position = "none") +
  facet_wrap(~ author, ncol = 3, scales = "free") +
  coord_flip() +
  labs(y = "TF-IDF values")
```



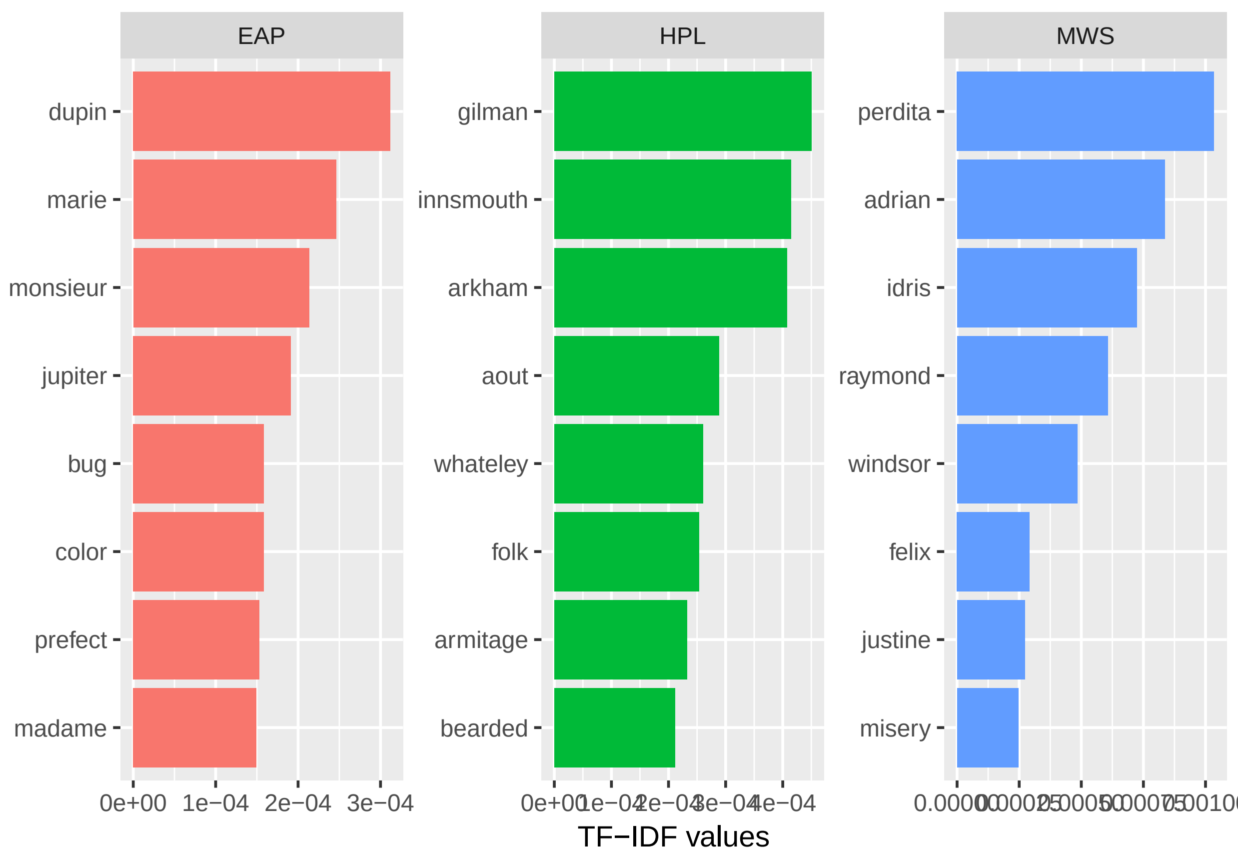
We find:

Beyond the specific place and character names (such as ???Raymond??? or ???Arkham???) there are a few interesting final words for each author: Poe has ???altogether???, ???minute???, ???antagonist??? and also

???machine??? which fits into his technical vocabulary.

Lovecraft???s sentences most characteristingly end on ???moonlight???, ???region???, or ???thing???. And for Shelley it???s a (Halloween-themed) roller coaster of emotions: ???misery???, ???love???, ???wretchedness???, and ???sympathy???. This is very consistent with the overall emphasis she puts on the microcosm of feelings that express the fears and struggles of her protagonists. ## Gender balance of the texts we could plot the difference in the frequencies of the words ???woman??? vs ???man???, ???she??? vs ???he???, ???her??? vs ???him???, and general gender indicators for the 3 authors:

```r
tf_idf_last %>%
  arrange(desc(tf_idf)) %>%
  mutate(word = factor(word, levels = rev(unique(word)))) %>%
  group_by(author) %>%
  top_n(8, tf_idf) %>%
  ungroup() %>%
  ggplot(aes(word, tf_idf, fill = author)) +
  geom_col() +
  labs(x = NULL, y = "tf-idf") +
  theme(legend.position = "none") +
  facet_wrap(~ author, ncol = 3, scales = "free") +
  coord_flip() +
  labs(y = "TF-IDF values")
```



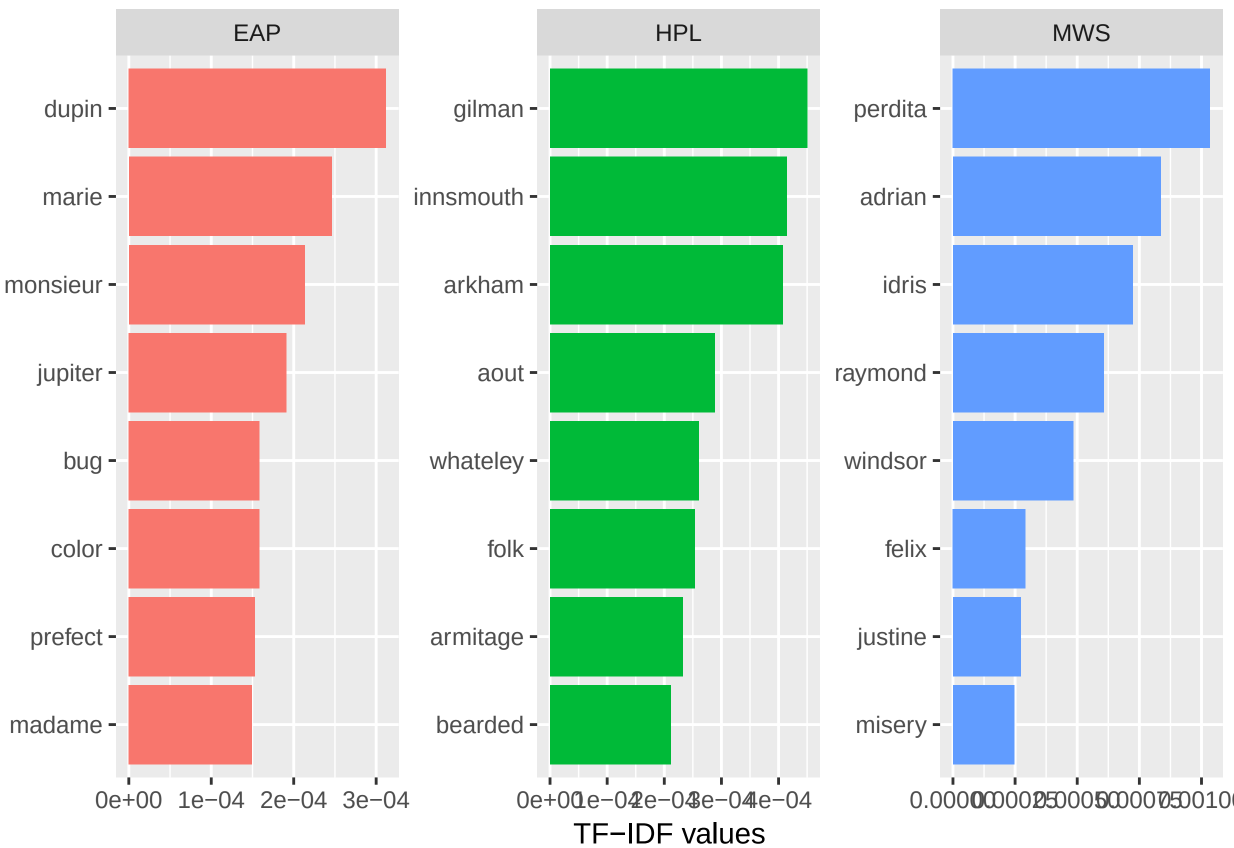
We find:

Not much femininity in Lovecraft. Interestingly, he has (just about) the most mentions of ???woman??? but not much of them seem to be doing something.

The dominance of ???her??? over ???him??? in Shelley???s (and Poe???s) work is remarkable. Of course, ???her??? can be the counterpart of ???him??? as well as ???his???, but I think that this alone can???t explain the full effect.

Perhaps unsurprisingly, Mary Shelley uses far more ???female??? words than her two male counterparts. The clear differences here between Poe and Lovecraft show a promising amount of distinguishing power.

```
tf_idf_last %>%
  arrange(desc(tf_idf)) %>%
  mutate(word = factor(word, levels = rev(unique(word)))) %>%
  group_by(author) %>%
  top_n(8, tf_idf) %>%
  ungroup() %>%
  ggplot(aes(word, tf_idf, fill = author)) +
  geom_col() +
  labs(x = NULL, y = "tf-idf") +
  theme(legend.position = "none") +
  facet_wrap(~ author, ncol = 3, scales = "free") +
  coord_flip() +
  labs(y = "TF-IDF values")
```



We find:

The overall numbers reflect what we had seen above for the gender balance; i.e. Lovecraft rarely using ???she???. The fact that his facet for the word ???she??? contains more bars than for the others is simply due to all the short bars sharing a rank with 2 occurences.

Men (or male entities) in Lovecraft???s works appear to be more associated with ???did??? and ???could??? rather than ???is??? and ???has???, as we find it for Poe. Interestingly, ???did??? and ???could??? are also the terms more frequently following the word ???she??? in Shelley???s work, while Poe???s top 5 includes ???must???.