Section 13: Text Mining

MTH 365: Introduction to Data Science

November 22, 2021

Recommended Reading

- *Tidy Text Mining with R*: Julia Silge and David Robinson (https://www.tidytextmining.com/)

```
library(tidyverse)
library(mdsr)
library(RColorBrewer)
library(tidytext)
```

Jeopardy!

Example: Reddit user u/PandaPython1 posted web scraped data from the Jeopardy Archive (http://www.jarchive.com) containing the questions, categories, answers, clue locations, and earnings from all Jeopardy shows through the 2017 season. The data is saved as `Jeopardy2017.csv`.

```
Jeopardy2017 <- read.csv("Jeopardy2017.csv")
names(Jeopardy2017)
```

```
##  [1] "index_number"   "Unnamed..0"     "index_number.1" "show"
##  [5] "show_date"      "contestant1"    "contestant2"    "contestant3"
##  [9] "clue_location"  "x_cord"         "y_cord"         "clue_number"
## [13] "clue_value"     "categories"     "question"       "answer"
## [17] "q_type"         "daily_double"   "cont1_response" "cont2_response"
## [21] "cont3_response" "earnings1"      "earnings2"      "earnings3"
```

If you're a future Jeopardy contestant, you might want to limit your studying:

1. What categories are most common?
2. What types of answers are most common?

```
Jeopardy2017 %>% summarize(n=n_distinct(categories))
```

```
##       n
## 1 38823
```

Options:

1. Review all 38,823 categories for classification manually
2. Use text mining!

Text mining: the process of deriving high-quality information from text.

Most common categories and answers

What are the most common Jeopardy categories? Answers?

```
category_df <- tibble(question=1:nrow(Jeopardy2017),
          category=as.character(Jeopardy2017$categories))
tidy_category <- category_df %>%
  unnest_tokens(word, category)
tidy_category %>% group_by(word) %>%
  summarize(n=n())
```

```
## # A tibble: 14,984 x 2
##    word            n
##    <chr>       <int>
##  1 __             10
##  2 ___            80
##  3 ____          408
##  4 _____         168
##  5 ______          4
##  6 _______        15
##  7 ______e         2
##  8 _____in         5
##  9 _____ing        5
## 10 ____for____     5
## # ... with 14,974 more rows
```

```
tidy_category %>% group_by(word) %>%
  summarize(n=n()) %>%
  arrange(desc(n))
```

```
## # A tibble: 14,984 x 2
##    word        n
##    <chr>   <int>
##  1 the     47670
##  2 of      13228
##  3 a       12837
##  4 in      11497
##  5 words    6780
##  6 world    6031
##  7 history  5869
##  8 to       5126
##  9 on       4936
## 10 tv       4706
## # ... with 14,974 more rows
```

How do we remove the "nuisance" words?

Stop words

Stop words: usually means the most common, short function words, such as the, is, at, which, and on. Those words does not have a meaningful concept in the text thus we usually will avoid them in the text mining.

```r
data(stop_words)
tidy_category <- tidy_category %>% anti_join(stop_words)
```

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

- What do you think `anti_join()` does?

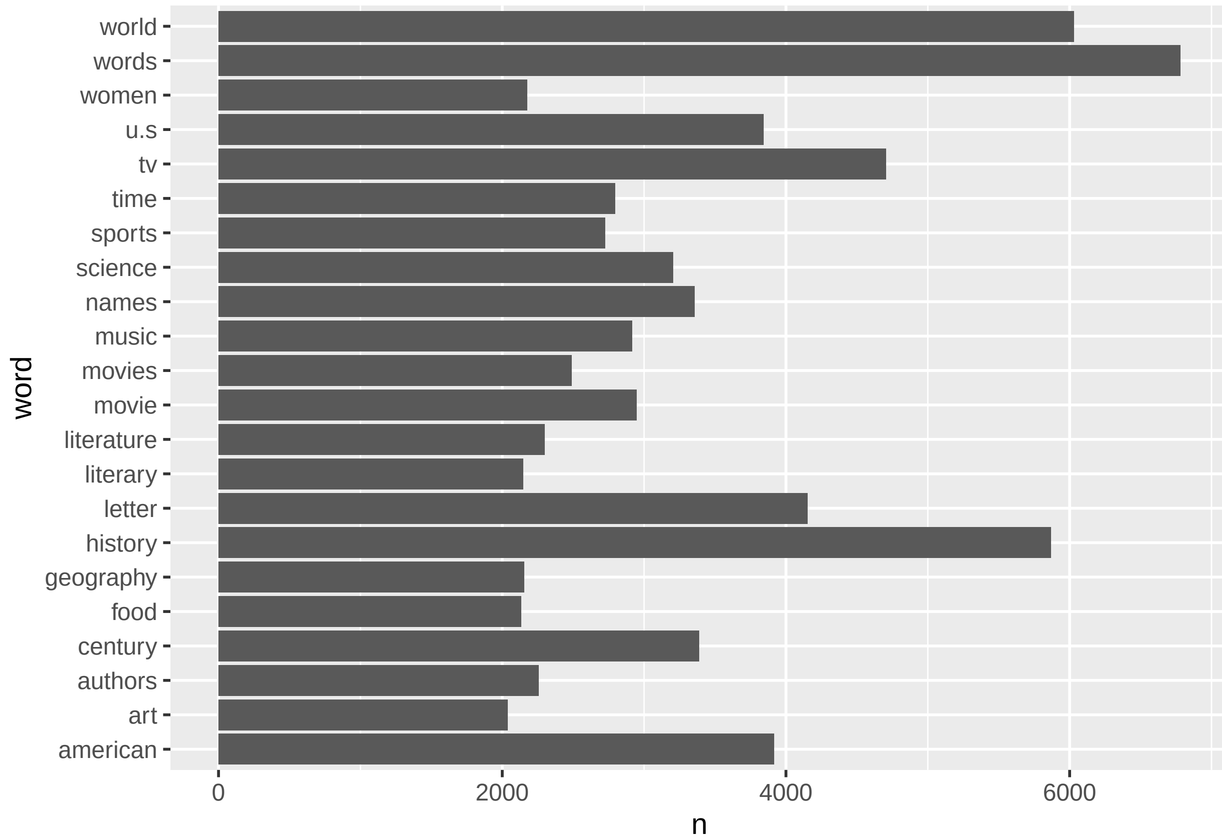
  Answer:

```r
tidy_category %>% group_by(word) %>%
  summarize(n=n()) %>%
  arrange(desc(n))
```

```
## # A tibble: 14,428 x 2
##    word         n
##    <chr>    <int>
##  1 words     6780
##  2 world     6031
##  3 history   5869
##  4 tv        4706
##  5 letter    4153
##  6 american  3916
##  7 u.s       3843
##  8 century   3387
##  9 names     3357
## 10 science   3206
## # ... with 14,418 more rows
```

Now, back to the most common categories.

```r
top_categories <- tidy_category %>% group_by(word) %>%
  summarize(n=n()) %>%
  filter(n>=2000)
top_categories %>% ggplot(aes(x=word, y=n)) +
  geom_col() + coord_flip()
```



Most Jeopardy categories aren't just one word.

Bigram: a pair of consecutive written units such as letters, syllables, or words.

```
category_bigrams <- category_df %>%
  unnest_tokens(bigram, category, token = "ngrams", n = 2)
category_bigrams %>% group_by(bigram) %>%
  summarize(n=n()) %>% arrange(desc(n))
```

```
## # A tibble: 44,211 x 2
##    bigram              n
##    <chr>           <int>
##  1 <NA>            61555
##  2 of the           3869
##  3 in the           3393
##  4 letter words     2375
##  5 on the           1841
##  6 the world        1343
##  7 crossword clues  1341
##  8 20th century     1270
##  9 rhyme time       1135
## 10 19th century      996
## # ... with 44,201 more rows
```

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

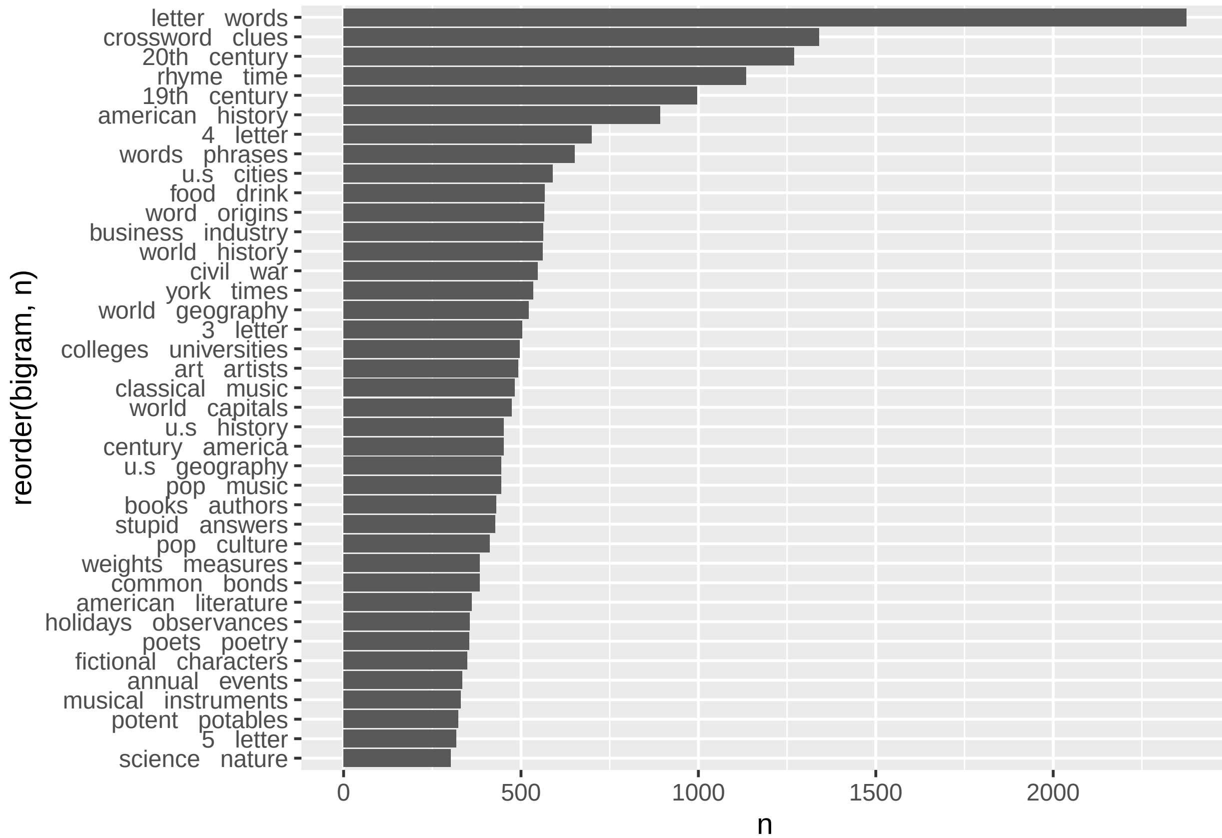
```r
bigrams_filtered <- bigrams_separated %>%
  filter(!word1 %in% stop_words$word) %>%
  filter(!word2 %in% stop_words$word)

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

```r
bigram_counts
```

```
## # A tibble: 20,138 x 3
##    word1     word2       n
##    <chr>     <chr>   <int>
##  1 letter    words    2375
##  2 crossword clues    1341
##  3 20th      century  1270
##  4 rhyme     time     1135
##  5 19th      century   996
##  6 american  history   892
##  7 4         letter    700
##  8 words     phrases   651
##  9 u.s       cities    589
## 10 food      drink     567
## # ... with 20,128 more rows
```

```r
top_bigrams <- bigram_counts %>% filter(n>=300) %>% mutate(bigram = paste(word1, ' ', word2))
top_bigrams %>% ggplot(aes(x=reorder(bigram, n), y=n)) + geom_col() + coord_flip()
```



What are the most common single-word question categories?

```
Jeopardy2017 %>% summarize(n=n_distinct(categories))
```

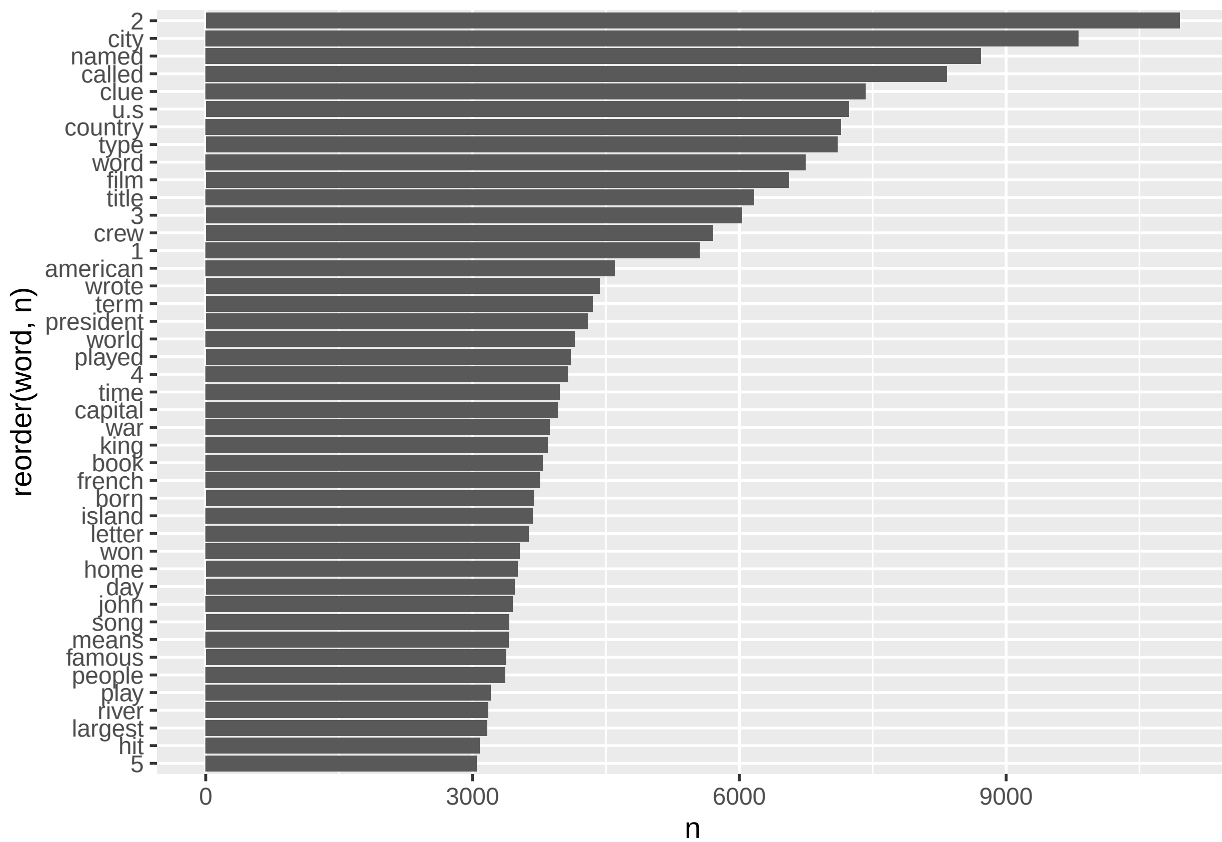
```
##       n
## 1 38823
```

```
question_df <- tibble(number=1:nrow(Jeopardy2017),
                      question=as.character(Jeopardy2017$question))
tidy_question <- question_df %>% unnest_tokens(word, question)

tidy_question <- tidy_question %>% anti_join(stop_words)
```

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

```
top_questions <- tidy_question %>% group_by(word) %>% summarize(n=n()) %>% filter(n>=3000)
top_questions %>% ggplot(aes(x=reorder(word, n), y=n)) + geom_col() + coord_flip()
```



Based on bigrams, what are the most common categories?

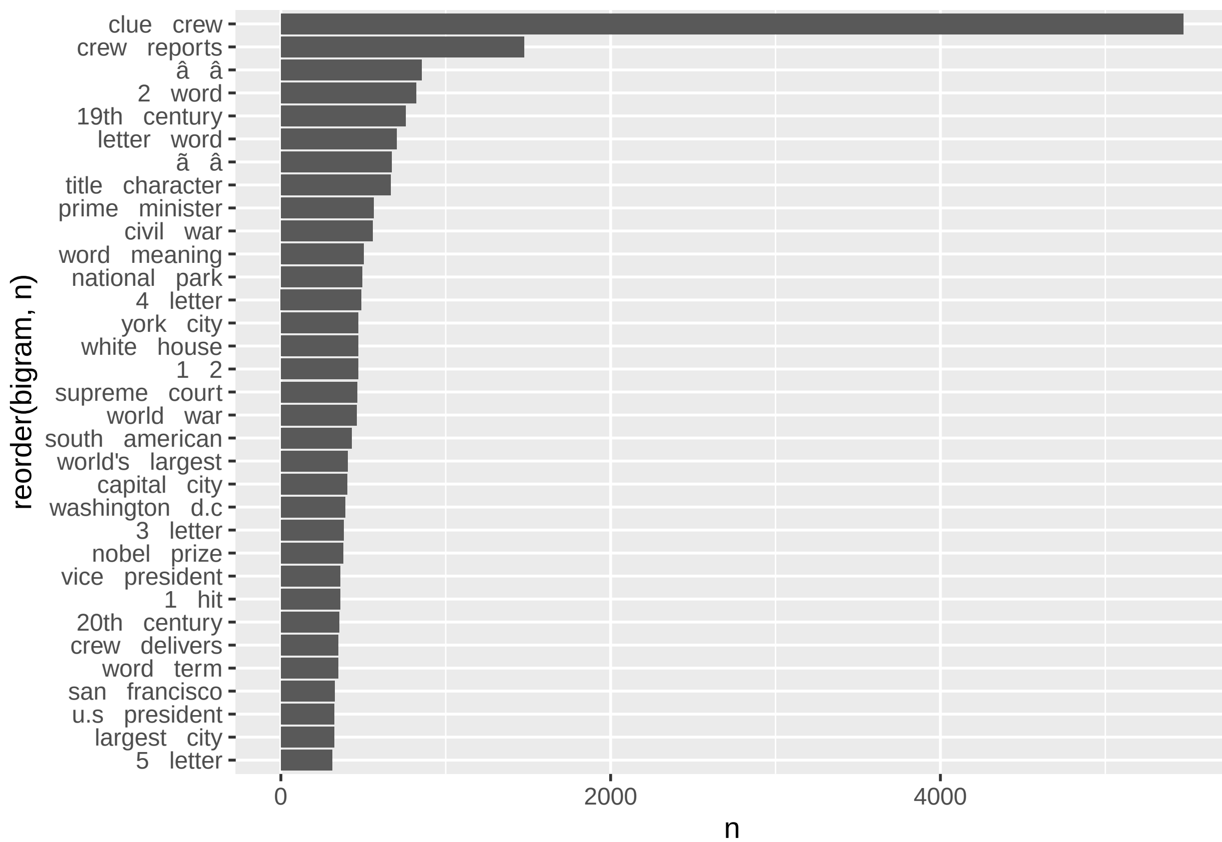
```
question_bigrams <- question_df %>%
  unnest_tokens(bigram, question, token = "ngrams", n = 2)

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

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

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

top_bigrams <- bigram_counts %>% filter(n>=300) %>% mutate(bigram = paste(word1, ' ', word2))
top_bigrams %>% ggplot(aes(x=reorder(bigram, n), y=n)) + geom_col() + coord_flip()
```



Most common answers

Use text mining to investigate the most common Jeopardy answers. If you're planning an appearance on Jeopardy, what should you brush up on?