ST720 Data Science

tf-idf

Seung Jun Shin (sjshin@krea.ac.kr)

Department of Statistics, Korea University

Introduction

- How to quantify what a document is about?
- Can we do this by looking at the words?
- One measure of how important a word is its term frequency (tf).
- There are words in a document occur many times but not important.
- Some eleminated words are more important in some documents than others.
- One way is to use a term's inverse document frequency (idf).

$$\text{idf(term)} = \log\left(\frac{n_{\text{documments}}}{n_{\text{documments containing term}}}\right)$$

- tf-idf (the two quantities multiplied together) is the frequency of a term adjusted for how rarely it is used.

Term frequency in Jane Austen's novels

- What are the most commonly used words in Jane Austen's novels?

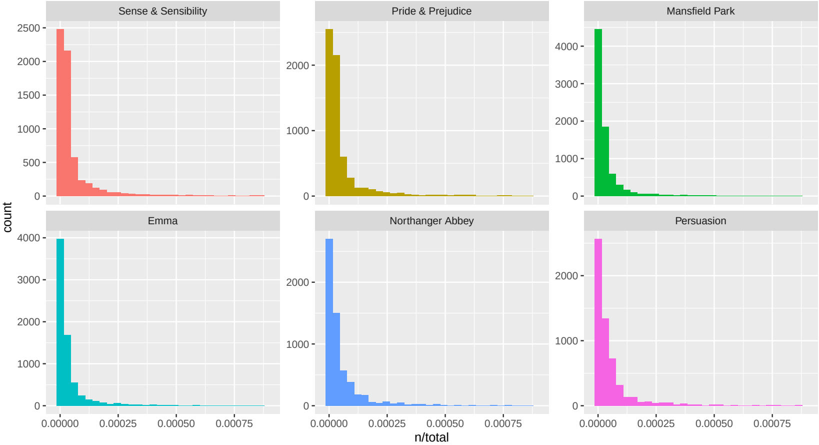
```
book_words <- austen_books() %>%
  unnest_tokens(word, text) %>%
  count(book, word, sort = TRUE)
total_words <- book_words %>%
  group_by(book) %>%
  summarize(total = sum(n))
book_words <- left_join(book_words, total_words)
print(book_words, n = 4)
```

```
## # A tibble: 40,379 x 4
##   book              word      n  total
##   <fct>             <chr> <int>  <int>
## 1 Mansfield Park the    6206 160460
## 2 Mansfield Park to     5475 160460
## 3 Mansfield Park and    5438 160460
## 4 Emma              to     5239 160996
## # ... with 4.038e+04 more rows
```

Term frequency in Jane Austen's novels

- Distribution of `n/total` for each novel.

```
ggplot(book_words, aes(n/total, fill = book)) +
  geom_histogram(show.legend = FALSE) +
  xlim(NA, 0.0009) +
  facet_wrap(~book, ncol = 3, scales = "free_y")
```



Zipf's Law

- Distributions with a long tale is typical in language.
- Zipf's Law states

$$\text{tf} \propto \frac{1}{\text{rank}}$$

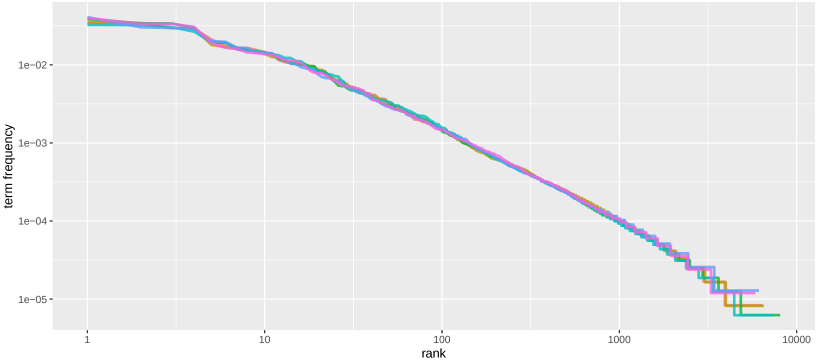
```
freq_by_rank <- book_words %>%
  group_by(book) %>%
  mutate(rank = row_number(),
         `term frequency` = n/total)
print(freq_by_rank, n = 3)
```

```
## # A tibble: 40,379 x 6
## # Groups:   book [6]
##   book              word      n  total  rank `term frequency`
##   <fct>             <chr> <int>  <int> <int>              <dbl>
## 1 Mansfield Park the    6206 160460     1             0.0387
## 2 Mansfield Park to     5475 160460     2             0.0341
## 3 Mansfield Park and    5438 160460     3             0.0339
## # ... with 4.038e+04 more rows
```

Zipf's Law

- Zipf's law is often visualized in log-scale.

```
freq_by_rank %>%
  ggplot(aes(rank, `term frequency`, color = book)) +
  geom_line(size = 1.1, alpha = 0.8, show.legend = FALSE) +
  scale_x_log10() + scale_y_log10()
```



The bind_tf_idf function

- tf-idf finds the words that are important (i.e., common) in a text, but not too common.

```
book_words <- book_words %>%
  bind_tf_idf(word, book, n)
print(book_words, n = 5)
```

```
## # A tibble: 40,379 x 7
##   book              word      n  total     tf   idf tf_idf
##   <fct>             <chr> <int>  <int>  <dbl> <dbl>  <dbl>
## 1 Mansfield Park the     6206 160460 0.0387     0      0
## 2 Mansfield Park to      5475 160460 0.0341     0      0
## 3 Mansfield Park and     5438 160460 0.0339     0      0
## 4 Emma           to      5239 160996 0.0325     0      0
## 5 Emma           the     5201 160996 0.0323     0      0
## # ... with 4.037e+04 more rows
```

- tf-idf are zero for extremely common words.

The bind_tf_idf function

- Let's look at terms with high tf-idf in Jane Austen's works.

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

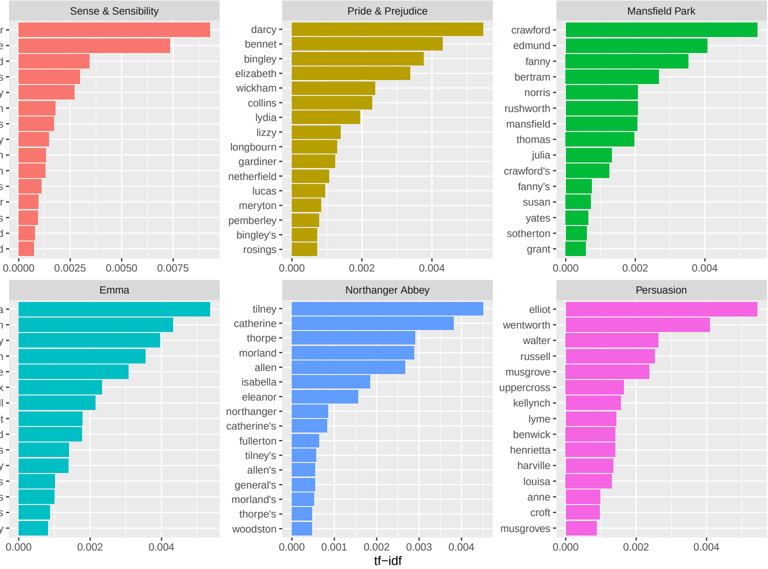
```
## # A tibble: 40,379 x 6
##    book                word          n      tf   idf  tf_idf
##    <fct>               <chr>     <int>   <dbl> <dbl>   <dbl>
##  1 Sense & Sensibility elinor      623 0.00519  1.79 0.00931
##  2 Sense & Sensibility marianne    492 0.00410  1.79 0.00735
##  3 Mansfield Park      crawford    493 0.00307  1.79 0.00551
##  4 Pride & Prejudice   darcy       373 0.00305  1.79 0.00547
##  5 Persuasion          elliot      254 0.00304  1.79 0.00544
##  6 Emma                emma        786 0.00488  1.10 0.00536
##  7 Northanger Abbey    tilney      196 0.00252  1.79 0.00452
##  8 Emma                weston      389 0.00242  1.79 0.00433
##  9 Pride & Prejudice   bennet      294 0.00241  1.79 0.00431
## 10 Persuasion          wentworth   191 0.00228  1.79 0.00409
## # ... with 40,369 more rows
```

The bind_tf_idf function

```r
book_words %>%
  arrange(desc(tf_idf)) %>%
  mutate(word = factor(word, levels = rev(unique(word)))) %>%
  group_by(book) %>%
  top_n(15) %>%
  ungroup() %>%
  ggplot(aes(word, tf_idf, fill = book)) +
  geom_col(show.legend = FALSE) +
  labs(x = NULL, y = "tf-idf") +
  facet_wrap(~book, ncol = 2, scales = "free") +
  coord_flip()
```

The `bind_tf_idf` function

- Let's visualize!



A corpus of physics texts

- Let's download some classic physics texts from Project Gutenberg and see what terms are important in these works, as measured by tf-idf.
  - Discourse on Floating Bodies by Galileo Galilei
  - Treatise on Light by Christiaan Huygens
  - Experiments with Alternate Currents of High Potential and High Frequency by Nikola Tesla
  - Relativity: The Special and General Theory by Albert Einstein.

```r
physics <- gutenberg_download(c(37729, 14725, 13476, 30155),
                              meta_fields = "author")
```

Raw counts

```
physics_words <- physics %>%
  unnest_tokens(word, text) %>%
  count(author, word, sort = TRUE)

print(physics_words, n = 5)
```

```
## # A tibble: 12,671 x 3
##   author                word      n
##   <chr>                 <chr> <int>
## 1 Galilei, Galileo      the    3760
## 2 Tesla, Nikola         the    3604
## 3 Huygens, Christiaan the    3553
## 4 Einstein, Albert      the    2993
## 5 Galilei, Galileo      of     2049
## # ... with 1.267e+04 more rows
```

tf-idf

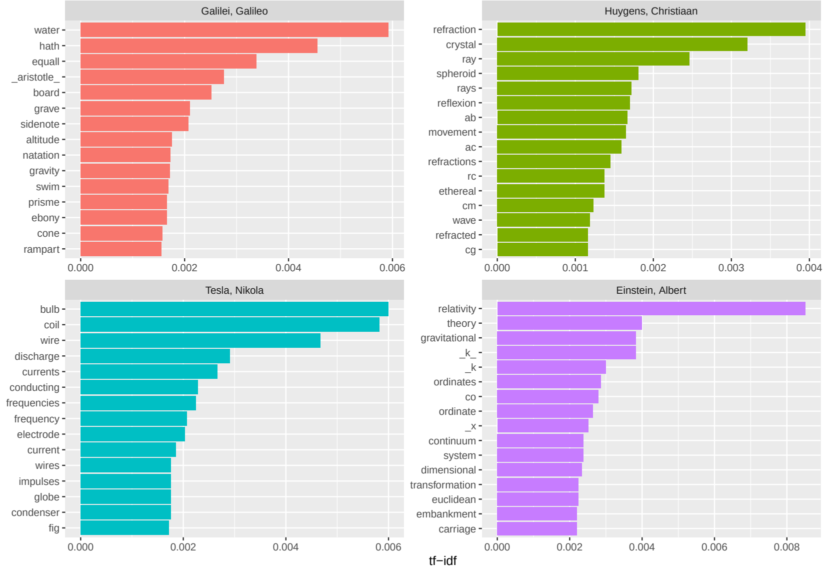
```
plot_physics <- physics_words %>%
  bind_tf_idf(word, author, n) %>%
  mutate(word = fct_reorder(word, tf_idf)) %>%
  mutate(author = factor(author, levels = c("Galilei, Galileo",
                                            "Huygens, Christiaan
                                            "Tesla, Nikola",
                                            "Einstein, Albert"))

print(plot_physics, n = 5)
```

```
## # A tibble: 12,671 x 6
##   author              word      n     tf   idf tf_idf
##   <fct>               <fct> <int>  <dbl> <dbl>  <dbl>
## 1 Galilei, Galileo    the    3760 0.0935     0      0
## 2 Tesla, Nikola       the    3604 0.0913     0      0
## 3 Huygens, Christiaan the    3553 0.0928     0      0
## 4 Einstein, Albert    the    2993 0.0952     0      0
## 5 Galilei, Galileo    of     2049 0.0510     0      0
## # ... with 1.267e+04 more rows
```

tf-idf

```r
plot_physics %>%
  group_by(author) %>%
  top_n(15, tf_idf) %>%
  ungroup() %>%
  mutate(word = reorder(word, tf_idf)) %>%
  ggplot(aes(word, tf_idf, fill = author)) +
  geom_col(show.legend = FALSE) +
  labs(x = NULL, y = "tf-idf") +
  facet_wrap(~author, ncol = 2, scales = "free") +
  coord_flip()
```

tf-idf



Galilei, Galileo
water
hath
equall
aristotle
board
grave
sidenote
altitude
natation
gravity
swim
prisme
ebony
cone
rampart
0.000
0.002
0.004
0.006
Huygens, Christiaan
refraction
crystal
ray
spheroid
rays
reflexion
ab
movement
ac
refractions
rc
ethereal
cm
wave
refracted
cg
0.000
0.001
0.002
0.003
0.004
Tesla, Nikola
bulb
coil
wire
discharge
currents
conducting
frequencies
frequency
electrode
current
wires
impulses
globe
condenser
fig
0.000
0.002
0.004
0.006
Einstein, Albert
relativity
theory
gravitational
_k_
_k
ordinates
co
ordinate
_x
continuum
system
dimensional
transformation
euclidean
embankment
carriage
0.000
0.002
0.004
0.006
0.008
tf−idf

tf-idf matrix

```
plot_physics %>% select(author, word, tf_idf) %>%
                 dcast(word ~ author) %>% as_tibble() %>%
      rename(GG = "Galilei, Galileo", HC = "Huygens, Christiaan"
             TN = "Tesla, Nikola",    EA = "Einstein, Albert")
```

```
## Using tf_idf as value column: use value.var to override.
```

```
## # A tibble: 8,068 x 5
##    word      GG    HC    TN    EA
##    <fct> <dbl> <dbl> <dbl> <dbl>
##  1 1         0     0     0     0
##  2 10        0     0     0     0
##  3 11        0     0     0     0
##  4 12        0     0     0     0
##  5 13        0     0     0     0
##  6 14        0     0     0     0
##  7 15        0     0     0     0
##  8 16        0     0     0     0
##  9 17        0     0     0     0
## 10 18        0     0     0     0
## # ... with 8,058 more rows
```

Summary

- Using term frequency and inverse document frequency allows us to find words that are characteristic for one document in corpus.
- Exploring term frequency gives us insight into how language is used in a collection of natural language, and `dplyr` verbs like `count()` and `rank()` give us tools to reason about term frequency.