Workshop #11

Tidying text data with tidytext

Introduction to Data Science
November 2022

Júlia Cots Capell & Viktorija Ruzelyte

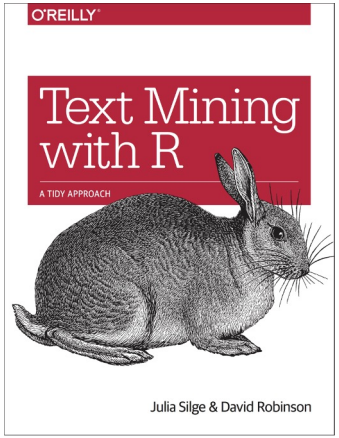
Overview

1. Tidytext: key facts
2. Tidyverse universe
3. Tidy format (tokens)
4. Functions of the tidytext package (unnest_tokens, get_sentiments)
5. Applying functions from other packages (dplyr, ggplot, wordcloud)
6. Correlation of word-usage between books
7. Conclusion
8. Further exercises

What is tidytext?

Julia Silge

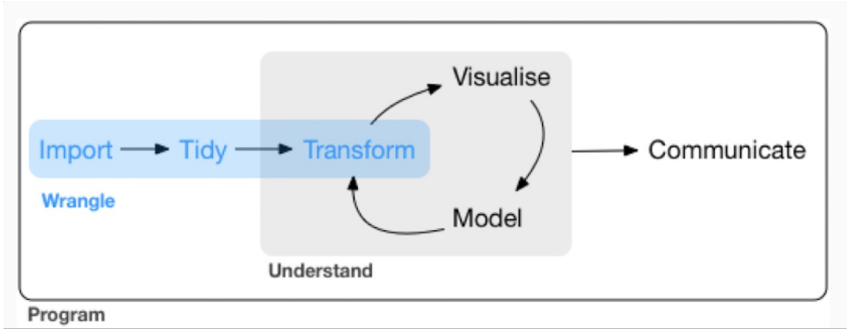
David Robinson



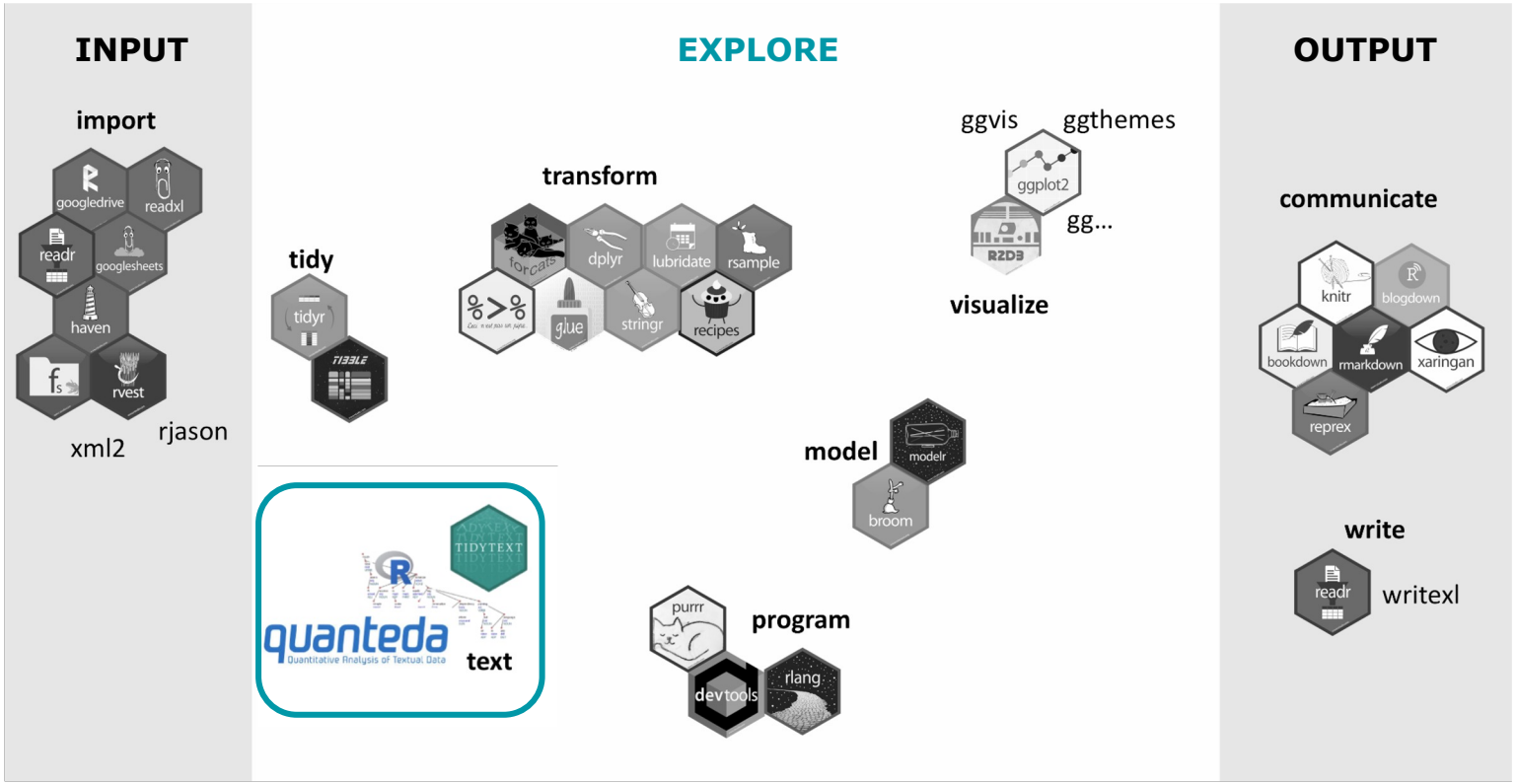
- **2016**.
- Observed **difficulty** to apply methods for data wrangling and visualisations to text.
- Tidytext provides a way to convert textual data into a "**tidy format**" which makes it better to apply the existing tools.
- It requires using "**tidy data principles**", by turning large datasets into data frames of text that can be easily manipulated, summarised and visualised.

Where are we in the tidyverse universe?

Making data "**tidy**"



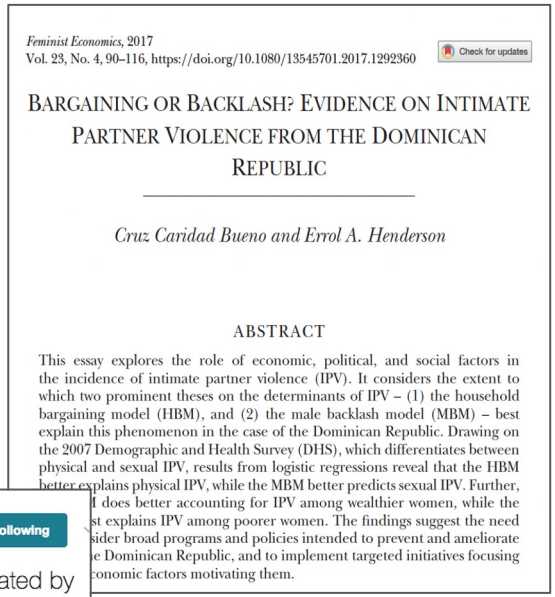
Where are we in the tidyverse universe?



The tidy format

- Textual data is written information understood as complete **blocks of text**, such as sentences, or paragraphs.

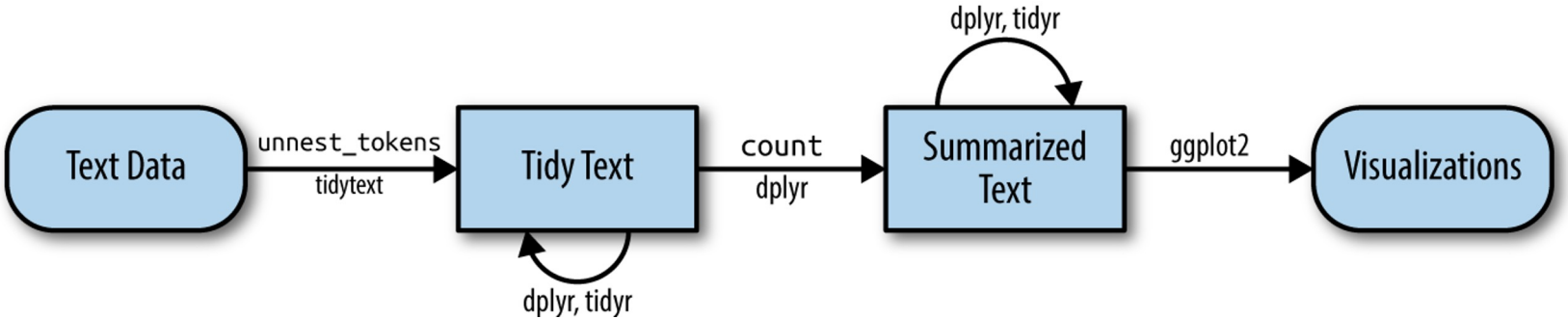
  For example...



The tidy format

- The tidy format asks us to place:
  - Each variable in a column
  - Each observation in a row
  - Each observational unit in a table

- In textual data the **token** is any **meaningful unit of text** (usually words) = tidytext allows us to "tokenise" the data and get one-word-per-row.



Tokens

- The way text is ordinarily stored in R is as a **string**, and it is managed as a **character** variable.

- Example from Assignment 3 from our course:

```
 [1] "Midterm elections 2022: ‘Red wave’ fails to materialise as Kentucky rejects anti-abortion measure"
 [2] "Control of House remains unknown as Democrats beat midterm expectations"
 [3] "US midterm election results"
 [4] "Early lessons from the US midterm elections as votes are still being counted"
 [5] "Trump has little to say as Republicans fail to deliver"
 [6] "Russia-Ukraine war: Russia orders troops to leave key Ukrainian city of Kherson"
 [7] "China’s top climate official urges US to ‘clear barriers’ to talks"
 [8] "Unseen Kristallnacht photos published 84 years after pogrom"
 [9] "Former UK health secretary covered with bugs and sludge in I’m a Celebrity preview"
[10] "Facebook owner to sack 11,000 workers after revenue collapse "
```

```
Values
  articles_headlines  chr [1:87]
```

→ further analysis

Functions of the tidytext package

- Character vector
- Tibble
- Unnest_tokens

```
text <- c("i carry your heart with me i carry it in,
          my heart i am never without it anywhere,
           i go you go,my dear;and whatever is done,
           by only me is your doing,my darling,
             i fear,
             no fate for you are my fate,my sweet i want,
             no world for beautiful you are my world,my true,
             and it's you are whatever a moon has always meant,
             and whatever a sun will always sing is you,
              here is the deepest secret nobody knows,
             here is the root of the root and the bud of the bud,
               and the sky of the sky of a tree called life;which grows,
               higher than soul can hope or mind can hide,
             and this is the wonder thats keeping the stars apart,
              i carry your heart i carry it in my heart")
```

```
text_df <- tibble(line = 1:length(text), text = text)

text_df
```

Functions of the tidytext package

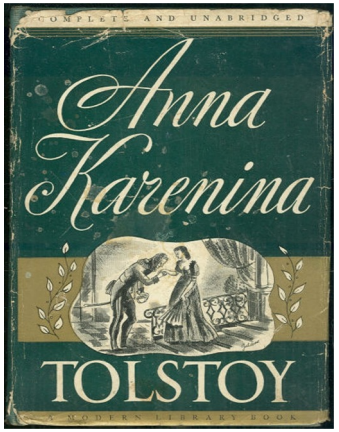
```
library(tidytext)

text_df %>%
  unnest_tokens(word, text) %>%
  count(word, sort = TRUE)
```

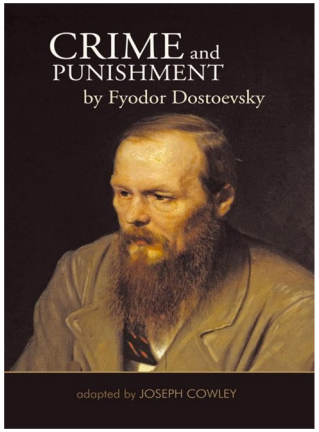
```
# A tibble: 72 × 2
   word      n
   <chr> <int>
 1 the       9
 2 i         8
 3 my        8
 4 and       6
 5 is        6
 6 you       5
 7 carry     4
 8 heart     4
 9 of        4
10 a         3
# … with 62 more rows
```

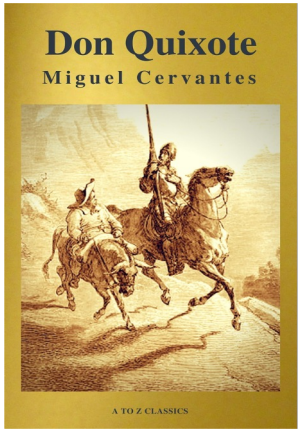
Functions of the tidytext package

Larger data sets

- Gutenberg Project
- Read_html
- Xpath








```
crimepunishment_url <- read_html("https://www.gutenberg.org/cache/epub/2554/pg2554-images.html")
annakarenina_url <- read_html("https://www.gutenberg.org/cache/epub/1399/pg1399-images.html")
donquixote_url <- read_html("https://www.gutenberg.org/cache/epub/996/pg996-images.html")


crimepunishment_extraction <-html_text(html_elements(crimepunishment_url, xpath = "//body"))
annakarenina_extraction <-html_text(html_elements(annakarenina_url, xpath = "//body"))
donquixote_extraction <-html_text(html_elements(donquixote_url, xpath = "//p"))
```

Crime and Punishment - unnest_tokens

```
tidy_crimepunishment <- crimepunishment_df %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words)

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

	word	n
	<chr>	<int>
1	raskolnikov	725
2	don't	465
3	it's	451
4	time	385
5	sonia	370
6	that's	354
7	razumihin	324
8	dounia	302
9	looked	293
10	suddenly	293

… with 9,261 more rows

Combining the books

```
frequency <- bind_rows(mutate(tidy_crimepunishment, author = "Crime and Punishment"),
                       mutate(tidy_annakarenina, author = "Anna Karenina"),
                       mutate(tidy_donquixote, author = "Don Quixote")) %>%
  mutate(word = str_extract(word, "[a-z']+")) %>%
  count(author, word) %>%
  group_by(author) %>%
  mutate(proportion = n / sum(n)) %>%
  select(-n) %>%
  pivot_wider(names_from = author, values_from = proportion) %>%
  pivot_longer(`Crime and Punishment`:`Anna Karenina`,
               names_to = "author", values_to = "proportion")
```

All books – sentiment analysis with get_sentiments

168	agreeable	positive
169	agreeableness	positive
170	agreeably	positive
171	aground	negative
172	ail	negative
173	ailing	negative
174	ailment	negative
175	aimless	negative
176	alarm	negative
177	alarmed	negative
178	alarming	negative
179	alarmingly	negative
180	alienate	negative
181	alienated	negative
182	alienation	negative
183	all-around	positive
184	allegation	negative
185	allegations	negative

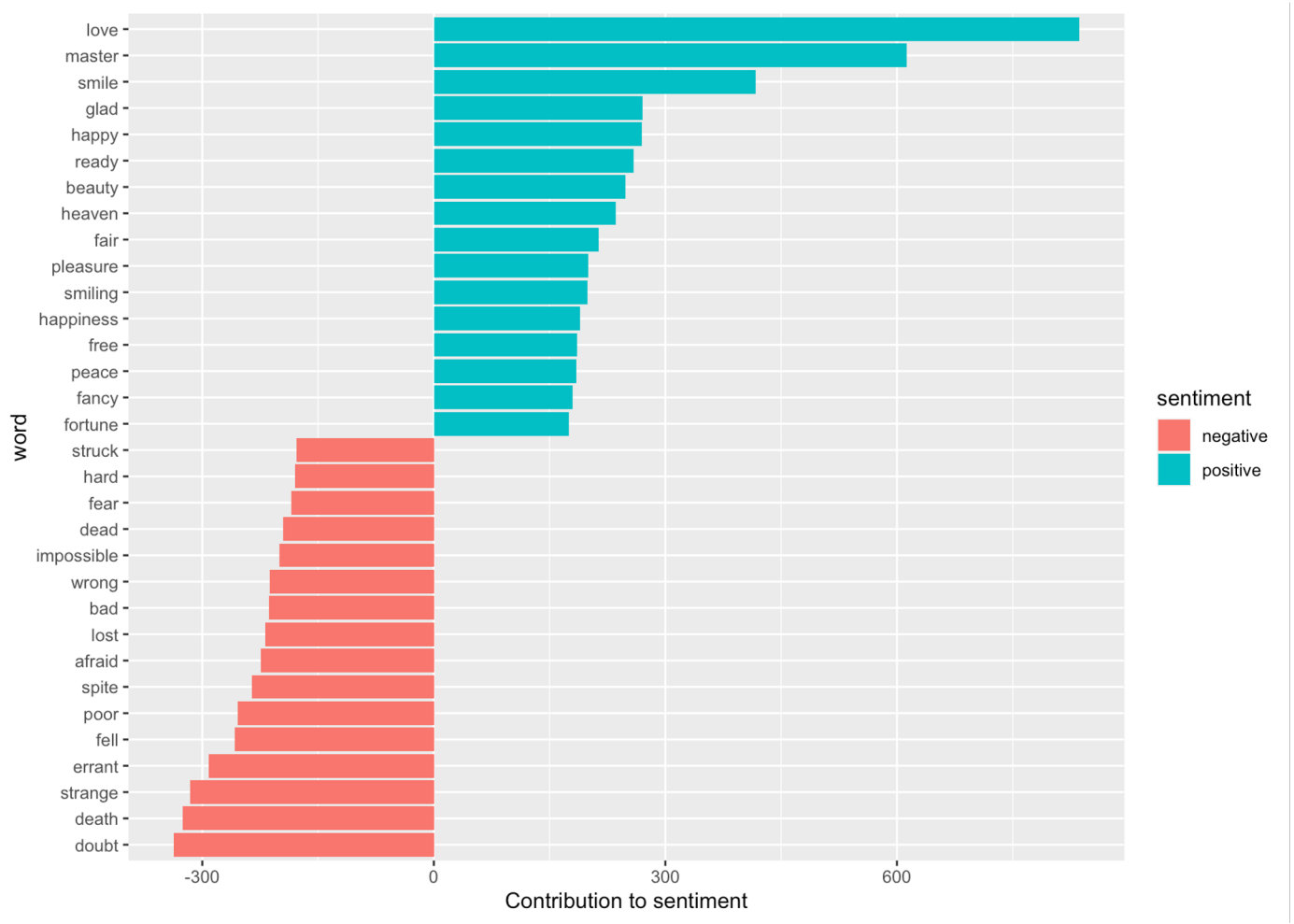
```
bing <- get_sentiments("bing")

bing_word_counts <- combined %>%
  inner_join(bing) %>%
  count(word, sentiment, sort = TRUE)
```

```
# A tibble: 3,482 × 3
   word    sentiment     n
   <chr>   <chr>     <int>
 1 love    positive    836
 2 master  positive    613
 3 smile   positive    417
 4 doubt   negative    337
 5 death   negative    325
 6 strange negative    316
 7 errant  negative    292
 8 glad    positive    270
 9 happy   positive    269
10 ready   positive    259
# … with 3,472 more rows
```

All books – sentiment analysis & ggplot

OPTION A. Common chart.

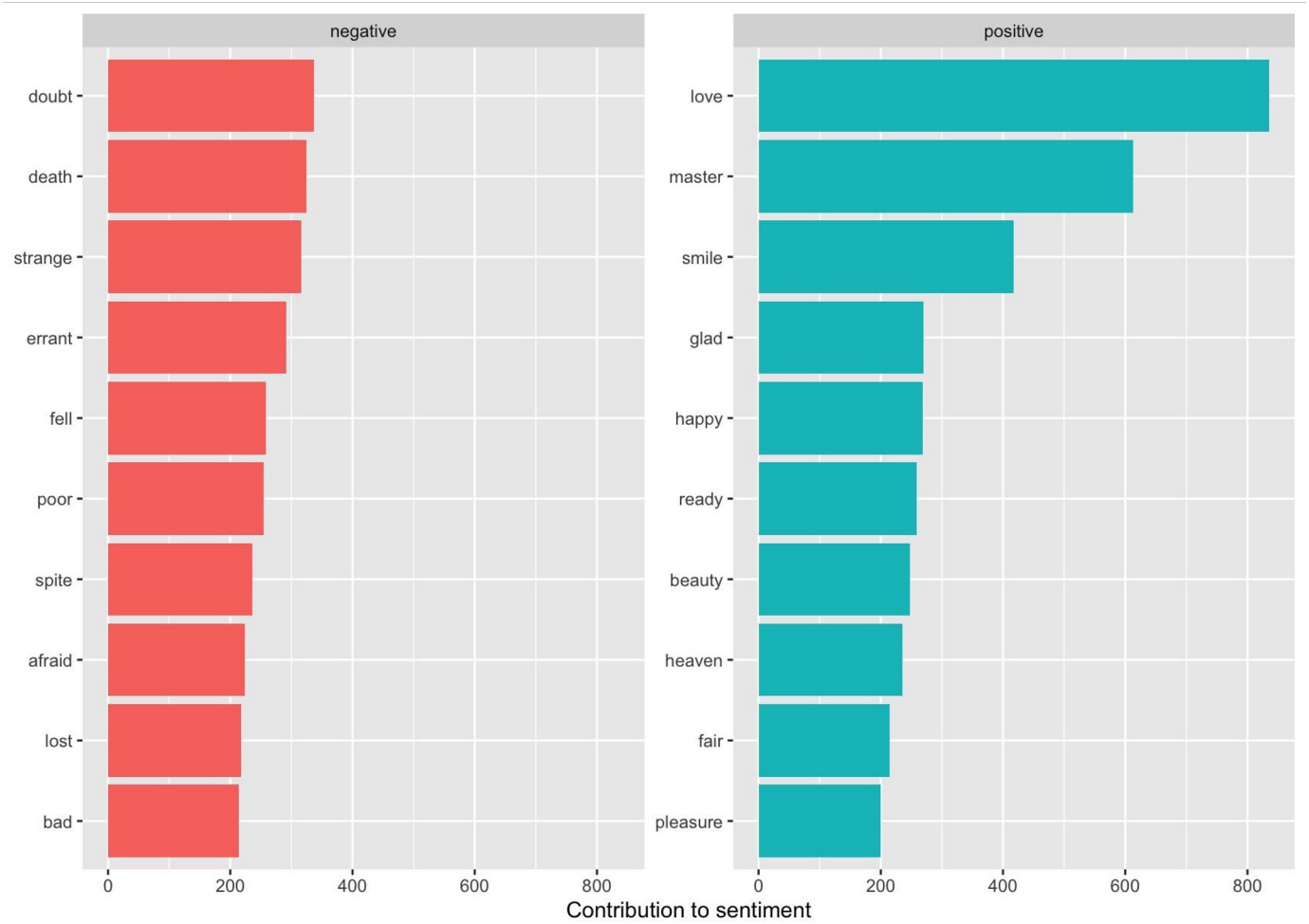
```
bing_word_counts %>%
  filter(n > 170) %>%
  mutate(n = ifelse(sentiment == "negative", -n, n)) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col() +
  coord_flip() +
  labs(y = "Contribution to sentiment")
```


love
master
smile
glad
happy
ready
beauty
heaven
fair
pleasure
smiling
happiness
free
peace
fancy
fortune
struck
hard
fear
dead
impossible
wrong
bad
lost
afraid
spite
poor
fell
errant
strange
death
doubt
word
-300
0
300
600
Contribution to sentiment
sentiment
negative
positive

All books - sentiments & ggplot

OPTION B. Separate chart.

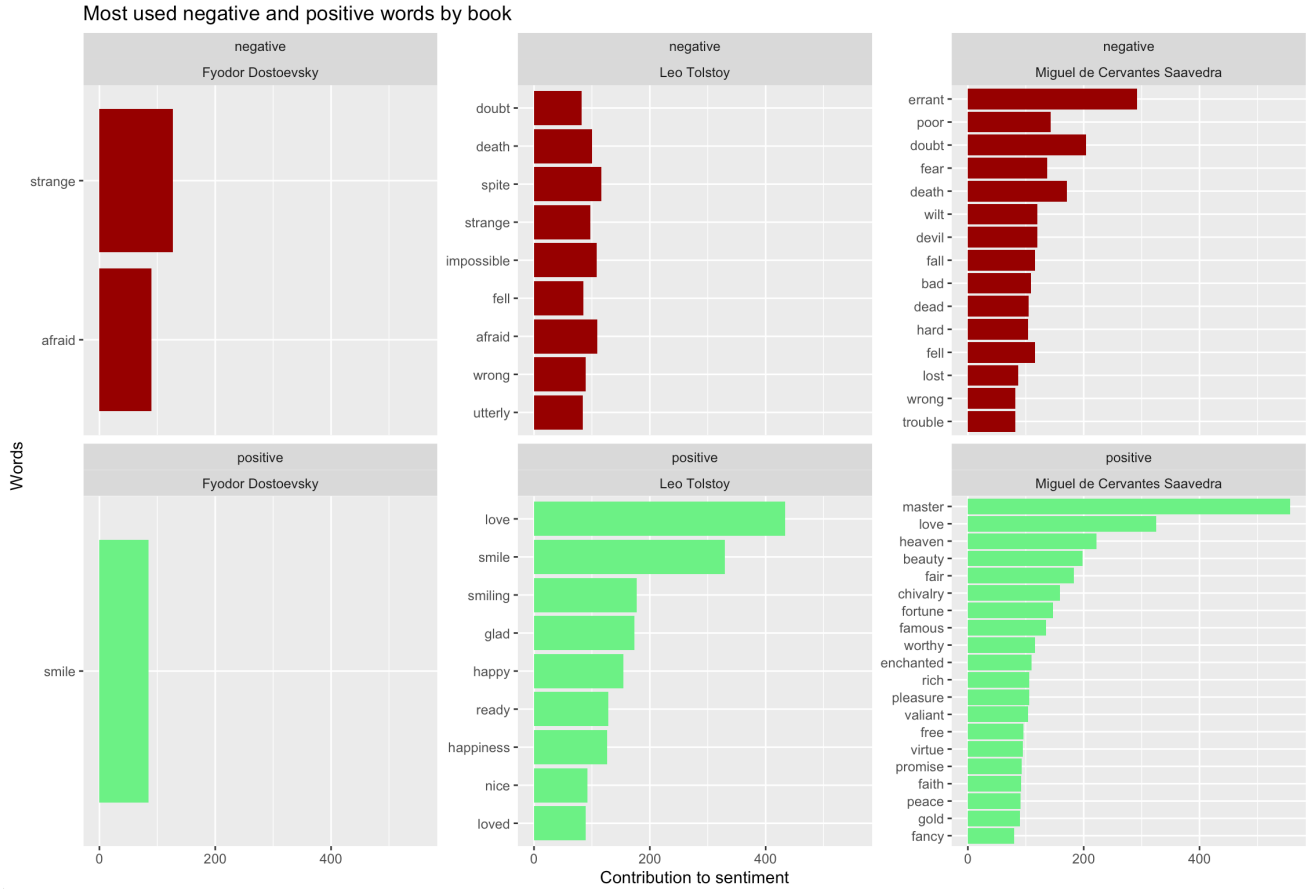
```
bing_word_counts %>%
  group_by(sentiment) %>%
  slice_max(n, n = 10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(x = "Contribution to sentiment",
       y = NULL)
```


negative
doubt
death
strange
errant
fell
poor
spite
afraid
lost
bad
0
200
400
600
800
positive
love
master
smile
glad
happy
ready
beauty
heaven
fair
pleasure
0
200
400
600
800
Contribution to sentiment

Books separately – sentiments & ggplot

```
bing_word_counts %>%
  filter(n>=85) %>%
  group_by(sentiment) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(n, word, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment + author, scales = "free_y") +
  scale_fill_manual(values= c("dark red", "light green")) +
  labs(title = "Most used negative and positive words by book",
         x = "Contribution to sentiment",
       y = "Words")
```

Books separately – sentiments & ggplot



All books - wordcloud

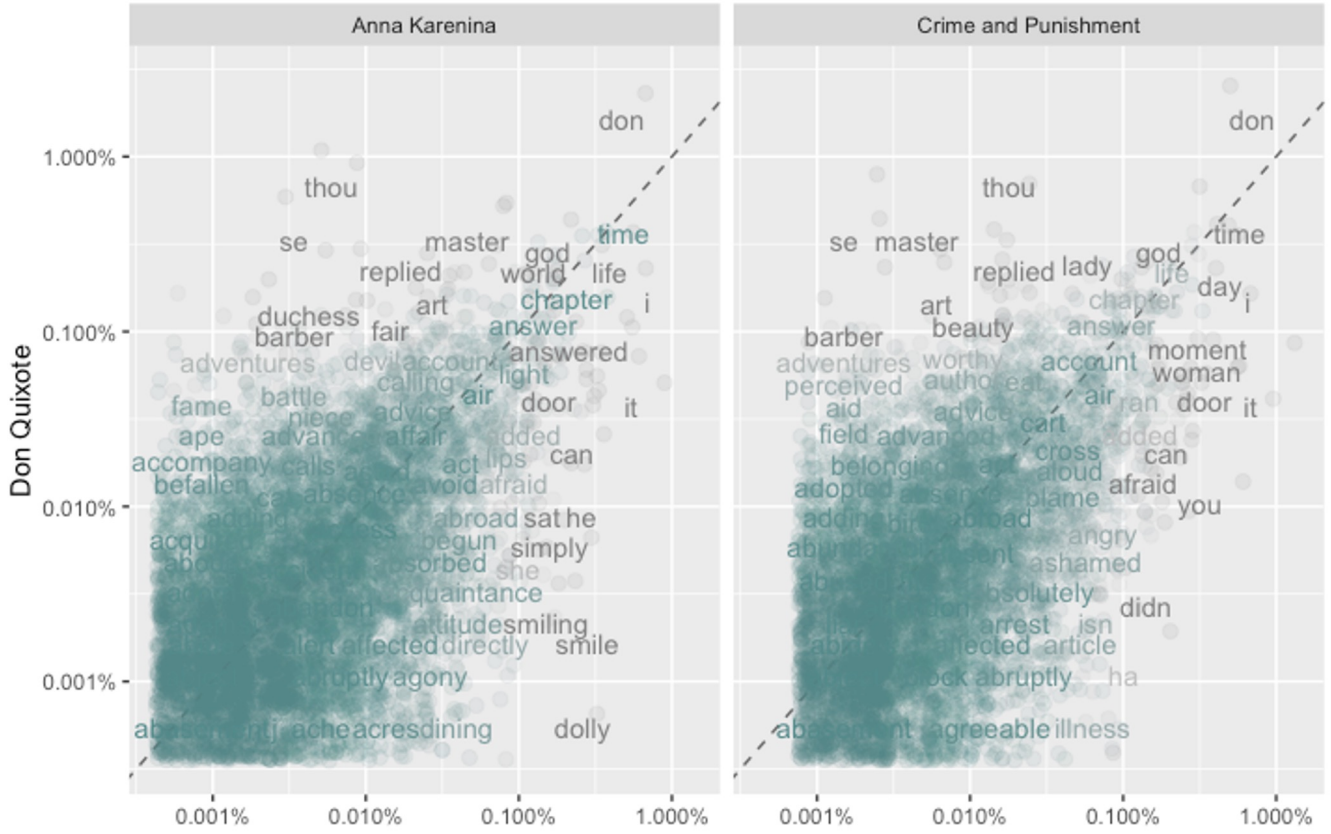
```
library(wordcloud)

combined %>%
  count(word) %>%
  with(wordcloud(word, n, max.words = 100))

library(reshape2)

combined %>%
  inner_join(bing) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("#F8766D", "#00BFC4"),
                   max.words = 100)
```

All books - wordcloud

All books - are books (the words used) correlated?

```
ggplot1 <- ggplot(frequency, aes(x = proportion, y = `Don Quixote`,
                        color = abs(`Don Quixote` - proportion))) +
  geom_abline(color = "gray40", lty = 2) +
  geom_jitter(alpha = 0.1, size = 2.5, width = 0.3, height = 0.3) +
  geom_text(aes(label = word), check_overlap = TRUE, vjust = 1.5) +
  scale_x_log10(labels = percent_format()) +
  scale_y_log10(labels = percent_format()) +
  scale_color_gradient(limits = c(0, 0.001),
                       low = "darkslategray4", high = "gray75") +
  facet_wrap(~author, ncol = 2) +
  theme(legend.position="none") +
  labs(y = "Don Quixote", x = NULL)
```

All books - are books (the words used) correlated?



All books - are books (the words used) correlated?

All books - are books (the words used) correlated?

```
cor.test(data = frequency[frequency$author == "Crime and Punishment",],
         ~ proportion + `Don Quixote`)

cor.test(data = frequency[frequency$author == "Anna Karenina",],
         ~ proportion + `Don Quixote`)
```

```
        Pearson's product-moment correlation

data:  proportion and Don Quixote
t = 45.371, df = 5970, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.4872524 0.5249789
sample estimates:
      cor
0.5063579
```

```
        Pearson's product-moment correlation

data:  proportion and Don Quixote
t = 47.577, df = 7075, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.4744780 0.5097856
sample estimates:
      cor
0.4923343
```

TIDYTEXT

Your chance to practise has arrived 👏🏽

Are you ready to analyse their inauguration speeches?

📣

Inauguration speeches

TIDYTEXT

References

R packages:

Silge J, Robinson D (2016). "tidytext: Text Mining and Analysis Using Tidy Data Principles in R." _JOSS_,*1*(3). doi:10.21105/joss.00037 <https://doi.org/10.21105/joss.00037>, <http://dx.doi.org/10.21105/joss.00037>.

Wickham H (2022). _stringr: Simple, Consistent Wrappers for Common String Operations_. R package version 1.4.1, <https://CRAN.R-project.org/package=stringr>.

Wickham H. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016.

Wickham H (2022). _rvest: Easily Harvest (Scrape) Web Pages_. R package version 1.0.3, <https://CRAN.R-project.org/package=rvest>.

Wickham H, Averick M, Bryan J, Chang W, McGowan LD, François R, Grolemund G, Hayes A, Henry L, Hester J, Kuhn M, Pedersen TL, Miller E, Bache SM, Müller K, Ooms J, Robinson D, Seidel DP, Spinu V, Takahashi K, Vaughan D, Wilke C, Woo K, Yutani H (2019). "Welcome to the tidyverse." _Journal of Open Source Software_, *4*(43), 1686. doi:10.21105/joss.01686 <https://doi.org/10.21105/joss.01686>.

Wickham H, François R, Henry L, Müller K (2022). _dplyr: A Grammar of Data Manipulation_. R package version 1.0.10, <https://CRAN.R-project.org/package=dplyr>.

Bibliography:

Robinson, Julia Silge and David. Introduction to Tidytext, 19 Aug. 2022, https://cran.rproject.org/web/packages/tidytext/vignettes/tidytext.html.

Robinson, Julia Silge and David. "Welcome to Text Mining with r: Text Mining with R." Welcome to Text Mining with R | Text Mining with R, https://www.tidytextmining.com/.

"Tidy Text." YouTube, YouTube, 1 Mar. 2022, https://www.youtube.com/watch?v=Udp2WlvuWHo&t=645s.

"Text Mining, the Tidy Way." YouTube, YouTube, 16 May 2017, https://www.youtube.com/watch?v=0poJP8WQxew&t=400s.