The tidy text format

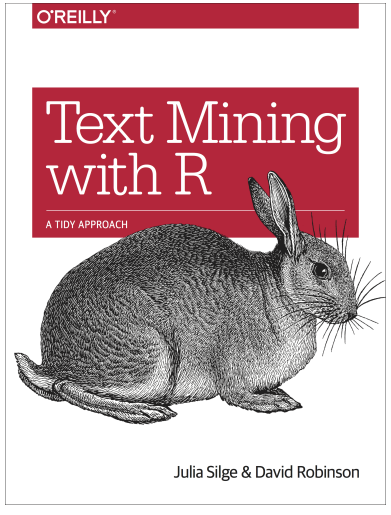
Rei Sanchez-Arias, Ph.D.

The tidy text format: definitions, basic functions, and examples

Tidy text mining

Tidy data

"Text Mining with R: A Tidy Approach" by Julia Silge and David Robinson



Recall that our definition of **tidy data** is:

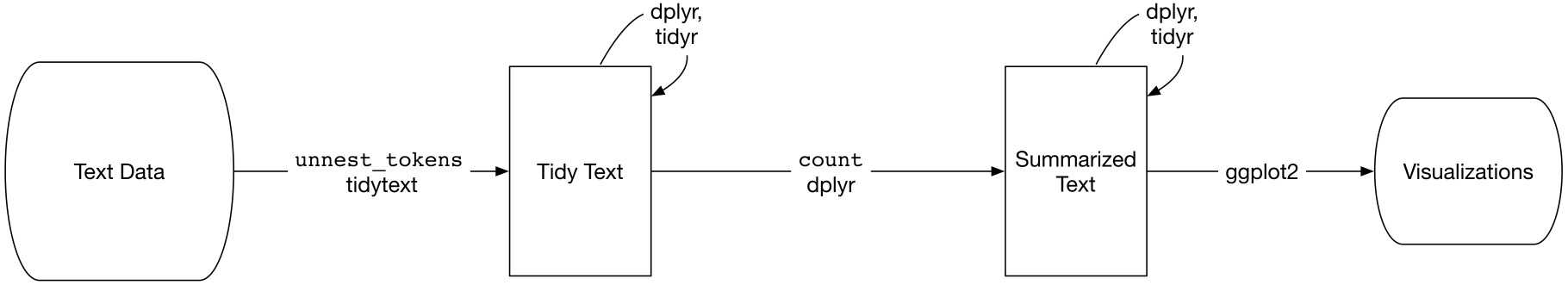
- Each variable is a column
- Each observation is a row
- Each type of observational unit is a table

> Examples and materials in this set of slides are adapted from the book by Silge and Robinson

One-token-per-row

Tidy text format: a table with **one-token-per-row**. A **token** is a meaningful unit of text, such as a word, that we are interested in using for analysis, and **tokenization** is the process of splitting text into tokens. The token that is stored in each row is most often a single word, but can also be an n-gram, sentence, or paragraph.

The `tidytext` package, provides a functionality to tokenize by commonly used units of text and convert to a one-term-per-row format.



A note on other approaches

The tidy text format framework is worth contrasting with the ways text is often stored in text mining approaches:

- **String:** Text can, of course, be stored as strings, i.e., character vectors, and often text data is first read into memory in this form.
- **Corpus:** These types of objects typically contain raw strings annotated with additional metadata and details.
- **Document-term matrix:** This is a sparse matrix describing a collection (i.e., a **corpus**) of documents with one row for each document and one column for each term. The value in the matrix is typically word count or TF-IDF (term frequency-inverse document frequency).

The `unnest_tokens()` function

```
queen <- c(
  "Buddy, you are a boy, make a big noise",
  "Playing in the street, gonna be a big man someday",
  "You got mud on your face, you big disgrace",
  "Kicking your can all over the place, singing"
)
```

This is a typical character vector that we might want to analyze. In order to turn it into a tidy text dataset, we first need to put it into a data frame.

```
library(tidyverse)
text_df <- tibble(line = 1:4, text = queen)
```

```
## # A tibble: 4 x 2
##    line text
##   <int> <chr>
## 1     1 Buddy, you are a boy, make a big noise
## 2     2 Playing in the street, gonna be a big man someda
## 3     3 You got mud on your face, you big disgrace
## 4     4 Kicking your can all over the place, singing
```

(a *tibble* has a convenient print method, will not convert strings to factors, and does not use row names)

The `unnest_tokens()` function (cont.)

```
library(tidytext)
text_df %>%
  unnest_tokens(output = word,
                input = text)
```

```
## # A tibble: 36 x 2
##     line word
##    <int> <chr>
##  1     1 buddy
##  2     1 you
##  3     1 are
##  4     1 a
##  5     1 boy
##  6     1 make
##  7     1 a
##  8     1 big
##  9     1 noise
## 10     2 playing
## # … with 26 more rows
```

The two basic arguments to `unnest_tokens()` used here are column names. First we have the output column name that will be created as the text is unnested into it (`word`, in this case), and then the input column that the text comes from (`text`, in this case).

Remember that `text_df` above has a column called `text` that contains the data of interest.

The `gutenbergr` package

gutenbergr

```
library(gutenbergr)
```

The `gutenbergr` package provides access to the public domain works from the Project Gutenberg collection. The package includes tools both for downloading books (stripping out the unhelpful header/footer information), and a complete dataset of Project Gutenberg metadata that can be used to find works of interest. You can use the function `gutenberg_download()` to download one or more works from Project Gutenberg by `ID`.

You can also use other functions to explore metadata, pair Gutenberg ID with title, author, language, etc., or gather information about authors.

Word frequencies

A common task in text mining is to look at **word frequencies**.

Consider some science fiction and fantasy novels by H.G. Wells, who lived in the late 19th and early 20th centuries. Let us get "The Time Machine", "The War of the Worlds", "The Invisible Man", and "The Island of Doctor Moreau". We can access these works using `gutenberg_download()` and the Project Gutenberg ID numbers for each novel.

```
hgwells <- gutenberg_download(c(35, 36, 5230, 159))
```

Use `unnest_tokens()` from `tidytext` and `anti_join(x, y)` from `dplyr` (return all rows from `x` where there are not matching values in `y`, keeping just columns from `x`)

```
tidy_hgwells <- hgwells %>%
                  unnest_tokens(word, text) %>%
                  anti_join(stop_words)
```

Word frequencies (cont.)

Most common words in these novels of H.G. Wells

```
tidy_hgwells %>%
  count(word, sort = TRUE) %>%
  head(7)
```

```
## # A tibble: 7 x 2
##   word       n
##   <chr>  <int>
## 1 time     454
## 2 people   302
## 3 door     260
## 4 heard    249
## 5 black    232
## 6 stood    229
## 7 white    222
```

We use tools from the `tidyverse` to *reshape* the dataframe in the form we need it for visualization:

```
freq_hgwells <- tidy_hgwells %>%
  mutate(author = "H.G. Wells") %>%
  mutate(word =
           str_extract(
             word,
             "[a-z']+")
         ) %>%
  count(author, word) %>%
  group_by(author) %>%
  mutate(proportion = n/sum(n)) %>%
  select(-n)
```

Word frequency and proportion

Examine the new object

```
head(freq_hgwells, 8)
```

```
## # A tibble: 8 x 3
## # Groups:   author [1]
##   author      word        proportion
##   <chr>       <chr>            <dbl>
## 1 H.G. Wells a              0.0000150
## 2 H.G. Wells aback          0.0000150
## 3 H.G. Wells abandon        0.0000150
## 4 H.G. Wells abandoned      0.000180
## 5 H.G. Wells abandoning     0.0000450
## 6 H.G. Wells abandonment    0.0000150
## 7 H.G. Wells abart          0.0000150
## 8 H.G. Wells abbey          0.0000300
```

We use `str_extract()` here because the UTF-8 encoded texts from Project Gutenberg have some examples of words with underscores around them to indicate emphasis (like italics). The tokenizer treated these as words, but we do not want to count "*any*" separately from "any".

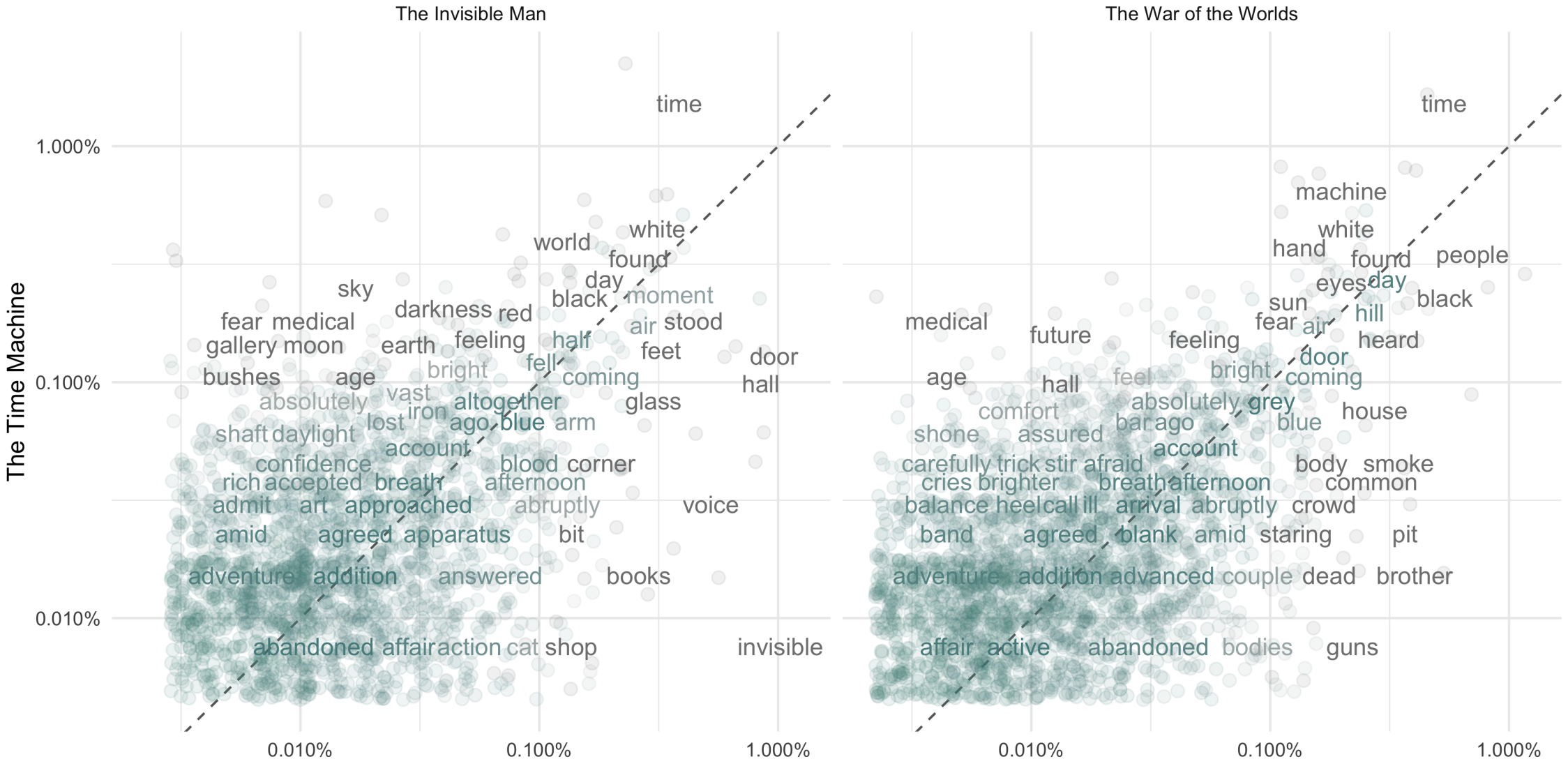
We perform an analysis comparing the proportion (number of times a word was used) of words in the different books we are studying by H.G. Wells

Comparing proportions

```
wells_freq <- tidy_hgwells %>%
  mutate(book = case_when(
    gutenberg_id == 35 ~ "The Time Machine",
    gutenberg_id == 36 ~ "The War of the Worlds",
    gutenberg_id == 5230 ~ "The Invisible Man",
    gutenberg_id == 159 ~ "The Island of Doctor Moreau"
    ) ) %>%
  mutate(word = str_extract(word, "[a-z']+")) %>%
  count(book, word) %>%
  group_by(book) %>%
  mutate(proportion = n/sum(n)) %>%
  select(-n) %>%
  pivot_wider(names_from = book, values_from  = proportion) %>%
  pivot_longer(
   cols = c(`The War of the Worlds`, `The Invisible Man`, `The Island of Doctor Moreau`),
   names_to = "book", values_to = "proportion")
```

Comparing proportions (cont.)

```
# let us consider two of the books when compared with "The Time Machine"
wells_freq %>%
  filter(book %in% c("The War of the Worlds", "The Invisible Man")) %>%
  ggplot(aes(x = proportion,
             y = `The Time Machine`,
             color = abs(`The Time Machine` - 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 = scales::percent_format()) +
  scale_y_log10(labels = scales::percent_format()) +
  scale_color_gradient(limits = c(0, 0.001),
                       low = "darkslategray4",
                       high = "gray75") +
  facet_wrap(~book, ncol = 2) +
  theme_minimal() +
  theme(legend.position="none") +
  labs(y = "The Time Machine", x = NULL)
```


The Invisible Man
The War of the Worlds
The Time Machine
1.000%
0.100%
0.010%
0.010%
0.100%
1.000%
time
white
world
found
day
black
moment
sky
darkness
red
air
stood
fear
medical
gallery
moon
earth
feeling
half
feet
door
bushes
age
bright
fell
coming
hall
absolutely
vast
altogether
glass
iron
lost
ago
blue
arm
shaft
daylight
account
confidence
blood
corner
rich
accepted
breath
afternoon
admit
art
approached
abruptly
voice
amid
agreed
apparatus
bit
adventure
addition
answered
books
abandoned
affair
action
cat
shop
invisible
machine
hand
people
eyes
sun
hill
heard
future
feeling
fear
air
door
bright
coming
medical
age
hall
feel
absolutely
grey
house
comfort
bar
ago
blue
shone
assured
account
carefully
trick
stir
afraid
body
smoke
cries
brighter
breath
afternoon
common
balance
heel
call
ill
arrival
abruptly
crowd
band
agreed
blank
amid
staring
pit
adventure
addition
advanced
couple
dead
brother
affair
active
abandoned
bodies
guns

stopwords

Stop-words ?

Use the `stopwords()` function to find a list of common stop-words in different languages

```
stopwords::stopwords(
  language = "english") %>%
  sample(10)
```

```
##  [1] "which"   "she's"   "ours"    "whi
##  [8] "further" "if"      "below"
```

```
stopwords::stopwords(
  language = "french") %>%
  sample(10)
```

```
##  [1] "soyons" "à"      "eue"    "été"
##  [9] "es"     "tes"
```

```
# another sample
stopwords::stopwords(
  language = "spanish") %>%
  sample(10)
```

```
##  [1] "estuvierais" "tuyo"          "haya"
##  [6] "entre"       "tu"            "antes
```

```
# another sample
stopwords::stopwords(
  language = "portuguese") %>%
  sample(10)
```

```
##  [1] "seríamos"  "esteve"    "estivera"
##  [7] "mas"       "a"         "pelas"
```

Stop-words ? (2)

```
# languages available
stopwords::stopwords_getlanguages(
  source = "snowball" )
```

```
##  [1] "da" "de" "en" "es" "fi" "fr" "hu" "ir" "it" "nl" "no" "pt" "ro" "ru" "sv"
```

```
# possible sources
stopwords::stopwords_getsources()
```

```
## [1] "snowball"      "stopwords-iso" "misc"          "smart"
## [5] "marimo"        "ancient"       "nltk"
```

Stop-words ? (3)

```
# possible languages available in source
stopwords::stopwords_getlanguages(
  source = "stopwords-iso")
```

```
##  [1] "af" "ar" "hy" "eu" "bn" "br" "bg" "ca" "zh" "hr" "cs" "da" "nl" "en" "eo"
## [16] "et" "fi" "fr" "gl" "de" "el" "ha" "he" "hi" "hu" "id" "ga" "it" "ja" "ko"
## [31] "ku" "la" "lt" "lv" "ms" "mr" "no" "fa" "pl" "pt" "ro" "ru" "sk" "sl" "so"
## [46] "st" "es" "sw" "sv" "th" "tl" "tr" "uk" "ur" "vi" "yo" "zu"
```

```
# japanese stop words
stopwords::stopwords(
  language = "ja",
  source = "stopwords-iso") %>%
  sample(10)
```

```
##  [1] "ず"      "あの人" "どの"   "ば"      "いう"   "そこ"   "ので"   "とき"
##  [9] "ほど"   "たち"
```