Intro to text mining in R

R workshops 2021
Baruch College

Víctor Peña, January 2021

Logistics

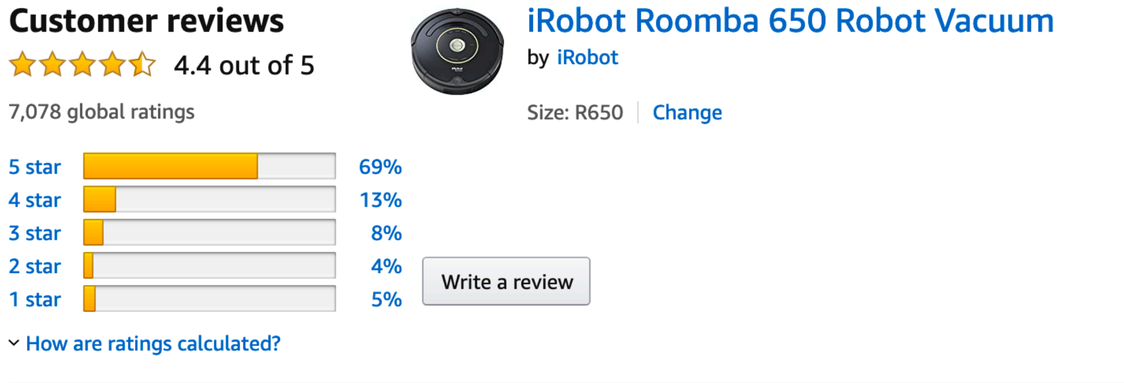
- Course website: https://vicpena.github.io/workshops/2021/advanced
- My email: victor.pena@baruch.cuny.edu
- I'll cover some sections of Text Mining with R, by Silge and Robinson and the Datacamp courses
  - Introduction to text analysis in R
  - Introduction to Natural Language Processing in R
  - String manipulation with stringr
- I'll post the code we write in our sessions on the course website

Topics

- Through case studies, we'll cover:
  - Analyzing word frequencies: word counts, tf-idf
  - Analyzing relationships between words: n-grams, correlations
  - Sentiment analysis
  - Topic modeling
  - Working with text data in R

First case study: Roomba reviews

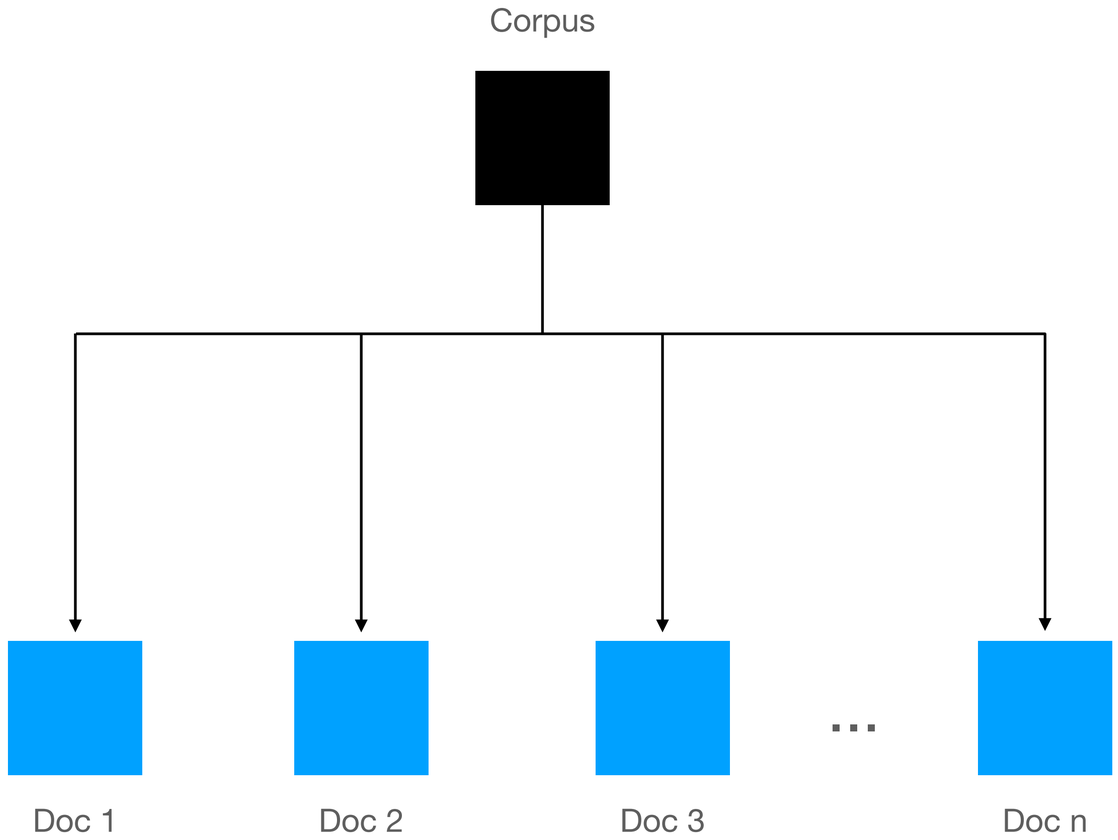
- We'll analyze a dataset which contains Amazon reviews for Roombas
- The dataset is clean and nicely formatted... which is not the usual starting point in text analysis
- The data can be found on the course website



Analyzing word/document frequencies

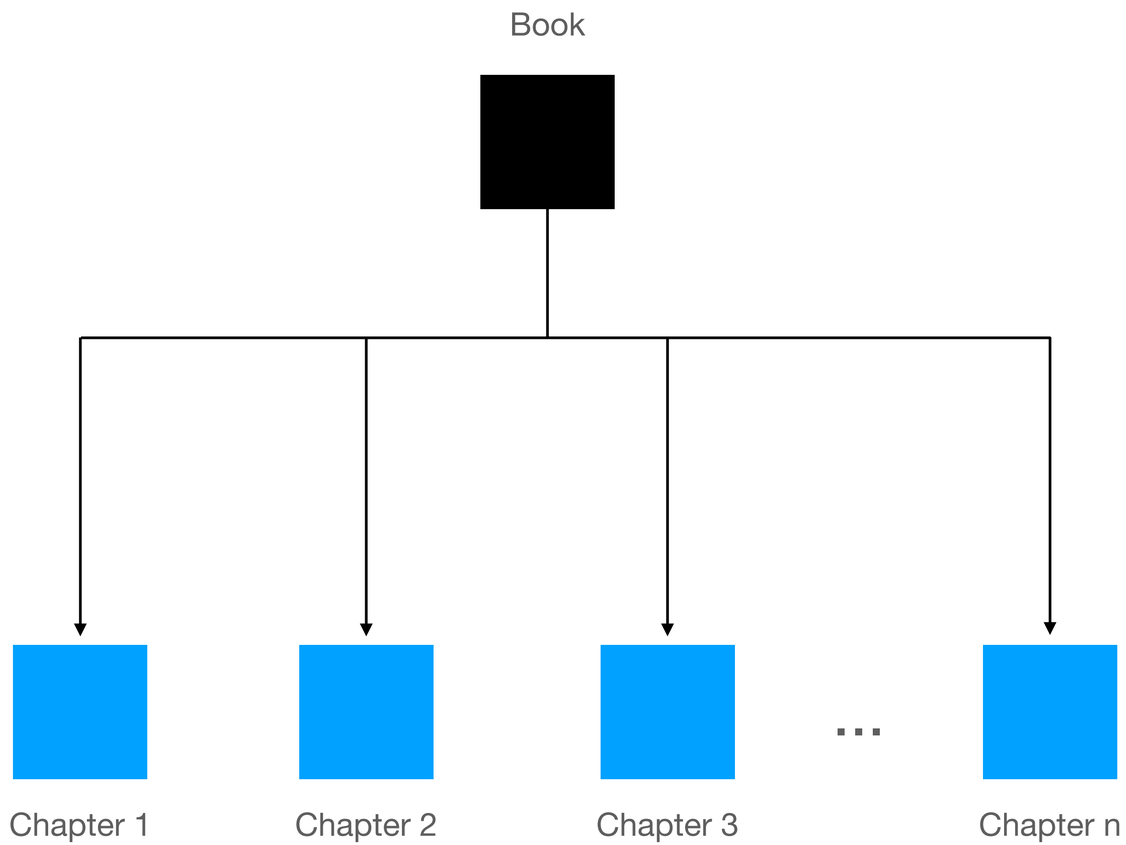
Conceptual framework

- We have a corpus of documents
- We want to know how different or similar the different documents are
- For example, are there certain words that are used more often in certain documents?
- Can we quantify "distance" or "correlation" between documents, in some way?



Example: book

- **Corpus:** book itself
- **Documents:** chapters
- **Questions we'll answer:**
  - Are there certain words that appear more often in certain chapters?
  - Which chapters are most similar?



TF-IDF is TF times IDF

- **TF:** term frequency of a word in a document

TF(word, document) = (# times word appears in document)/(# words in document)

- **IDF:** log "inverse document frequency" of a word in the corpus

IDF(word) = log[(# documents)/(# documents containing word)]

  - If a word appears in all documents => IDF(word) = log(1) = 0
  - If a word does not appear in any document => IDF(word) = log(#docs/0) = Infinity
  - If IDF(word) is high, the word only appears in a few documents; if IDF(word) is low, it appears in most documents

- **TF-IDF:** simply multiply TF times IDF

TF-IDF(word, document) = TF(word, document) * IDF(word)

TF(word, document) = (# times word appears in document)/(# words in document)

IDF(word) = log[(# documents)/(# documents containing word)]

- **Document 1**
  - cat, dog, cat, cat
- **Document 2**
  - dog, dog, dog, mouse
- **Document 3**
  - beetle, mouse, dog, aye-aye

TF-IDF(cat, doc 1)?

- TF(cat, doc 1) = 3/4
- IDF(cat) = log(3/1)
- TF-IDF ~ (3/4)*log(3/1) ~ 0.8239

TF(word, document) = (# times word appears in document)/(# words in document)

IDF(word) = log[(# documents)/(# documents containing word)]

- **Document 1**
  - cat, dog, cat, cat
- **Document 2**
  - dog, dog, dog, mouse
- **Document 3**
  - beetle, mouse, dog, aye-aye

TF-IDF(cat, doc 2)?

- TF(cat, doc 2) = 0/4 = 0
- IDF(cat) = log(3/1)
- TF-IDF(cat, doc 2) = 0

If a term doesn't appear in a document

its TF-IDF is going to be 0

TF(word, document) = (# times word appears in document)/(# words in document)

IDF(word) = log[(# documents)/(# documents containing word)]

- **Document 1**
  - cat, dog, cat, cat
- **Document 2**
  - dog, dog, dog, mouse
- **Document 3**
  - beetle, mouse, dog, aye-aye

TF-IDF(dog, doc 2)?

- TF(dog, doc 2) = 3/4
- log-odds: log(TF/(1-TF))
- IDF(dog) = log(3/3) = 0
- TF-IDF(dog, doc 2) = (3/4)*0 = 0

If a term appears in all documents, then

TF-IDF is going to be 0

TF(word, document) = (# times word appears in document)/(# words in document)

IDF(word) = log[(# documents)/(# documents containing word)]

- **Document 1**
  - cat, dog, cat, cat
- **Document 2**
  - dog, dog, dog, mouse
- **Document 3**
  - beetle, mouse, dog, aye-aye

TF-IDF(aye-aye, document 3)?

- TF(aye-aye, doc 3) = 1/4
- IDF(aye-aye) = log(3/1)
- TF-IDF = (1/4)*log(3) ~ 0.275

Odds and ends

- If a word in a document has a high TF-IDF, it means that that word appears "***weirdly often***" in that document
  - Another way of saying the same thing: ***the word is "highly specific to that document"***
- In R, we won't have to do these computations by hand
  - We can use bind_tf_idf in library(tidytext)

Let's find TF-IDFs in a real example

- Let's work with the text of Animal Farm, by George Orwell