LING 410X: Language as Data

Semester: Spring '18

Instructor: Sowmya Vajjala

Iowa State University, USA

23 January 2018

Outline

- Review of last week
- Some advice on coping with R, doing automated text analysis
- What is a corpus?
- Reading different formats of data

Review of last week

Recap

- Getting the most frequent words in a text
- Looking for regular expression based patterns in text
- grep/grepl functions - looking for occurrences of a given pattern/regular expression
- sub/gsub functions - for substituting one pattern with another
- stringr library, str_count - function.

Last class' exercise

- Alter 3 lines such that the program works with most of Gutenberg.org projects
- I first identified one pattern (startsWith (*** START OF THE PROJECT))

Last class' exercise

- Alter 3 lines such that the program works with most of Gutenberg.org projects
- I first identified one pattern (startsWith (*** START OF THE PROJECT))
- I realized after class that it is inconsistent, and does not occur like that in all texts.
- So, I changed to something that seemed more consistent.

Original code: Specific to that particular text

```
english <- scan("DollsHouse-Eng.txt", what = "character", sep = "\n")
english.start <- which(english == "DRAMATIS PERSONAE")
english.end <- which(english ==
      "(The sound of a door shutting is heard from below.)")
actual_english <- english[english.start:english.end]
actual_english_string <- paste(actual_english, collapse = " ")
english_lower <- tolower(actual_english_string)
english_words <- strsplit(english_lower, "\\W+")
sorted_freqs_english <- sort(table(english_words), decreasing = TRUE)
plot(sorted_freqs_english[1:10], type="b")
```

First Solution, to make it work for all gutenberg.org texts

```
english <- scan("DollsHouse-Eng.txt", what = "character", sep = "\n")
meta.top.end <- which(startsWith(english,"*** START OF THIS PROJECT GUTENBERG EBOOK"))
meta.bottom.start <- which(startsWith(english, "End of the Project Gutenberg EBook"))
actual_english <- english[meta.top.end+1:meta.bottom.start+1]
actual_english_string <- paste(actual_english, collapse = " ")
english_lower <- tolower(actual_english_string)
english_words <- strsplit(english_lower, "\\W+")
sorted_freqs_english <- sort(table(english_words), decreasing = TRUE)
plot(sorted_freqs_english[1:10], type="b")
```

Second Solution

```
library(stringr)
english <- scan("DollsHouse-Eng.txt", what = "character", sep = "\n")
meta.top.end <- which(str_detect(english,"START OF THIS PROJECT"))
meta.bottom.start <- which(str_detect(english,"END OF THIS PROJECT"))
actual_english <- english[meta.top.end+1:meta.bottom.start-1]
actual_english_string <- paste(actual_english, collapse = " ")
english_lower <- tolower(actual_english_string)
english_words <- strsplit(english_lower, "\\W+")
sorted_freqs_english <- sort(table(english_words), decreasing = TRUE)
plot(sorted_freqs_english[1:10], type="b")
```

Does this work?

- Seems to work, but what if some book does not use upper case for those words? (Note: Gutenberg.org books are volunteer typing/proof reading efforts)

Does this work?

- Seems to work, but what if some book does not use upper case for those words? (Note: Gutenberg.org books are volunteer typing/proof reading efforts)
- We should perhaps look for some way of making the pattern case-insensitive
- Or there is some other innovative solution.

Does this work?

- Seems to work, but what if some book does not use upper case for those words? (Note: Gutenberg.org books are volunteer typing/proof reading efforts)
- We should perhaps look for some way of making the pattern case-insensitive
- Or there is some other innovative solution.
- Working with real-world data is this process of trial and error.
- So, don't get frustrated. Persistence is the key.

General advice on working through this course

- Follow the hands-on sessions - ask as many questions as you want, use the discussion forum
- Get your hands dirty with lot of error messages in R studio
- Carefully work through all the tutorial documents I share, or textbook lessons.
- It needs some extra time, and is not a regular humanities/social sciences class - accept that.
- Believe that you will become text analysis novice-gurus soon!

Last week's exercise and discussion

- In Doll's House, how many times did: "you know" appear?
  str_count(dollshousetext,"you know") - gives 32
- How many times did numbers appear in the text (99712 is to be counted as 1 number. Not 5)?
  str_count(dollshousetext, "\\d+") - gives 10

Note: dollshousetext here - is the output after paste() line in our original code i.e., the actual_english_string after removing all metadata.

setwd()

- How do we get the path? (from R-studio, from file browser on your computer i.e., my computer, finder etc)
- Special caution for windows computers about paths: Forward vs Backward slash symbol
- folder vs file
- Why bother about setwd()? When is it not relevant?

setwd()

- How do we get the path? (from R-studio, from file browser on your computer i.e., my computer, finder etc)
- Special caution for windows computers about paths: Forward vs Backward slash symbol
- folder vs file
- Why bother about setwd()? When is it not relevant?

Suggestions:
Go back and do Lesson 2 in Swirl().
Another useful ref:
`https://www.statmethods.net/interface/workspace.html`

What is a corpus?

Corpus

- As a loose definition, we call the texts we want to analyze as the corpus.
- A single piece of text is not a corpus. Corpus is a collection of such single texts.
- It can be 10 texts. It can be 100 texts. It can be 1000 texts.

Corpus

- As a loose definition, we call the texts we want to analyze as the corpus.
- A single piece of text is not a corpus. Corpus is a collection of such single texts.
- It can be 10 texts. It can be 100 texts. It can be 1000 texts.
- A collection of speeches converted into text (e.g., transcripts) is also a text corpus.
- A collection of tweets is a corpus.

Corpus

- As a loose definition, we call the texts we want to analyze as the corpus.
- A single piece of text is not a corpus. Corpus is a collection of such single texts.
- It can be 10 texts. It can be 100 texts. It can be 1000 texts.
- A collection of speeches converted into text (e.g., transcripts) is also a text corpus.
- A collection of tweets is a corpus.
- As a working definition, we consider any collections of text we choose to work on to solve some problem as a corpus.

How do we get hold of a corpus?

- You can a pre-existing corpus if that suits what you want to work on.
- Creating our own corpus to address our own problem:
  - Data collection (downloading, scanning, web scraping etc)
  - Data preparation/pre-processing
  - Deciding on how to extract the information you need
  - Writing R code
  - Getting the results and analyzing them.

Corpus pre-processing

- Loading the corpus into R environment
- Reading different formats of data in a way R can understand
- Doing any other processing steps as needed (lower casing, sentence splitting etc)

Loading corpus into R

Before getting into this, we should know what are the different formats of text we can see online:

- plain text files - files that you can open and read in most of the text editors starting with basic ones such as notepad.
- PDF files - files you usually read in a pdf reader
- Word documents and the likes -MS word etc.
- XML documents
- HTML documents
- tweets, tables etc (we will discuss as they come)

PDF files

- In the last class, we saw how to easily open a text file using scan() function in R and start doing some stuff like counting word frequencies right away.
- What happens if there is a PDF file instead of a text file in that scan() call. Did anyone try?

PDF files

- In the last class, we saw how to easily open a text file using scan() function in R and start doing some stuff like counting word frequencies right away.
- What happens if there is a PDF file instead of a text file in that scan() call. Did anyone try?
- Sometimes, you may end up with having only pdf versions of documents as your corpus (e.g., a corpus of research articles on topic X from XYZ journal)
- What should we do in such cases?

PDF files

- In the last class, we saw how to easily open a text file using scan() function in R and start doing some stuff like counting word frequencies right away.
- What happens if there is a PDF file instead of a text file in that scan() call. Did anyone try?
- Sometimes, you may end up with having only pdf versions of documents as your corpus (e.g., a corpus of research articles on topic X from XYZ journal)
- What should we do in such cases?
- R supports pdf to text extraction. One of the libraries that can do this is: pdftools

Pdftools

- Install pdftools library by going to Tools $--> $ Install Packages and typing pdftools there.
- Once installed, here is how you get a plain text version of a pdf file:

```
library(pdftools)
txt <- pdf_text("somefilename.pdf")
```

note: I am assuming this file is in my current working directory. Else, we need to give full file path.

Pdftools

- Install pdftools library by going to Tools $--> $ Install Packages and typing pdftools there.
- Once installed, here is how you get a plain text version of a pdf file:

```
library(pdftools)
txt <- pdf_text("somefilename.pdf")
```

  note: I am assuming this file is in my current working directory. Else, we need to give full file path.
- This will not give you a cleanly formatted text string, you need to write extra code for that. This may also not work perhaps with all pdfs. Be aware of that.
- If you end up using this for your work, make sure it works on your pdfs, and know what additional cleaning/formatting you should do within R to get some clean text.
- More information on this library: `https://cran.r-project.org/web/packages/pdftools/pdftools.pdf`

Doc files and qdaptools library

- Install qdaptools library by going to Tools $-- >$ Install Packages and typing qdaptools there.
- Once installed, here is how you get a plain text version of a docx file:

```
library(qdapTools)
txt <- read_docx("somefile.docx")
```

More functions: `https://cran.r-project.org/web/packages/qdapTools/qdapTools.pdf` - have a look at read_docx section here.

XML files

- XML stands for eXtensible Markup Language.
- It is a way to store and data with a tag structure.. so we can add a lot of metadata to text in this format.
- Although not extremely common, you may often end up having access to only XML versions of some corpus and asked to do something with it.
- This can be done using XML library (note the upper case).

Reading XML files in R

```
library(XML)
books <- xmlTreeParse("Books.xml", useInternalNodes = TRUE)
book_titles <- sapply(getNodeSet(books, "//catalog/book/title/text()"), xmlValue)
author_names <- sapply(getNodeSet(books, "//catalog/book/author/text()"), xmlValue)
description <- sapply(getNodeSet(books, "//catalog/book/description/text()"), xmlValue)
output <- data.frame(book_titles,author_names,description)
write.table(output, "xmlout.csv", sep="\t")
```

more on XML

- I just took a simple example, but you can do many more complex things with XML package in R.
- If you end up working with lot of xml files for some reason in future, remember to visit the package documentation: `https://cran.r-project.org/web/packages/XML/XML.pdf`
- Another tutorial on XML parsing in R: `https://www.stat.berkeley.edu/~statcur/Workshop2/Presentations/XML.pdf`
- I think Chapter 10 in textbook is complex to understand - so pointing to these resources.
- Literature students - you may see more corpora in XML format, if we have to go by the author.

HTML

- HyperText Markup Language - used to display webpages
- the markup tells a webpage how to display text (font, bold, url, image etc)
- Looks similar to XML, but different, and it serves a different purpose.

Reading HTML files in R

```
library(XML)
url <- "http://radar.oreilly.com/2011/09/building-data-science-teams.html"
doc <- htmlParse(url, useInternal = TRUE)
links <-xpathSApply(doc, "//a/@href")
```

Assignment 1 - reminder/description

- deadline: 27th Jan
- Two questions - first one is writing a brief lit. review
- Second question - small questions in R, which you should be able to do if you followed the class.

Next Class

- Continuation of today's topic
- Some hands on practice
- Storing your lines of code in .R files