LING 410X: Language as Data

Semester: Spring '18

Instructor: Sowmya Vajjala

Iowa State University, USA

13 February 2018

Class Outline

- Recap of last week and how to continue on those topics
- Searching for words and their context in text.
- Getting ngrams from text
- Assignment 3 description

Note: Assignment 2 discussion on Thursday, not Today.

Recap

Recap of last week

- Text analysis concepts:
  - Lexical variety: mean word frequency, type-token ratio, hapax richness (percentage of words that appear only once)

Recap of last week

- Text analysis concepts:
  - Lexical variety: mean word frequency, type-token ratio, hapax richness (percentage of words that appear only once)
- R concepts:
  1. Data structures in R: vectors, lists, matrices, arrays, dataframes, factors

Recap of last week

- Text analysis concepts:
  - Lexical variety: mean word frequency, type-token ratio, hapax richness (percentage of words that appear only once)
- R concepts:
  1. Data structures in R: vectors, lists, matrices, arrays, dataframes, factors
  2. Usage of built-in R functions such as: lapply, unname, sum, mean, scale

Recap of last week

- Text analysis concepts:
  - Lexical variety: mean word frequency, type-token ratio, hapax richness (percentage of words that appear only once)
- R concepts:
  1. Data structures in R: vectors, lists, matrices, arrays, dataframes, factors
  2. Usage of built-in R functions such as: lapply, unname, sum, mean, scale
  3. Writing our own R functions

Recap of last week

- Text analysis concepts:
  - Lexical variety: mean word frequency, type-token ratio, hapax richness (percentage of words that appear only once)
- R concepts:
  1. Data structures in R: vectors, lists, matrices, arrays, dataframes, factors
  2. Usage of built-in R functions such as: lapply, unname, sum, mean, scale
  3. Writing our own R functions
- Reporting analysis process and results using R Markdown

How to work further on these

- I covered Chapters 6 and 7 from the text book during last week. Go through those chapters, do the exercises at the end of the chapter.
- Revisit the slides, and understand the lines of R code shown in class.
- Try to do it yourself, taking some other text file or
- converting the programs to work on a collection of files instead of one file (How??)
- Do the lab exercises whenever you find time, if you did not finish them in lab.
- Think in terms of functions you can create and reuse for the rest of the class.
- Ask questions, use discussion forums and office hours.

Key Words In Context (KWIC)

(based on Chapters 8 and 9 in the textbook)

What is KWIC?

- KWIC is a way of searching through a text where we do not end our question at: "Where is word X appearing?" but ask for more: "What is the context in which X is appearing?"
- What does context mean?

What is KWIC?

- KWIC is a way of searching through a text where we do not end our question at: "Where is word X appearing?" but ask for more: "What is the context in which X is appearing?"
- What does context mean?
- "Context" just refers to the words surrounding X. e.g., two words before and three words after X.
- The analysis methods we looked at so far saw words as individual entities without bothering about their context.
- This week, we will study how to extract this context information in R.

Before we move on ...

What is this function doing?

```
processfile <- function(file_name)
{
  text <- scan(file_name, what = "character", sep = "\n")
  text_as_string <- tolower(paste(text, collapse = " "))
  text_as_string <- gsub("([[:punct:]])", " ", text_as_string)
  text_as_string <-  gsub(" +", " ", text_as_string)
  return(unlist(strsplit(text_as_string, " ")))
}
```

What will processfile(”somefilepath”) give me?

Continuing from last slide:

Try to understand what is happening in this:

```
words <- processfile("DollsHouse-Eng.txt")
helmer <- which(words == "helmer")
for(i in 1:length(helmer))
{
  start <- helmer[i]-2
  end <- helmer[i]+2
  cat(paste(words[start:end], sep=" "))
  cat("\n")
  #What does cat do??
}
```

Converting the previous slide into a function

```
getKwic <- function(wordsvector, word)
{
  word_index <- which(wordsvector == word)
  for(i in 1:length(word_index))
  {
    start <- word_index[i]-2
    end <- word_index[i]+2
    cat(paste(wordsvector[start:end], sep=" "))
    cat("\n")
  }
}
```

Interact with the user

- Why?

Interact with the user

- Why?
- How? - using readline() function.

```
file <- readline("Enter the name of the file you want to search in: ")
word <- readline("Enter the word for which you want the context: ")
context0 <- readline("Enter the number of words before and after
                     you want to print: ")
context <- strtoi(readline("Enter the number of words before
                    and after you want to print: "))
```

- What will happen now?
- What is the difference between context0 and context?

Modified getKwic() function

- The previous getKwic() function we discussed printed out words with a context window of 2 words.
- It has to be changed to take context from the user input.

```
getKwic <- function(wordsvector, word, context)
{
  word_index <- which(wordsvector == word)
  for(i in 1:length(word_index))
  {
    start <- word_index[i] - context
    end <- word_index[i] + context
    cat(paste(wordsvector[start:end], sep=" "))
    cat("\n")
  }
}
```

Everything looks good so far

... but

- What if the the word we are searching for is the very first word?

Everything looks good so far

... but

- What if the the word we are searching for is the very first word?
- a small detour: if I have a vector:
  $words < -c(''Robert'', ''Rose'', ''Ryan'', ''Richard'')$
- What are:

```
words[1]
words[2]
words[0]
words[5]
words[-1]
words[-2]
words[1:3]
words[-2:3]
words[1:7]
```

with that knowledge ...

- Let us come back to the same question: What if the the word we are searching for is the very first word?

with that knowledge ...

- Let us come back to the same question: What if the the word we are searching for is the very first word?
- If I am looking for the first word, and am looking for a context of say 3 words, I will get an error at the very beginning, and the R programs stops.
- If I am looking for the last word, and am looking for a context of say 3 words. What happens???
- What should we do to avoid these situations??

final getKwic() function

```
getKwic <- function(wordsvector, word, context)
{
  word_index <- which(wordsvector == word)
  for(i in 1:length(word_index))
  {
    start <- word_index[i] - context
    if(start < 1)
    {
      start <- 1
    }
    end <- word_index[i] + context
    if(end >= length(wordsvector))
    {
      end <- length(wordsvector)
    }
   # cat(start, end) #This prints only positions, not actual words.
    cat(paste(wordsvector[start:end], sep=" "))
    cat("\n")
  }
}
```

Note: Final R file for today (KWIC.R) and will be uploaded to Canvas.

How to continue from here?

- Improve the pre-processing
- Accept a different context length for before and after (e.g., 2 words before, 3 words after instead of 2 words before and 2 words after)
- Make this work with more than one file
- Add the functionality of taking the path to a folder, reading all .txt files from that folder, and do this for each file.
- Take a doc or pdf, extract plain-text from that and do this
- Exercises 8.1, 8.2 and 9.1, 9.2 in the textbook (solutions are available in the supplementary material).

.... and so on. (You will do some of this on thursday)

Ngram analysis

What are Ngrams?

- an n-gram is a ordered sequence of words. n- refers to the number of words in the sequence.
- unigrams - single words, bigrams - two word sequences, trigrams - three word sequences, 4grams - four word sequences and so on.
- If I have this sentence: "This is an example sentence", what are all possible ngrams in this??

What are Ngrams?

- an n-gram is a ordered sequence of words. n- refers to the number of words in the sequence.
- unigrams - single words, bigrams - two word sequences, trigrams - three word sequences, 4grams - four word sequences and so on.
- If I have this sentence: "This is an example sentence", what are all possible ngrams in this??
- Until now, we looked at only words (their frequencies, their position of appearance, their context of appearance)
- n-gram analysis is about moving beyond words and looking for patterns of word sequences.

ngram package in R

- This package provides a lot of functions to automatically create and analyse ngrams from text strings (STRINGS).
- The package has several advanced functionalities, we don't need at this point.
- We will only talk about how to extract ngrams of varying sizes from the text, count their frequencies etc.
- installation: install "ngram" package following the usual procedure.
- Enthusiastic students can have a look at the documentation for this package to know about all functionalities it has.

working with ngram package

```
library(ngram)
#usual pre-processing for the file first.
dollshouse_text <- scan("DollsHouse-Eng.txt", what = "character", sep = "\n")
dollshouse_string <- paste(dollshouse_text, collapse = " ")
dollshouse_string <- tolower(dollshouse_string)
dollshouse_string <- gsub("[[:punct:]]", " ", dollshouse_string)
dollshouse_string <- gsub(" +", " ", dollshouse_string)

#Three most useful functions for us:
trigrams <- ngram(dollshouse_string,n=3)
trigrams_vector <- get.ngrams(trigrams)
head(get.phrasetable(trigrams),n = 10)
```

note: Ngram.R file is on Canvas.

A detour into vectors (again!)

- R has functions such as union() and intersect() which takes two vectors and returns:
  - union(vector1, vector2) returns a vector that has all items that occurred in either vector1 or vector2.
  - intersection(vector1, vector2) returns a vector that has all items that occured in BOTH the vectors.

Union-Intersection example

```
a = c(1,4,44,5,12)
b = c(1,3,5,44,2)
c <- intersect(a,b)
d <- union(a,b)
c
[1]  1 44  5
d
[1]  1  4 44  5 12  3  2
```

Note: These can be string vectors as well.
Hint: You can use one of these functions to get part of the answer in Question 2 of Assignment 3.

Assignment 3 Description

- Topics: Last week, and This Week's content (Chapters 6–9 in Textbook)
- Questions: 2 questions (4% + 6% of your grade)
- Deadline: 24th February
- Description: On Canvas

Thursday

- Assignment 2 discussion
- Practice exercises for using what we learned today (KWIC and N-grams)
- To do: Read this article "Data Mining reveals the rise of ISIS propaganda on Twitter" (`https://goo.gl/QqTT9k`). We will start the class with a discussion on that on thursday