16 Jan 2018 - Most Frequent Words in a Text

Today, I am going to show you how to read a text file into R environment, calculate the number of characters, words, and plot the frequencies of words in the text. I will use two texts for this purpose: "A Doll's House", a play by a Norwegian author Henrik Ibsen in two versions: English and Esperanto translations. Both the translations are freely available on Gutenberg.org. This is loosely based on Chapter 2 of the textbook. I have simplified a little bit (I hope!)

First, let me start with setting my working directory to where these files are on my computer:

```
setwd("/home/bangaru/Dropbox/ClassroomSlides-BothCourses/LING410X/Week2Materials")
```

This folder has two .txt files, as you can see. First task is to "Load" these files into my R environment so that I can start analysing them. We do this using the scan function in R (there are other ways too. We will discuss later.) When you do this, you ofcourse should set your working directory to where your files are. Not my path! :)

```
english <- scan("DollsHouse-Eng.txt", what = "character", sep = "\n")
esperanto <- scan("DollsHouse-Esperanto.txt", what = "character", sep = "\n", encoding = "UTF-8")
```

What we are doing here is saying R to read these files character by character, and breakup the text into lines based on wherever a newline character is seen in text. Newline character is indicated by "\n" and store each file as a character vector.

```
english[1]
```

```
## [1] "The Project Gutenberg EBook of A Doll's House, by Henrik Ibsen"
```

```
esperanto[1]
```

```
## [1] "The Project Gutenberg EBook of Puphejmo, by Henrik Ibsen"
```

-these print the first lines of both english and esperanto vectors.

There is a lot of metadata at the beginning and end of Gutenberg.org texts which indicate licensing information and other such stuff. We don't need it here. One simple way to eliminate those words especially when you have only one or two text files is to just inspect the documents manually, look for the starting line, and then tell R to start from there.

```
english.start <- which(english ==  "DRAMATIS PERSONAE")
english.end <- which(english == "(The sound of a door shutting is heard from below.)")
english.start
```

```
## [1] 15
```

```
english.end
```

```
## [1] 2621
```

```
actual_english <- english[english.start:english.end]

esperanto.start <- which(esperanto == "Tradukis")
esperanto.end <- which(esperanto == "(El malsupre oni aŭdas la bruon de pordo, kiu frape ŝlosiĝas.)")
actual_esperanto <- esperanto[esperanto.start:esperanto.end]
```

These two variables now contain the actual play in English and Esperanto respectively, with all the line breaks preserved. However, if our task is to just have a look at words and frequencies, we don't need line breaks. We can use the paste() function to eliminate line breaks and convert the whole text into one string.

```
actual_english_string <- paste(actual_english, collapse = " ")
actual_esperanto_string <- paste(actual_esperanto, collapse = " ")
```

Now that I have all the text in one string, here is what I want to do: I want to lowercase the text (why?)

```
english_lower <- tolower(actual_english_string)
esperanto_lower <- tolower(actual_esperanto_string)
```

Next task is to split the string into words. "\W+" is a regular expression that matches "one or more non word characters" in a long string. We use that to split the string into words, such that all punctuation markers and white spaces are removed, and not considered for our further calculations.

```
english_words <- strsplit(english_lower, "\\W+")
esperanto_words <- strsplit(esperanto_lower, "\\W+")
```

Now, from here, I convert it into word-frequency table like this:

```
sorted_freqs_english <- sort(table(english_words), decreasing = TRUE)
sorted_freqs_esperanto <- sort(table(esperanto_words), decreasing = TRUE)
```

We can get the frequency of individual words in these texts using the below commands:

```
sorted_freqs_english["nora"]
```

```
## nora
##  709
```

```
sorted_freqs_english["helmer"]
```

```
## helmer
##    332
```

```
sorted_freqs_esperanto["nora"]
```

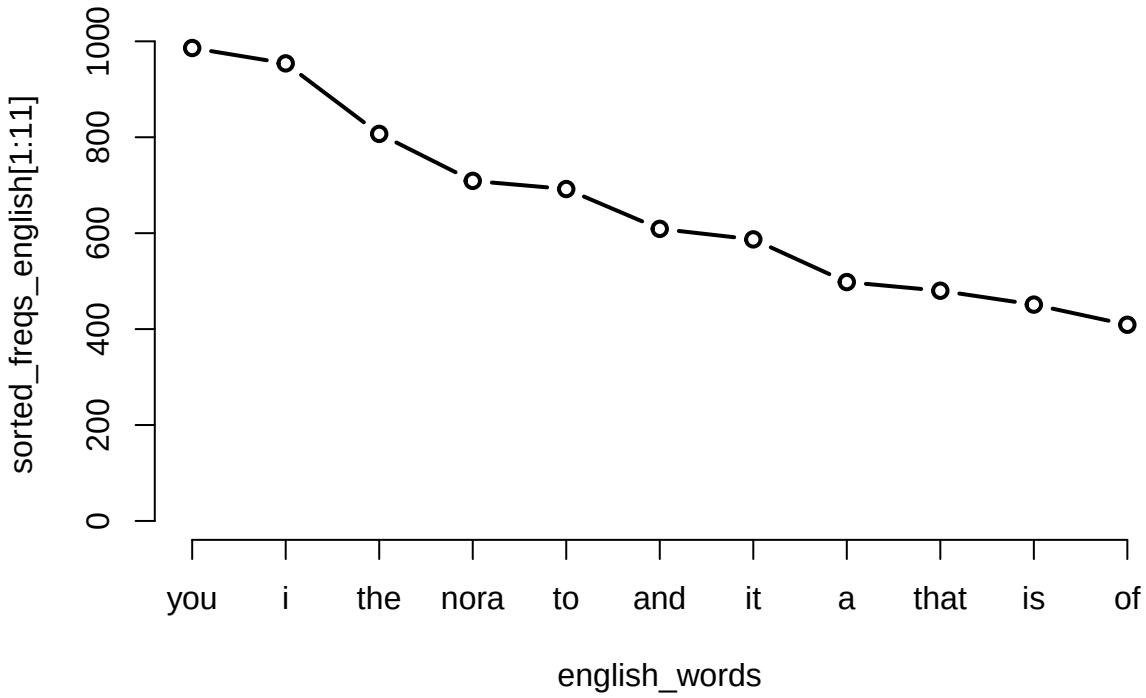
```
## nora
##  702
```

```
sorted_freqs_esperanto["helmer"]
```

```
## helmer
##    305
```

Now to the final plots (type = b indicates join the points by lines.)

```
plot(sorted_freqs_english[1:11], type="b")
```




```
plot(sorted_freqs_esperanto[1:11], type="b")
```