Quantitative Text Analysis
Exercise 3: Descriptive Statistics for Textual Data

23rd July 2014, Essex Summer School

Kenneth Benoit and Paul Nulty

In today's lab we will explore various descriptive statistics of text with R and `quanteda`.

Instructions

1. **Keywords-in-context**

   (a) We've already seen a basic way of searching text in R and on the terminal using `grep`. `quanteda` provides a keyword-in-context function that is easily usable and configurable to explore texts in a descriptive way. Type `?kwic` to view the documentation.

   (b) For speed, we will work with a subset of the corpus. The subset command will extract a sub-corpus that matches an expression. Use the command below to extract the 2010 speeches:

   ```
   data(iebudgets)
   iebudgets2010 <- subset(iebudgets, year==2010)
   ```

   (c) Load the Irish budget debate speeches and experiment with the `kwic` function, following the syntax specified on the help page for `kwic`. `kwic` can be used either on a character vector or a corpus object. Note that the `kwic` function returns a dataframe containing the document name and pre-keyword and post-keyword text for each result, so that you could assign this return value to a new object if you wished to save it. Try assigning the return value from `kwic` to a new object and then examine the dataframe you have assigned by clicking on it in the environment pane in RStudio.

   (d) Use the `kwic` function to discover the context of the word 'toxic'. Is this associated with environmental pollution?

   (e) Examine the context of words related to "disaster". Hint: you can use the stem of the word along with setting the `regex` argument to `TRUE`.

2. **Descriptive statistics**
   (Hint: for this section, note the following standard R functions: `colSums`, `rowSums`, `sort`, and `length`.)

   (a) We can extract basic descriptive statistics from a corpus from its document feature matrix. Make a dfm from the 2010 subset of the Irish budget speeches corpus.

   (b) Use standard R commands to calculate the total number of word types per document and, the total number of word tokens per document. Hint: Total word types can be obtained by coercing the Boolean `TRUE` value from the condition that a (word type > 0) to the integer 1 and summing these values across rows using `rowSums`.

   (c) What is the most frequent word in the corpus? (You can calculate this either (i) by sorting the sums of the columns in the dfm using standard R functions, or (ii) sorting the dfm with the `quanteda` function `dfmSort`.

(d) The `summary` function returns a dataframe containing a column indicating the number of sentences in each document. How many sentences occur in the corpus in total?

(e) `summary` `quanteda` provides a function to count syllables in a word — `countSyllables`. Try the function at the prompt. The code below will apply this function to all the words in the corpus, to give you a count of the total syllables in the corpus.

```
# count syllables from texts in the 2010 speech corpus
textSyls <- countSyllables(getTexts(iebudgets2010))
# sum the syllable counts
totalSyls <- sum(textSyls)
```

(f) One of the best known readability measures is the Flesch-Kincaid index. The formula is:

$$206.835 - 1.015\left(\frac{totaltokens}{totalsentences}\right) - 84.6\left(\frac{totalsyllables}{totaltokens}\right)$$

You should now have the values for these variables — calculate the Flesch-Kincaid index of the Irish budget speeches.

3. **Lexical Diversity over Time**

(a) We can plot the type-token ratio of the Irish budget speeches over time. To do this, begin by extracting a subset of iebudgets that contains only the first speaker from each year:

```
data(iebudgets)
finMins <- subset(iebudgets, no=="01")
```

(b) Get the type-token ratio for each text from this subset, and plot the resulting vector of ttrs.

4. **Zipf's Law**

(a) Zipf's Law states that the log of the rank of a words in a word frequency list has a linear relationship with the log of the frequency. Run the R code below to generate a plot demonstrating this for the Irish budget speeches (where the variable `total` contains the sorted total document frequencies).

```
# demonstrate Zipf's law - plot log frequency by log rank
plot(log10(1:100), log10(total[1:100]),
     xlab="log(rank)", ylab="log(frequency)", main="Top 100 Words")
```

(b) Zipf's law also suggests that the regression slope will be approximately -1.0. Check this using `lm` for linear regression.

```
# regression to check if slope is approx -1.0
regression <- lm(log10(total[1:100]) ~ log10(1:100))
summary(regression)
confint(regression)
```