Co-Occurrence of Words in the Shakespeare Drama Corpus

The following co-occurrence script aims to discover the semantic proximity of two words throughout the Shakespeare Drama Corpus. At the end, it will take in a word of the user's choice and find the top ten closest terms by proximity.

Global parameters

Set working directory by pointing to the location on your computer where you have stored the files. Below, we have chosen to Save the folder "RAnalysis" on the Desktop on a Mac. It contains all the other R scripts, texts, notebooks, and results. If you have branched the Github, simply note where you have save the folder. If you are on a PC, you will need to use an absolute path such as "C:Users:XXX."

Hint: Can't figure out your syntax? Click Session > Set Working Directory > Choose Directory, then select the Text_Analysis directory in which you are working. This will set your working directory in the console below while you are working here, but make sure to copy the path into the "setwd" command below to keep the directory constant if you close this script and reopen later. ***

```
setwd("~/Desktop/R/Text_Analysis/RNotebooks")
```

Include necessary packages for notebook R's extensibility comes in large part from packages. Packages are groups of functions, data, and algorithms that allow users to easily carry out processes without recreating the wheel. Some packages are included in the basic installation of R, others created by R users are available for download. Make sure to have the following packages installed before beginning so that they can be accessed while running the scripts. By calling library(packagename) they will be automatically loaded and ready to use.

The first three packages are used to render the RNotebook you are currently viewing:

knitr - Creates a formatted report from the script provided

markdown - a package used to render textual documents from plain text to others such as R and XML

rmarkdown - similar to markdown but specifically to render R documents

The next two packages are used within the co-occurrence script:

tm - this package provides tools (functions) for performing various types of text mining. In this script, we will use tm to performing text cleaning in order to have uniform data for analysis. Check out this link for the documentation!

RWeka - another package providing tools for

```
library(knitr)
library(markdown)
library(rmarkdown)
library(tm)
```

```
## Loading required package: NLP
```

```
library(RWeka)
```

```
## Warning: package 'RWeka' was built under R version 3.2.4
```

```
corpus <- Corpus(DirSource("~/Desktop/R/Text_Analysis/data/shakespeareFolger"))
```

Create a corpus

Clean the corpus To clean the corpus in this example, we are using the tm package's built in tools to: 1) lowercase all words, 2) remove stopwords ("a" "an" "the"), 3) remove any other words the tm package might not consider a stopword (here Shakespeare's use of "tis" and "hath" would not appear in the modern English stopword list) 4) remove punctuation 5) strip out any whitespace between words

```
corpus <- tm_map(corpus, content_transformer(tolower))
corpus <- tm_map(corpus, removeWords, stopwords("SMART"))
corpus <- tm_map(corpus, removeWords, c("tis", "hath"))
corpus <- tm_map(corpus, removePunctuation)
corpus <- tm_map(corpus, stripWhitespace)
```

Processing After the corpus has been cleaned, we can now begin to process the text. The next step uses the DocumentTermMatrix() function again from the tm package. This function creates a matrix where each document is a row and the terms in the text make up the columns, saved here as "dtm." The rest of the matrix consists of the frequencies for each term within each document.

```
dtm <- DocumentTermMatrix(corpus)
```

Once we've created the Document-Term Matrix, we need to find the overall frequency of each term across the corpus. Here, we get the sums of the columns (colSums) within the Document-Term Matrix and save it as a sorted numeric vector called "freq."

```
freq <- sort(colSums(as.matrix(dtm)), decreasing = TRUE)
```

Results Finally, we use the tm function findAssocs to find the top ten associations with any word we choose. The example below is the word "father" but you can choose any word as you are learning, or can even have multiple lines returning multiple associations (uncomment the line below the "father" association line)

```
findAssocs(dtm, "father", .6)
```

```
## $father
##        son     escape       gale      means   mildness       nero
##       0.70       0.69       0.68       0.68       0.64       0.64
##       rest      straw unconstant      burst       buzz      bandy
##       0.64       0.64       0.64       0.63       0.63       0.62
##      fares       life
##       0.61       0.61
```

```
#findAssocs(dtm, "love", .6)
```