Top Ten Most Frequent Words in the Hamlet

The following script finds the ten most frequent words in Shakespeare's Hamlet. Many times top ten analyses are used to find the top ten words overall, but this particular script strips out stopwords so the resulting top ten words shed light on the top ten thematic words in the text.

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 package is used within the top ten frequency 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!

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

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

Scan in the text

Uses R's "scan" function to read in the text and is then saved as a variable called "text_raw"

```
text_raw<-scan("~/Desktop/R/Text_Analysis/data/shakespeareFolger/Hamlet.txt", what="character", sep="\n
```

Save the text as a corpus object

The "Corpus" function from tm reads in the vector we scanned earlier and saves it as a corpus object.

```
corpus <- Corpus(VectorSource(text_raw))
```

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) Build a stopword list that uses a standard English list concatenated with an Early Modern list that has words like "hath" and "thy" in it. 4) remove punctuation 5) strip out any whitespace between words 6) converts the corpus object to a Plain Text document

```
corpus <- tm_map(corpus, content_transformer(tolower))
#Add early modern stopwords by adding "myStopWords"
myStopWords <- scan("~/Desktop/R/Text_Analysis/data/earlyModernStopword.txt", what="character", sep="\n
corpus <- tm_map(corpus, removeWords, c(stopwords("SMART"), myStopWords))
#To change the stopword list, use other dictionaries available with the tm package
#Add early modern stopwords by u adding "myStopWords" to line 19
myStopWords <- scan("~/Desktop/R/Text_Analysis/data/earlyModernStopword.txt", what="character", sep="\n
corpus <- tm_map(corpus, removeWords, c(stopwords("SMART"), myStopWords))
corpus <- tm_map(corpus, removePunctuation)
corpus <- tm_map(corpus, stripWhitespace)
corpus <- tm_map(corpus, PlainTextDocument)
```

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 can plot and view the data as a line graph with the top ten words along the x-axis and the frequency of appearance along the y-axis.

```
plot(head(freq, 10), type="b", lwd=2, col="blue", col.lab="red",
     main="Hamlet, Entire Play", xlab="Top Ten Words", ylab="Number of Occurences", xaxt="n",)
axis(1,1:10, labels=names(head(freq, 10)))
```