tidydata_attempt

Michael Pearson

12/21/2017

Let's begin by loading the libraries and reading the files on my computer where the Swiftkey zips have been unzipped.

```
knitr::opts_chunk$set(echo = TRUE)
library(tidytext, quietly = TRUE)
library(dplyr, quietly = TRUE)
library(readr, quietly = TRUE)
library(R.utils, quietly = TRUE)
library(tm, quietly = TRUE)
library(SnowballC, quietly = TRUE)
library(ggplot2, quietly = TRUE)
library(tidyr, quietly = TRUE)
library(ggraph, quietly = TRUE)
library(igraph, quietly = TRUE)
library(data.table, quietly = TRUE)
eng_news <- read_file("/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone Project/
eng_blogs <- read_file("/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone Projec
eng_twitter <- read_file("/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone Proj
blog_us <- file("/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone Project/files/
tweet_us <- file("/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone Project/files
news_us <- file("/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone Project/files/
newslines <- countLines("/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone Proje
bloglines <- countLines("/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone Proje
tweetlines <- countLines("/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone Proj
blog25 <- readLines(blog_us, 25)
close(blog_us)
tweet25 <- readLines(tweet_us, 25)
close(tweet_us)
news25 <- readLines(news_us, 25)
close(news_us)
```

Exploring the data

Some basic examination of the three text sources: the News, the Blogs, the Tweets.

The news file has 205,243,643 characters, and 1,010,242 lines of text. The news file is 196.2775 MegaBytes.

The blog file has 208,623,081 characters, and 899,288 of text. The blog file is 200.4242 MegaBytes

The twitter file has 11,790,868 characters and 2,360,148 of text. The twitter file is 159.3641 MegaBytes

Beginning of text processing

```
newzchar <- nchar(news25)
sumnew <- sum(newzchar)
```

```
tweetchar <- nchar(tweet25)
sumtweet <- sum(tweetchar)
blogchar <- nchar(blog25)
sumblog <- sum(blogchar)
```

Begin the exploration by loading the full texts of the Swiftkey files...

```
tweet_us <- file("/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone Project/files
tweet_all <- readLines(tweet_us, n= tweetlines, warn = FALSE, encoding = "UTF=8", skipNul = TRUE)
close(tweet_us)
love_it <- length(grep("love", tweet_all))
hate_it <- length(grep("hate", tweet_all))
phrase_it <- length(grep("A computer once beat me at chess, but it was no match for me at kickboxing",
blog_us <- file("/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone Project/files/
blog_all <- readLines(blog_us, n= bloglines, warn = FALSE, encoding = "UTF=8", skipNul = TRUE)
close(blog_us)
news_us <- file("/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone Project/files/
news_all <- readLines(news_us, n = newslines, warn = FALSE, encoding = "UTF=8", skipNul = TRUE)
romance <- length(grep("romantic date", news_all))
close(news_us)
long_tweet <- max(nchar(tweet_all[1:tweetlines]))
long_news <- max(nchar(news_all[1:newslines]))
long_blog <- max(nchar(blog_all[1:bloglines]))
```

More exploration of the texts

Some of these are from the quiz for the first week of the Capstone Project...

The phrase "A computer once beat me at chess, but it was no match for me at kickboxing" occurs 3 times in the twitter sample.

The word 'love' occurs 90,956 times in the twitter sample.

The word 'hate' occurs 22,138 times in the twitter sample.

The longest line in the twitter sample is 140 characters. Duh!

The longest line in the blog sample is 40,833 characters.

The longest line in the news sample is 11,384 characters.

The phrase "romantic date" occurs 5 times.

Getting rid of the profanity

I got a list of profanity from Google: full-list-of-bad-words-banned-by-google-txt-file_2013_11_26_04_53_31_867.txt

I will use this file to filter profanity from my sample of the corpus.

Now let's create a sample of 12% of the text, load it into a corpus.

Then we will remove profanity (the 'badwords' file from Google), tidy the corpus using the 'tidytext' package - which follows tidy data procedures and makes one variable per column.

I use the file "full-list-of-bad-words-banned-by-google-txt-file_2013_11_26_04_53_31_867.txt" to create a list of profanity to remove from the samples.

We will remove non alphabetic characters, remove blanks, and then count word frequencies, and create tidy data frames for bigrams and trigrams.

```
##badwords <- readLines("/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone Proje
set.seed(1151960)
samp_per <- 0.20
sam_twit <- tweet_all[sample(1:length(tweet_all),samp_per*length(tweet_all))]
write_lines(sam_twit, "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone Project/
sam_news <- news_all[sample(1:length(news_all),samp_per*length(news_all))]
write_lines(sam_news, "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone Project/
sam_blog <- blog_all[sample(1:length(blog_all),samp_per*length(blog_all))]
write_lines(sam_blog, "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone Project/
sam_Corpus <- VCorpus(DirSource("/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capsto
```

```
## Warning in as.POSIXlt.POSIXct(Sys.time(), tz = "GMT"): unknown timezone
## 'zone/tz/2017c.1.0/zoneinfo/America/Los_Angeles'
```

```
sam_tidy <- tidy(sam_Corpus)
data("stop_words")
tidy_sentences <- data.table(sam_tidy) %>% unnest_tokens(sentences, text, token = "sentences")
text_tokens <- data.table(sam_tidy) %>% unnest_tokens(word, text, token = "words")
text_tokens$word <- gsub("[^[:alpha:] | ^[:punct:]]" , " ", text_tokens$word)
text_tokens$word   <- gsub("-", " ", text_tokens$word)
tidy_sentences$sentences <- gsub("'ve ", " have ", tidy_sentences$sentences)
tidy_sentences$sentences <- gsub("'ll ", " will ", tidy_sentences$sentences)
tidy_sentences$sentences <- gsub("'re ", " are ", tidy_sentences$sentences)
tidy_sentences$sentences <- gsub("'d ", " had ", tidy_sentences$sentences)
tidy_sentences$sentences <- gsub(" i'm ", "i am ", tidy_sentences$sentences)
tidy_sentences$sentences <- gsub(" im ", "i am ", tidy_sentences$sentences)
tidy_sentences$sentences <- gsub(" won't ", " will not ", tidy_sentences$sentences)
tidy_sentences$sentences <- gsub("n't ", " not ", tidy_sentences$sentences)
tidy_sentences$sentences <- gsub(" ur ", " your ", tidy_sentences$sentences)
tidy_sentences$sentences <- gsub(" tits ", " breasts ", tidy_sentences$sentences)
tidy_sentences$sentences <- gsub(" fuck ", " intercourse ", tidy_sentences$sentences)
tidy_sentences$sentences <- gsub(" shit ", " feces ", tidy_sentences$sentences)
tidy_sentences$sentences <- gsub(" piss ", " urine ", tidy_sentences$sentences)
tidy_sentences$sentences <- gsub(" cunt ", " vagina ", tidy_sentences$sentences)
tidy_sentences$sentences <- gsub(" pussy ", " vagina ", tidy_sentences$sentences)
tidy_sentences$sentences <- gsub("[^[:alpha:]]", " ", tidy_sentences$sentences)
tidy_sentences$sentences <- gsub("-", " ", tidy_sentences$sentences)
tidy_sentences$sentences <- gsub("  ", " ", tidy_sentences$sentences)
write.csv(tidy_sentences$sentences, file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Cours
```

And now the sample space

```
samp_per <- 2*samp_per
set.seed(12212017)
sam_twit <- tweet_all[sample(1:length(tweet_all),samp_per*length(tweet_all))]
write_lines(sam_twit, "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone Project/
sam_news <- news_all[sample(1:length(news_all),samp_per*length(news_all))]
write_lines(sam_news, "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone Project/
```

```
sam_blog <- blog_all[sample(1:length(blog_all),samp_per*length(blog_all))]
write_lines(sam_blog, "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone Project/
test_corpus <- VCorpus(DirSource("/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capsto
test_tidy <- tidy(test_corpus)
test_sentences <- data.table(test_tidy) %>% unnest_tokens(sentences, text, token = "sentences")
test_sentences$sentences <- gsub("'ve ", " have ", test_sentences$sentences)
test_sentences$sentences <- gsub("'ll ", " will ", test_sentences$sentences)
test_sentences$sentences <- gsub("'re ", " are ", test_sentences$sentences)
test_sentences$sentences <- gsub("'d ", " had ", test_sentences$sentences)
test_sentences$sentences <- gsub(" i'm ", " i am ", test_sentences$sentences)
test_sentences$sentences <- gsub(" im ", " i am ", test_sentences$sentences)
test_sentences$sentences <- gsub(" won't ", " will not ", test_sentences$sentences)
test_sentences$sentences <- gsub("n't ", " not ", test_sentences$sentences)
test_sentences$sentences <- gsub(" ur ", " your ", test_sentences$sentences)
test_sentences$sentences <- gsub("[^[:alpha:]]", " ", test_sentences$sentences)
test_sentences$sentences <- gsub("-", " ", test_sentences$sentences)
test_sentences$sentences <- gsub("  ", " ", test_sentences$sentences)
write.csv(test_sentences$sentences, file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Cours
```

back to making a corpus

```
text_tokens <- subset(text_tokens, word !="")
write.csv(text_tokens, file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone
word_sen <- tidy_sentences %>% unnest_tokens(word, sentences) %>% mutate(word, wordStem(word))
write.csv(word_sen, file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone Pro
##stopit <- stop_words[stop_words$lexicon=="snowball",]
text_count <- text_tokens %>% count(word, sort = TRUE)
write.csv(text_count, file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone P
rm(text_count)
word_count <- word_sen %>% count(word, sort = TRUE)
write.csv(word_count,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone Pr
rm(word_count)
```

make the n-grams for the sample corpus

```
text_bigrams <- sam_tidy %>% unnest_tokens(bigram, text, token = "ngrams", n=2)
bigram_count <- text_bigrams %>% count(bigram, sort = TRUE)
text_bigrams <- text_bigrams$bigram
write.csv(text_bigrams,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone
rm(text_bigrams)
write.csv(bigram_count,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone
rm(bigram_count)
sen_bigrams <- tidy_sentences %>% unnest_tokens(bigrams, sentences, token = "ngrams", n = 2)
sen_bigram_count <- sen_bigrams %>% count(bigrams, sort = TRUE)
sen_bigrams <- sen_bigrams$bigrams
write.csv(sen_bigrams,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone P
rm(sen_bigrams)
write.csv(sen_bigram_count,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capst
rm(sen_bigram_count)
test_bigrams <- test_sentences %>% unnest_tokens(bigrams, sentences, token = "ngrams", n = 2)
```

```
test_bigram_count <- test_bigrams %>% count(bigrams, sort = TRUE)
test_bigrams <- test_bigrams$bigrams
write.csv(test_bigrams,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone
rm(test_bigrams)
write.csv(test_bigram_count,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Caps
rm(test_bigram_count)
```

trigrams

```
text_trigrams <- sam_tidy %>% unnest_tokens(trigram, text, token = "ngrams", n=3)
trigram_count <- text_trigrams %>% count(trigram, sort = TRUE)
text_trigrams <- text_trigrams$trigram
write.csv(text_trigrams,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone
rm(text_trigrams)
write.csv(trigram_count,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone
rm(trigram_count)
sen_trigrams <- tidy_sentences %>% unnest_tokens(trigrams, sentences, token = "ngrams", n = 3)
sen_trigram_count <- sen_trigrams %>% count(trigrams, sort = TRUE)
sen_trigrams <- sen_trigrams$trigrams
write.csv(sen_trigrams,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone
rm(sen_trigrams)
write.csv(sen_trigram_count,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Caps
rm(sen_trigram_count)
test_trigrams <- test_sentences %>% unnest_tokens(trigrams, sentences, token = "ngrams", n = 3)
test_trigram_count <- test_trigrams %>% count(trigrams, sort = TRUE)
test_trigrams <- test_trigrams$trigrams
write.csv(test_trigrams,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone
rm(test_trigrams)
write.csv(test_trigram_count,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Cap
rm(test_trigram_count)
```

quadgrams

```
text_quadgrams <- sam_tidy %>% unnest_tokens(quadgram, text, token = "ngrams", n=4)
quadgram_count <- text_quadgrams %>% count(quadgram, sort = TRUE)
text_quadgrams <- text_quadgrams$quadgram
write.csv(text_quadgrams,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capsto
rm(text_quadgrams)
write.csv(quadgram_count,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capsto
rm(quadgram_count)
sen_quadgrams <- tidy_sentences %>% unnest_tokens(quadgrams, sentences, token = "ngrams", n = 4)
sen_quadgram_count <- sen_quadgrams %>% count(quadgrams, sort = TRUE)
sen_quadgrams <- sen_quadgrams$quadgrams
write.csv(sen_quadgrams,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone
rm(sen_quadgrams)
write.csv(sen_quadgram_count,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Cap
rm(sen_quadgram_count)
test_quadgrams <- test_sentences %>% unnest_tokens(quadgrams, sentences, token = "ngrams", n = 4)
test_quadgram_count <- test_quadgrams %>% count(quadgrams, sort = TRUE)
test_quadgrams <- test_quadgrams$quadgrams
```

```
write.csv(test_quadgrams,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstor
rm(test_quadgrams)
write.csv(test_quadgram_count,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Ca
rm(test_quadgram_count)
```

quingrams

```
text_quingrams <- sam_tidy %>% unnest_tokens(quingram, text, token = "ngrams", n=5)
quingram_count <- text_quingrams %>% count(quingram, sort = TRUE)
text_quingrams <- text_quingrams$quingram
write.csv(text_quingrams ,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capst
rm(text_quingrams)
write.csv(quingram_count ,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capst
rm(quingram_count)
sen_quingrams <- tidy_sentences %>% unnest_tokens(quingrams, sentences, token = "ngrams", n = 5)
sen_quingram_count <- sen_quingrams %>% count(quingrams, sort = TRUE)
sen_quingrams <- sen_quingrams$quingrams
write.csv(sen_quingrams,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstone
rm(sen_quingrams)
write.csv(sen_quingram_count,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Cap
rm(sen_quingram_count)
test_quingrams <- test_sentences %>% unnest_tokens(quingrams, sentences, token = "ngrams", n = 5)
test_quingram_count <- test_quingrams %>% count(quingrams, sort = TRUE)
test_quingrams <- test_quingrams$quingrams
write.csv(test_quingrams,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Capstor
rm(test_quingrams)
write.csv(test_quingram_count,file = "/Users/mutecypher/Documents/Documents - Michael's iMac/Coursera/Ca
rm(test_quingram_count)
```

Count the words

Take a quick look at the head of the text count table.

```
text_count <- text_tokens %>% count(word, sort = TRUE)
head(text_count)
```

```
## # A tibble: 6 x 2
##    word      n
##   <chr>  <int>
## 1   the 951837
## 2    to 551948
## 3   and 483152
## 4     a 477001
## 5    of 402024
## 6     i 331233
```

Addressing some of the review criteria. . .

Find the number of words that would cover 90% the lexicon

Find the number of words that would cover 98% of it.

Total words are 20,454,558

I will now do the calculation to determine both of the above benchmarks.

```
text_count$sum <- cumsum(text_count$n)
ninetyfive <- which(text_count$sum > sum(text_count$n)*0.95)
ninetyeight <- which(text_count$sum > sum(text_count$n)*0.98)
```

Needed words are...

It would take 17,497 to cover 95% of all word instances of the 20,454,558 words in the sample corpus.

It would take 46,555 words to cover 98% of all word instances in the sample corpus.

Let's do some graphing, plotting and histogramming

Here's a bar plot of the 25 most common words in the sample

```
samp_plot <- text_count[1:25,]
g <- ggplot(data = samp_plot, aes(x=factor(word,levels = word),  y= n)) + geom_bar(stat="identity", fil
g<- g + theme(axis.text.x = element_text(angle = 90))
g
```

Now let's bar plot the top 25 most frequent bigrams

```
bigram_count[1:25,]
bigram_samp <- bigram_count[1:25,]
bigram_plot <- ggplot(data = bigram_samp, aes(x=factor(bigram, levels=bigram), y= n)) + geom_bar(stat="
bigram_plot <- bigram_plot + theme(axis.text.x = element_text(angle = 90))
bigram_plot
```

And now let's bar plot the top 25 most frequent trigrams

```
trigram_count[1:25,]
trigram_samp <- trigram_count[1:25,]
trigram_plot <- ggplot(data = trigram_samp, aes(x=factor(trigram, levels = trigram), y= n)) + geom_bar(
trigram_plot <- trigram_plot + theme(axis.text.x = element_text(angle = 90))
trigram_plot
```

Word Clouds

Let's make a word cloud of the top 50 words in the sample corpus

```
library(wordcloud)
text_tokens %>% count(word) %>% with(wordcloud(word, n, max.words = 50, rot.per = 0.55, colors = brewer
```

Bigram word clouds

Let's make a wordcloud using the top 25 bigrams

```
bigram_count  %>% with(wordcloud(bigram, n, max.words = 25, rot.per = 0.45, colors = brewer.pal(7, "Spe
```

Trigram word clouds

Let's make a wordcloud of the top 25 trigrams

```
trigram_count  %>% with(wordcloud(trigram, n, max.words = 25, rot.per = 0.35, colors = brewer.pal(11, "
```

Make a Markov chain of a few parts.