Text Data Report

Lynna

Sunday, March 29, 2015

Summary

The goal of this project is just to display that I've gotten used to working with the data and that you are on track to create your prediction algorithm. The content of this report includes: 1. Demonstration that I've downloaded the data and successfully loaded it in. 2. Basic report of summary statistics about the data sets. 3. Any interesting findings.

Loading the data

```
setwd("~/coursera/capstone/final/en_US")

con1 <- file("en_US.twitter.txt", "r")
con2 <- file("en_US.news.txt", "r")
con3 <- file("en_US.blogs.txt", "r")

twitter <- readLines(con1)
news <- readLines(con2)
```

```
## Warning in readLines(con2): incomplete final line found on
## 'en_US.news.txt'
```

```
blogs <- readLines(con3)
```

Summary statistics about the data sets

Twitter data is the largest amount of lines, however, blogs data have the most word count!

```
summary(twitter) #Twitter file has 2,360,148 lines
```

```
##    Length     Class      Mode
##   2360148 character character
```

```
sum(sapply(strsplit(blogs, " "), length)) #Blog file has 37,334,131 words
```

```
## [1] 37334131
```

Here is a summary table:

```
##             Length    Class       Mode        Word_Count
## twitterData "2360148" "character" "character" "30373543"
## newsData    "77259"   "character" "character" "2643969"
## blogsData   "899288"  "character" "character" "37334131"
```

Interesting findings

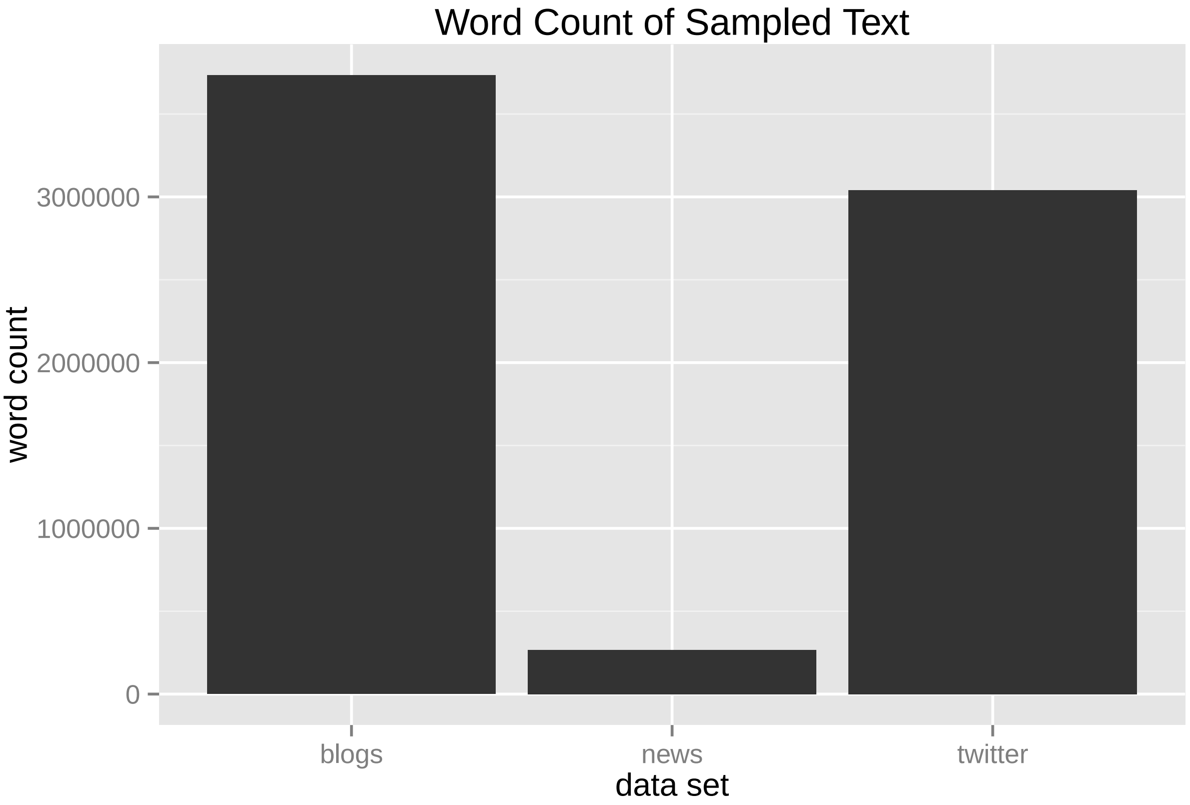
Since the data sets are so large, I took samples from each of the data set to do some data exploration. After taking line samples, we explore how many words are in each data set.

```r
#random line samples
randomNumT <- sample(length(twitter), length(twitter)*0.10) #10% of 2.3 million lines
twitter_training <- twitter[c(randomNumT)]

randomNumN <- sample(length(news), length(news)*0.10) #10% of 77,259 lines
news_training <- news[c(randomNumN)]

randomNumB <- sample(length(blogs), length(blogs)*0.10) #10% of 899,288 lines
blogs_training <- blogs[c(randomNumB)]
```

```
## Loading required package: plyr
##
## Attaching package: 'reshape'
##
## The following objects are masked from 'package:plyr':
##
##     rename, round_any
##
## Mapping a variable to y and also using stat="bin".
##   With stat="bin", it will attempt to set the y value to the count of cases in each group.
##   This can result in unexpected behavior and will not be allowed in a future version of ggplot2.
##   If you want y to represent counts of cases, use stat="bin" and don't map a variable to y.
##   If you want y to represent values in the data, use stat="identity".
##   See ?geom_bar for examples. (Deprecated; last used in version 0.9.2)
```



Even though twitter training data has the most lines, blogs training has the most words. Here is a table summary of the above graph:

```
##           value
## twitter 3039996
## news     265908
## blogs   3733305
```

For fun, let's see how many times the word "love" occurs in the blogs compared to in the twitter training data.

```
sum(grepl("[lL][oO][vV][eE]", blogs_training))
```

```
## [1] 5468
```

```
sum(grepl("[lL][oO][vV][eE]", twitter_training))
```

```
## [1] 11788
```

The word love showed up 11,753 times in the twitter training data set, and less than half as many times in the blogs data set!

```
sum(grepl("[lL][oO][vV][eE]", news_training))
```

```
## [1] 115
```

I didn't expect there to be any occurance of "love" in the news training data set, but there are some!

The end.