Capstone_Week2_Pjt

Banu Priya

24th Sep 2019

## Synopsis

This is the Milestone Report for week 2 of the Coursera Data Science Capstone project.

The objective of the report is to understand the various statistical propertties of dataset to build a prediction model for the final data product using Shiny Application.

The model will be trained using unified document corpus from the following sources:

a. Blogs
b. News
c. Twitter

The provided text data has four different languages. This project will only focus on the English language.

Environment Setup

Loading initial packages and clearing the global workspace (including hidden objects).

```r
library(knitr)
```

```
## Warning: package 'knitr' was built under R version 3.6.1
```

```r
rm(list = ls(all.names = TRUE))
setwd("C:/Users/b.s.priya/Documents/Data Science_R/Capstone")
```

Loading the Data

Download, unzip and load the training data.

```r
trainURL <- "https://d396qusza40orc.cloudfront.net/dsscapstone/dataset/Coursera-SwiftKey.zip"
trainDataFile <- "data/Coursera-SwiftKey.zip"
if (!file.exists('data')) {
  dir.create('data')
}
if (!file.exists("C:/Users/b.s.priya/Documents/Data Science_R/Capstone")) {
  tempFile <- tempfile()
  download.file(trainURL, tempFile)
  unzip(tempFile, exdir = "data")
  unlink(tempFile)
}

## blogs
blogsFileName <- "C:/Users/b.s.priya/Documents/Data Science_R/Capstone/en_US.blogs.txt"
con <- file(blogsFileName, open = "r")
```

Table 1:

File	FileSize	Lines	Characters	Words	WPL.Min	WPL.Mean	WPL.Max
en_US.blogs.txt	200 MB	899288	206824505	37570839	0	42	6726
en_US.news.txt	196 MB	77259	15639408	2651432	1	35	1123
en_US.twitter.txt	159 MB	2360148	162096241	30451170	1	13	47

```r
blogs <- readLines(con, encoding = "UTF-8", skipNul = TRUE)
close(con)

## news
newsFileName <- "C:/Users/b.s.priya/Documents/Data Science_R/Capstone/en_US.news.txt"
con <- file(newsFileName, open = "r")
news <- readLines(con, encoding = "UTF-8", skipNul = TRUE)
```

```
## Warning in readLines(con, encoding = "UTF-8", skipNul = TRUE): incomplete
## final line found on 'C:/Users/b.s.priya/Documents/Data Science_R/Capstone/
## en_US.news.txt'
```

```r
close(con)

## twitter
twitterFileName <- "C:/Users/b.s.priya/Documents/Data Science_R/Capstone/en_US.twitter.txt"
con <- file(twitterFileName, open = "r")
twitter <- readLines(con, encoding = "UTF-8", skipNul = TRUE)
close(con)
rm(con)
```

Basic Summary of Data

Prior to building the unified document corpus and cleaning the data, a basic summary of the three text corpora is being provided which includes file sizes, number of lines, number of characters, and number of words for each source file. Also included are basic statistics on the number of words per line (min, mean, and max).

Initial Data Summary

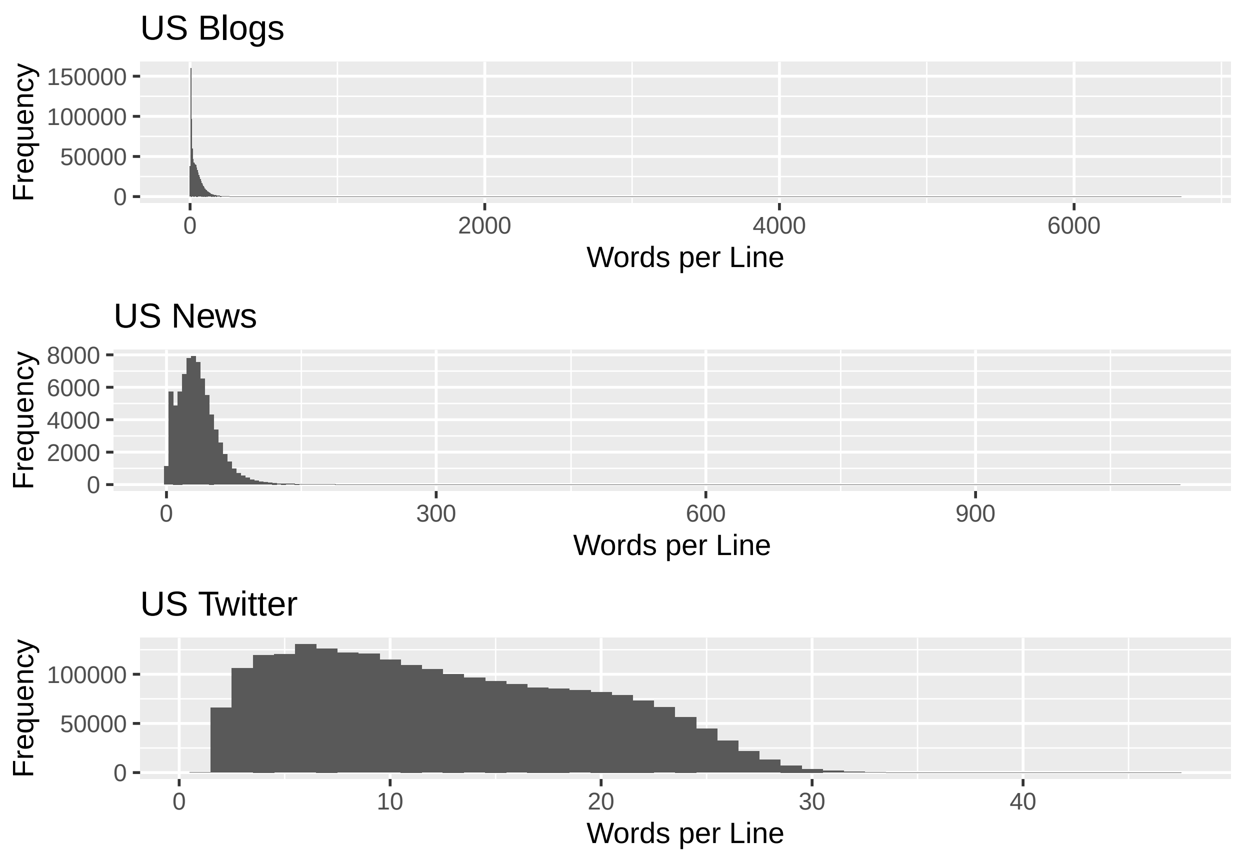
```
## Warning: package 'kableExtra' was built under R version 3.6.1
```

An initial investigation of the data shows that on average, each text has a relatively low number of words per line on. Blogs tend to have more words per line, followed by news and then twitter which has the least words per line. The lower number of words per line for the Twitter data is expected given that a tweet is limited to a certain number of characters. Even when Twitter doubled its character count from 140 to 280 characters in 2017, research shows that only 1% of tweets hit the 280-character limit, and only 12% of tweets are longer than 140 characters. Perhaps after so many years, users were simply trained to the 140-character limit.

Another important observation in this initial investigation shows that the text files are fairly large. To improve processing time, a sample size of 1% will be obtained from all three data sets and then combined into a unified document corpus for subsequent analyses later in this report as part of preparing the data.

Histogram of Words per Line

```
## Warning: package 'gridExtra' was built under R version 3.6.1
```



The relatively low number of words in the three source files charted earlier in this section is also visible in the histogram plots shown above. This observation seems to support a general trend towards short and concise communications that may be useful later in the project.

Prepare the Data

Prior to performing exploratory data analysis, the three data sets will be sampled at 1% to improve performance. All non-English characters will be removed from the subset of data and then combined into a single data set. The combined sample data set will be written to disk which contains 33,365 lines and 702,126 words.

The next step is to create a corpus from the sampled data set. A custom function named `buildCorpus` will be employed to perform the following transformation steps for each document:

1. Remove URL, Twitter handles and email patterns by converting them to spaces using a custom content transformer
2. Convert all words to lowercase
3. Remove common English stop words
4. Remove punctuation marks
5. Remove numbers
6. Trim whitespace

Table 2: First 10 Documents

freezer
liquid bottom bin usually run food decomposing can often smell sour can acidic plants recommend use liquid water plants
backs similar popular set except yellow instead green vertical just typical stats biography fun trivia questions answerssee n
chapter filing betsey johnson llc will see chains stores close person familiar matter said steven madden ltd owns betsey joh
fold unfinished edge front back panel place velcro folded edges sew sew velcro top panel onto wrong side fabric velcro bott
difference can viewed quantitatively also come undone garnered little yellow post slightly less recent selection book ruth ac
dont think authentic leadership skill position essential anyone wants bring best work life
fast
gave authority speak mankind know fact many humans dedicated pissdrinkers convinced fluids therapeutic properties celeb
chairman joint chiefs staff gen martin dempsey ordered inquiry april day canceled perspectives islam islamic radicalism tra

7. Remove profanity
8. Convert to plain text documents

The corpus will then be written to disk in two formats: a serialized R object in RDS format and as a text file. Finally, the first 10 documents (lines) from the corpus will be displayed.

Appendix

A.1 Basic Data Summary

Basic summary of the three text corpora.

```r
library(stringi)
library(kableExtra)
# assign sample size
sampleSize = 0.01
# file size
fileSizeMB <- round(file.info(c(blogsFileName,
                                newsFileName,
                                twitterFileName))$size / 1024 ^ 2)
# num lines per file
numLines <- sapply(list(blogs, news, twitter), length)
# num characters per file
numChars <- sapply(list(nchar(blogs), nchar(news), nchar(twitter)), sum)
# num words per file
numWords <- sapply(list(blogs, news, twitter), stri_stats_latex)[4,]
# words per line
wpl <- lapply(list(blogs, news, twitter), function(x) stri_count_words(x))
# words per line summary
wplSummary = sapply(list(blogs, news, twitter),
                    function(x) summary(stri_count_words(x))[c('Min.', 'Mean', 'Max.')])
rownames(wplSummary) = c('WPL.Min', 'WPL.Mean', 'WPL.Max')
summary <- data.frame(
  File = c("en_US.blogs.txt", "en_US.news.txt", "en_US.twitter.txt"),
  FileSize = paste(fileSizeMB, " MB"),
  Lines = numLines,
  Characters = numChars,
  Words = numWords,
  t(rbind(round(wplSummary)))
)
kable(summary,
      row.names = FALSE,
      align = c("l", rep("r", 7)),
      caption = "") %>% kable_styling(position = "left")
```

A.2 Histogram of Words per Line

Histogram of words per line for the three text corpora.

```
library(ggplot2)
library(gridExtra)
plot1 <- qplot(wpl[[1]],
               geom = "histogram",
               main = "US Blogs",
               xlab = "Words per Line",
               ylab = "Frequency",
               binwidth = 5)
plot2 <- qplot(wpl[[2]],
               geom = "histogram",
               main = "US News",
               xlab = "Words per Line",
               ylab = "Frequency",
               binwidth = 5)
plot3 <- qplot(wpl[[3]],
               geom = "histogram",
               main = "US Twitter",
               xlab = "Words per Line",
               ylab = "Frequency",
               binwidth = 1)
plotList = list(plot1, plot2, plot3)
do.call(grid.arrange, c(plotList, list(ncol = 1)))
```

A.3 Sample and Clean the Data

```
# set seed for reproducability
set.seed(101)

# sample all three data sets
sampleBlogs <- sample(blogs, length(blogs) * sampleSize, replace = FALSE)
sampleNews <- sample(news, length(news) * sampleSize, replace = FALSE)
sampleTwitter <- sample(twitter, length(twitter) * sampleSize, replace = FALSE)

# remove all non-English characters from the sampled data
sampleBlogs <- iconv(sampleBlogs, "latin1", "ASCII", sub = "")
sampleNews <- iconv(sampleNews, "latin1", "ASCII", sub = "")
sampleTwitter <- iconv(sampleTwitter, "latin1", "ASCII", sub = "")

# combine all three data sets into a single data set and write to disk
sampleData <- c(sampleBlogs, sampleNews, sampleTwitter)
```

```r
sampleDataFileName <- "C:/Users/b.s.priya/Documents/Data Science_R/Capstone/en_US.sample.txt"
con <- file(sampleDataFileName, open = "w")
writeLines(sampleData, con)
close(con)

# get number of lines and words from the sample data set
sampleDataLines <- length(sampleData);
sampleDataWords <- sum(stri_count_words(sampleData))

# remove variables no longer needed to free up memory
rm(blogs, news, twitter, sampleBlogs, sampleNews, sampleTwitter)
```

A.4 Build Corpus

```r
library(tm)
# download bad words file
badWordsURL <- "https://www.freewebheaders.com/wordpress/wp-content/uploads/full-list-of-bad-words_text-
badWordsFile <- "C:/Users/b.s.priya/Documents/Data Science_R/Capstone/full-list-of-bad-words_text-file_
if (!file.exists('data')) {
  dir.create('data')
}
if (!file.exists(badWordsFile)) {
  tempFile <- tempfile()
  download.file(badWordsURL, tempFile)
  unzip(tempFile, exdir = "data")
  unlink(tempFile)
}
buildCorpus <- function (dataSet) {
  docs <- VCorpus(VectorSource(dataSet))
  toSpace <- content_transformer(function(x, pattern) gsub(pattern, " ", x))
  docs <- tm_map(docs, toSpace, "(f|ht)tp(s?)://(.*)[.][a-z]+")
  docs <- tm_map(docs, toSpace, "@[^\\s]+")
  docs <- tm_map(docs, toSpace, "\\b[A-Z a-z 0-9._ - ]*[@](.*?)[.]{1,3} \\b")
  docs <- tm_map(docs, tolower)
  docs <- tm_map(docs, removeWords, stopwords("english"))
  docs <- tm_map(docs, removePunctuation)
  docs <- tm_map(docs, removeNumbers)
  docs <- tm_map(docs, stripWhitespace)
  # remove profane words from the sample data set
  con <- file(badWordsFile, open = "r")
  profanity <- readLines(con, encoding = "UTF-8", skipNul = TRUE)
  close(con)
  profanity <- iconv(profanity, "latin1", "ASCII", sub = "")
  docs <- tm_map(docs, removeWords, profanity)
  docs <- tm_map(docs, removeWords, c("cyalisx",
                                      "cyberfuc",
                                      "cyberfuck",
                                      "cyberfucked",
                                      "cyberfucker",
                                      "cyberfuckers",
                                      "cyberfucking"))
  # convert the text to plaintext
  docs <- tm_map(docs, PlainTextDocument)
  return(docs)
}


# build the corpus and write to disk (RDS)
corpus <- buildCorpus(sampleData)
saveRDS(corpus, file = "C:/Users/b.s.priya/Documents/Data Science_R/Capstone/en_US.corpus.rds")
# convert corpus to a dataframe and write lines/words to disk (text)
corpusText <- data.frame(text = unlist(sapply(corpus, '[', "content")), stringsAsFactors = FALSE)
con <- file("C:/Users/b.s.priya/Documents/Data Science_R/Capstone/en_US.corpus.txt", open = "w")
writeLines(corpusText$text, con)
close(con)
kable(head(corpusText$text, 10),
      row.names = FALSE,
      col.names = NULL,
      align = c("l"),
      caption = "First 10 Documents") %>% kable_styling(position = "left")
# remove variables no longer needed to free up memory
rm(sampleData)
```