Capstone project

Matthew Rees

4 November 2014

# Text Prediction Application

# Introduction

Over the past decade, there has been a dramatic increase in the usage of electronic devices for email, social networking and other activities. Errors typing on such devices are far from uncommon, and can have considerable implications concerning the efficient use of such devices for communication.

Text prediction requires estimation of the next character or word given a string of the input history. This may represent a useful solution to the problem of mistyping words.

In this project we aim to develop a text predictive algorithm derived from large data sets composed of different source material blogs, twitter etc. to develop an application.

# Task 1 - Data acquisition and cleaning

Data was obtained from HC corpora (www.corpora.heliohost.org). It contains data from four languages (english, german, finish and russian) of blogs, news and twitter data.

**Tasks to accomplish:**

The aim of this initial report is to perform an exploratory analysis of text data from HC corpora, providing a basic understanding of; - The distribution of words - The relationship between the words - The differences in frequencies of words - The presents of word pairs and triplets in the data.

This aim required the accomplishment of three primary tasks: - Exploratory analysis.

* Data cleaning
* Profanity filtering - removing profanity and other words you do not want to predict.

# 1. Data processing

library(wordcloud)  
library(qdap)  
library(stringr)  
library(RWeka)  
library(tm)

Analysis of the corpora database; - Basic statistics of the data set

corpus.b <- readLines("~/Desktop/Capstone Project/final/en\_US/en\_US.blogs.txt")  
nlines.b <- length(corpus.b)  
nwords.b <- sum(unlist(sapply(corpus.b, str\_count, pattern = " "))) + nlines.b  
  
corpus.n <- readLines("~/Desktop/Capstone Project/final/en\_US/en\_US.news.txt")  
nlines.n <- length(corpus.n)  
nwords.n <- sum(unlist(sapply(corpus.n, str\_count, pattern = " "))) + nlines.n  
  
corpus.t <- readLines("~/Desktop/Capstone Project/final/en\_US/en\_US.twitter.txt")

## Warning: line 167155 appears to contain an embedded nul  
## Warning: line 268547 appears to contain an embedded nul  
## Warning: line 1274086 appears to contain an embedded nul  
## Warning: line 1759032 appears to contain an embedded nul

nlines.t <- length(corpus.t)  
nwords.t <- sum(unlist(sapply(corpus.t, str\_count, pattern = " "))) + nlines.t  
  
df.lines <- data.frame(file = factor(c("blogs", "news", "twitter")), count = c(nlines.b, nlines.n, nlines.t))  
  
df.words <- data.frame(file = factor(c("blogs", "news", "twitter")), count = c(nwords.b, nwords.n, nwords.t))  
  
df.comb <- data.frame(File = factor(c("Blogs", "News", "Twitter")), Count\_of\_lines = c(nlines.b, nlines.n, nlines.t), Count\_of\_words = c(nwords.b, nwords.n, nwords.t))  
df

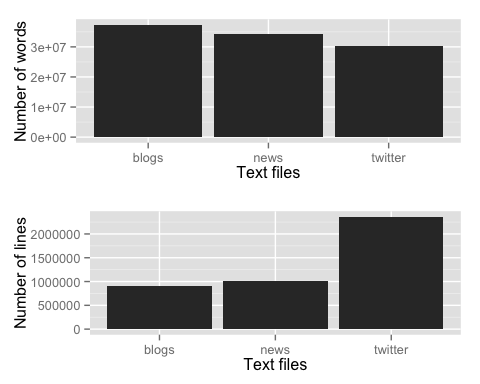
## function (x, df1, df2, ncp, log = FALSE)   
## {  
## if (missing(ncp))   
## .External(C\_df, x, df1, df2, log)  
## else .External(C\_dnf, x, df1, df2, ncp, log)  
## }  
## <bytecode: 0x126988ea8>  
## <environment: namespace:stats>

Key information that we can discern form this exploration is...

* The number of words per line is highest for the *blogs* data set and lowest for the *twitter* data set, as may be anticipated given the nature of these media.
* The mean line length for the *blogs* data set is 229.99
* The mean line lengths for the *news* dataset is 201.16
* The mean line length for the *twitter* dataset is 68.68.

These statistics can be better assessed with a plot demonstrating the number of lines and of words in the three text files. .

## Loading required package: grid



As can be seen, the *blogs* data set has roughly nine hundred thousand lines, the *news* data set has roughly seventy thousand lines, and the *twitter* file has roughly two point four million lines

# Data cleaning

The *stringi* and *tm* packages were used to transform the text into a format conducive to tokenisation, which is the next step in the project. The following steps were undertaken....

* Convert all characters to lower space
* Replace i with I
* Remove punctuation except for ".", and "'", which are essential for structure and understanding of the sentence.
* Remove white spaces
* Remove dot spaces
* Remove numbers
* Split lines by the postitioning of dots

library(stringi)  
  
directory\_source <- DirSource("~/Desktop/Capstone Project/final/en\_US", encoding = "UTF-8")  
  
corpus <- Corpus(directory\_source,readerControl=list(reader=readPlain))

## Warning: line 167155 appears to contain an embedded nul  
## Warning: line 268547 appears to contain an embedded nul  
## Warning: line 1274086 appears to contain an embedded nul  
## Warning: line 1759032 appears to contain an embedded nul

# Convert all letering to lower case  
corpus.clean <- tm\_map(corpus, function(x) stri\_trans\_tolower(x[[1]]))  
  
# Replacing 'i' with 'I', using the tm package, this was simpler to perform through multiple substitutions than through a single line of code.  
corpus.clean <- tm\_map(corpus.clean, function(x) gsub("^i ", "I ", x))  
corpus.clean <- tm\_map(corpus.clean, function(x) gsub("^i'", "I'", x))  
corpus.clean <- tm\_map(corpus.clean, function(x) gsub(" i ", " I ", x))  
corpus.clean <- tm\_map(corpus.clean, function(x) gsub(" i'", " I'", x))  
corpus.clean <- tm\_map(corpus.clean, function(x) gsub(".i ", ". I ", x))  
corpus.clean <- tm\_map(corpus.clean, function(x) gsub(".i'", ". I'", x))  
  
# Remove punctuation with exception of . and ''' using the tm package.  
corpus.clean <- tm\_map(corpus.clean, function(x) gsub("[!?,.]+", ".", x))  
corpus.clean <- tm\_map(corpus.clean, function(x) gsub('[])(;:#%$^\*\\~{}[&+=@/"`|<>\_]+', "", x))  
  
# Remove white spaces, using the tm package.  
corpus.clean <- tm\_map(corpus.clean, stripWhitespace)  
  
# Remove dot spaces for later split  
corpus.clean <- tm\_map(corpus.clean, function(x) gsub(" \\.", ".", x))  
corpus.clean <- tm\_map(corpus.clean, function(x) gsub("\\. ", ".", x))  
  
# Remove numbers, using the tm package.  
corpus.clean <- tm\_map(corpus.clean, removeNumbers)  
  
# Splitting by dots  
corpus.clean <- tm\_map(corpus.clean, function(x) strsplit(x, "\\."))

Ultimately this process will convert a blog such as ...

head(corpus$ en\_US.blogs.txt, n = 1)

## [1] "In the years thereafter, most of the Oil fields and platforms were named after pagan “gods”."

Into the following...

head(corpus.clean$ en\_US.blogs.txt, n = 1)

## [[1]]  
## [1] "in the years thereafter"   
## [2] "most of the oil fields and platforms were named after pagan “gods”"

# Profanity filtering

A decision was made not to exclude profanities until the prediction component of the project, due to the impact this may have on the understanding of the sentence.

# Tokenisation

This is the next step in the capstone project, although the code has not yet been written, however it will begin with segmenting text into words based on the presence of spaces, and will be followed by the production of unigram, bigram and trigram with an Ngramtokenizer in the RWeka package.

Fin.