Rpub test

Kevin Spring

11/15/2014

This is just to test that publishing to Rpubs is working.

Introduction

The purpose of this project is to predict a word a user is typing and also predict a successive word after a word is typed. The final product will be a web application that allows a user to input text into a text field. The application will predict what word they are typing and display a few words that are predicted. The user can click on these words for faster user input if the words are correctly predicted or continue to type. As the user continues to input characters the application will continue to display predicted words on the sequence of character inputs. After a word is inputted or selected by the user, the application will predict what next word the user will input and display a list of possible predicted words or phrases. The user can select the application predicted text or continue to input characters.

Data Manipulation

The data used in this analysis is text from english language twitter feeds, blogs, and news. The twitter, news, and blog text data has 2,360,148, 1,010,242, and 899,288 lines of code, respectively.

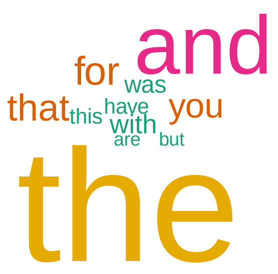
The training data set was limited to 50,000, 20,000, and 20,000 lines of code for the twitter, news, and blog text data, respectively. This training set was selected by taking a random sample of each of the twitter, news, and blog text data sets. Before we can accuratly assess how many words are in each line of text the data needs to be cleaned. I used the R programming language with the `tm` package to analyze the text data.

Results

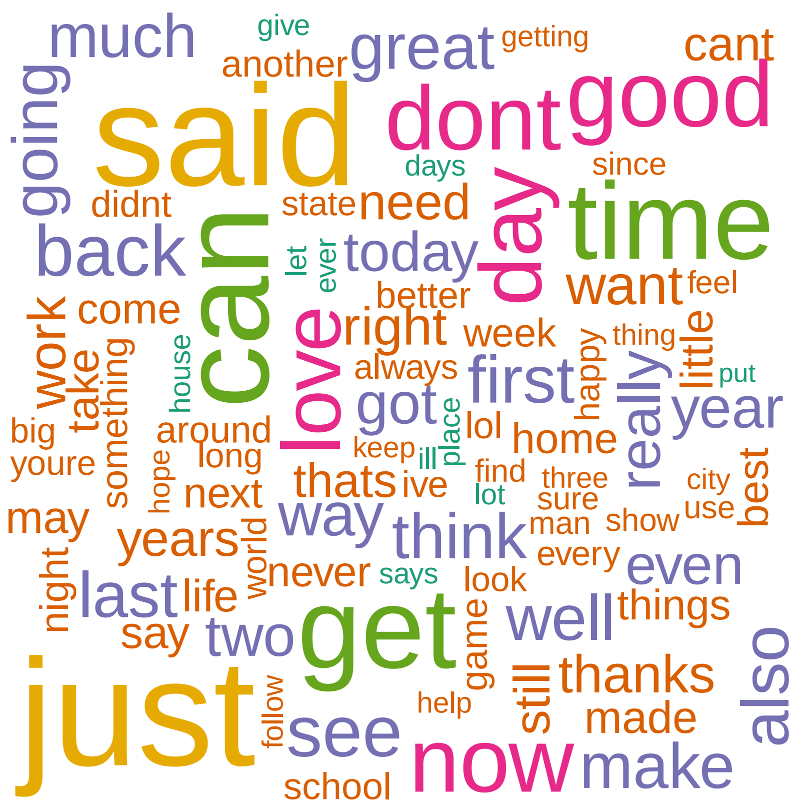
The first step is to remove any non-Latin characters in the data set. Althought this data is classified as English, there is non-Latin characters in the data. The characters removed includes emoticons, Chinese hanzi, and Japanese kana. Arabic numbers are also removed from the training data. There were a total of 85521 terms in the combined twitter, news, and blog text data.

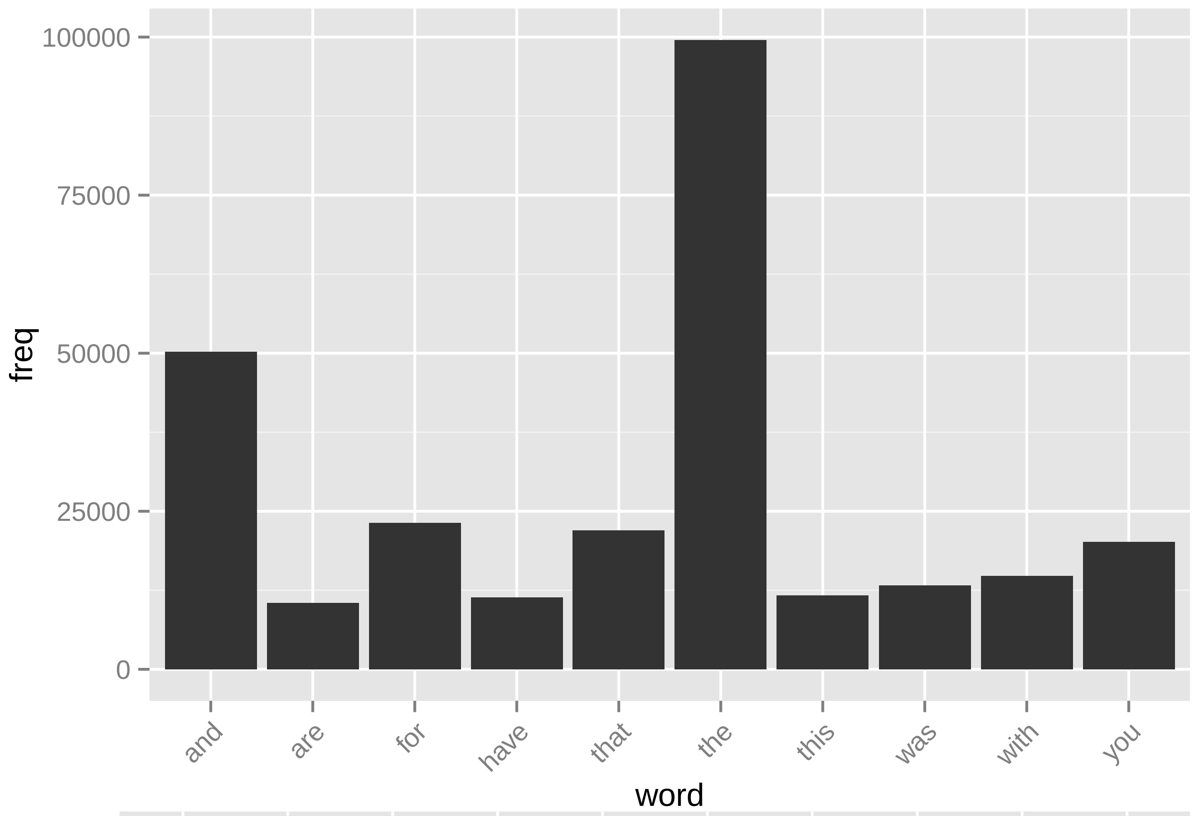
A further reduction of words such as onomatopoeias and profanity filters the data set to 2108 terms. Profanity is commonly used in the corpus but serves no prediction value to this project. The same profanity word can be used as a verb, noun, adverb, or adjective and thus prediction is limited for these words. As a user interface tool, showing profanity for predicted words if the user did not want to see may cause the user to use a competitor's product. If the user uses profanity then those words can be added to the dictionary of the final tool to accurately predict and suit the user's language style.

As shown in Table 1, the ten most common terms are stop words. Table 2 shows the ten most common words with stop words removed from the corpus. Unlike most test data analysis, I will be keeping stop words in the corpus. Term frequency was measured using the number of times a term is used in a document.

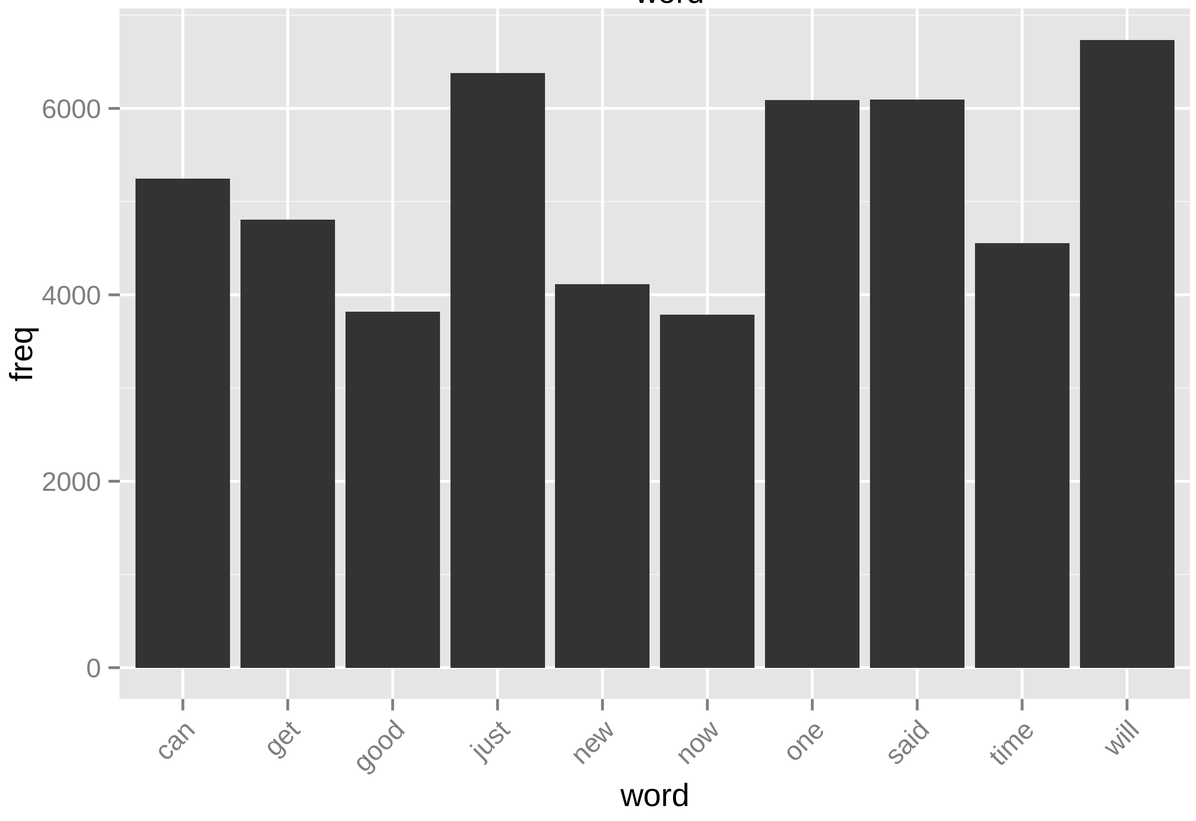


```
## Warning in wordcloud(d$word, d$freq, min.freq = 10000, rot.per = 0.2,
## max.words = 100, : people could not be fit on page. It will not be
## plotted.

## Warning in wordcloud(d$word, d$freq, min.freq = 10000, rot.per = 0.2,
## max.words = 100, : one could not be fit on page. It will not be plotted.

## Warning in wordcloud(d$word, d$freq, min.freq = 10000, rot.per = 0.2,
## max.words = 100, : new could not be fit on page. It will not be plotted.

## Warning in wordcloud(d$word, d$freq, min.freq = 10000, rot.per = 0.2,
## max.words = 100, : know could not be fit on page. It will not be plotted.

## Warning in wordcloud(d$word, d$freq, min.freq = 10000, rot.per = 0.2,
## max.words = 100, : many could not be fit on page. It will not be plotted.

## Warning in wordcloud(d$word, d$freq, min.freq = 10000, rot.per = 0.2,
## max.words = 100, : will could not be fit on page. It will not be plotted.
```





freq
100000
75000
50000
25000
0
and
are
for
have
that
the
this
was
with
you
word



freq
6000
4000
2000
0
can
get
good
just
new
now
one
said
time
will
word

N-gram tokenization

A bigram is two words that appear in a sequence. For example, “happy birthday” would be a bigram. Table 3 shows the top ten bigrams in the corpus after sparse terms were removed.

Stop word removal

Stop words are common words used in languages such as function words such as, the, is, at, which, and on. While it is important to accurately predict these words in this project, the inclusion of these words in the main prediction algorithm would over estimate the user wanting to type these words versus other words. See Fig X to compare the most common words with and without stop word removal. A second prediction algorithm can be used to accurately predict stop word use on a bigram model.

Correlated terms

If two words always appear together then the correlation would be 1.0. If they never appear together the correlation would be 0. This correlation is a measurement of how closely associated the words are in the corpus.

Conclusions