'John Hopkins University Data Science Specialiazition Capstone Project: Milestone Report 1

Steffen Hartleib

July 21, 2015

## Overview

The project assignment is to create an app that takes a string of words from a user and predicts the next word.  
This initial report explores the data and briefly describes the next steps in building the model.

## The raw data

The corpus is three large sets of English language data:

n\_US.news.txt -- 196M  
en\_US.blogs.txt -- 196M  
en\_US.twitter.txt -- 196M

These files are too large to process efficently in an app, so for each one I drew a random sample of 1% of the lines:

## Line counts and size of sample files

## News Blogs Twitter  
## Size(MB) 1 10 0  
## LineCount 5000 45183 5000

These sample files make up the corpora for the rest of the analysis.

## Cleaning the data

* removed punctuation
* removed any non alphabetical charcters
* kept dashes and apostrophes within words
* added periods back to most frequent abreviations
* converted all words to lower case
* removed profanties

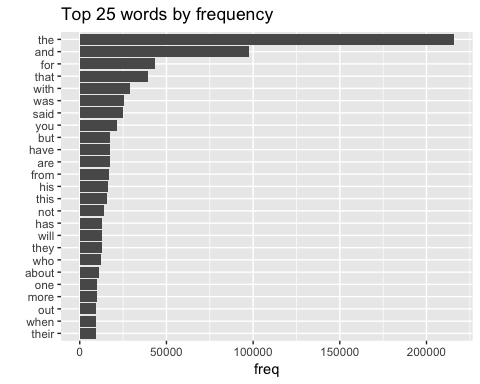
## Tokinizing the data sets

created frequency tables of:  
\* single words  
\* 2-word phrases (bigrams)  
\* 3-word phrases(trigrams)

## Single words

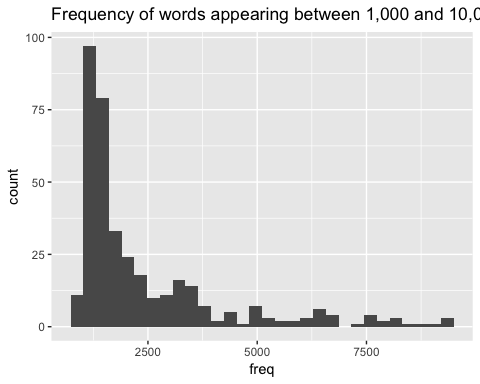
* The corpus of sample files includes:
* 1.3 Million instances made up of about 72k distinct words
* The average frequency is 19
* The the median frequency is 2
* The data is highly skewed by a few highfrequency words such as "the", "and".
* Since the assignemnt is to predict the next word, I did not remove these stop words.

## Here are the 25 most frequent words



## Frequency distribution

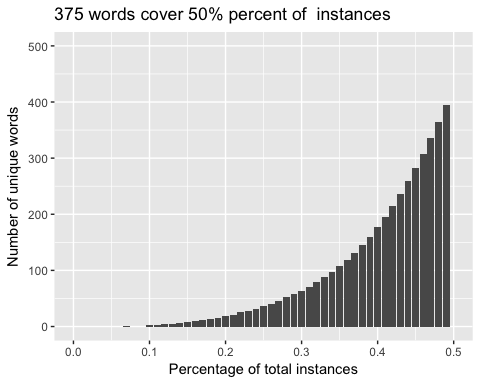
To make the chart more legible I zoomed in on frequencies between 1,000 and 10,000. Still the data remains highl skewed.



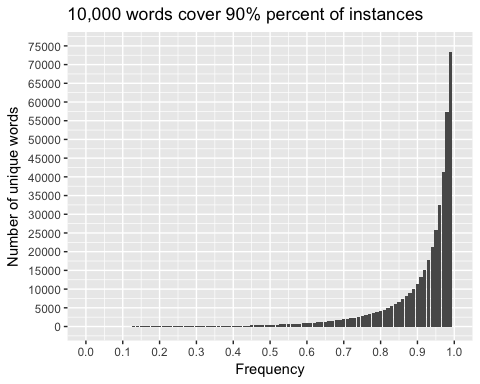
## How many unique words does it take to cover 50% and 90% of the instances?

Because the data is so heavily skewed towards high frequency words, it takes only 375 words to cover 50% of the instances. 10,000 words will get you to 90% coverage:

## Warning: Removed 50 rows containing missing values (position\_stack).



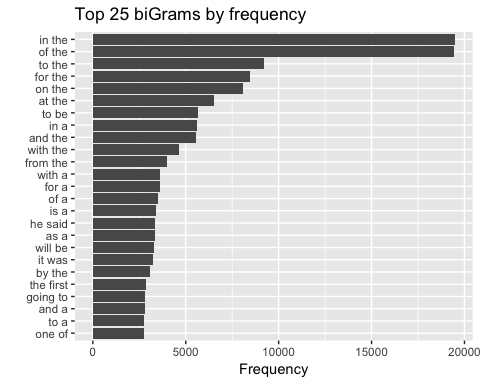
## Warning: Removed 1 rows containing missing values (position\_stack).



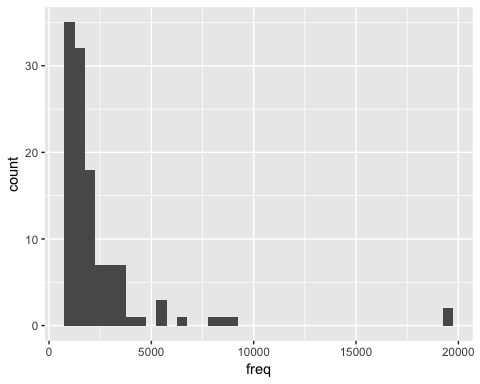
## Two word phrases (biGrams)

There are 1.6 million instances of 680k unique phrases  
Then mean frequency is 2.4 and the median is 1  
This means data is significantly less skewed than single words.

## Here are the 25 most frequent biGrams:



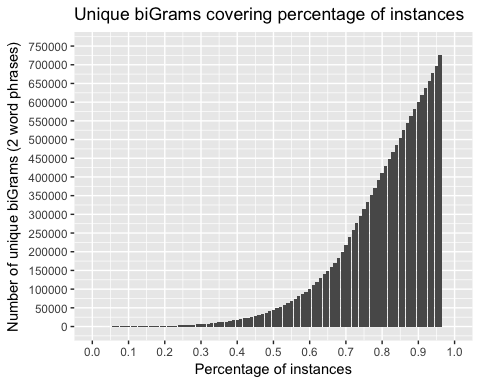
#### frequency distribution:

Again, to make the chart more legible, I zoomed in on frequencies over 1,000 

## How many unique biGrams will cover 50% and 90% of the instances?

Since the biGrams much less skewed than the single words it takes 50,000 unique bigrams to cover 50% of the instances, and a whopping 500,000 to cover 90% of the instances. (better wrap up the exploratory analysis and start working on the model...)

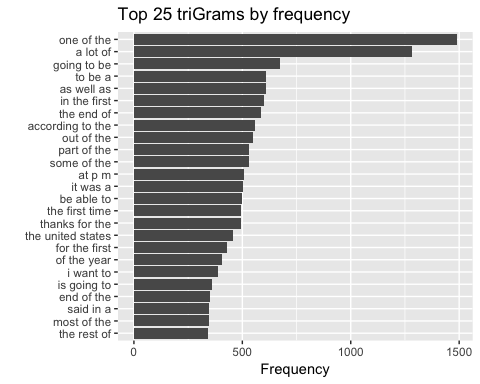
## Warning: Removed 4 rows containing missing values (position\_stack).



## Three word phrases (triGrams)

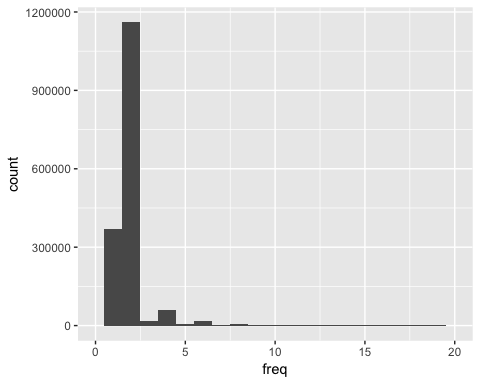
There are 730k unique triGrams, and 1.20 million instances.  
The mean frequency is 1.2 and the median of 1. So while the data is less skewed, it's also very narrowly distributed around the mean of one. In other words, most phrases appear only once.

## Here are the top 25 trigrams:



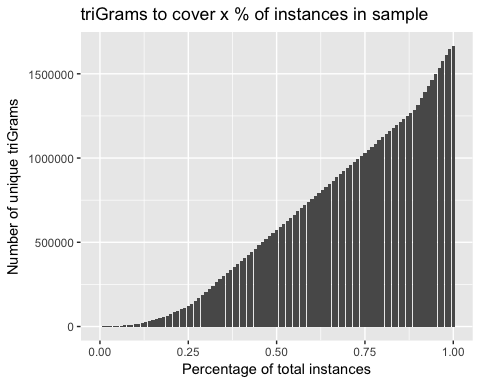
## Frequency distribution of trigrams

## Warning: Removed 5601 rows containing non-finite values (stat\_bin).



## How many unique triGrams will cover 50% and 90% of the instances?

Once you cover 25% of the unique instances the relationship becomes linear. This means we'll need to include roughly 50% unique triGrams to cover 50% of the instances, 75% of unique triGrams to cover 75% of instances etc.



)