Quantifying Literature's Quality

New Mexico
Supercomputing Challenge
Final Report
April 2, 2014

Team 67
Los Alamos High School

Team Members:
Tabitha Welch

Teacher:
Lee Goodwin

Project Mentor:
Paul Welch

Abstract

This project aims to better understand what makes a particular piece of literature sell well and appeal to a large audience. This year I focused on average sentence length, author vocabulary size, and word frequency for a data set of 100 fiction books or short stories. Three codes were written in Perl to determine all of these factors. The first calculated average sentence length. The second removed all punctuation from the text. The third created a list of words used by the book's author and how many times each word on that list appeared. I used normalized download counts from Project Gutenberg as a measure of each book's popularity. Histograms of many of my metrics, such as number of sentences and vocabulary size, showed clear mathematical patterns. In addition, a plot of normalized vocabulary size vs. total number of sentences revealed that there is an exponentially decaying relationship between these two factors. Histograms of word frequencies for different books looked nearly identical, but this is likely due to “background noise;” that is, some words are always frequently used in the English language. It does not seem that popularity is closely related to sentence length or vocabulary size, but more tests are necessary to verify this conclusion.

Problem

The goal of this project is to better understand what makes various pieces of literature popular. Complexity of literature may be related to what makes a particular piece last long and appeal to a specific audience. However, complexity is not directly equivalent to quality. I am especially focusing on the size of vocabularies, average sentence length, and word frequency this year. I believe that to some extent, a book will be more popular if it has a large author vocabulary and low word frequency. Books with these characteristics will have generally more varied language and may therefore be more interesting to read. I have acquired 100 fiction writing samples from Project Gutenberg, an online source of ebooks whose copyrights have expired, for my data.

Methods

Each of my 100 books is either a short story or novel. The books were chosen from a random list of all books on Project Gutenberg. I developed three codes to analyze this data. My first code was written to calculate the average number of words per sentence in each sample. This code simply counted the total number of words and divided it by the number of sentences in the text. A second code stripped each of my data samples of all punctuation marks, leaving only the words that the author used. The third code could then be run on the modified data. This final code was a "dictionary" code; it created a list or "dictionary" of all the words that appeared in a particular sample. Each word was recorded in the dictionary only once.

Algorithm of the Sentence Length Code

The code for average sentence length was the simplest of the three codes. Individual words in the sample were separated at every space in the text. An if statement within a for loop checked the words for periods and counted both periods and total words. The average sentence length was calculated when the code divided number of words by number of periods. It should be noted that the sentence length code does not consider all end punctuation, but only periods. In the future I will modify the code to include punctuation such as the exclamation point and question mark.

Algorithm of the Punctuation Code

The punctuation code took into account almost all punctuation marks used in written language, including uncommon ones such as the backslash and underscore. It assembled a new file that contained a punctuation-free version of the data sample. For each punctuation mark, a separate if statement was created. The if statement checked for its particular punctuation in the text. Each time the punctuation appeared, the text was split in that location. In other words, it separated punctuation from words, and only words remained in the new file. The separated punctuation mark was then tagged, indicating that it had already been recognized. This process applied to any punctuation mark that only appeared at the

beginning of a word or only appeared at the end of a word (for example, a comma or period). For punctuation that might have appeared at either the beginning or end of a word (for example, quotation marks), an if-else statement was used. This statement served the same purpose, but it checked both ends of a word. The code only removed one punctuation mark per word, so it was necessary to run it multiple times on modified files to eliminate all punctuation. In addition, a few marks or symbols were not included due to difficulties or complications that they caused. For instance, including apostrophes would have separated all possessives and contractions into two words. “Isn't” would have become “Isn t,” and so forth. In future work I will attempt to correct these issues.

Algorithm of the Dictionary Code

The dictionary code made use of hash tables to create separate dictionaries for each sample. Each word (a string) encountered was placed into a hash table. However, if the word had been previously encountered, it was not repeated in the table. The code converted all uppercase letters to lowercase ones from the beginning, eliminating repetition of words because of case. For example, the word “of” was recorded instead of the words “of” and “Of.” The dictionary code produced two files: the list of words itself and a list of numbers that represented how many times each word on the list appeared in the text.

Model Verification

In order to verify that my codes worked properly, I prepared a short data test set. The set consisted of a slightly modified paragraph of *Pride and Prejudice*. I added extra punctuation so that all parts of the punctuation code would be tested and a few extra words. The test set was short enough that I could perform the calculations myself and compare them to the computer's.

Results

I obtained results centered around six metrics. These were downloads in 30 days, average sentence length, vocabulary size, total number of words, total number of sentences, and word

frequency. Because each sample was a different length, the author's vocabulary size had to be normalized for each one. I accomplished this by dividing the number of different words (dictionary length) by the number of total words in the text. I made a few plots to visualize any possible correlations between download counts and each of my variables (see Figures 1,2). The download counts, obtained from Project Gutenberg, were the number of times that the ebook had been downloaded from that database in the last 30 days. They were the sample's "grade" or measure of popularity. The plot of normalized vocabulary vs. download count (Figure 1) and the plot of average sentence length vs. download count (Figure 2) were two of the most relevant of these plots to test my original hypothesis. However, there was no strong relationship visible. Most of the data is clustered in one area of the graph, with only a few outliers that give a shape to it. The y-axes of Figures 1 and 2 are scaled logarithmically to make the data more easily visible. Noticing that vocabularies tended to be smaller for shorter samples, I plotted the normalized vocabulary vs. the number of sentences in Figures 3 and 4. Surprisingly, an exponential relationship emerged. The longer the sample is, the smaller its normalized vocabulary is.

I also made histograms of several of my metrics (Figures 5-10). These included Gutenberg downloads, total number of words, total number of sentences, sentence length, vocabulary size (dictionary length), and normalized vocabulary size. The histograms showed a clear pattern for each metric, with one or two peaks on each graph. I used a few samples as references to connect these patterns with the download counts and therefore popularity. There were two main peaks on the histogram of logarithm of the downloads (Figure 5), and both looked like normal distributions. The first and larger of these peaks represented average books and short stories that were not generally well-known, and the second represented famous classics such as *Les Miserables*. The total words histogram in Figure 6 also had two peaks; these were short stories and novels. Novels were much more frequent in the data set than short stories. As expected, the total sentences histogram in Figure 7 followed a very similar pattern. The histogram of average sentence length (Figure 8) had only one

peak, which was roughly normal. It should be noted that very popular books such as *Les Miserables* and *Huckleberry Finn* fell on the the peak of the graph's curve, indicating that their average sentence length is no different from that of most other books. The histogram of raw vocabulary size (Figure 9) had two main sections, which were short stories and novels. This fits with the theory that an exponential relationship exists between vocabulary size and sample length. However, there seems to be little correlation between popularity and raw vocabulary size; *Huckleberry Finn* fell in the vicinity of *An Open-Eyed Conspiracy*, which was the least popular book in the data set. Finally, the histogram of normalized vocabulary size in Figure 10 had a single peak with most of the data concentrated around a normalized vocabulary of 0.1. Because of how the vocabulary was normalized, *Les Miserables*, which had the highest raw vocabulary (see Figure 9), had the lowest normalized vocabulary. For histograms of total number of words (Figure 6), number of sentences (Figure 7), raw vocabulary (Figure 9), and normalized vocabulary (Figure 10), frequently downloaded books such as *Huckleberry Finn* were on the highest peak, while less popular books such as *An Open-Eyed Conspiracy* were lower on these curves.

I made histograms for individual samples of the frequencies of word counts obtained from my dictionary code; a few of these are shown in Figures 11-13. The x-axis (“Word Count”) on each histogram shows the number of times a particular word appeared, while the y-axis (“Frequency”) shows how many words appeared that number of times. I created them in the hopes of obtaining a unique “signature” for each sample that might reveal something about its popularity. These histograms need modification in order to better understand these results. Some English words, such as “the” and “I,” will always be present in writing regardless of author or popularity. In the future I will attempt to cut out these very frequent words that form “background noise” so that the author's individual word usage can be more clearly seen.

Conclusions

My original hypothesis was probably incorrect. There does not seem to be a strong correlation between average sentence length or author vocabulary size and popularity. However, more data is needed to verify this conclusion. Perhaps a trend will develop if I run the codes on a greater number of books. A few outlying data points seen on the plots in Figures 1 and 2 suggest this possibility. A trend might also appear if I attempted to plot the data differently. There is certainly an unexpected trend between normalized vocabulary *v* and total number of sentences *s*. This is an exponential function, and it seems to indicate that the shorter a book is, the smaller its vocabulary is likely to be (Figures 3 and 4). In other words, $v \sim \exp(-s)$. Histograms of my metrics separated the data set into two distinct groups but showed little correlation to book popularity (see Figures 5-10). A great deal of work remains to be done on this project, and many improvements can still be made.

Significant Achievements

My most significant achievement while working on the project this year was without a doubt the ability to write and run basic code in the Perl and C programming languages. Although my final code was written in Perl, I spent a considerable amount of time learning C as well. The dictionary code was the most difficult and most significant code that I wrote this year. I obtained a very interesting result that shows an exponential relationship between normalized vocabulary size and total number of sentences. This was a completely unexpected correlation.

Software

All final codes for the project were written in Perl. I had originally attempted to write them in C, but Perl proved more appropriate for my goals. I used R and LibreOffice Calc to create my final plots, charts, and histograms of the data.

Plots, Histograms, and Charts

Figure 1. This plot shows number of downloads when compared to normalized vocabulary. There is little or no correlation between the two factors.

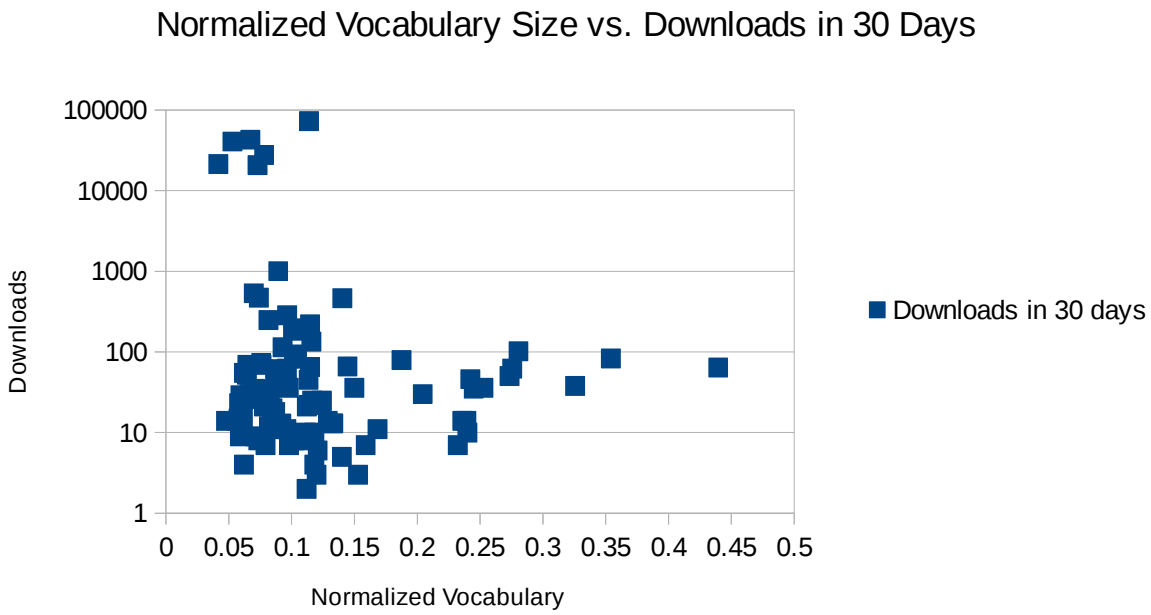

Figure 2. Similar to Figure 1, this plot shows downloads when compared to average sentence length. Again, there does not appear to be a strong correlation here.

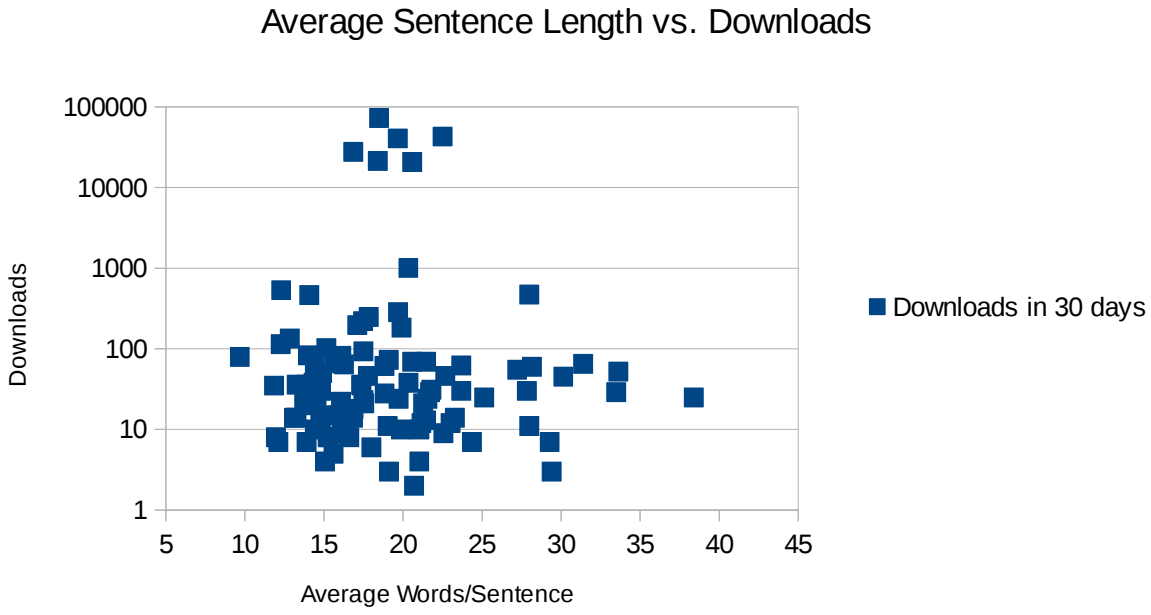

Figure 3. This plot of normalized vocabulary size vs. total number of sentences shows a surprising relationship of exponential decay.

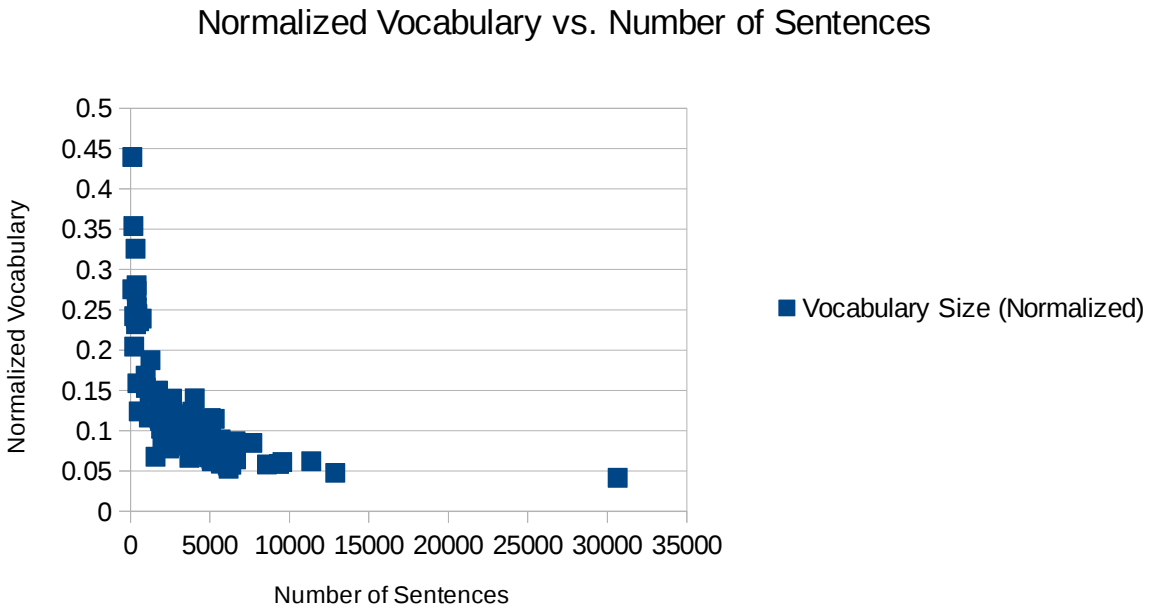

Figure 4. This plot shows the same results as Figure 3. The natural logarithm of each data point was graphed to illustrate a strong relationship.

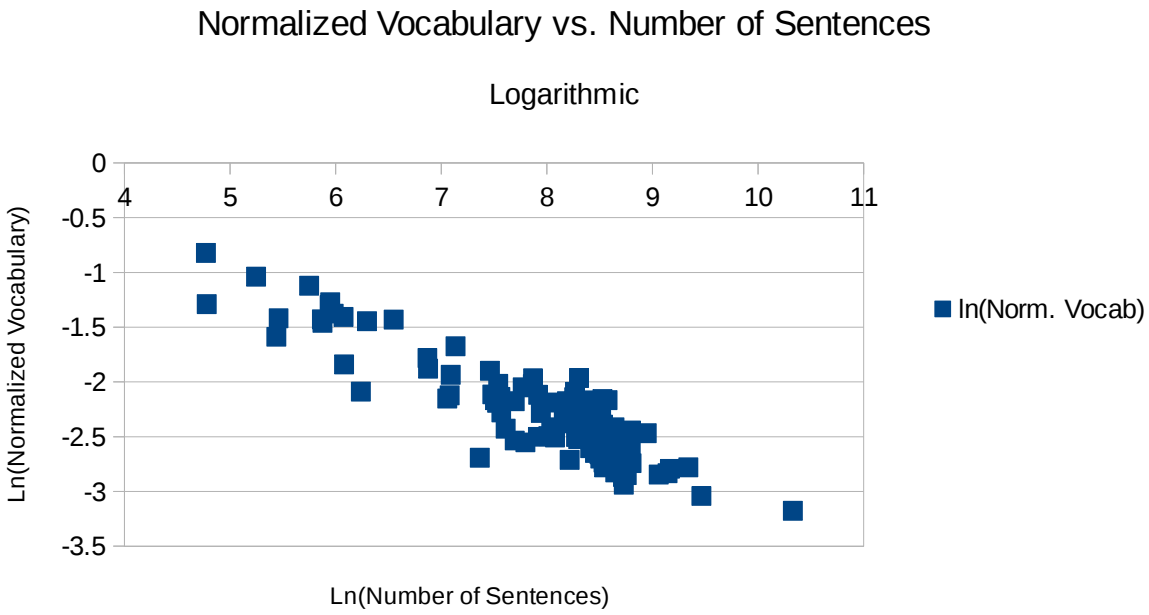

Figure 5. Here, the download counts for the whole data set are represented in a histogram. A few sample bins are labeled with example book titles for reference.

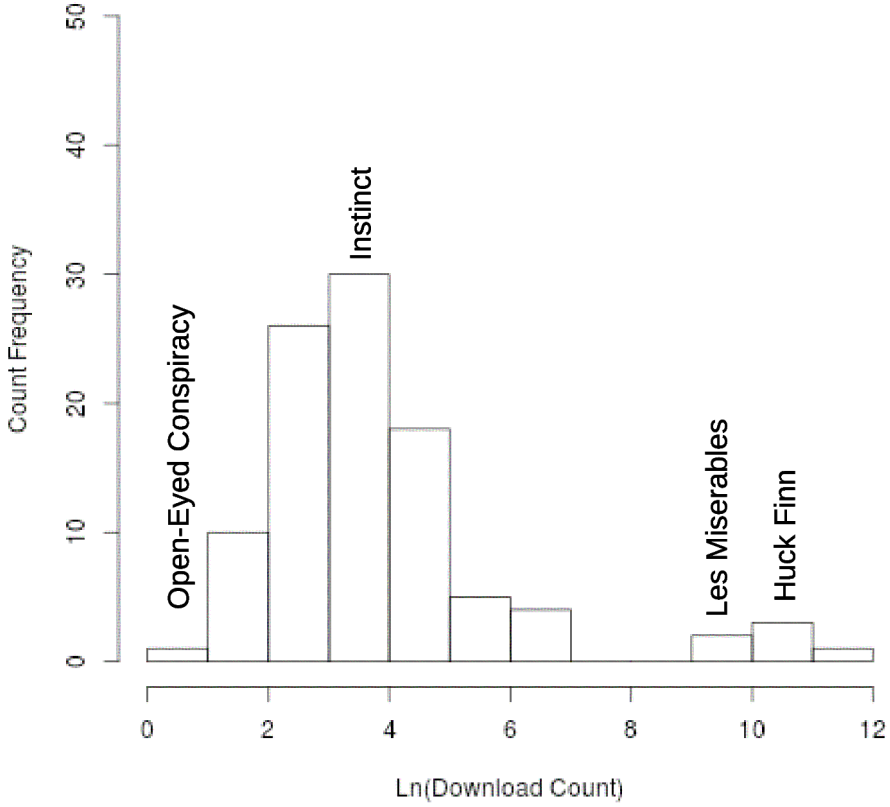

Figure 6. This histogram represents the length in words of all books in the data set. The data has been divided into two categories ("novels" and "short stories"), and reference points are again used.

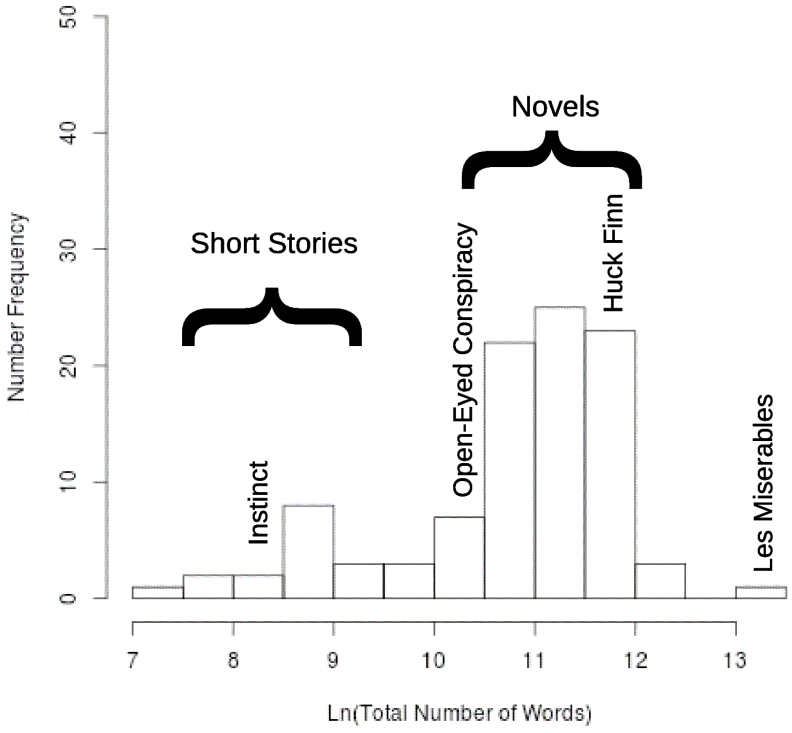

Figure 7. This histogram of total number of sentences for each book follows a pattern similar to that of

the total words histogram.

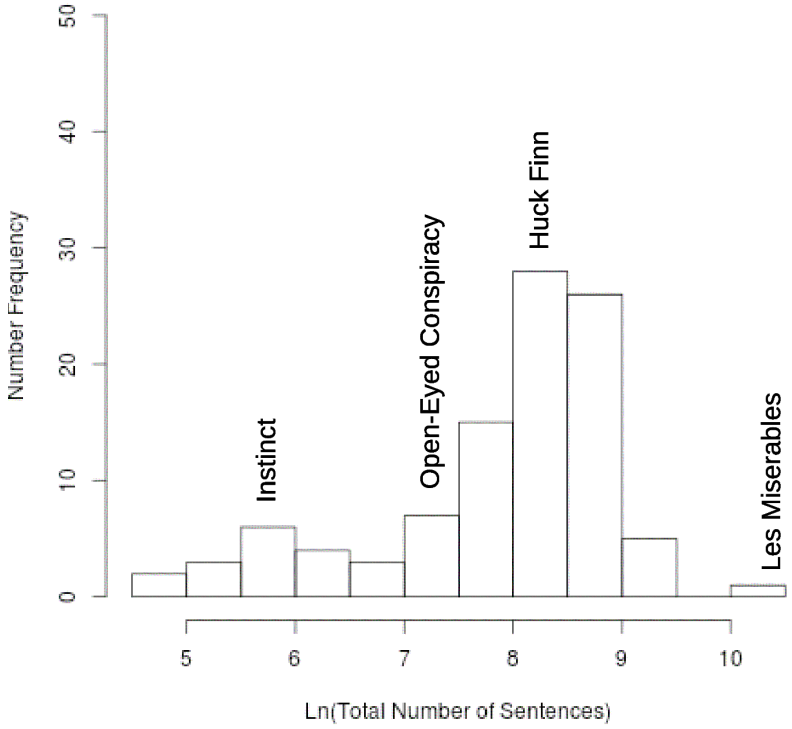

Figure 8. This histogram shows average sentence length for each book in the data set. The length ranged from 5 to 40 words per sentence. It should be noted, however, that well-known books such as Les Miserables have the same sentence lengths as much more obscure titles.

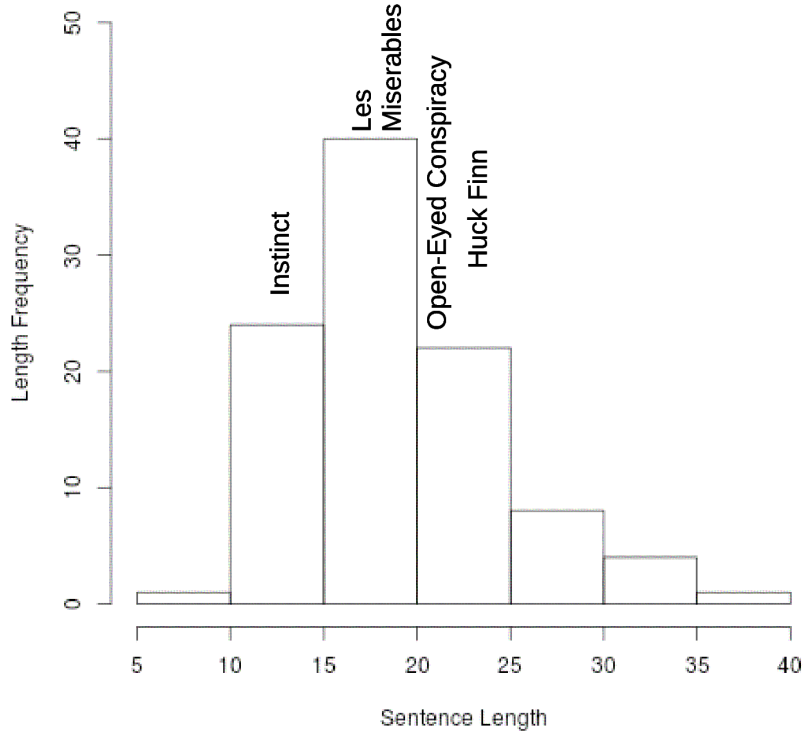

Figure 9. This histogram represents the raw vocabulary size of all books in the data set. These can again be divided into two basic groups: short story and novel.

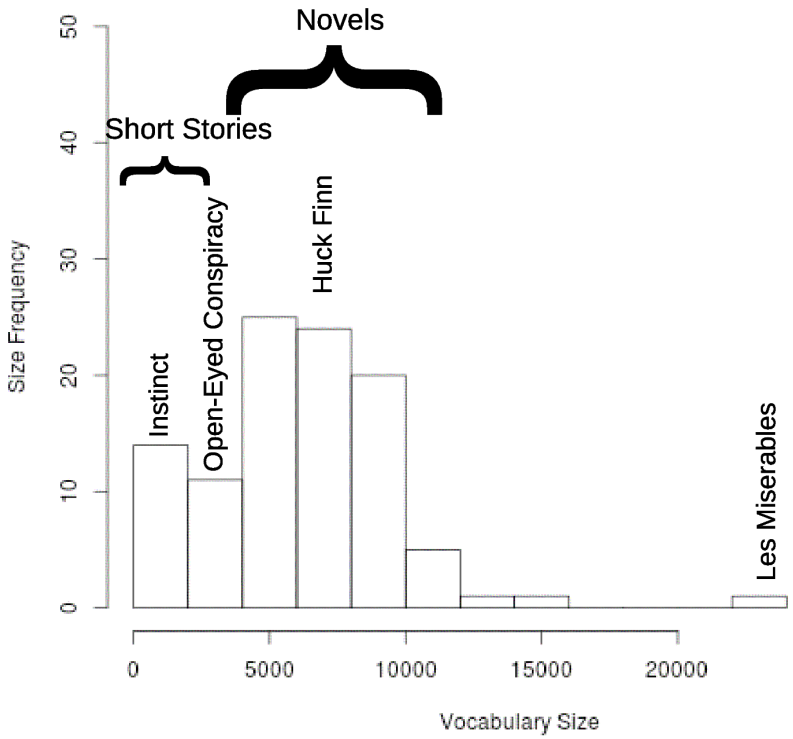

Figure 10. Here, normalized vocabulary sizes are represented in a histogram. Unlike the raw vocabulary histogram, books with very large vocabularies (such as Les Miserables) are shown closer to 0.0 on the x-axis.

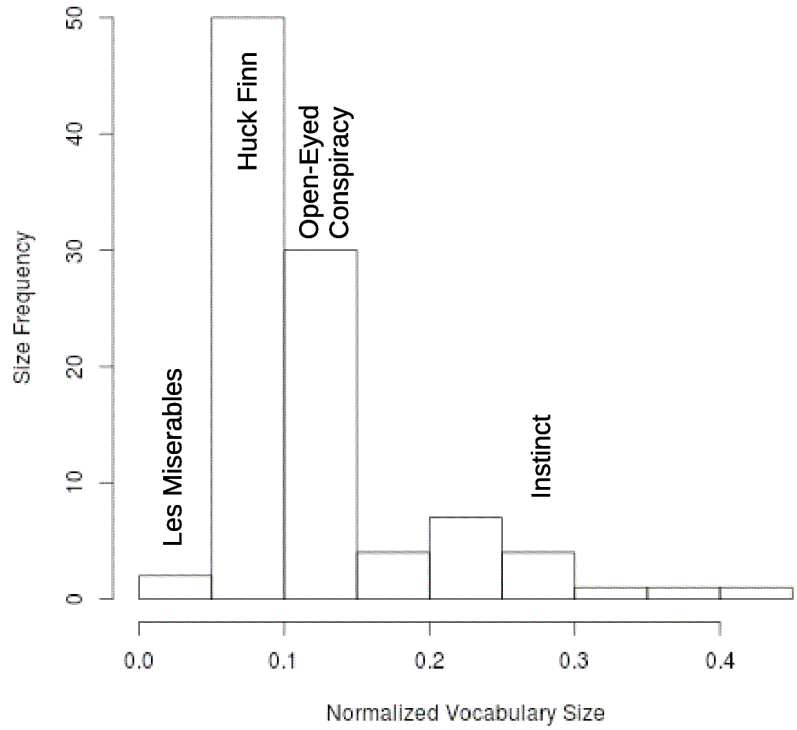

Figure 11. Here, the frequencies for all word counts in *Huckleberry Finn* are shown in a histogram. The x-axis shows the number of times a particular word appeared, while the y-axis shows how many words appeared that number of times.

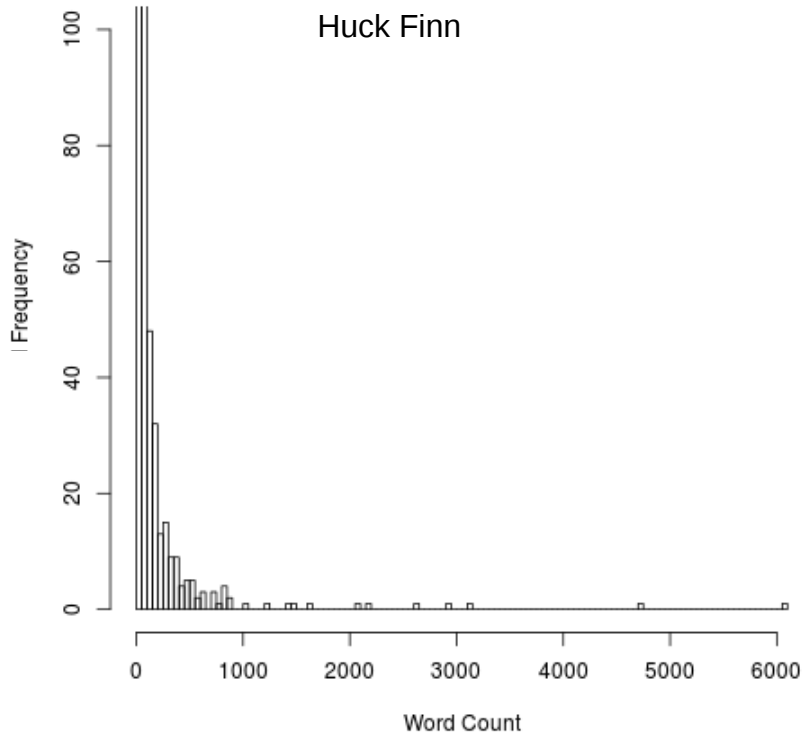

Figure 12. This histogram shows word count frequencies for *The Open-Eyed Conspiracy.* Although this text is much less popular than *Huckleberry Finn*, the two histograms look almost identical. Perhaps when background noise is eliminated, a more unique curve will emerge.

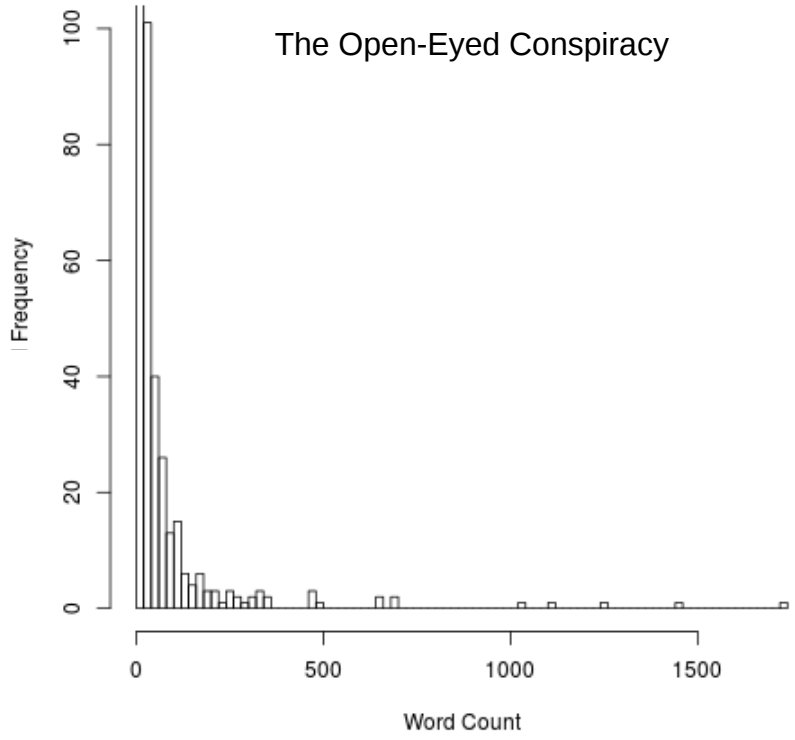

Figure 13. Here, word count frequencies for *Instinct* are shown in a histogram. *Instinct* is much shorter than either *Huckleberry Finn* or *An Open-Eyed Conspiracy*, but again the graph's shape is the same.

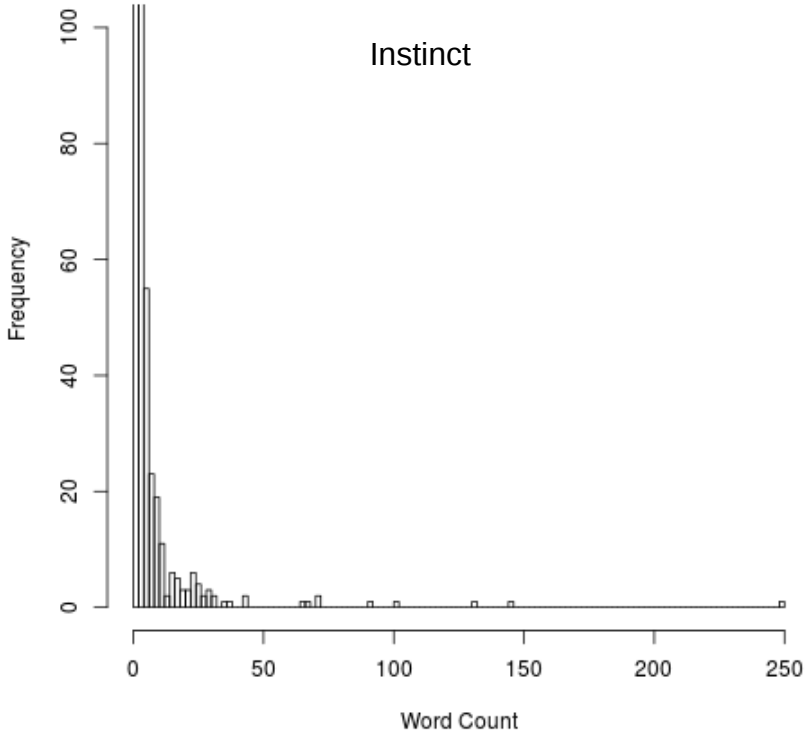

Table 1. List of All Data Books

A Houseful of Girls
A Love Episode
A Mountain Woman
A Red Wallflower
A Tale of Two Cities
Alarm Clock
Alive In The Jungle
An Open - Eyed Conspiracy: An Idyl of Saratoga
Bred in the Bone
Buffalo Bill's Spy Trailer
Cast Adrift
Chanticleer
Crome Yellow
David Dunne
Dora Deane, Or, The East India Uncle
Elsie at Home
Expediter
Fame and Fortune
Flames
Gabriel and the Hour Book
How It All Came Round
Huckleberry Finn
In Her Own Right

In The Brooding Wild
Instinct
Judith of the Plains
King Spruce
Les Miserables
Little Lost Sister
Little Mittens for the Little Darlings
Love and Mr. Lewisham
Mary Gray
My Fair Planet
Ned Garth
Old Man Curry: Race Track Stories
Operation Earthworm
Peveril of the Peak
Polly Oliver's Problem
Poor Jack
Precaution: A Novel
Pride and Prejudice
Rewards and Fairies
Squinty the Comical Pig
Sunny Boy in the Country
Sustained Honor
Swiss Family Robinson
That Affair Next Door
The Adventure Club Afloat
The Adventures of Harry Revel
The Adventures of Sherlock Holmes
The Black Bag
The Captives
The Chums of Scranton High
The Dope on Mars
The Dragon of Wantley
The Efficiency Expert
The Eye of Zeitoon
The Goat and Her Kid
The Good Neighbors
The Grizzly King
The Home in the Valley
The Hunters
The Inhabited
The Inner Sisterhood
The Invader
The Iron Woman
The Island of Faith
The King in Yellow
The Lost City
The Mayor of Casterbridge
The Mystery of the Four Fingers
The Old Folks' Party

The Patchwork Girl of Oz
The Prince and the Page: A Story of the Last Crusade
The Puppet Crown
The Radio Boys on the Mexican Border
The Red House Mystery
The Rover Boys At College
The Severed Hand
The Silver Butterfly
The Slave of Silence
The Spinners
The Spinster
The Splendid Folly
The Story of Red Feather: A Tale of the American Frontier
The Sword Maker
The Tragedy of the Chain Pier
The Turmoil: A Novel
The Valley of Decision
The Witness
The Young Lieutenant
Wanted - 7 Fearless Engineers!
Wayside Courtships
We Didn't Do Anything Wrong, Hardly
Wessex Tales
With Airship and Submarine
With Wolfe in Canada
Wolves of the Sea
Woman Triumphant
Wood Magic: A Fable

References

Kernighan, Brian W. and Ritchie, Dennis M. *The C Programming Language, Second Edition*. Upper Saddle River, NJ: Prentice Hall, 1978. Print.

"Free Ebooks - Project Gutenberg." *Project Gutenberg*. N.p., n.d. Web. 31 Dec. 2013.

"What Is a Lexile Measure?" *What Is a Lexile Measure?* MetaMetrics Inc., 2013. Web. 31 Dec. 2013.

"Most Common English Words." Wikipedia. Wikimedia Foundation, 25 Oct. 2013. Web. 18 Mar. 2014.

Wall, Larry, Tom Christiansen, and Randal L. Schwartz. *Programming Perl: Second Edition*. Beijing, China: O'Reilly, 1996. Print.

Acknowledgements

I would like to thank my mentor Paul Welch for his frequent helpful guidance on the project throughout this school year. I would also like to thank Mr. Lee Goodwin for being a wonderful teacher sponsor this year and the Supercomputing staff for providing such a unique opportunity.