Literature Analysis

Divya Iyer, Brennan Ayres, Tijana Cosic, John Berry

Introduction

Our Literary Analysis project is structured to provide insights into an author's literary signature. We aimed to learn more about the way an author composes their publications in order to compare with the writing styles of others. We wanted to provide a way for anyone to gain an understanding of an author's style and sentiment from just a glance. We chose to do this project because it was explained interestingly to the class and appeared to have a clear goal in mind.

We think some potential applications could include sentiment and style analysis on types of publications other than novels. Perhaps it could be used to detect bias in the media, or to recommend similar books to someone who enjoys reading a certain type of book. More generally, literature analysis has the unique ability to show trends in a body of text that a human could not detect on their own. It serves as a tool for someone to better understand the author's tone and writing style which may go completely unnoticed by a novice reader. In the same way a toddler is unable to discern sarcasm and other nuances of language, we may be incapable of picking up on trends in a body of text that could give useful insight into the author's signature.

Technical Approach

Our project used text mining and data visualization concepts in order to compare books and show the similarities and differences between them. In order to analyze the text from books, we used the Gutenberg python wrapper package to take advantage of Gutenberg's API. With over 60,000 eBooks, the Gutenberg library was the perfect free and online library to conduct our analyses. Having connected to the Gutenberg API, we used the elasticsearch database to store the cleaned versions of the books we retrieved to have easier accessibility for our analysis methods.

Once stored, we built several methods that analyzed complexity, sentiment, and more, to glean high-level information about each book. This provided us with a foundation to conduct future analyses where we review two or more books side-by-side.

Our data consisted of historical novels that we tokenized and stored as strings and lists of strings. The biggest issue of data integrity that we had to resolve was in the cleaning of text - level-setting all of our data with regard to capitalization, punctuation, line spacing, line breaks, etc.

Methodology

Our framework consists of eight major components. The first component was the text storage (text_storage.py). This piece of the framework retrieved the desired book by index from the Gutenberg library and added it to the elasticsearch database for more accessible use for the analysis methods. Regarding the analysis, there are six components that address various forms of analysis for literature.

The main piece of the sentiment analysis (sentiment_analysis.py) section is essentially the line graph that is presented. The sentiment analysis component of our framework has the ability to divide the sentiment analysis scores into any number of sections and it takes in a list of numbers representing the indices of the desired books from the Gutenberg library, a string representing either “pos”, “neg”, “neutral”, or “compound” sentiment, and it takes in the number representing the number of desired sections. This outputs a line graph displaying the progression of the desired sentiment score for the given list of books.

Another analysis component is finding the most common overlapping words in two books. This piece takes in two integers representing the index of the desired books from the Gutenberg library and it also takes in an integer representing the number of desired common words. This outputs two bar graphs for visualizing the most common words that are present in each book and also are present in both books. As we will address in our results section, this was not an effective analysis method and we elected to enhance and replace this with another analysis component known as theme_words.py.

The analysis method that acted to replace finding the most common words was finding the most frequently used genre specific words in a list of books of any length. This component takes in a list of genre-specific words and a list of the indices of the desired books from the Gutenberg library. This component presents a grouped bar graph of the normalized frequency of words from the given list of genre specific words that are present in each of the books.

Our Zipf’s Law analysis allowed us to evaluate the ‘normalcy’ of a piece of writing, as determined by its adherence to Zipf’s Law. Zipf’s Law describes the inverse relationship between rank and frequency, and is often used as a level-setter for data analysts to determine how efficiently a message/idea/story is conveyed in a piece of writing (in other words, the level of optimization of a piece of writing). As is generally true, we found prepositions (e.g., “the”, “a”) occurring most frequently along the curve (lowest rank), and more specific words like characters, adjectives, and action verbs (e.g., “Frankenstein”, “looked”). (Appendix. Exhibit 1)

Our Heaps’ Law Analysis allowed us to evaluate the level of sophistication of a body of text as measured by its number of unique words per overall document length. In accordance with this law, bodies of text should see an increasing number of unique words at a decreasing rate as the body of text increases. Despite this proposition, we found in our research that even at a given novel’s conclusion there seems to be a growth rate in the number of unique words that is somewhat linear. This result indicates that not all bodies of text show an asymptotic curve in the number of distinct words over the length of the text.

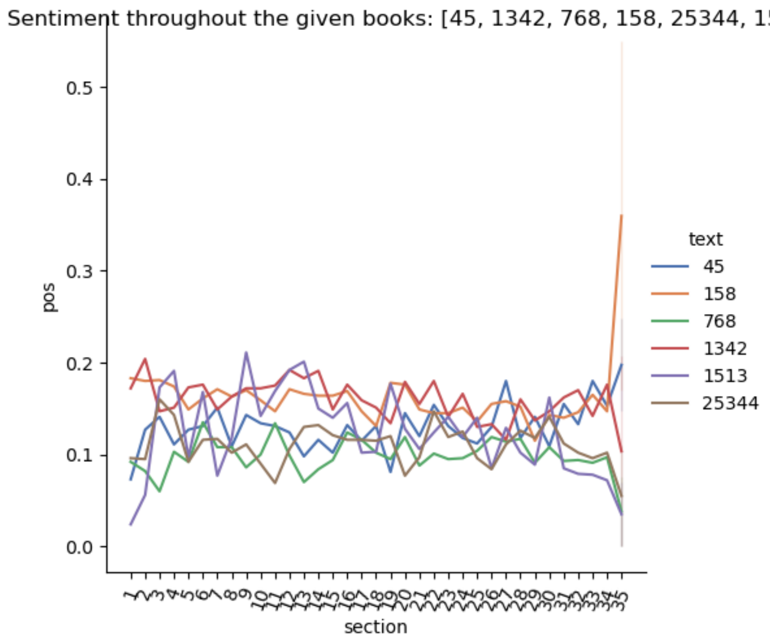
Finally, we designed word cloud graphers as helpful, stylistic visuals of our most common word analyses. These visuals offer an interesting depiction of word frequency, using word size as a function of frequency. Furthermore, filtered correctly (by removing all relevant stop words, for instance), these graphs can provide viewers with potential high-level themes of a story.

Results and Analysis

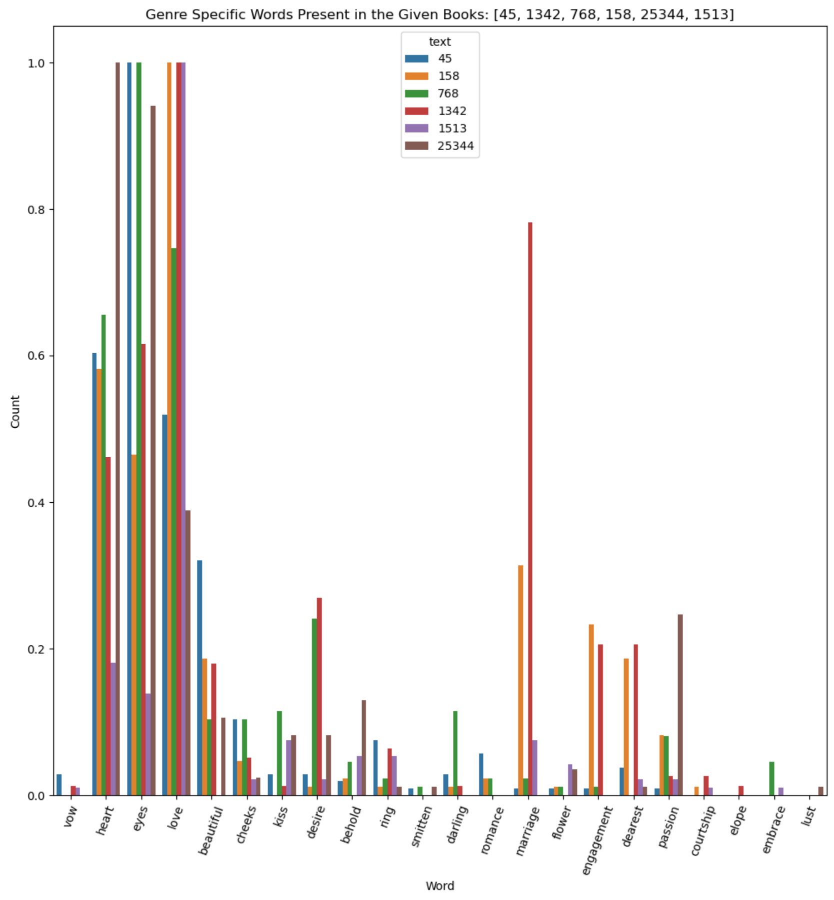
Link to GitHub Repo: https://github.ccs.neu.edu/bfayres/ds3500_lit_analysis

I. Romance Books

Title of Books used and Corresponding Index Number: Anne of Green Gables (45), Pride and Prejudice (1342), Wuthering Heights (768), Emma (158), Scarlet Letter (25344), Romeo and Juliet (1513)



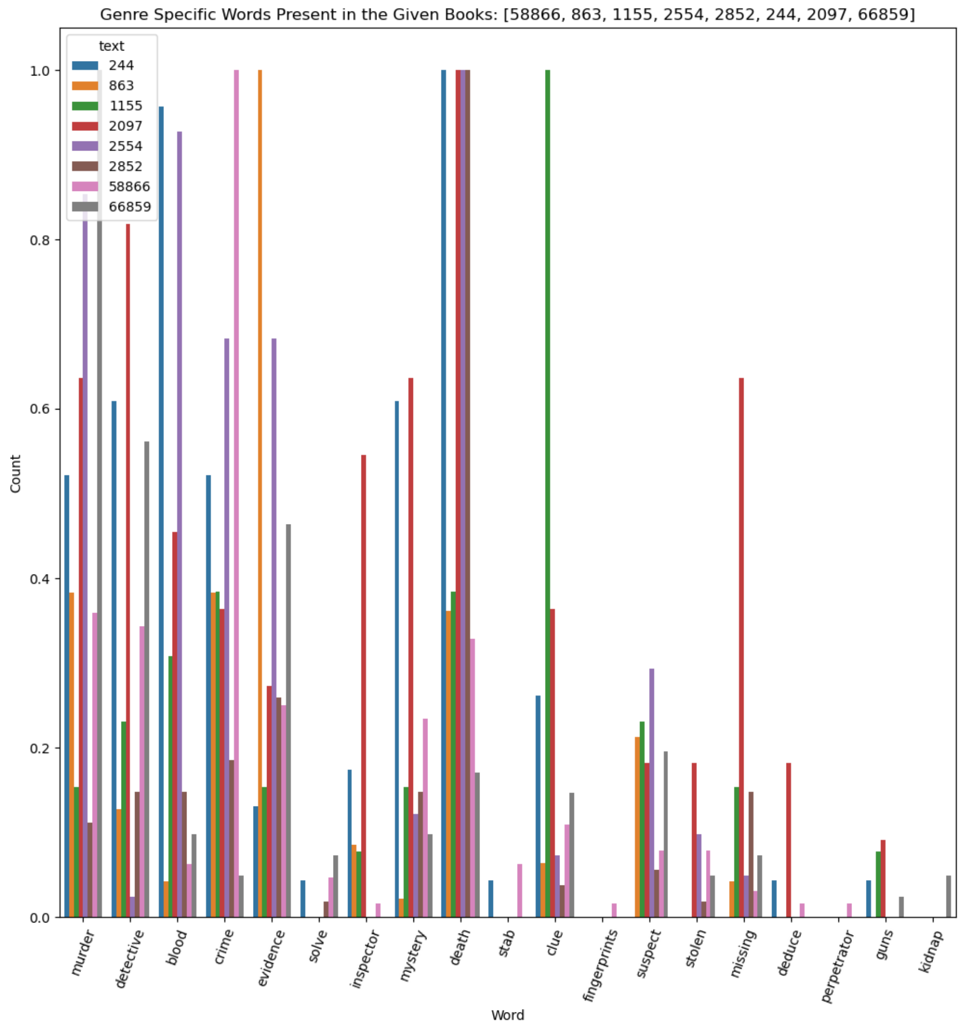
This visual displays the positive sentiment progression throughout 35 approximately evenly divided sections for a list of books categorized in the romance genre. The books used were Anne of Green Gables (45), Pride and Prejudice (1342), Wuthering Heights (768), Emma (158), Scarlet Letter (25344), and Romeo and Juliet (1513). Over the course of these sections, it can be seen that each book generally follows a similar trend. For each book, the positive sentiment stays overall between approximately 0.05 and 0.25. A point that stands out is that the positive sentiment score for Emma (158) drastically increases to a little under 0.4 in the last section, whereas the positive sentiment score for most of the other books, specifically Pride and Prejudice, Romeo and Juliet, Wuthering Heights, and Scarlet Letter decreased in the last section. Another interesting thing to note is that the range of positive sentiment scores for almost all books was greater in the first half of the books than in the second half. This is made clear by the smaller y-values as the section numbers (x-axis) increase. This could be because at the beginning of most of these romance books, the plot is more positive than in the end, considering that some of these books end in tragedy.



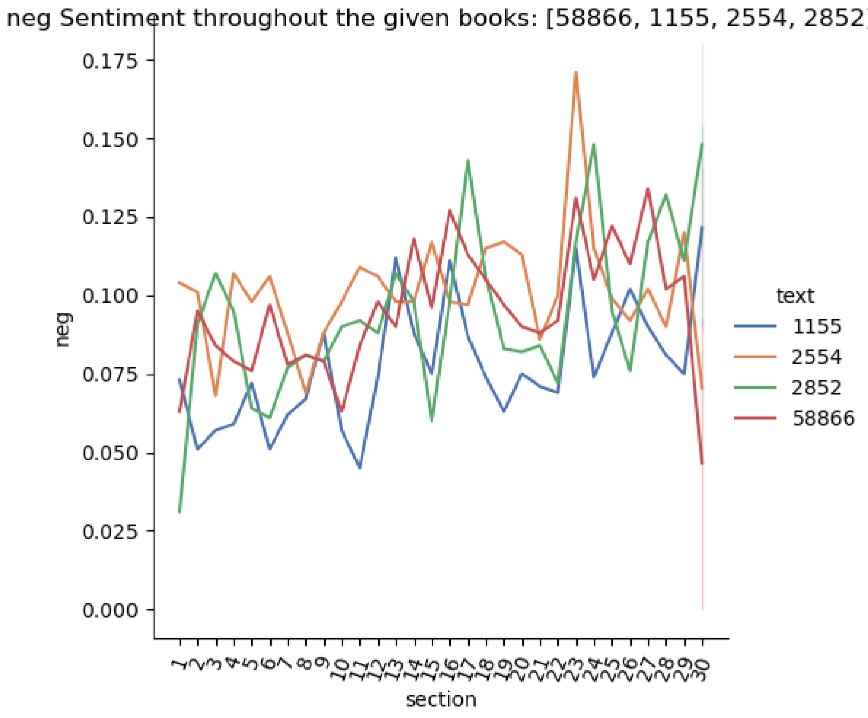
This visual displays the normalized amount of times each word in the given list of genre-specific words appears in each book provided. The genre associated with this visual was romance and the books used were Anne of Green Gables (45), Emma (158), Wuthering Heights (768), Pride and Prejudice (1342), Romeo and Juliet (1513), and Scarlet Letter (25344). The most occurring genre-specific words used across these books are ‘heart’, ‘eyes’, and ‘love’, as every book in the list uses these words. All of these words are expected to be used in romance related books. A point that stands out from this graph is that Pride and Prejudice (1342) uses the word “marriage” significantly more frequently than any other book. It can also be interesting to look at what kinds of words are being used the most within these books. Most of these books use nouns having to do with romance, such as the ones mentioned earlier (‘heart’, ‘eyes’, ‘love’), as well as ‘marriage’, ‘engagement’, and ‘passion.’ The verbs and adjectives, such as ‘smitten’, ‘elope’, ‘embrace’, and ‘lust’ are seen significantly less than the nouns throughout the course of these romance books.

II. Mystery Books

Title of Books used and Corresponding Index Number: The Murder on the Links (58866), The Mysterious Affair at Styles (863), The Secret Adversary (1155), Crime and Punishment (2554), Shadow in the House (2852), The Hound of the Baskervilles (244), A Study in Scarlet (2097), and The Sign of the Four (66859)



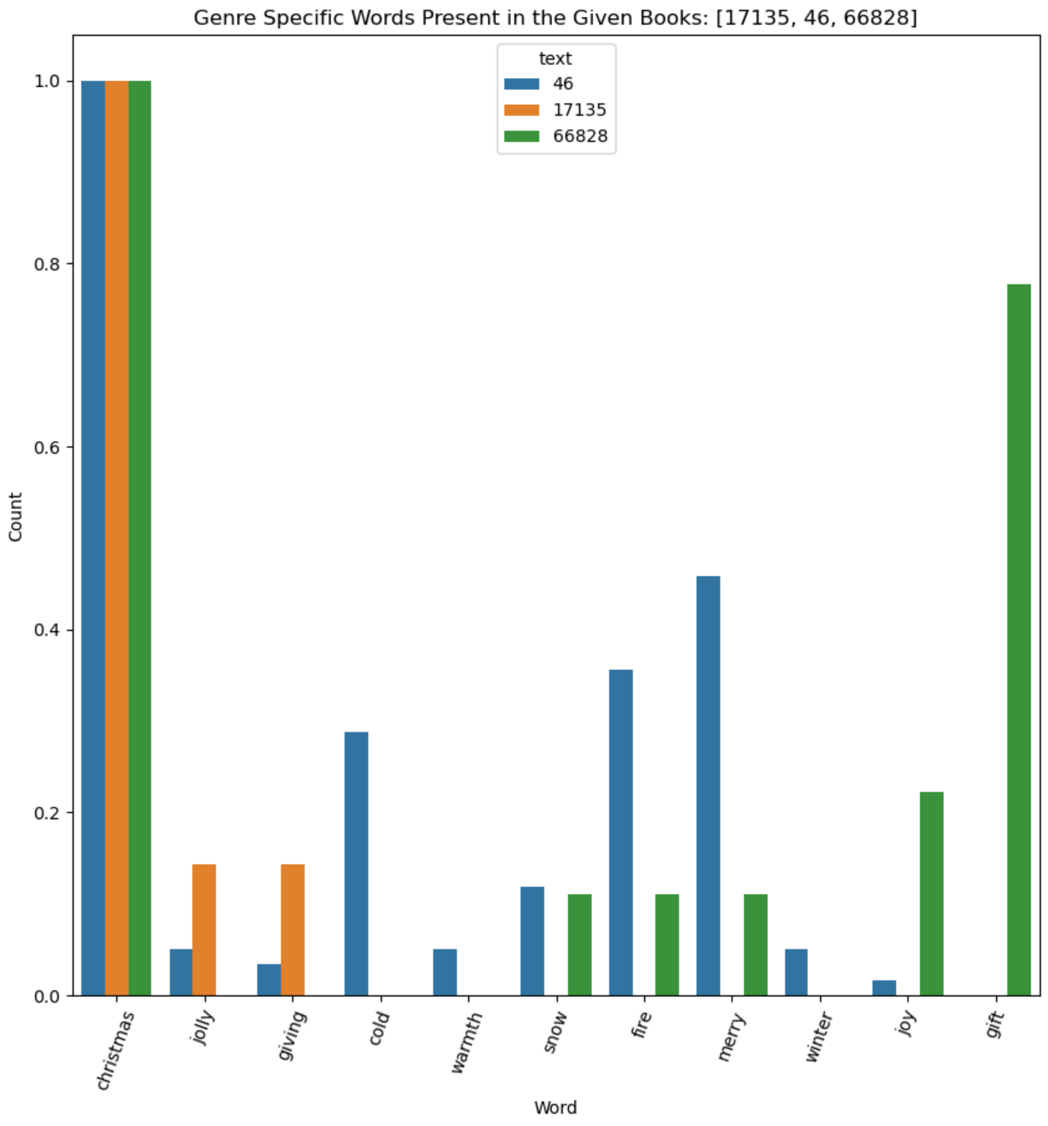
This visual displays the normalized amount of times each word in the given list of genre-specific words appears in each book provided. The genre associated with this visual was mystery and the books used were The Hound of the Baskervilles (244), Mysterious Affair at Styles (863), The Secret Adversary (1155), A Study in Scarlet (2097), Crime and Punishment (2554), Shadow in the House (2852), The Murder on the Links (58866), and The Sign of the Four (66859). Expectedly, the most prominent genre-specific word generally used across these books is "death." One noticeable point is that The Secret Adversary used the word "clue" significantly more frequently than any other books in the given list. The Mysterious Affair at Styles also uses the word "evidence" more frequently than the other books. From the visual, it's evident that there are some words that are used significantly across all books, yet there are also multiple words that aren't used at all by almost all the books. This goes to show a drastic difference/contrast in mystery books and how the amount of negative words in a specific genre can vary so much.



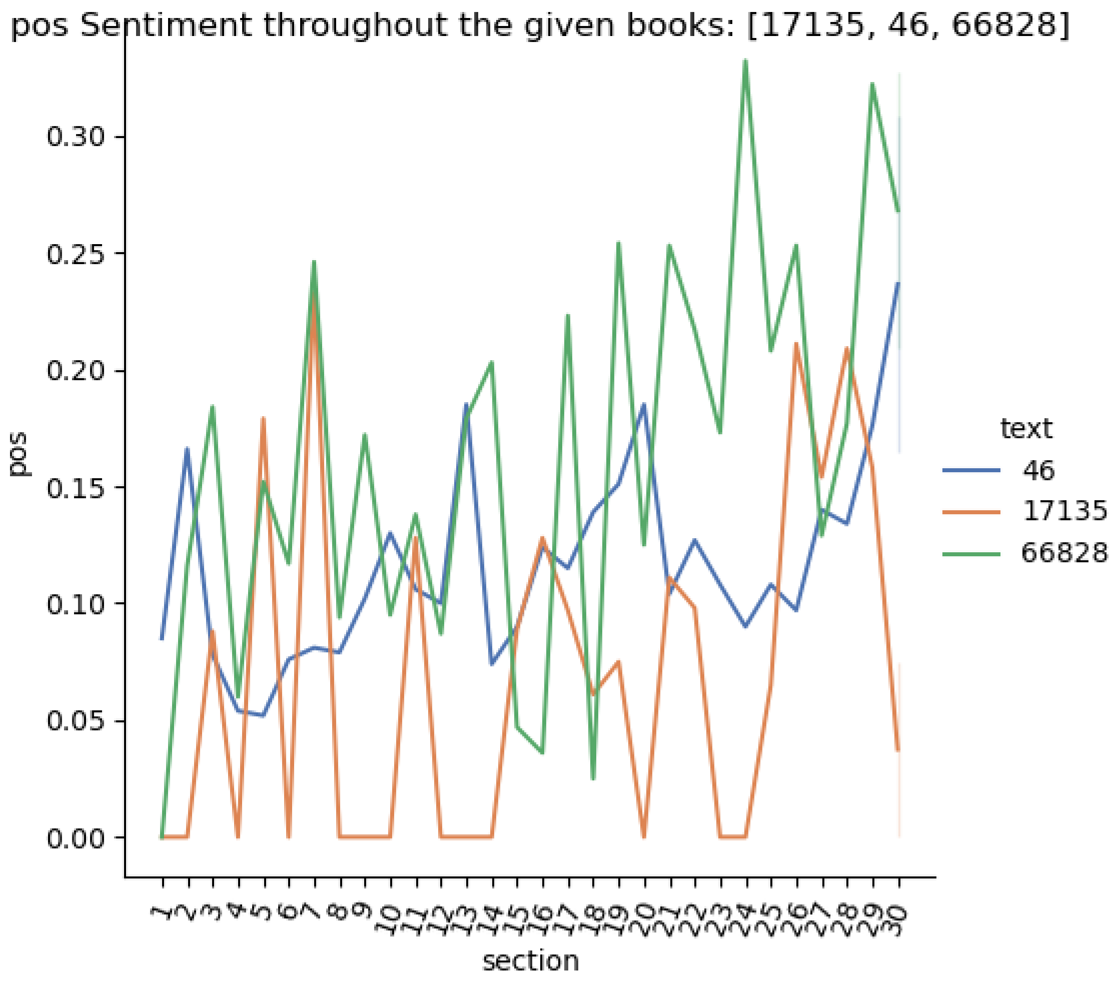
This visual displays the negative sentiment progression throughout 30 approximately evenly divided sections for a list of books categorized in the genre horror. The books used were The Secret Adversary (1155), Crime and Punishment (2554), Shadow in the House (2852), and The Murder on the Links (58866). Over the course of these sections, it can be seen that each book generally follows a sporadic trend. However, there tends to be a general peak between the 23rd and 26th section. In the beginning sections, the negative sentiment score for The Murder on the Links and the Shadow in the House both increased. However, the negative sentiment score for The Secret Adversary and Crime and Punishment increased in the first few sections. We also see that as the section numbers increase and we move farther along in the books, the general trend of negative sentiment scores tend to move in an upwards direction. In the end sections, it's clear that half of the books take a sharp decline in their negative sentiments, meaning they become less negative at the end. The other half of the books show a sharp increase in the negative sentiment scores, which means that they become more negative at the end.

III. Christmas Books

Title of Books used and Corresponding Index Number: A Christmas Carol (17135), Twas' the Night before Christmas (46), and A Pilgrim's First Christmas (66828)
christmas = [17135, 46, 66828]

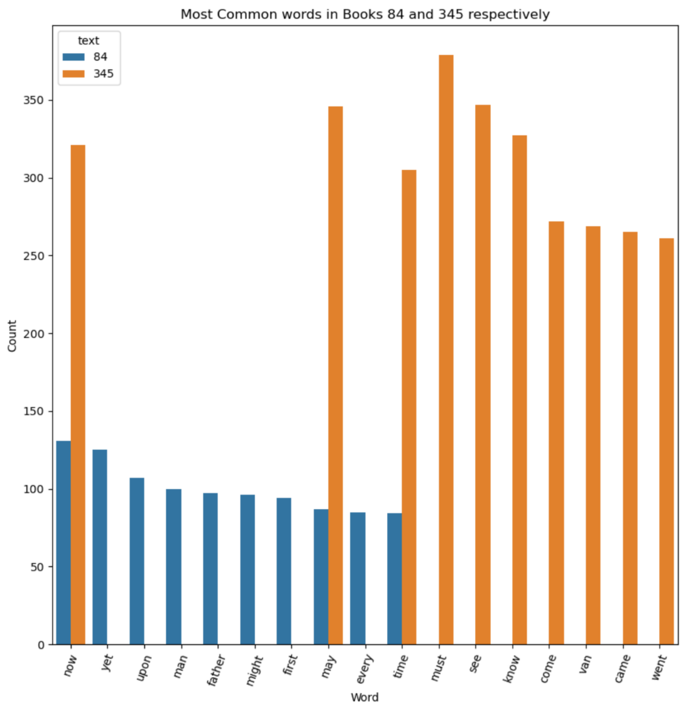


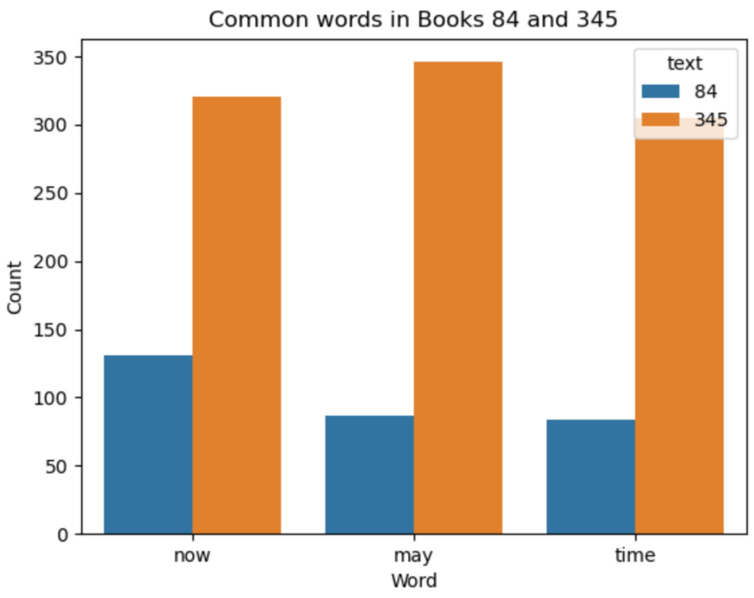
This visual displays the normalized amount of times each word in the given list of genre-specific words appears in each book provided. The theme associated with this visual was Christmas and the books used were Twas' the Night before Christmas (46), A Christmas Carol (17135), and A Pilgrim's First Christmas (66828). The most occurring genre-specific word used across these books is unsurprisingly ‘Christmas’, as every book in the list uses this word incredibly frequently. This is also the only word in the genre-specific list that all three books have in common. Words such as “snow”, “fire”, “merry”, and “joy” are only used in the Twas’ the Night before Christmas book, as well as the A Pilgrim’s First Christmas book. The words “jolly” and “giving” are only used in Twas’ the Night before Christmas and A Christmas Carol. It’s interesting how all of the other words don’t even compare to the word ‘Christmas.’ This can be interpreted to mean that most Christmas movies



This visual displays the positive sentiment progression throughout 30 approximately evenly divided sections for a list of books categorized in the genre horror. The theme associated with this visual was Christmas and the books used were Twas' the Night before Christmas (46), A Christmas Carol (17135), and A Pilgrim's First Christmas (66828). Over the course of these sections, it can be seen that each book generally follows a drastically sporadic trend, having multiple peaks. There are several sharp peaks especially in Twas' the Night before Christmas and A Christmas Carol. The positive sentiment score for each book generally increases in the initial sections, but the positive scores in the ending sections for A Pilgrim's First Christmas and Twas' the Night before Christmas both decrease. The positive sentiment score for Twas' the Night before Christmas increases in the ending sections. An interesting point is that in approximately the seventh section, both A Christmas Carol and A Pilgrim's First Christmas follow an incredibly similar progression in their positive sentiment scores.

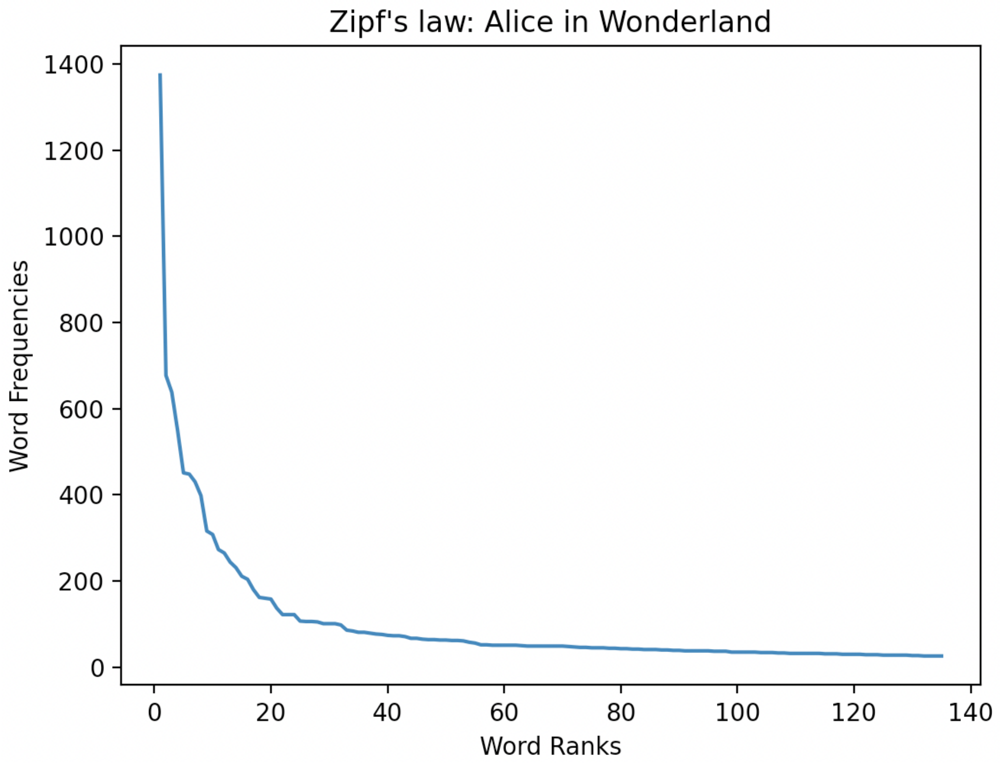
An analysis component that we created but unfortunately did not end up being effective was determining the most common words between two books.

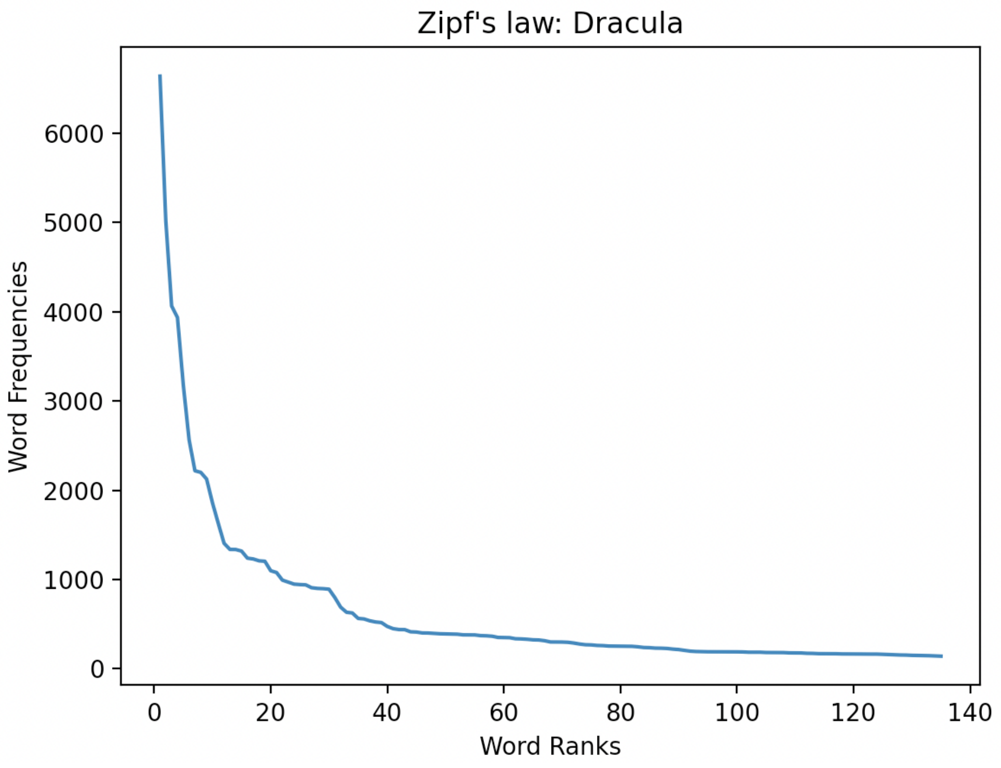




The two graphs above display the most common words in Frankenstein (84) and Dracula (345) respectively as well as the most common words that are present in both books. It can be seen that the words in the visual do not provide any worthwhile insight for the analysis of these books.

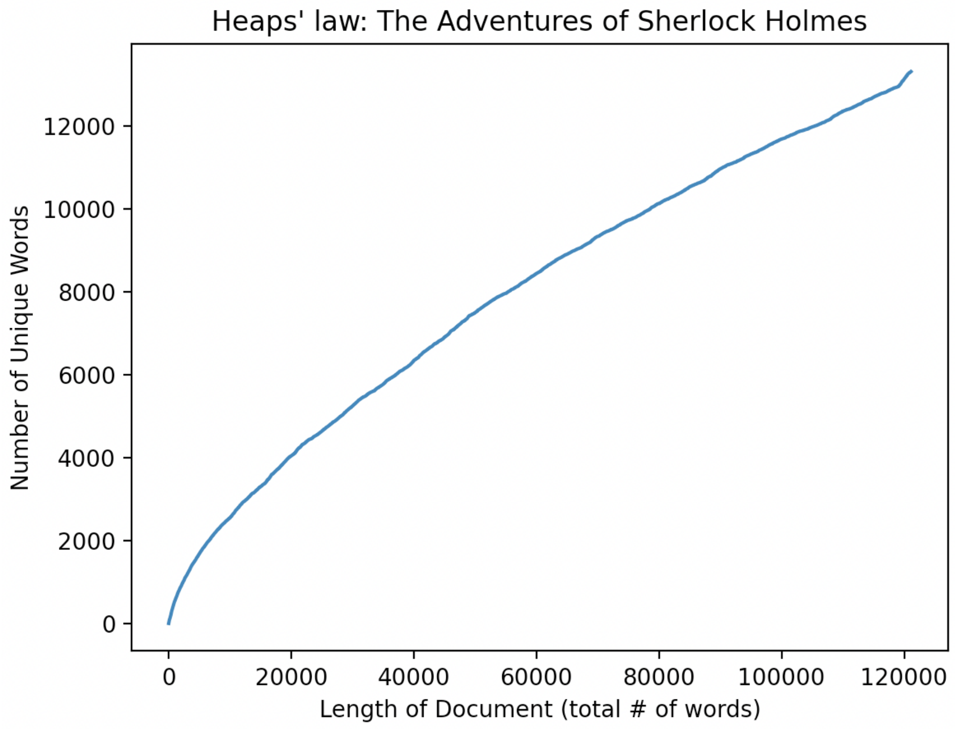
IV. Zipf’s Law Analysis Using Alice In Wonderland

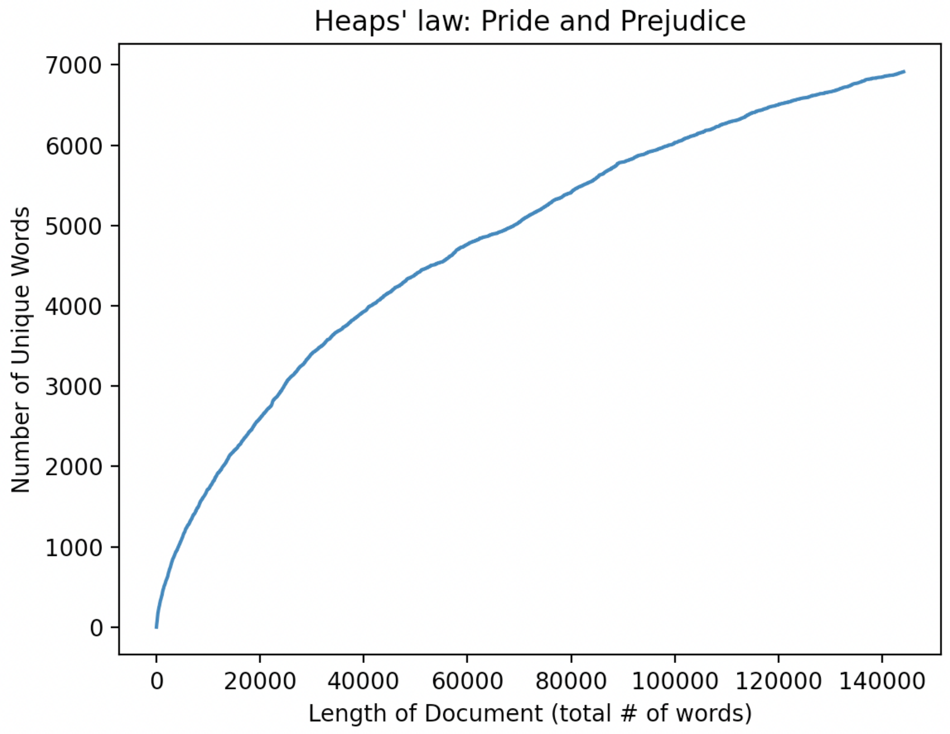




As is evident in the two Zipfian graphs displayed above (representing text data for Alice In Wonderland and Dracula, respectively) both texts closely follow the exponential curve described by the Zipf formula. In plain english, this curve is characterized by word frequencies that roughly decrease by half for each increase by one of a word's rank. These visualizations reveal the levels of optimization conveyed in both authors' story-telling abilities. The fewer deviations there are from a truly exponential curve, the closer the author's writing comes to maximizing its storytelling optimization.

V. Heaps’ Law Analysis Using Sherlock Holmes and Pride And Prejudice





Visualized in the two graphs above are Alice In Wonderland's and Sherlock Holmes' compliance with Heaps Law. This compliance is visualized by the positively sloped lines in each graph that show an exponential decay. This exponential decay represents the decrease in the number of new unique words given a one word increase in a novel's length. Hypothetically speaking, in an analysis of one novel, this decay would approach an asymptotic limit as the length of the novel approaches infinity.

Future Work

Given more time, there would be other functionalities that would be added. In order to address the issue with finding the most common words in two books, we could add the functionality of a tf-idf analysis. This is a numerical statistic that would reflect how important a word is to the book. This would allow for the elimination of irrelevant words that lack significant meaning for the analysis of the book.

Furthermore, it would be helpful to supplement our complexity analysis (by way of Zipf's Law and Heaps' Law), with a Flesch Reading Score analysis and comparison. The former two laws help us to perform an objective analysis of a piece of writing, whereas the Flesch scoring allows us to interpret complexity through the lens of a human reader. This two-pronged analysis would help us to triangulate a more accurate and thorough conclusion about the complexity of a piece of writing.

Additionally, it would be helpful to generalize and parameterize our Zipf's and Heaps' analyses to plot several curves on the same graph. This would reveal compliance with these laws between genres, time periods, and even between human and computer authors. Furthermore, we could generalize this code in conjunction with our text splitting code to compare different sections of a text's compliance with either law, indicating whether complexity evolves (increases or decreases) throughout a story, poem, speech, etc.

Finally, a downstream evolution of our word clouds could be helpful in supplementing our word-based sentiment analysis (for instance, colorizing the words in a word cloud by emotional association - positive or negative, happy or sad, etc.). This would enhance the visual impact of the graphs, as well as enhance their usability as an analytical tool and presentation element.

SOURCES

https://www.pythontutorial.net/python-basics/python-read-text-file/

https://pypi.org/project/textstat/

https://stackoverflow.com/questions/8369219/how-to-read-a-text-file-into-a-string-variable-and-strip-newlines

https://www.nltk.org/data.html

https://stackoverflow.com/questions/47103811/how-do-i-download-vader-lexicon-through-nltk-in-jupyter
https://stackoverflow.com/questions/40325980/how-is-the-vader-compound-polarity-score-calculated-in-python-nltk

https://stackabuse.com/removing-stop-words-from-strings-in-python/

https://www.nltk.org/api/nltk.sentiment.html

https://www.nltk.org/howto/sentiment.html

https://towardsdatascience.com/sentimental-analysis-using-vader-a3415fef7664

https://www.codedrome.com/zipfs-law-in-python/

https://towardsdatascience.com/simple-wordcloud-in-python-2ae54a9f58e5

https://stackoverflow.com/questions/40871180/elasticsearch-py-search-using-source-filter

https://theaidigest.in/extract-data-from-elasticsearch-using-python/

https://kb.objectrocket.com/elasticsearch/how-to-query-elasticsearch-documents-in-python-268

https://www.analyticsvidhya.com/blog/2021/05/how-to-build-word-cloud-in-python/

https://aclanthology.org/W98-1218.pdf

https://iq.opengenus.org/heaps-law-in-nlp/

https://stackoverflow.com/questions/9860606/heaps-law-in-python

https://nlp.stanford.edu/IR-book/html/htmledition/heaps-law-estimating-the-number-of-terms-1.html

https://towardsdatascience.com/very-simple-python-script-for-extracting-most-common-words-from-a-story-1e3570d0b9d0

https://gist.github.com/larsyencken/1440509#file-stopwords-txt

https://stackoverflow.com/questions/50912819/how-to-combine-two-bar-chart-of-two-files-in-one-diagram-in-matplotlib-pandas

https://penandthepad.com/vocabulary-words-writing-scary-stories-8153801.html

https://www.geeksforgeeks.org/normalize-a-column-in-pandas/