Predictive Text

Aiden Metz
Data 301
California Polytechnic State University

Abstract- In this project I explore making a text prediction function. I collected a large amount of text from classic novels, and used it to build Markov Chains in order to determine the probability of each following word.

Introduction

This research paper is found in the world and I am not a word to say that he was so much as he had been made by his own.

Sounds funny? That's because that wasn't written by a human. In this project I am going to explore using machine learning to predict text.

Predictive text is used in many applications today. Most cell phone keyboards, email services, and word processing softwares utilize some sort of predictive text. Data scientists are also designing chatbots and virtual assistants. These have the added difficulty of having to answer questions correctly as well as form good sentences. Beyond that many scientists believe that one day AI will be so advanced that it will be able to write a best selling book.

Data Collection

In order to build this model I needed a large database of natural english language. The patent on many classical books has expired so I was able to easily access the full text file for these books from Project Gutenberg.

I found a page with their 100 most popular books. I was able to successfully web scrape the links from that page. Once I had the links I was able to programmatically load all of the texts. Not all of the links worked but in all, I got about 80 books, from titles such as Pride and Prejudice and Mobey Dick.

Data Cleaning

There were a few steps included in cleaning the data from the project gutenberg library. It was encoded in UTF-8 so I had to account for that as well as to pull out all of the newlines and other type-in symbols. I also made all characters lowercase and removed punctuation. I realize that this will make it impossible for my algorithm to know how to complete a sentence, But I feel that it would be more feasible to limit the scope of the problem to just trying to figure out a next word.

Data Formatting

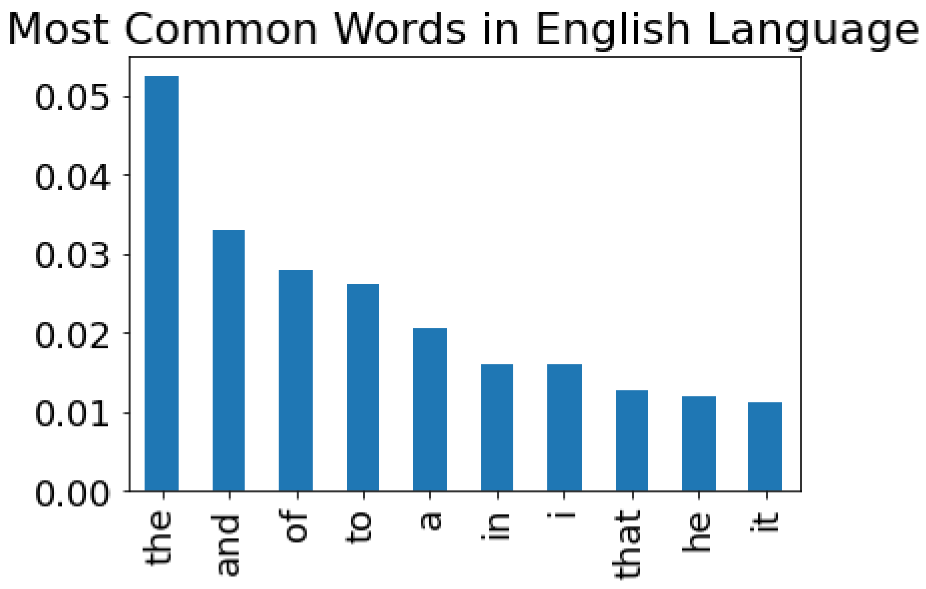
Now that I had a large corpus of text, I had to put it into a format in which I could do some analysis. I decided to first break all of the text into bigrams, and then group by first word. That way I am able to determine all of the counts, and respective probabilities of all of the words that come after a given word. I then broke the data into trigrams and grouped by first two words to find the probabilities of the words that came after. I carried this same analysis out on 4 and 5 grams as well. I ended up with over 17 million n-gram groups of various lengths all with a corresponding counter for the following words.

Predictive Text

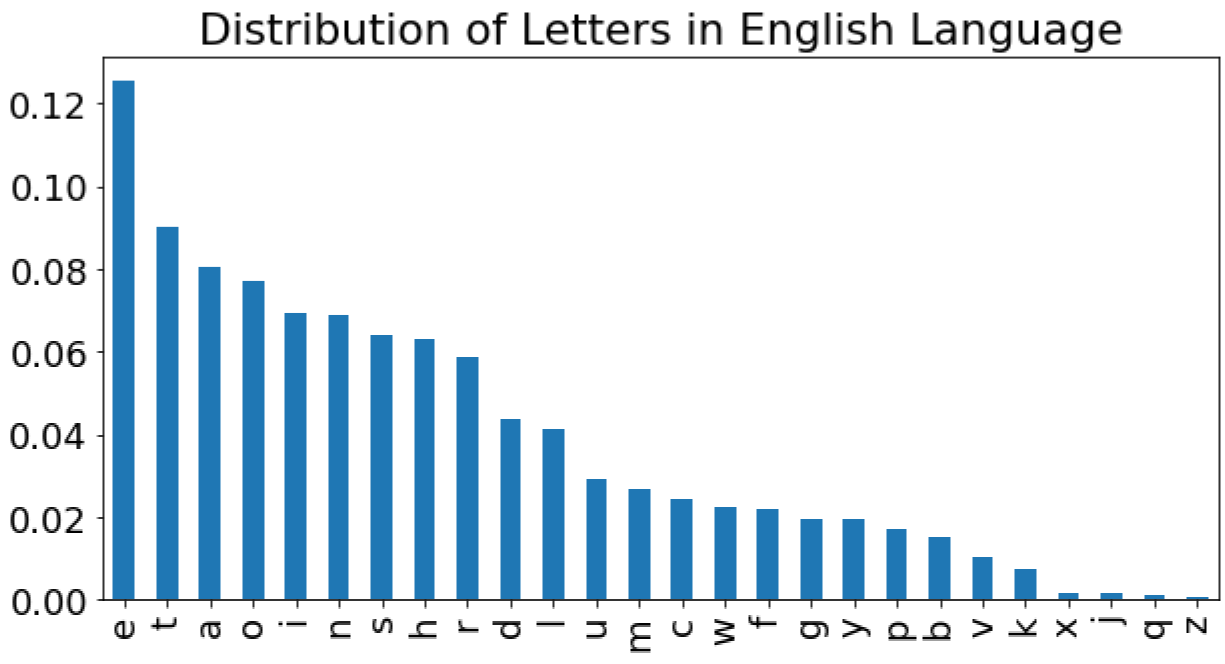
Aiden Metz
Data 301
California Polytechnic State University

Data Exploration

Now that the text is in a quantitative format, there is quite a bit of interesting exploration that can happen. Firstly, the most common words in the english language.



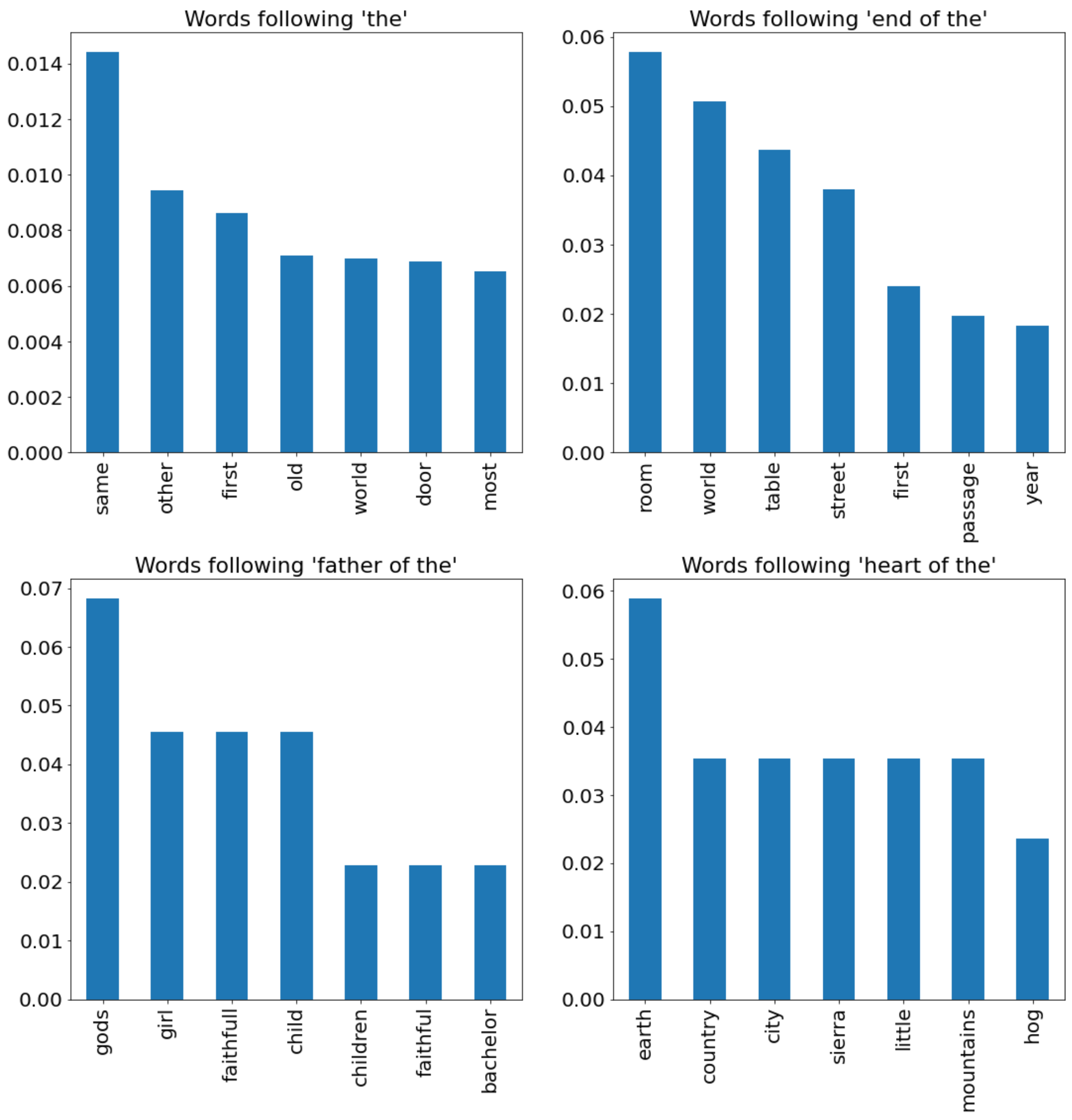
I also thought that it would be interesting to find the distribution of letters counts. The letter that surprised me the most was “t” because it was actually more common than most of the vowels.



Predictive Text

Aiden Metz
Data 301
California Polytechnic State University

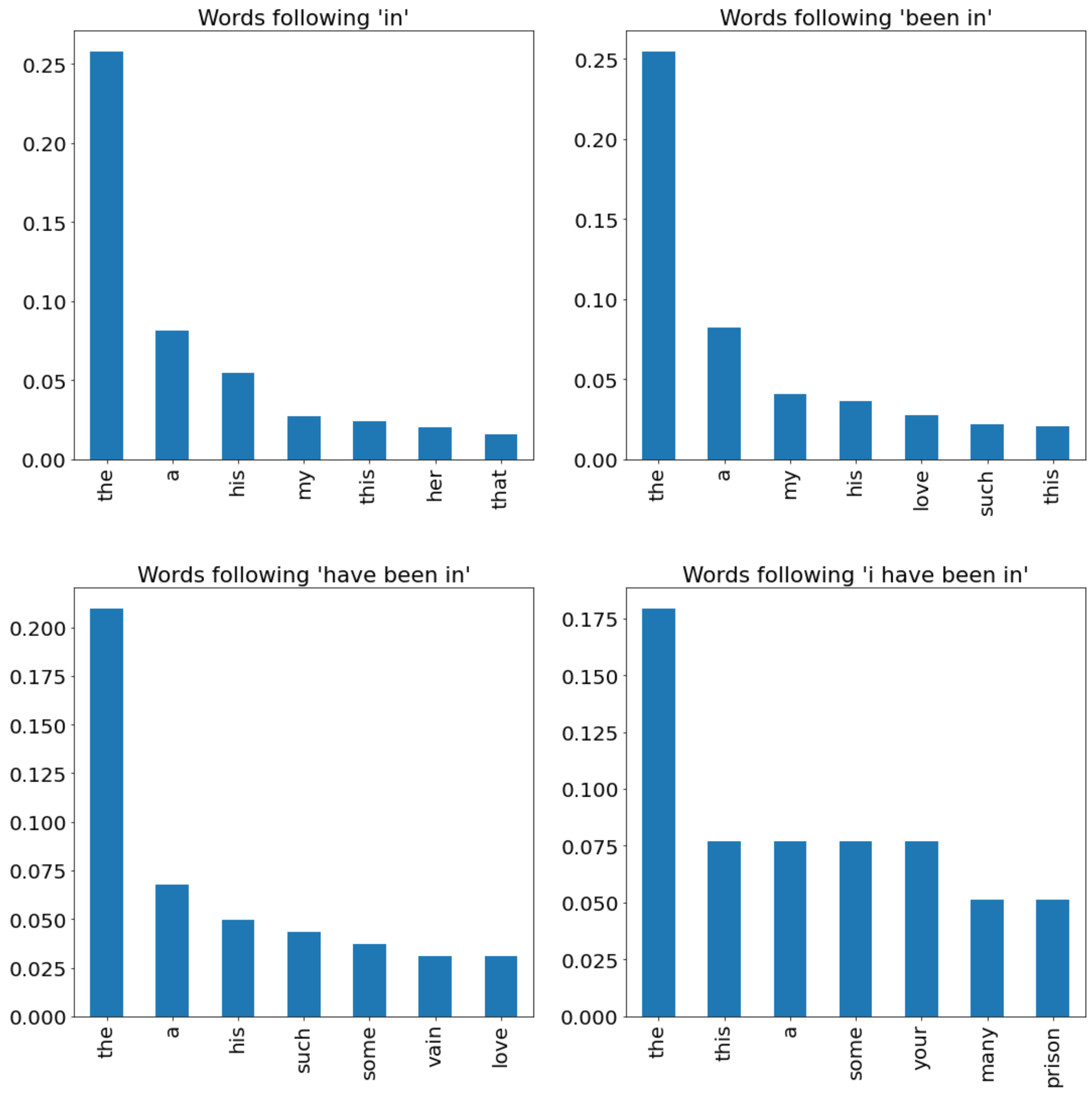
Next, I wanted to illustrate how I was able to look for words following certain groups of words. Understandably, common words like ‘the’, ‘and’, or ‘of’ come up alot, and work in so many places, that it could guess only common words and never really be wrong. However, I wanted to find an example where I could force the model to predict a less common word to show if it was actually learning. I settled on a sentence of the form: “_____ of the”. This forces a less common word as not many common words fit into the next spot. Depending on what I put in the blank, there could be multiple correct answers, but there are many words that I can put in the blank that do not share correct answers with each other. Here are the predictions for following just “the” as well as a few versions of the “____ of the” statement.



Predictive Text

Aiden Metz
Data 301
California Polytechnic State University

Finally, I wanted to demonstrate how a prediction can be made for the following word from any string between 1 and 4 words. In this manner I am able to favor looking at longer strings when making predictions, but I will always be able to match at least one word unless it is a super uncommon one that did not appear in my dataset at all. If I wanted to make a prediction for the string “I have been in” I can look at any of the previous N words. It is clear to see “the” is always the most common. After that however, the plots start to have a bit of disagreement about what comes next.



Predictive Text

Aiden Metz
Data 301
California Polytechnic State University

Machine Learning

This type of analysis is known as a Markov Chain. A Markov Chain is a useful model for sequential data. Essentially what a markov chain is is a directed graph where the nodes represent a state and the edges have an associated probability. In this context each node or state represents a word or a group of words and the edges all point to the words that came after the words in the current node, as well as the likelihood for each of those words. That is what makes it so easy to return the next most common word, or a list of multiple good suggestions.

I wanted to create a program that is as similar to a smartphone's text suggestion as possible. This means giving 3 possible suggestions for any input string. I wrote the function to only grab the first 3 unique predictions that it comes across, and then I just had it check the longer strings before the shorter ones in order to favor them.

I was able to create a function that made a prediction for the next word relatively well, but it was still a little bit of a departure from the user interface that I was picturing. I added a loop that constantly scans a text document as it is being written, and then outputs its predictions for the next word.

How could this be further improved

I think that both the data and the model could be further improved. The data was in classical writing style which was noticeable at points when making predictions. I think that it would be much more accurate if there was a database as a starting point, but then as the user used it, was able to add all of their new text to the database, and give the users text more of a priority, the model might be able to learn the users more unique writing style over time.

Markov chains are an excellent choice for this type of task, however there are models that can do much better. One of the best for sequential tasks such as text prediction are Long-Short term memory recurrent neural networks(LSTM). LSTMs are special layers in neural networks, which have special gates that allow them to methodically remember and forget data on previous predictions that it made. This has enabled these models to be trained to predict not the next word, but the next character, and still be able to form correctly spelled words and even correctly punctuated and capitalized sentences.

Conclusion

Overall I am extremely happy with where I got to on this project. The function is able to produce some good predictions for the majority of inputs. It isn't able to produce more than a couple of words in a row on its own that make much sense, as seen in the introduction, but it does still achieve its primary purpose well.