Book Genre Classification
Springboard Capstone 2
Evan Hintz

This project will focus on unsupervised machine learning models to classify various book texts into genres. Since book genres already exist I'm very curious to see the patterns that a machine learning model will pull out of texts as opposed to attempting to classify them into pre-existing themes. The end goal is to be able to offer companies a new and potentially better scheme for labeling books which will aid in consumer recommendation systems. Theoretically, an unsupervised model will be able to see patterns that may not be included in the traditional book genres.

The data used will be collected from Project Gutenberg, an online library of open-source texts (expired copyrights, or granted rights). Generally, this means that the texts will be significantly older. A selection of various genres will be picked but focused on fiction novels at this time although this is restricted to the books available which in this case will mean fewer fantasy and more gothic.

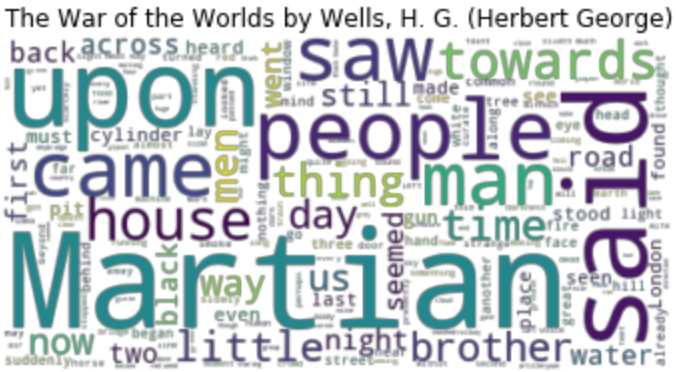
First the data must be gathered from Project Gutenberg. Currently, forty hand-picked books are being used, all chosen from the website's "Top 100" books. Thirty will be used for training and the remaining ten for testing.

The initial dive into the data was using sentiment analysis which included looking at the raw polarity and subjectivity of each book.

Text polarity attempts classify the emotions expressed within the text as generally positive, neutral, or negative. It returns a number between -1 and 1. The following graph displays all book's polarity. It is interesting to note that even the lowest value (War of the Worlds) is still above 0 and the highest (Emma) is still quite far from 1.

Text subjectivity analysis tries to give a numerical value to describe how objective or subjective the text is. In other words, did the text consist of factual statements (objective) or more emotional and opinionated (subjective). This scale is between 0 and 1, 1 being the most subjective or emotional. As you can see in the below graph there isn't much range in the subjectivity of the books, where almost all books are just about at the midway point - although non-fiction texts will generally receive much lower scores.

Word clouds are a great way to see the main focus on any texts. The following clouds are of a few of the books with the more extreme statistics from the other graphs shown.



Understanding the differences in books is important. Using the length of characters in each it is easy to compare the length of books and the range of sizes. The following graph shows the huge variance from a children's book (Tale of Peter Rabbit) to the expanded novel of The Count of Monte Cristo.

The next step is to clean and prepare the texts for the machine learning models. In natural language processing there are options for a lot of different types of wrangling. Depending how the text is being used they can contain a lot of noise and unwanted variety. This can include uppercase letters, punctuation, as well as different forms of words like plural,or past-tense. Texts also have a lot of words that don't necessarily add any substance to classifying models; these are known as 'stop-words'. Stop-words include 'the', 'and', 'a', and usually changes for

different uses. These are usually just removed from texts to remove the noise and increase processing speed.

The case of letters can create issues as Python will read "Snakes" as a different word from "snakes" however this can also change the meaning of words; for example with acronyms like "US" as a country versus "us". 'Immigrate to the US with us!' Most often the simplest solution is to bring all the letters to lowercase and accept the inaccuracies that will occur.

Punctuation will also have a large effect on the meaning of a sentence and this is another topic that machine learning models will struggle with and will often have negative outcomes on predictability. Generally punctuation is completely removed.

To start, the books are tokenized. This process separates the text as a single string into individual words that will be easier to process and understand for the models. Another technique is the use of n-grams which are strings of n words that often improve analysis.

Example: "I'm not surprised!" tokenized as ('im'), ('not'), ('surprised')
or a 2-gram: ('im'), ('not surprised')

Each of these can have very different effects when used as input to a ML algorithm. n-grams may be implemented later to compare.

Since all the words have been individually wrapped into their own strings it is now easier to remove any stop-words that are present to reduce noise and increase processing speeds with (hopefully) minimum effects on classification.

Next, each word is tagged with it's part of speech. This step is necessary in order for the following lemmatization to occur properly. Each word within the texts will be tagged with the correlating part of speech: verb = 'VB', noun = 'NN', etc. This will help the model better understand the words and their context within the book.

Lemmatizing is a form of standardizing or normalizing the text so that it becomes easier for the models to compare between books. Since this project is focused on unsupervised machine learning to classify genres, language and context is important between texts so lemmatization was chosen over stemming. Stemming is generally faster but will oftentimes convert a word to a stem that is not an actual word. Whereas lemmatizing brings the word to its root while keeping it a full word and it does this using a corpus (a collection of words) which makes it significantly slower. While speed is not currently a concern at this point, the use of stemming may be explored later on if this is applied to a larger database. Lemmatizing also provides more readability if necessary while working through this project.

Following lemmatizing the work can transition to machine learning models to start classifying each book.