Introduction

Detecting sentiment from a piece of text is a key part of Natural Language Processing (NLP). It is a foundational part of recommender systems which track a user's affinities and aversions with the aim of matching the user with a product/ service that's helpful for them.[1, 2] Much work on sentiment analysis has been done on short-form text data such as those derived from micro-blogging social media sites. Because of the succinct nature of these social media posts, it is perhaps easier to glean points of view than from longer pieces of text. [3]

I would therefore like to perform sentiment parsing of long-form text in the form of book reviews. The aim is to see if different models are as capable in situations where the user leaves a more verbose review. Book cataloging websites might benefit from insights gained from this project in order to improve their recommendation system. If longer reviews do result in less reliable recommendations the website might decide to implement a word limit on reviews.

I will use book reviews scraped from various websites including Goodreads stored on Kaggle as CSV files. The goal is to use different machine learning models to parse sentiment (either positive or negative) from each review. For this supervised classification problem, I will compare the performance of a neural network (Long Short Term Memory and Recurrent Neural Network) to a Naive Bayes model.

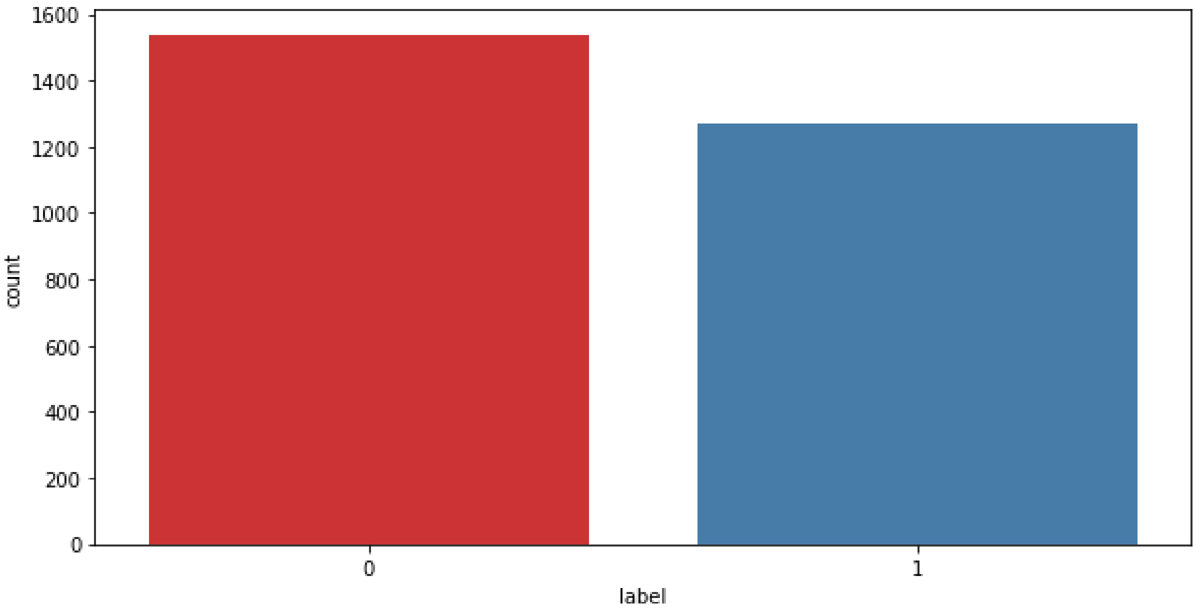
I decided on the Bernoulli Naive Bayes model because of its popularity in classifying text and its sometimes superior performance despite its less sophisticated assumptions[1]. Neural networks such as Long Short Term Memory (LSTM) and Recurrent Neural Network (RNN) are significantly more complex than Naive Bayes but they make up for this by being exemplary learners on sequential data such as text. The Long Short Term Memory (LSTM) neural network was chosen as a juxtaposition due to its ability to maintain information in memory for longer periods of time compared to the simple Recurrent Neural Network (RNN).[2] LSTM are also more capable of working with long range dependencies common in linguistics.[4]

Exploratory Data Analysis:

I obtained the first csv file containing reviews and their labels from the Kaggle website. The file contains two columns, one containing text of the review and the other column containing the rating (__label__1 for negative, and __label__2 for a positive rating). There were around 2,800 ratings. The dataset creator mined the ratings from different public websites but did not specify the particular source. [5]

The first task was to replace the labels. I exchanged the original __label__1 for zero to signify a negative rating and __label__2 to signify a positive rating. I then plotted the labels to get a better understanding of the proportions of each rating in the whole dataset.

a. Proportion of Positive vs. Negative Ratings in First Dataset



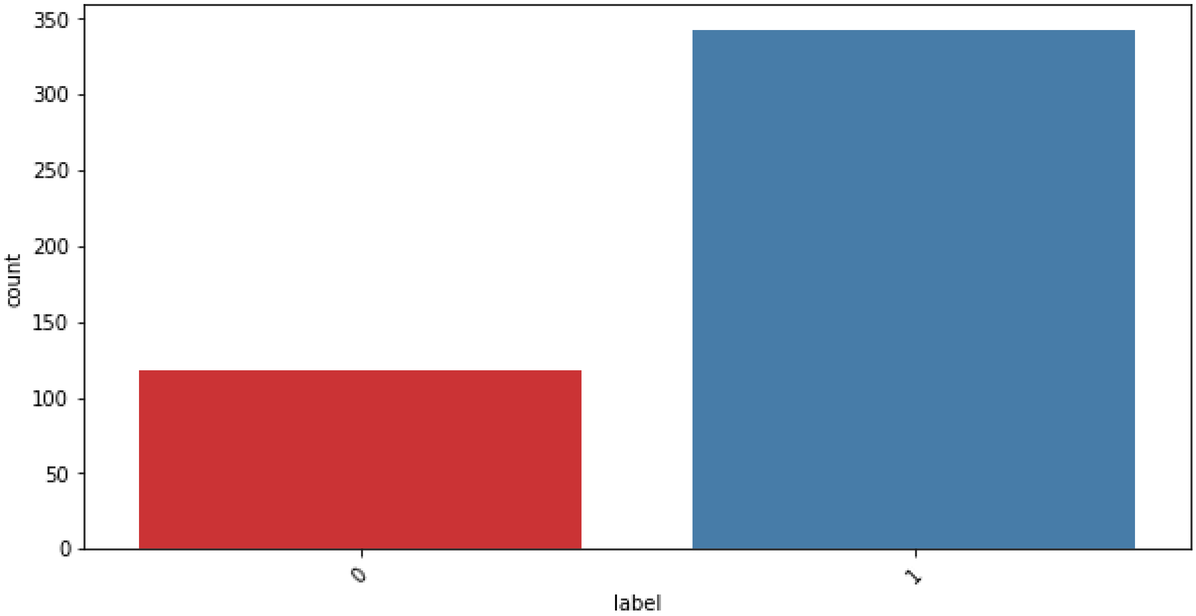
It is apparent that the dataset contains more negative ratings (a little over 1,500 instances) compared to positive ratings (almost 1,300 instances).

The second dataset was also obtained from Kaggle. It contains around 400 reviews obtained from the book review aggregator website Goodreads. The dataset had 14 columns one of which contained the text of the review and another which contained the rating of the review (from 0 to 5). [6] I sliced the dataset to only focus on the two columns containing the review text (column named 'review_text' and the rating review (column named 'review_rating').

Because the reviews were in a range from 0 (most negative) to 5 (most positive), I arbitrarily chose a cutoff of 3 with ratings less than 4 given a new label of zero for negative. The reviews given a rating of 4 and above were given a new label of 1 for positive in order to be consistent with the first dataset. I also renamed the 'review_text' column to 'text' and the 'review_rating' column to 'label' in order to preserve consistency.

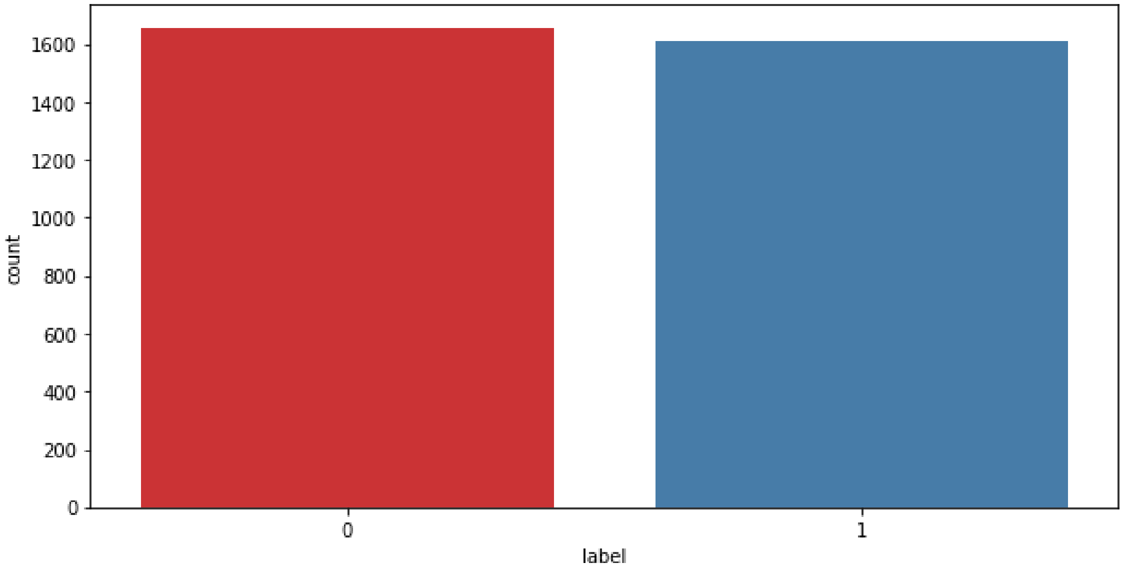
I plotted the 'label' column in order to visualize the proportions of the labels and obtained the following result.

b. Proportion of Positive vs. Negative Ratings in Second Dataset



I then concatenated the two datasets and the resulting data frame had about 3,000 rows and 2 columns. The reviews were in the 'text' column and the ratings were in the numerical column named 'label'. There were no missing values in this new data frame. To visualize the proportions of the two labels in the whole dataset, I plotted the 'label' column to obtain the next plot.

c. Proportion of Positive vs. Negative Ratings in Final Dataset



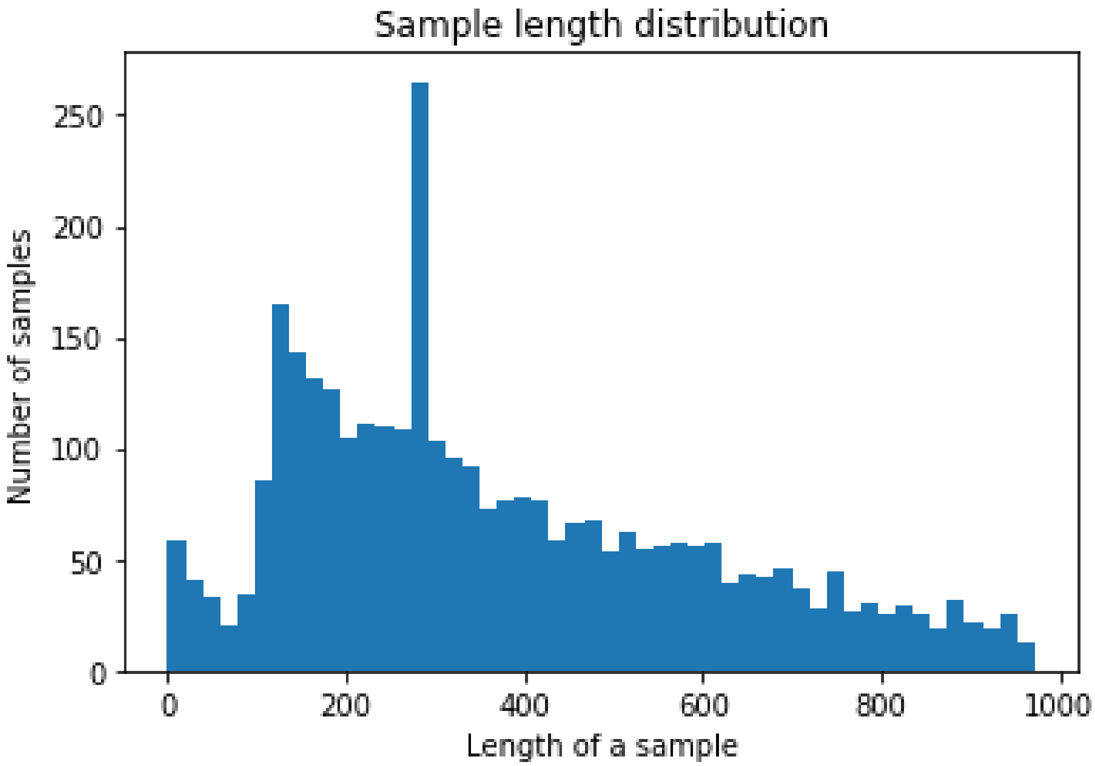
The proportions of negative (rated zero) vs positive (rated one) became more even with the addition of the second dataset of reviews. Obtaining value counts of the 'label' column revealed zero rated (negative) ratings to be 1, 656 while the one rated (positive) ratings were numbered at 1, 612.

Pre-processing Text

I ran the text through different functions to clean and lemmatize the test in preparation for the first model which was the Bernoulli Naive Bayes. The first function was to remove accented characters after noticing some of the reviews written in Spanish and Arabic. I decided to lemmatize the text instead of stemming in order to keep some of the context surrounding the word. [7] I used the default english language statistical model from the spaCy library to create nlp objects from each review. I then obtained the lemmas for each word that was not identified as a pronoun.

I then ran the text through another function sweeping for any left over special characters. I decided to keep digits in the reviews in the cases in order to have the models learn from entries where a reviewer might leave a numerical rating instead of a text rating. I then removed extra white space from the text before plotting the following sample distribution of the text.

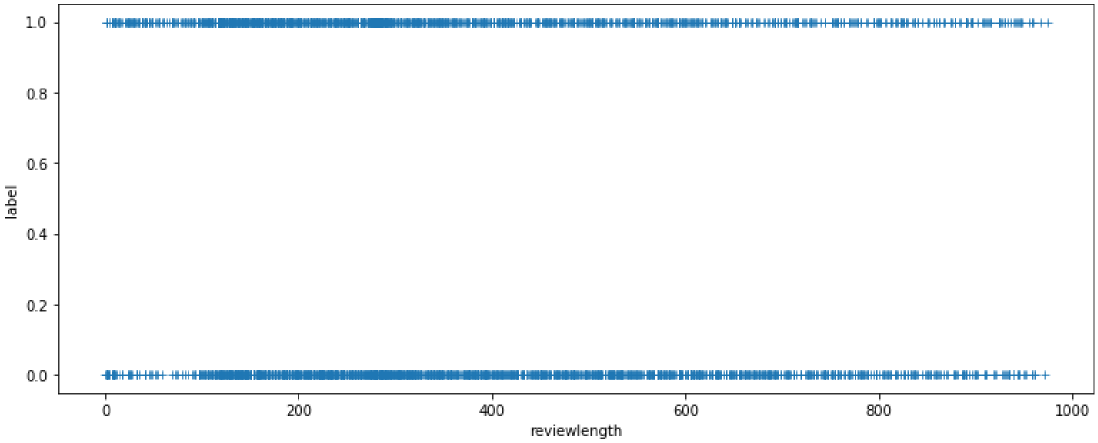
d. Distribution of Cleaned Text Length in final dataset



The resulting distribution does not follow any obviously identifiable pattern, but it is evident that the majority of reviews had a length of over 300 words (specifically, average was 374 words). The longest review had 974 words while the shortest review had no words. The text in this dataset seems to be longer than most social media texts used for sentiment analysis. For comparison, posts on the social media website Twitter which imposes a character limit contain an average of 33 characters.[8]

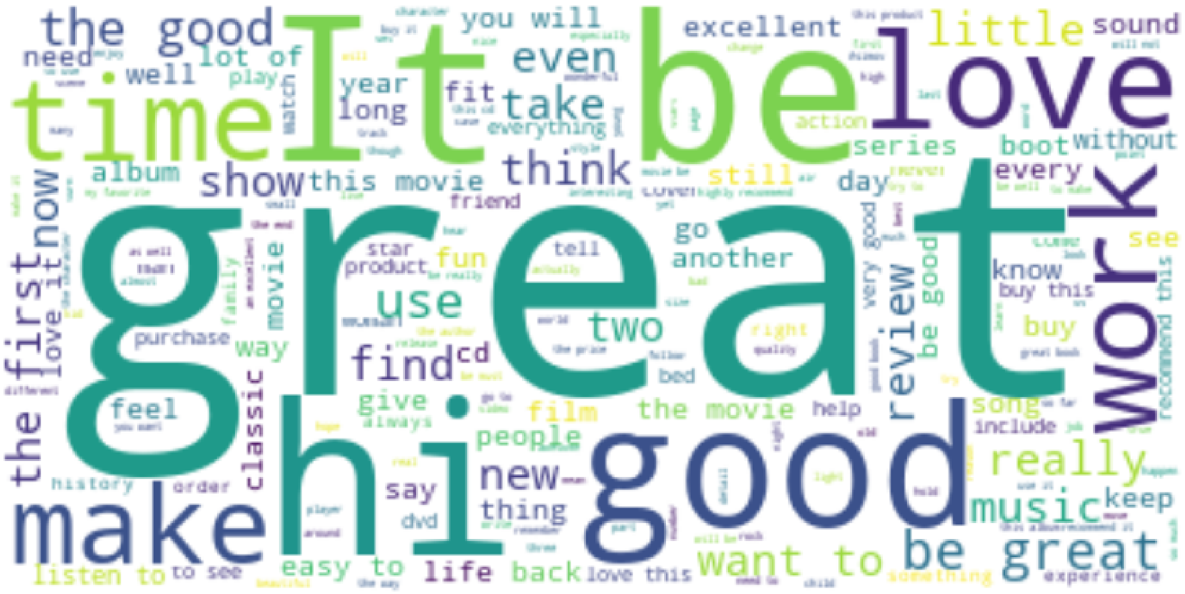
I then created a scatter plot to see whether longer reviews were more likely to be rated positive/ negative reviews compared to the shorter reviews. The plot shows no distinguishable relationship between the length of a review and its likelihood to be positive or negative.

e. Investigating the Correlation Between Review Length and Label

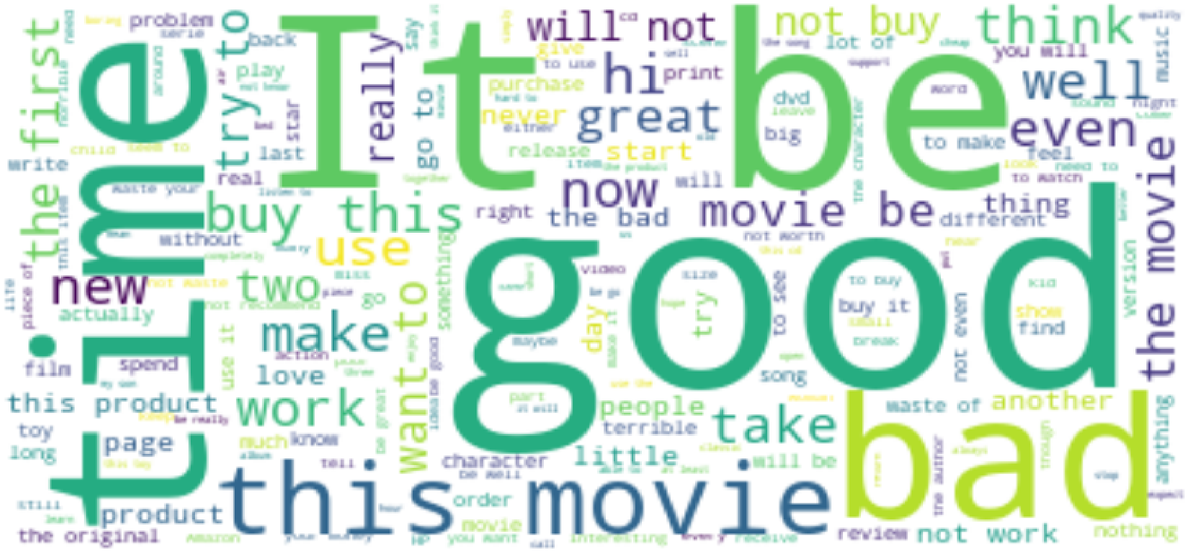


I then split the dataset into two with the first one containing only the positive reviews while the second one contained only the negative reviews. I then created a wordcloud of the text in the respective dataframes. After removing common stop words including “book”, “it”, “be” I obtained the following results. The most prominent words in the positive reviews are largely positive as is expected with the words “great” and “good” crowding out the rest of the words in the visualization. The word cloud for the negative reviews was less clear cut with the most prominent words being “time”, “good” and “bad”.

f. Visualising the Most Common Words in the Positive Reviews



g. Visualising the Most Common Words in the Negative Reviews



References:

1. https://www.dataquest.io/blog/naive-bayes-tutorial/
2. https://analyticsindiamag.com/how-to-implement-lstm-rnn-network-for-sentiment-analysis/
3. Bermingham, A. & Smeaton, A.(2010). Classifying Sentiment in Microblogs: Is Brevity an Advantage? [PDF File]. Proceedings of the 19th ACM International Conference on Information and Knowledge Management. p 1833. Retrieved from http://doras.dcu.ie/15663/1/cikm1079-bermingham.pdf
4. https://en.wikipedia.org/wiki/Long-range_dependence
5. https://www.kaggle.com/rakeshkakati/book-reviews
6. https://www.kaggle.com/san089/goodreads-dataset
7. https://en.wikipedia.org/wiki/Lemmatisation
8. https://techcrunch.com/2018/10/30/twitters-doubling-of-character-count-from-140-to-280-had-little-impact-on-length-of-tweets/