Sentiment Analysis of Long-Form Text

Teresiah Kahura

Project Proposal

- Much work on sentiment analysis has been done on short-form text data such as those derived from micro-blogging social media sites.
- Because these social media posts are succinct, it might be easier to glean points of view than from longer pieces of text. [1]
- The aim of this project is to perform sentiment parsing of long-form text in the form of book reviews to see if different models are as capable in situations where the user leaves a more verbose review.
- I will compare the performance of Bernoulli Naive Bayes to neural networks: Long Short Term Memory (LSTM)and simple Recurrent Neural Network (RNN).

Exploratory Data Analysis

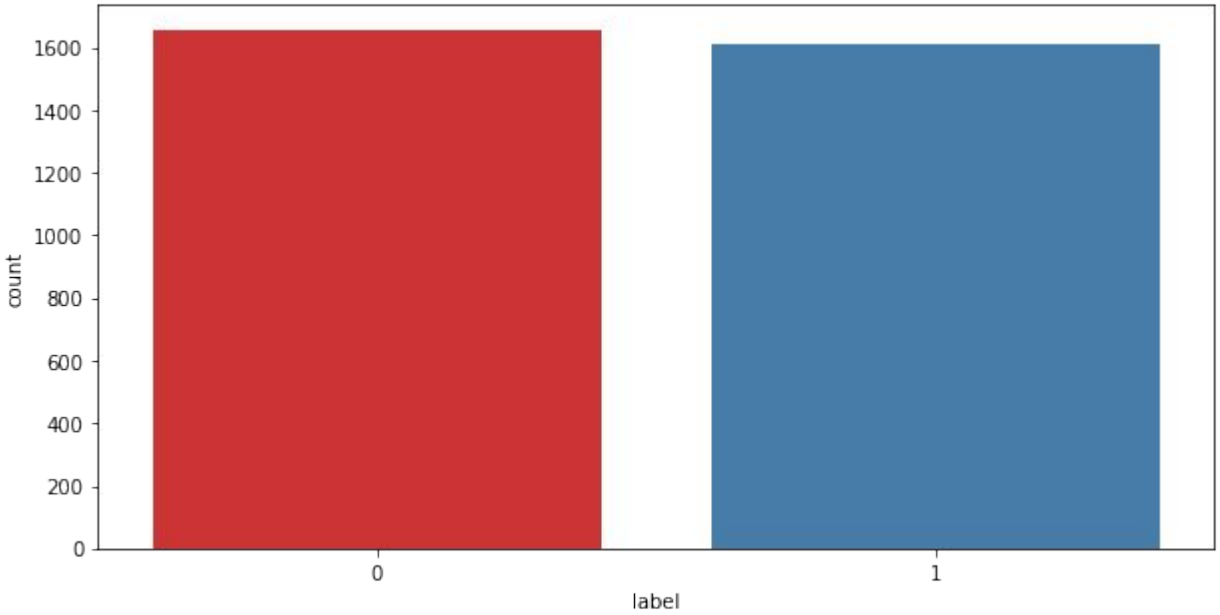
- I obtained the first csv file containing reviews and their labels from the Kaggle website.
    - Two columns, one containing text of the review and the other column containing the rating (__label__1 for negative, and __label__2 for a positive rating).
    - There were around 2,800 ratings. The dataset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

Exploratory Data Analysis

- The second dataset was also obtained from Kaggle. It contained around 400 reviews obtained from the book review aggregator website Goodreads.[3]
    - Contained 14 columns one of which contained the text of the review and another which contained the rating of the review (from 0 to 5).
    - I sliced the dataset to only focus on the two columns containing the review text (column re-named as ‘text’ and the rating review (column renamed as ‘label”).
    - Re-labeled ratings from 0-3 as negative (zero) and 4-5 as positive (one).

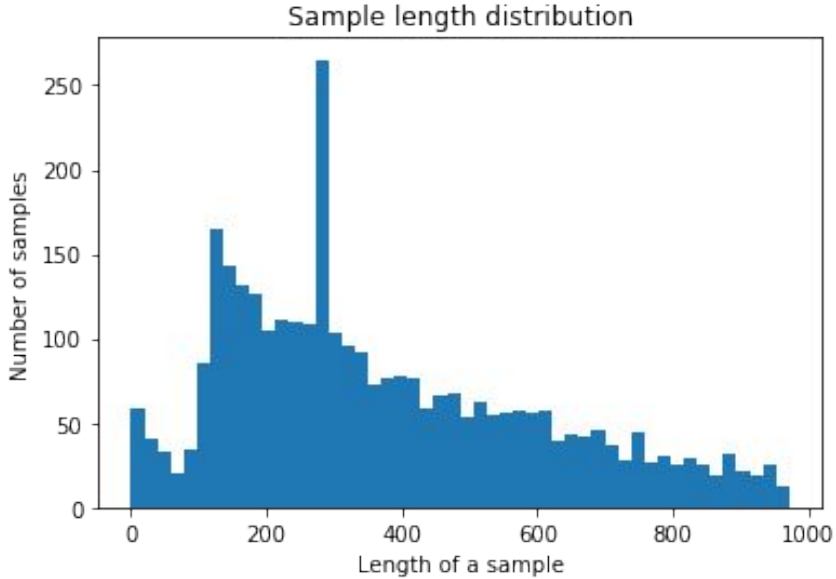
Exploratory Data Analysis

- After concatenating the two dataframes, I created a plot to visualize the proportions of the labels in the whole dataset.



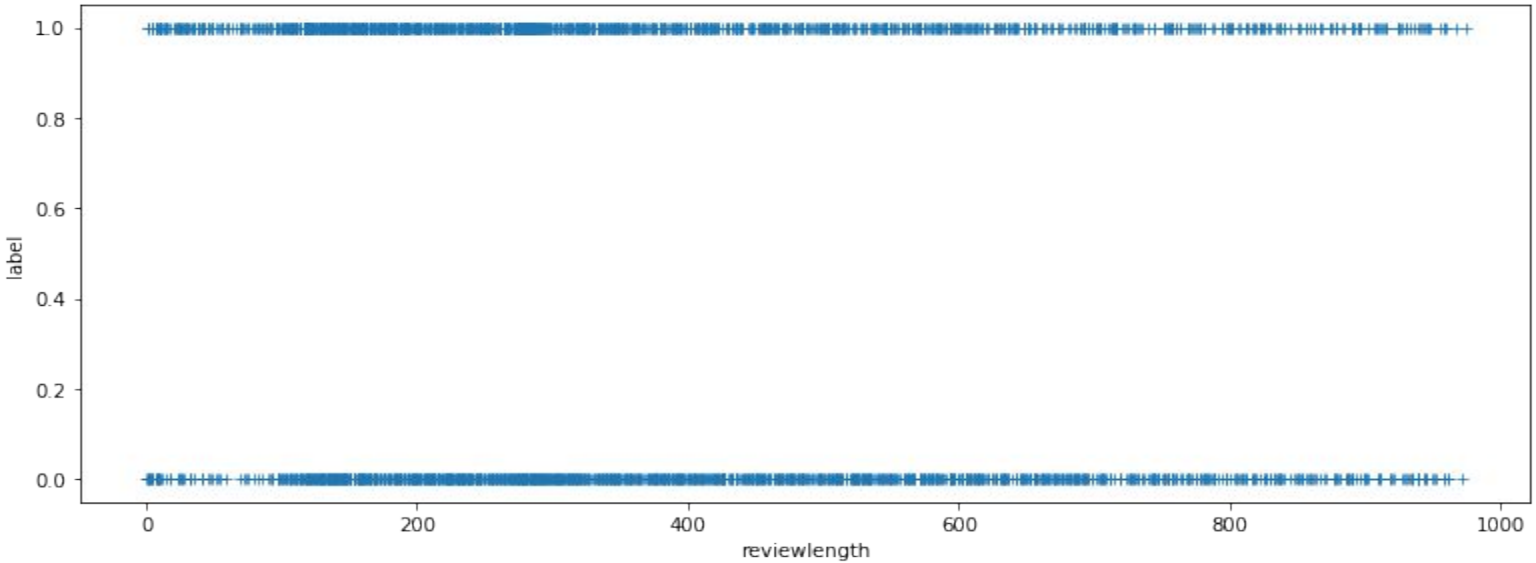
Preprocessing Text

- I ran the text through different functions to remove special characters and lemmatize the text (using spaCy) in preparation for the Bernoulli Naive Bayes model.
- I then created a plot to visualize the distribution of the review length (ave. 374 words).



Exploratory Data Analysis

- The longest text was close to 1000 words, so I decided to investigate whether or not longer texts were more likely to be rated positive or negative. The plot below shows no clear relationship between length and sentiment.



Exploratory Data Analysis

Positively rated text word cloud: positive words are prominent

Exploratory Data Analysis

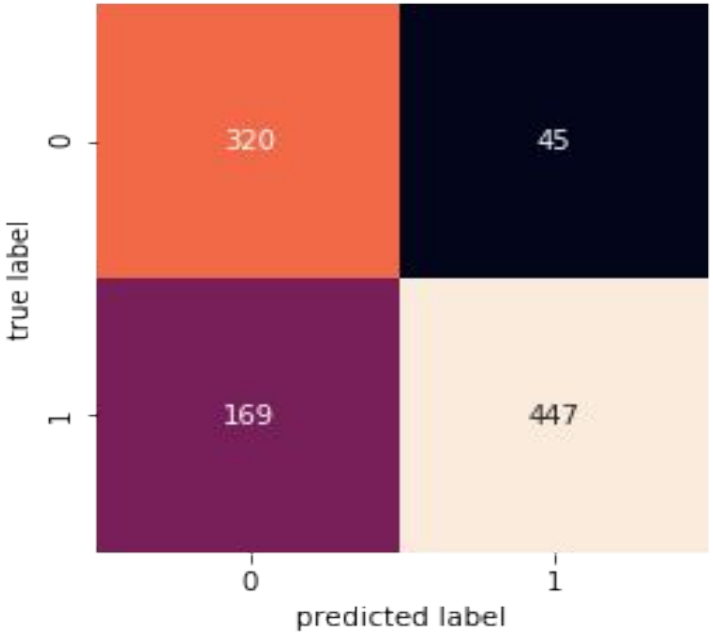
Negatively rated text word cloud: a mixed bag of prominent words

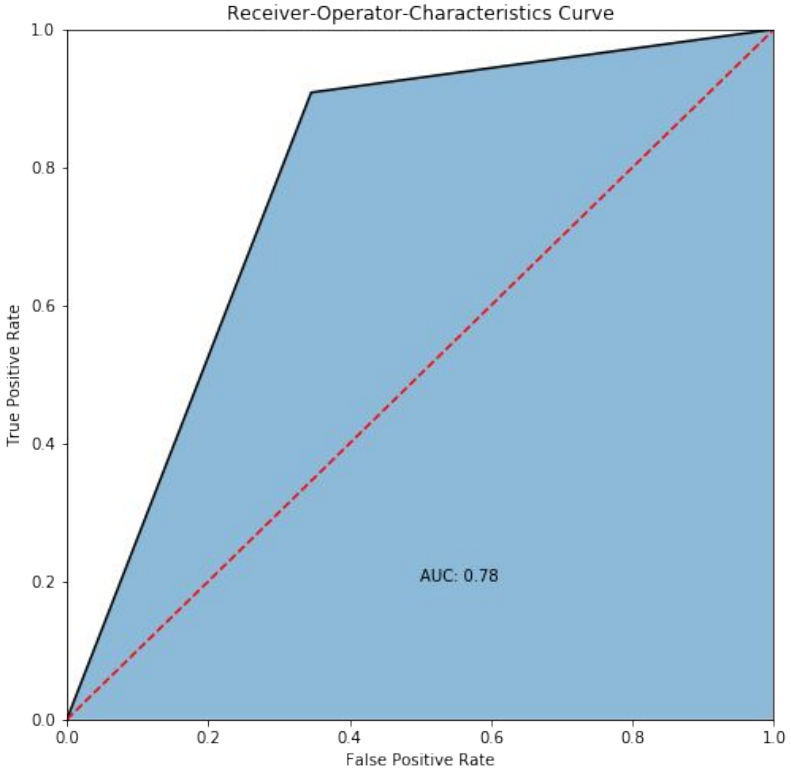
First Classification Model: Bernoulli Naive Bayes

- After vectorizing the preprocessed text (using CountVectorizer) and removing stop words, I divided the text into training and test sets along a 70% - 30% split.
- I then fit the classifier on the training data and thereafter used the test data to get a new set of predicted labels.
- This yielded 78% accuracy and 78% area under the curve (AUC). The classifier had more false negatives (Type II Error) compared to false positives (Type I Error).

Bernoulli Naive Bayes Performance

Confusion Matrix



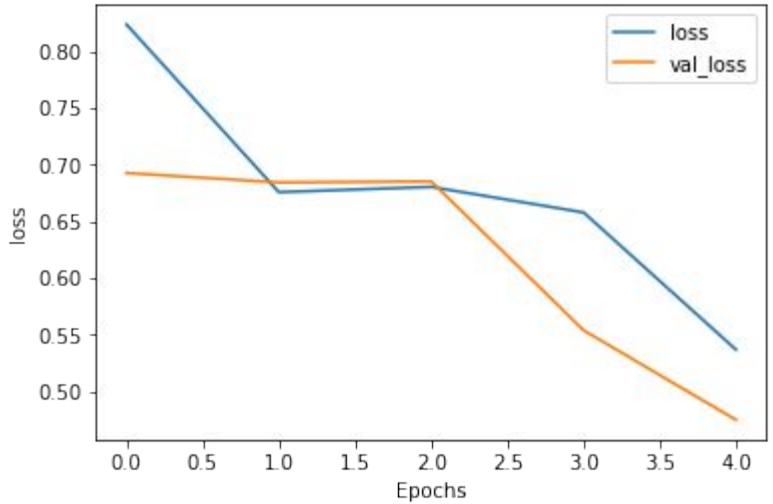


Second Classification Model: LSTM

- Tokenizer from the TensorFlow library in combination with spaCy for lemmatization to transform the words into vectors.
- In addition to removing special characters from the text, I transformed the column containing the labels into an array.
- After optimizing the model, I found the ideal train-test split to be an even 50%.
- Neural network layers (ran over 5 epochs):
    - Embedding layer
    - Two bidirectional LSTM layers
    - One dense layer for regularization
    - A 20% dropout layer
    - Final layer containing a sigmoid activation function (output 0/1)

LSTM Performance

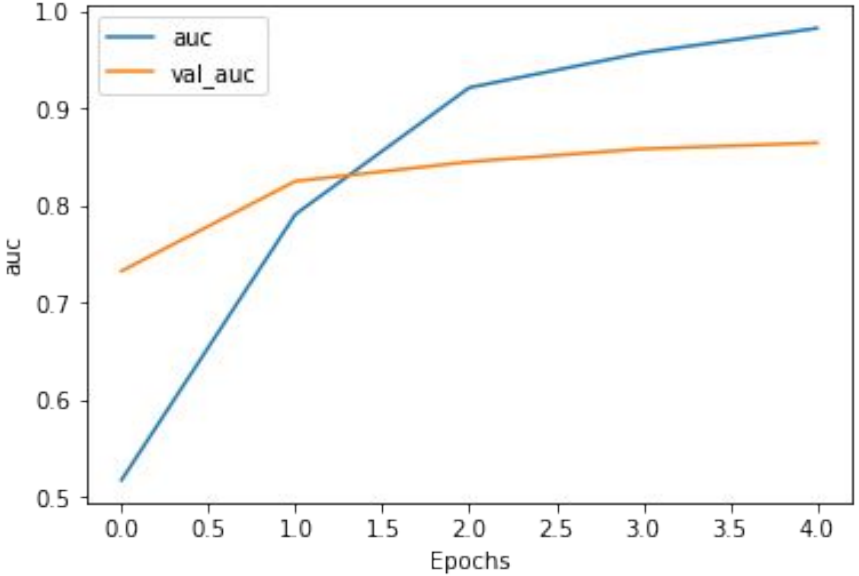
- Highest out of the three, most likely due to adding a regularization layer in order to visualize relationships in a sparse matrix.
- Because our dataset shrank after preprocessing to remove special characters, it is worth revisiting this model with a data set augmented with more English language text.



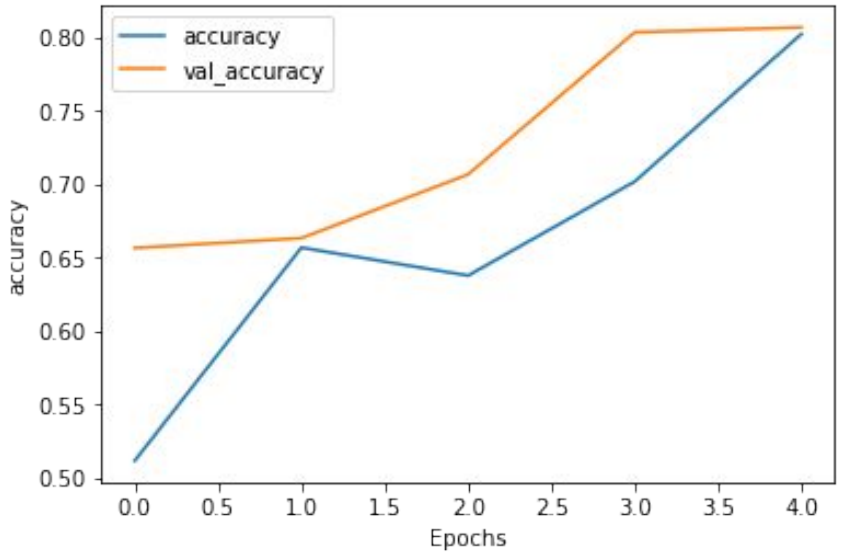
Loss visualized for training set and validation set (0.1%)

LSTM Performance

AUC curve on the training vs validation set (0.1%)



Accuracy curve on training vs. validation set (0.1%)



Third Classification Model: Simple RNN

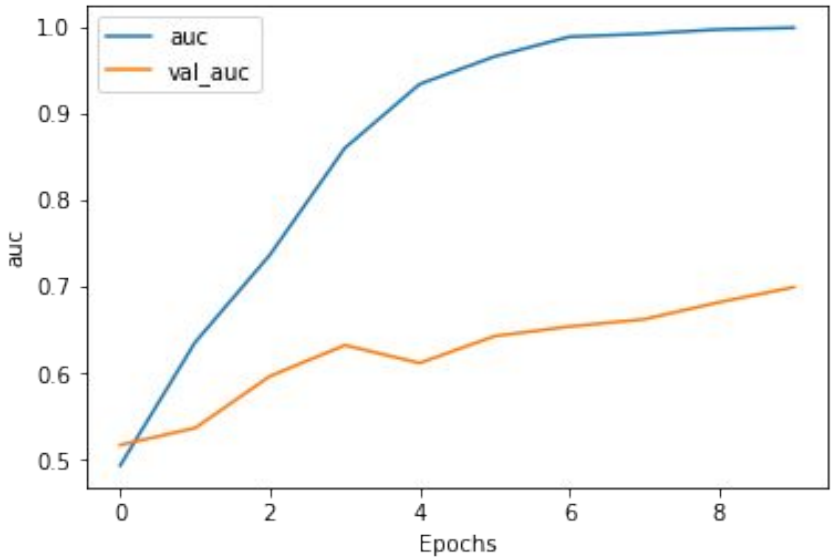
- The preprocessing steps for the Simple RNN were similar to those performed in preparation for running the LSTM model.
- Simple RNN layers (ran over 10 epochs):
  - Embedding layer
  - One bidirectional RNN layer
  - 10% dropout layer
  - Dense layer containing a sigmoid activation function (output 0/1)

Simple RNN Performance

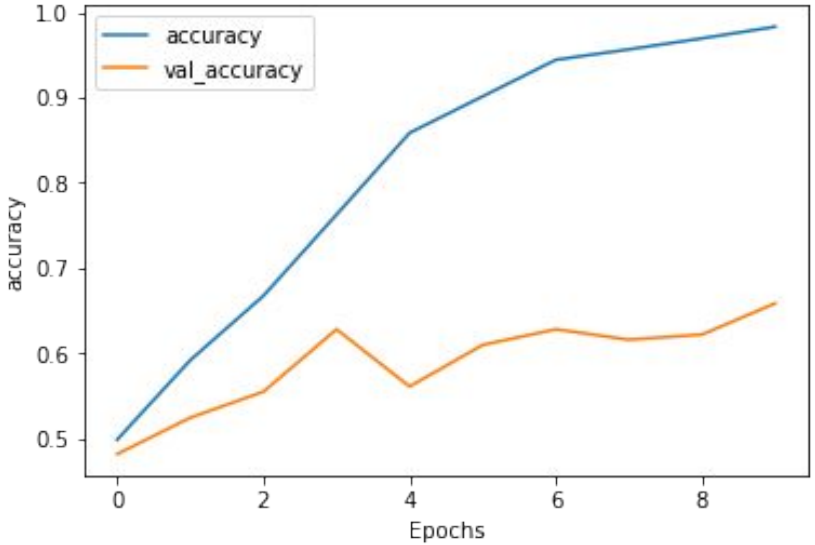
- This model performed the poorest out of all the ones tried on this dataset.
- This could have been due to the small corpus of text used for training.
    - After preprocessing to remove special characters, the reviews decreased from about 3,000 to almost 1,000 entries.
- The lack of a regularization layer might also have contributed to the model's lackluster ability to learn on a sparse matrix.

Simple RNN Performance

AUC Curve on the training vs validation set (0.1%)

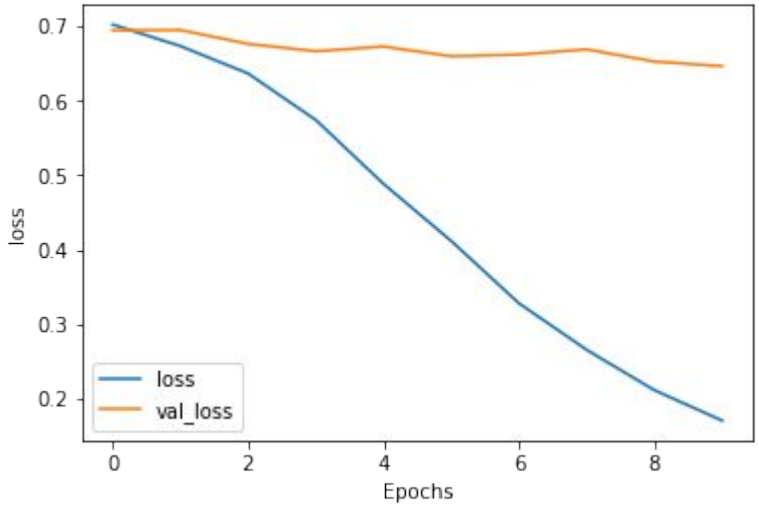


Accuracy curve on the training vs validation set (0.1%)



Simple RNN Performance

Loss visualized for training and validation set



Conclusions

- The size of the training corpus was a hindrance, especially for the simple RNN model.
- The Bernoulli Naive Bayes and LSTM models performed better but there was a high false negative rate, particularly in the case of the Naive Bayes classifier (puzzling given the prominence of positive words in the word cloud).
- This inability to correctly label positive reviews would hamper the effectiveness of the recommendation system.
- More work is needed to train the models with more English language text but a preliminary conclusion is that classification models do just as well parsing sentiment from long form text compared to short form text.