Question 3

First, the negative and positive training and testing sets were consolidated into their own arrays. The raw text of all these reviews had special characters, HTML tags and other punctuation elements like commas and exclamation marks. These elements are not necessary to do sentiment analysis on the reviews. They do not contribute any meaning to the analysis. As a result, they were removed using regular expression. The reviews were then all converted to lower case letters for consistency. This part of the process took the longest time. It can take up to 15 hours to finish cleaning the dataset when run locally.

The corpus was then normalized. This means that multiple forms of a word were condensed into its root word. For example, connected, connection and connecting are all forms of the root word connect. The forms do not provide further context into the meaning. Lemmatization and stemming were used to condense multiple forms of a word into its root. Both approaches were used to compare their performances when fitting the model. However, lemmatization is the more accurate method as it takes into account the context of the word before converting it to its root form [1]. Stemming essentially strips the last few letters from the word which can potentially lead to erroneous results, since it does not take the context into account [1]. Although lemmatization by definition is the more accurate approach, we found that stemming achieved better performance during training and testing. Python provides built-in functions to perform normalization of the corpus. One of the parameters of this function is the stop word parameter which was used. This essentially removes any stop words from the corpus. For example, ‘in’, ‘at’, ‘of’, ‘the’ and ‘a’ are all stop words. These words do not add any context to the meaning and so they were removed.

Finally, a multiple gram architecture was used to vectorize the corpus. A unigram model would consider each word in the sequence individually [2]. This can potentially alter the true meaning of a review. Consider the sequence, “movie was not loved by us.” If each word in that sequence is analyzed individually, the model will think it’s positive since love correlates to a positive sentiment. This can be alleviated with using multiple gram architecture like a bigram or trigram model. In this case, the words would be consolidated together in groups of two or three [2]. This can enhance the true sentiment of a review.

Four algorithms were chosen as the model to train and test the reviews. These include logistic regression, SVM, decision tree and multinomial naïve bayes. Logistic regression and SVM were chosen since they output a binary result (1 or 0). In this case, we are trying to predict the sentiment of a given review which can either be positive or negative (binary). Multinomial naïve bayes was used since it works well with discrete features such as word counts [3]. It usually uses integer feature counts which is done by vectorizing the corpus as described above [3]. The only hyperparameter that was changed in this analysis was the *c* value for SVM and logistic regression. The *c* value dictates the size of the hyperplane to divide the two classes. Larger values usually mean that the margin is smaller between the data points and the hyperplane while smaller values indicate larger margins. The *c* value ranged from 0.01 to 10.

The model was trained using the four algorithms described above. Logistic regression seemed to provide the best accuracy while training. It provided an accuracy of about 87.6% as opposed to the next closest accuracy which was 86.7% with multinomial naïve bayes. The model was then saved and used on the testing data which provided an accuracy of about 86%. These results were achieved with using lemmatization as one of the pre-processing techniques. Using stemming provided slightly better results. During training, logistic regression provided the highest accuracy of around 88.1%. This is also the case when predicting the test cases. Logistic regression performed the best with an accuracy of around 88.2%.

References

[1] What is the difference between lemmatization vs stemming? (1958, June 01). Retrieved July 15, 2021, from https://stackoverflow.com/questions/1787110/what-is-the-difference-between-lemmatization-vs-stemming

[2] Mohd Sanad Zaki RizviA computer science graduate. (2020, December 23). Language Model In NLP: Build Language Model in Python. Retrieved July 15, 2021, from https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-language-model-nlp-python-code/

[3] Nazrul, S. S. (2018, June 25). Multinomial Naive Bayes Classifier for Text Analysis (Python). Retrieved July 15, 2021, from https://towardsdatascience.com/multinomial-naive-bayes-classifier-for-text-analysis-python-8dd6825ece67