**Amazon Review Sentiment Analysis Report**

Introduction

Our program aims to teach us the fundamentals of NLP libraries, how we can break down a sentence into its individual words and gather the core meaning behind each word then piece them back together to get the meaning of those words together.

In this program, we will look at how can predict/ analyze the sentiment behind a sentence by using the previously mentioned words and their meanings to give us the context behind the text and overall sentiment.

Data

Naturally, we will need a dataset to test our NLP sentiment function on. We will be using an Excel CSV file of Amazon reviews which represents information regarding a purchase made on the Amazon shopping platform. These reviews are written by users of Amazon with a verified purchase.

A simple summary of the data is that it contains information about the item (what it is), the ratings/ reviews left by the user and product code/ links.

Data Preprocessing

First we must reduce the size of our dataset as it is currently 5000R by 25C. To help us with this, we must identify out the vital information we need for the sentiment analysis. In this case, things like the time, user information or extra product information such as links are redundant for this program. Therefore, we only kept 4 key columns, the name of the item, user recommendation, user rating and user review. We could do with just the review but will use the other columns to help us compare our results later on.

To help us with run time, we streamline the data by removing any missing values and limiting our data frame to the first 100 rows.

Then spacy has an excellent feature that allows us to identify stop words which are words like: is, are, the etc. Ultimately this does not alter the meaning of the review. So we remove all the stop words from our reviews.text column which contains our users review.

Evaluation of Results

Out of the first 100 reviews we analyzed, there were **85 Positive**, **8 Neutral** and **7 Negative** reviews. This information on its own does not tell us much we do not know how well our program did at predicting the sentiment of the review.

Whilst we could individually check if the 85 positive reviews were positive, that would take forever. So, remember the rating columns we left in our data frame. We used that to convert them into whether the user ratings were positive neutral or negative as a rational user would leave a good rating for a good review and vice versa. We marked ratings of 1/ 2 as negative, 3 as neutral and 4/5 as positive then compared them to our results.

We managed to get 79 matching results meaning our program was 79% accurate at predicting the sentiment of the 100 reviews.

Strengths and limitations of our model

I think our model did a good but not perfect job of predicting the sentiment of the review with an accuracy close to 80%. Furthermore, the length of our function and therefore code was not too long as such the run time was good for such a large dataset.

However, there are a few things I would improve. As the output sentiment score of our reviews is a float, categorizing the reviews based on 3 labels, positive neutral and negative affected our accuracy as there would be instances where a review would ever so slightly be positive when in reality it was more neutral. This could be fixed by adding more labels and small intervals so we can have very positive or slightly negative so in total have 5 labels. This would allow us to reduce the margin for error and create more accurate results in predicting the sentiment.

Overall, we were able to tell whether a review is positive neutral or negative so I am quite happy with the results.