Sentiment Analysis Report

Introduction

For this Capstone Project, the objective is to analyse a pair of Amazon Product Reviews and classify them as Positive, Negative or Neutral, and determine the similarity between both using a sentiment analysis based on a spaCy BlobText model.

The dataset used was a csv file containing information about 38,000 Amazon products, including name of product, ID, review score, review text… obtained from Kaggle database. From this csv file, the code would extract and clean the data from 'reviews.text' column, select a sample of 2 reviews to preprocess (tokenize) before running them through a sentiment analysis.

Finally, the code will print out the selected reviews with a Sentiment classification, and a Similarity score as a percentage.

Code Functionality

For the preprocessing process to clean the data, first we extracted the information to analyse from the 'reviews.text' column only. This was done to save processing and memory, and to avoid data type issues, as several columns were float types while others were string types. Once we have the column, we apply .dropna() to the column to remove any empty entries/cells. After this, we apply the custom function "cleaning_text(text)", in which we do the following checks:

- Verify that the row entry is a string and is not empty.
- Convert all characters to lowercase.
- Remove any punctuation and special characters.
- Remove any stop words.

Next step is to prepare the data for sentiment analysis. For this preprocessing step, we will tokenize each review using custom function "tokenize_text(text)", which filter out any stop words and punctuations.

Once the data has been cleaned and prepared for processing, we run it through a 'sentiment analysis' using the spaCy BlobText model. In this custom function "analyse_sentiment(tokenized_reviews)", first we determine the polarity of each tokenized review. If the polarity score is greater than 0, we assign 'Positive' sentiment; for smaller than 0, the sentiment is marked as 'Negative', therefor it will be assigned 'Neutral' if score is 0.

Finally, we run both tokenized reviews through a similarity check using the custom function "analyse_similarity(tokenized_review1, tokenized_review2)". This function will join all the tokens in each review into a string for calculations, which this is accomplished using .similarity().

Results and Observations

After running the program several times, the following message appears every time:
"*UserWarning: [W007] The model you're using has no word vectors loaded, so the result of the Doc.similarity method will be based on the tagger, parser and NER, which may not give useful similarity judgements. This may happen if you're using one of the small models, e.g. `en_core_web_sm`, which don't ship with word vectors and only use context-sensitive tensors. You can always add your own word vectors, or use one of the larger models instead if available.*"

Some reviews were marked as 'Positive', even though under further inspection, a 'Neutral' classification would be more appropriate, like this example:

"Wanted to travel without my laptop. Found the Fire to be perfect. Used it mainly for checking email and keeping up with Facebook posts of friends. Found the size of screen to be easy to read. Typing a tad uncomfortable until one has some experience using it. Have yet to use as an ereader."

This review shows some positive and negative aspects of the product, but is not sufficient for 'Positive' classification; specially when compared to this review:

"I love the Amazon Paper White Kindle. I can take it anywhere and it's my entire library at my finger tips. I can read at night, in the glaring sun and even in the constant changes of the outdoor light. This is the perfect reader for on the go."

Similarity score was 73.21%, highlighting why the first one came out 'Positive', even if the overall messaging is not there.

Overall, the program works as intended, but clearly shows the limitations of the model, as the assessment are very much surface level, but it is not doing a deeper interpretation of the context and all elements of the reviews.