HW4

IST664

Wenzhe Yang

**Sentiment Analysis**

Wenzhe Yang

College of Arts and Sciences, Syracuse University, New York USA

**1. Introduction**

1.1 Sentiment Analysis of Amazon Product Reviews

It is increasingly common that Internet users engage in various of online reviews. The availability of these review content offers researchers opportunities to better understand and model online social behavior. In this report, I conduct sentiment analysis to gain some understanding about the Amazon product reviews.

1.2 The Dataset

I analyze the review contents from Amazon Product Data provided by Julian McAuley at http://jmcauley.ucsd.edu/data/amazon/. This dataset contains product reviews and metadata from Amazon, including 142.8 million reviews spanning May 1996 - July 2014. It includes reviews (ratings, text, helpfulness votes), product metadata (descriptions, category information, price, brand, and image features), and links (also viewed/also bought graphs).

In this task I use 5-core subset of Baby/Clothing and Jewelry / Health and Personal Care.

5-core subsets mean that all users and items in the dataset have at least 5 reviews.

**2. Data Pre-processing**



Figure 1: processed data

In this CSV file, it has only one column “review” and each line is one review. Actually, each review contains multiple sentences, however I prefer to regard each review as a whole and I do not split the review into sentences and filter the symbols because in this task the words are the thing that matters the sentiment.

First I use the glob function and read all of the .txt file, and remain those lines which are start with “reviewText” by applying the regular expression '(?<=\\breviewText:\\b).\*. And then I filtered strange symbols and commas but remains n’t for the negation detection. each review text by delimiter “\t” and rewrite into a new csv file.

**3. Feature Extraction**

First, I tokenize each review with nltk tokenizer.

3.1 Top 2000 Frequent Words Feature

I edit a stop-words list which do not contain negation words and filter the stop words and nonalphabetic tokens. Then, I choose the top 2000 frequent words appear in the nltk corpus, sentence\_polarity as the features.

But I leave the original one that without this processing step and use the accuracy which uses original Top 2000 feature words as my base line in this task to see whether my edition improves the performance.

3.2 Negation Feature

I remove all the stop words apart from negation words such as nothing, never and those end with “n’t” to form the new stop words. And then choose the top 2000 frequent words from the remaining words.

3.3 Subjective Lexicon Feature

Apart from the top 2000 frequent words, I add positivecount which calculate the numbers of weak and positive plus two times of strong and positive words. Also, we add negativecount which calculate the numbers of weak and negative plus two times of strong and negative words.

3.4 Tfidf Feature:

Use TfidfVectorizer form Sklearn by employing NLTK toknizer and my own stop-word list. The tf–idf value increases proportionally to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain the word, which helps to adjust for the fact that some words appear more frequently in general. So I choose TfidfVectorizer to do the task because it performs well in classification tasks as it can extract features depend on how important important a word is to a document in a collection or corpus.

**4. Model and Results**

I choose Multinomial Naïve Bayes with sklearn and the Naïve Bayes with NLTK to compare their result between different feature extraction engineering.

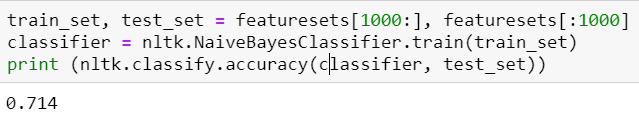
For the models that generated from Multinomial Naïve Bayes with sklearn, the evaluation each model with the average accuracy, 3 folds cross validation and a classification report which contains the precision, recall and f1 score for each category.

For the models that generated from Naïve Bayes with NLTK, I evaluate the results with the accuracy.

4.1 Top 2000 Frequent Words Feature

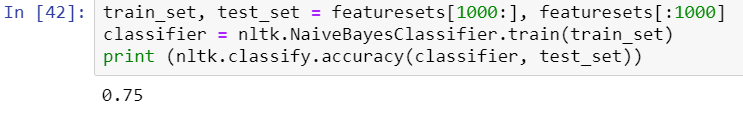
NLTK Naïve Bayes:

The original one that without filter stop words, the accuracy is 0.71.



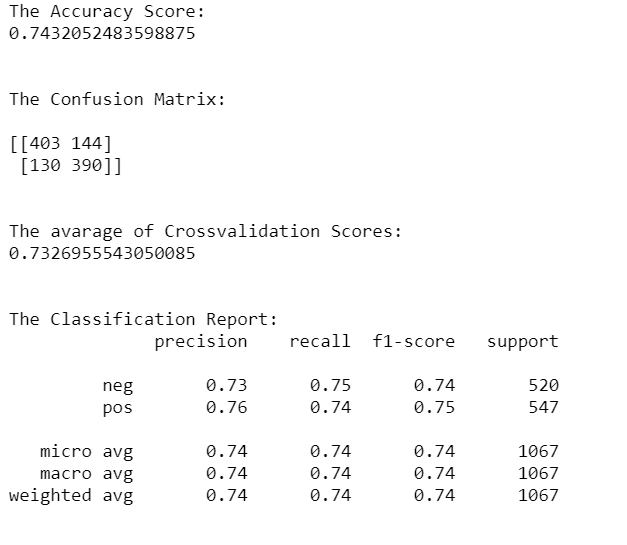
Modified Top 2000:

The features that generated after modifying tokens have better performance improves.



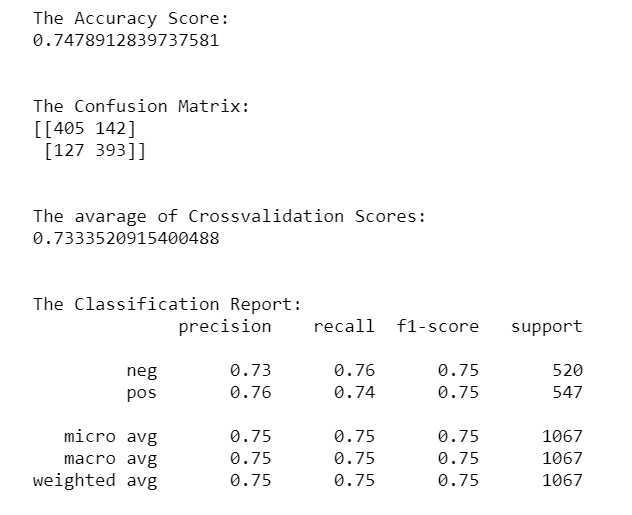
MultinomialNB with Sklearn:

Modified Top 2000:



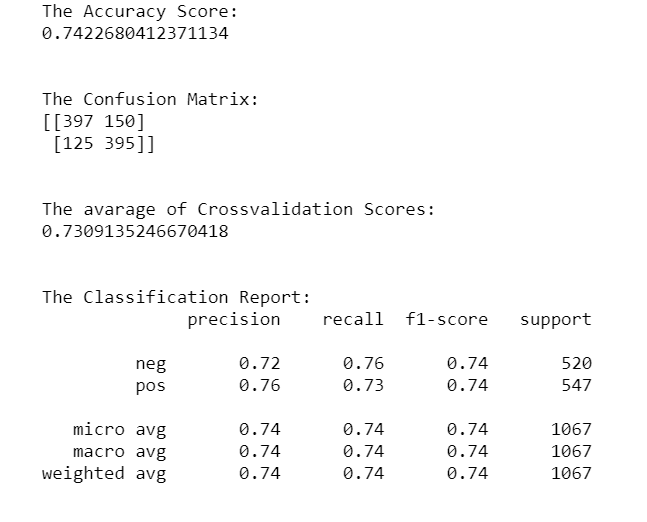
4.2 Negation Feature

MultinomialNB:



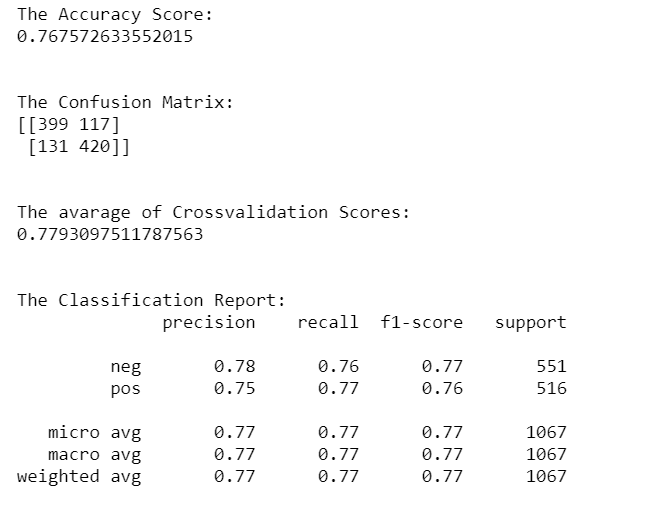
4.3 Subjective Lexicon Feature

MultinomialNB:



4.4 TfidfVectorizer Feature

MultinomialNB:

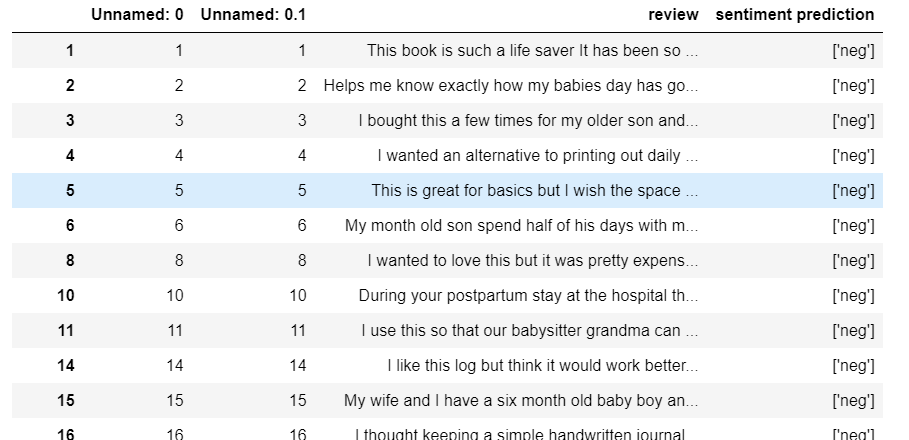


**5. Result Analysis**

The model of Multinomial Naïve Bayes classification with TfidfVectorizer performs better than all feature extraction transformation. And the TfidfVectorizer gets the best features among the three feature extractions. Finally, I choose the model of Multinomial Naïve Bayes with TfidfVectorizer to make the prediction of the review text and write the reviews and prediction results into the pred\_rev.CSV file.

**6. Screenshot of Prediction**

Negative sentence:



Positive sentence:

