The first step in organizing the data was simply to read the files into separate reviews. The files were split by next line characters, so I could split them that way. I then iterated through each set of reviews to collect individual words. I split them using the space character, then filtered out words that likely do not pertain to the sentiment of a review, like articles and conjunctions. After that, I simply collected all the words into a single list. I then used the C0unter method to create a dictionary where the keys are individual words, and the values are the number of times they appear. This helped me filter out words that only appear once in the whole data set. I also kept track of how many words there were total in the entire set. Thus, I could simply divide an individual word’s count by the total number of words to get the probability of that word appearing in either the Good Reviews dataset, or the Bad Reviews dataset.

The Bayes Classification formula in the context of this project would involve the following:

**P(Word | review classification) \* P(review classification)**

The probability that a word exists given a review classification was described earlier, where we can divide an individual word’s count in a dataset by the total number of words in that dataset. For example, if there are 30,000 words in the Good Reviews dataset, and the word ‘good’ appears 100 times in that dataset, then the probability of that word given the dataset is 100 / 30,000 = 0.33%. The same word may only appear 10 times in the Bad Reviews dataset, giving it a much lower probability. I ignored multiplying the probability of the review classification as either good or bad, because it is 0.5 in all cases given this dataset. For a particular review, I obtained two probabilities: the product of probabilities of each word in the Good Reviews dataset (e.g. P(“great” | Good Review) \* P(“film” | Good review) \* …), and the same for the Bad Reviews dataset. Whichever probability was higher was the one that I predicted, either as a good or bad review.

Using this method, I obtained these results:

DEVELOPMENT SET

Accuracy on GOOD REVIEWS: 75.625%

Accuracy on BAD REVIEWS: 69.125%

Accuracy on ALL REVIEWS: 72.375%

TEST SET

Accuracy on GOOD REVIEWS: 73.375%

Accuracy on BAD REVIEWS: 70.875%

Accuracy on ALL REVIEWS: 72.125%

An average of about 72% accuracy is high because the reviews are written by humans, where sentiment is very hard to determine computationally. Instead of multiplying the probabilities of individual words within each review, I attempted to select the highest probability word, and ignore other probabilities. The accuracy of that method was significantly lower. Even though I used a very stripped-down version of the Naïve Bayes Classification formula, I found the accuracy to be appreciable.