## Naive Bayes Classifier:

V1: Accuracy on training set: 0.90276
V1: Accuracy on testing set: 0.83724

My first modification to the Naive Bayes classifier was to remove punctuation and make all characters lowercase. This was done because I assumed that the punctuation was mostly unhelpful for determining the positive or negative valency of a review. Other than rare cases such as when exclamation marks are used to emphasize positive or negative sentiment, punctuation and capitalization are mostly a result of following syntax rules that do not contribute much to the semantics.
V2 Accuracy on training set: .92292
V2: Accuracy on testing set: .83788

My next modification was to change my function for computing the relative frequency of a given word across all the negative reviews and across all the positive reviews. I had noticed that the model seemed to be biased in favor of predicting negative reviews, despite the fact that there were an even amount of positive and negative reviews. I decided to change the denominator of the relative frequency algorithm. Initially, I had thought that the algorithm specified that when computing a negative weight for a word, we take the number of occurrences for a given word across all negative reviews and divide it by the number of occurrences of all words across all negative reviews. I discovered that when I divided by the number of occurrences of all words across all reviews, instead of the specific type of review that this weight was for. This mostly eliminated the bias of negative reviews. You can see this change in line 70 and 72. Either I was originally mistaken about the algorithm, or for some reason that I do not understand, this modification happens to work better for this specific model.

V3 Accuracy on training set: .92312
V3: Accuracy on testing set: .84228

## Perceptron Classifier:

V1: Accuracy on training set: 0.71312
V1: Accuracy on testing set: 0.7

My first modification does not noticeably improve the accuracy of the model, but allows it to run faster. It consisted of optimizing many of the functions involved in both the training and testing phase. I had originally reused code from my Naive Bayes classifier, which included the scoring function that first computed a BOW vector and then multiplied it by the weights vector and summed all values of the result. I realized that a more efficient way find the score of a given review, label, and weights was to take each word in the review, find the corresponding weights and add it to a total score. This was much faster and did not change the functionality of the program, which allowed me to train it with more epochs, thus improving accuracy.
V2 Accuracy on training set:  .93024
V2: Accuracy on testing set: .83284

My next modification was to remove the stop words from all training and testing reviews. The justification behind this was that stop words are less likely to contribute to the semantics of a review, and removing them also helps other words count as a larger proportion of the total words across all reviews, as well as increasing the speed of the program.
V3 Accuracy on training set:  .98264
V3: Accuracy on testing set: .84004

I decided on 10 epochs of training on the training data. This number was chosen because additional training had diminished returns, and the model seemed to plateau on around .84 accuracy on the testing set. I also assumed that it would be impossible to get 100% accuracy by continuously training on the same set of data, since the testing reviews will always include new words that aren't in the vocabulary that is built with the training reviews.