Gabriel Britain

Execution

To run the implemented Naive Bayes model, simply `cd` into the `python` directory, and run the following (assuming your default Python version is 2.7)

```
python NaiveBayes.py [-f] [-b] ../data/imdb1/
```

There are two optional parameters that can be provided to the application.

- `-f` - Filters out stopwords from the corpus
- `-b` - Caps word counts in each document at 1 (indicating the word is present)

Tasks

1. Complete
2. The impact that the removal of stop words has on the performance of this model is negligible. In fact, the removal of stop words decreased the average performance of this model slightly.
3. Complete
4. For sentiment analysis, there are a couple of different features that I can think of that might improve performance:
    - Independent of Bag-Of-Words features
        1. Stemming words could possibly improve performance by reducing the number of unique words to be considered. After all, `bad` and `badly` both are words with negative connotations, and don't need to be considered as independent features. In the same vein, lemmatizing could also prove to be useful.
        2. Including meta information about the review could also be useful, such as the length of the review, time it was posted, the average length of the words in the review, number of emojis used, etc. There's semantic meaning that's included in the use of the ☹ emoji that a Naive Bayes classifier wouldn't be able to pick up on.
    - Dependent on Bag-of-Words
        1. There were no *n*-grams used in this model. Using interpolation could provide a significant boost to the model's performance
        2. We could use methods more sophisticated than counting, such as TF-IDF, which would decrease the weight of commonly-used words like `the`, `and`, `a`, and increase the weights of seldom-used words which could provide a better picture to the overall sentiment of the review.

Results

Multinomial Naive Bayes Classification (Un-filtered)

```
[INFO]  Fold 0 Accuracy: 0.765000
[INFO]  Fold 1 Accuracy: 0.850000
[INFO]  Fold 2 Accuracy: 0.835000
[INFO]  Fold 3 Accuracy: 0.825000
[INFO]  Fold 4 Accuracy: 0.815000
[INFO]  Fold 5 Accuracy: 0.820000
[INFO]  Fold 6 Accuracy: 0.835000
[INFO]  Fold 7 Accuracy: 0.825000
[INFO]  Fold 8 Accuracy: 0.755000
[INFO]  Fold 9 Accuracy: 0.840000
[INFO]  Accuracy: 0.816500
```

Multinomial Naive Bayes Classification (Filtered)

```
[INFO]  Fold 0 Accuracy: 0.765000
[INFO]  Fold 1 Accuracy: 0.825000
[INFO]  Fold 2 Accuracy: 0.815000
[INFO]  Fold 3 Accuracy: 0.830000
[INFO]  Fold 4 Accuracy: 0.795000
[INFO]  Fold 5 Accuracy: 0.830000
[INFO]  Fold 6 Accuracy: 0.835000
[INFO]  Fold 7 Accuracy: 0.835000
[INFO]  Fold 8 Accuracy: 0.760000
[INFO]  Fold 9 Accuracy: 0.820000
[INFO]  Accuracy: 0.811000
```

Multinomial Naive Bayes Classification (Binarized)

```
[INFO]  Fold 0 Accuracy: 0.805000
[INFO]  Fold 1 Accuracy: 0.840000
[INFO]  Fold 2 Accuracy: 0.835000
[INFO]  Fold 3 Accuracy: 0.825000
[INFO]  Fold 4 Accuracy: 0.835000
[INFO]  Fold 5 Accuracy: 0.825000
[INFO]  Fold 6 Accuracy: 0.845000
[INFO]  Fold 7 Accuracy: 0.835000
[INFO]  Fold 8 Accuracy: 0.790000
[INFO]  Fold 9 Accuracy: 0.855000
[INFO]  Accuracy: 0.829000
```