04/27/2018
Giorgi Kharshiladze & Abhinav Pandey
Lab E Option 1 (Sentiment Analysis)
Report

Basic Functions implemented

addExample(): This function adds a new training example. We use a defaultdict data structure to store the words into two different dictionaries with their counts (positive and negative). We were initially running into index-value errors while using a normal dictionary, so we researched alternatives online and found that defaultdict has the advantage of not raising such errors. Based on the 'klass' of the document, we categorize words as either positive (pos) or negative (neg).

classify(): This function classifies test examples from the input test set as either positive (pos) or negative (neg). We found that the set() function offers a simple way to filter out words, so we used it to find all of the unique words in pos_dict and neg_dict to construct the vocabulary V. We then closely implemented the pseudocode given in Manning et. al. First, we determine the length of the overall vocabulary list (V). Then, we get the number of positive and negative counts stored in the dictionary (which is later used in the Naïve Baye's formula). We've then implemented the Naïve Bayes formula using logarithms (this simplifies the multiplications given in the pseudocode by turning them into log additions). The probability of the example being positive or negative is tracked and updated. Whichever probability is the highest is then the classification that is returned.

filterStopWords(): This function filters the input word list for stop words. This is simply done by checking if the current index word is in stopList, and removing it if it is. The filtered list is then returned.

Two Additional Improvements Explored

Boolean representation: To implement this feature, we simply found the unique words in the inputted words list for addExample() using a helper function. This feature is enabled only if the program is called with the –B argument.

Negation Features: To implement this, we implemented the negation() function which traverses the input words list and adds NOT_ to the word if it comes after 'never', 'not' or a word with 'n't'. The function also makes sure NOT_ isn't added to punctuation/symbols or to words that already have NOT_ in front of them.

Results:

```
giorgi@Giorgis-MacBook-Pro ~/Github/sentiment-analysis/python master python3 NaiveBayes.p
================================================================================
================================================================================
[INFO]  Performing 10-fold cross-validation on data set:        ../data/imdb1/
[INFO]  Fold 0 Accuracy: 0.770000
[INFO]  Fold 1 Accuracy: 0.820000
[INFO]  Fold 2 Accuracy: 0.820000
[INFO]  Fold 3 Accuracy: 0.815000
[INFO]  Fold 4 Accuracy: 0.810000
[INFO]  Fold 5 Accuracy: 0.825000
[INFO]  Fold 6 Accuracy: 0.830000
[INFO]  Fold 7 Accuracy: 0.825000
[INFO]  Fold 8 Accuracy: 0.750000
[INFO]  Fold 9 Accuracy: 0.835000
[INFO]  Accuracy: 0.810000
```

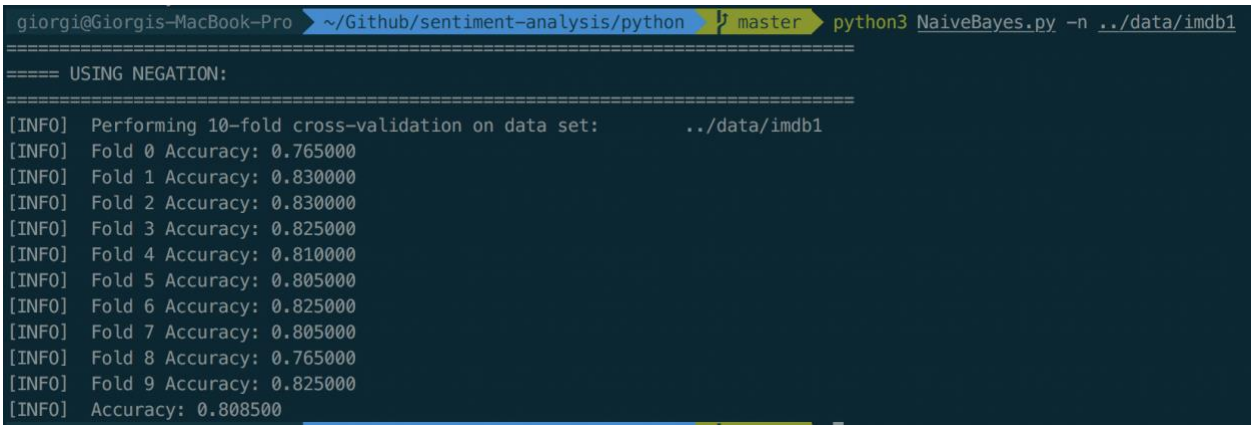
Normal Run

```
giorgi@Giorgis-MacBook-Pro ~/Github/sentiment-analysis/python master python3 NaiveBayes.py -f ../data/imdb1
================================================================================
===== USING FILTER_STOP_WORDS:
================================================================================
[INFO]  Performing 10-fold cross-validation on data set:        ../data/imdb1
[INFO]  Fold 0 Accuracy: 0.775000
[INFO]  Fold 1 Accuracy: 0.805000
[INFO]  Fold 2 Accuracy: 0.830000
[INFO]  Fold 3 Accuracy: 0.835000
[INFO]  Fold 4 Accuracy: 0.795000
[INFO]  Fold 5 Accuracy: 0.815000
[INFO]  Fold 6 Accuracy: 0.815000
[INFO]  Fold 7 Accuracy: 0.830000
[INFO]  Fold 8 Accuracy: 0.755000
[INFO]  Fold 9 Accuracy: 0.810000
[INFO]  Accuracy: 0.806500
```

Stopword Filtering

```
giorgi@Giorgis-MacBook-Pro ~/Github/sentiment-analysis/python master python3 NaiveBayes.py -b ../data/imdb1
================================================================================
===== USING BOOLEAN - True:
================================================================================
[INFO]  Performing 10-fold cross-validation on data set:        ../data/imdb1
[INFO]  Fold 0 Accuracy: 0.770000
[INFO]  Fold 1 Accuracy: 0.820000
[INFO]  Fold 2 Accuracy: 0.820000
[INFO]  Fold 3 Accuracy: 0.815000
[INFO]  Fold 4 Accuracy: 0.810000
[INFO]  Fold 5 Accuracy: 0.825000
[INFO]  Fold 6 Accuracy: 0.830000
[INFO]  Fold 7 Accuracy: 0.825000
[INFO]  Fold 8 Accuracy: 0.750000
[INFO]  Fold 9 Accuracy: 0.835000
[INFO]  Accuracy: 0.810000
```

Run with Boolean=True



Run with Negation

The results indicate that implementing of the new features decreased(not changed) accuracy slightly.