James D. Harris

IST 664

Final Project – Kaggle Movie Reviews

**Project Description**

For my final project for IST 664, I chose the Kaggle movie review competition data set. The task was to train and test a sentiment classification process, and to report the results. I used a combination of Python scripts to process and tokenize the text. To build an initial, baseline classifier, I ran the provided classifyKaggle.py, sentiment\_read\_LIWC\_pos\_neg\_words.py, and sentiment\_read\_subjectivity.py, all provided in the Syracuse University learning management system, along with numerous other classifiers and feature selectors. This baseline classifier used a “bag of words” feature, which collects all the words in the corpus. The script called for the 1,500 most common words in the corpus to be used. Out of the 156,060 phrases available, I used a random selection of 10,000 phrases to generate featuresets, and each featureset had an initial vocabulary size of 500. After I generated each featureset, I used cross-validation to learn precision, recall, and F1 scores, and then I used NLTK’s naïve Bayes classifier to generate an accuracy score.

I generated the following featuresets:

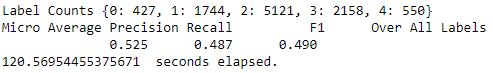
1. Bag of words / unigram (baseline)
2. Bigram measures
3. Using a sentiment lexicon (SL)
4. Using the LIWC sentiment lexicon
5. Using a combination of SL and LIWC
6. Part of speech tagging
7. Negation
8. Using Bing Liu’s Opinion Lexicon\*

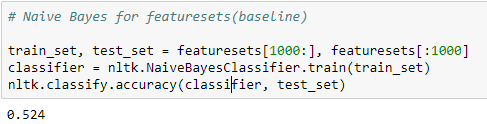
\* Bing Liu’s Opinion Lexicon, and its documentation, can be found at <https://www.cs.uic.edu/~liub/>. I generated this featureset to fulfill the requirements to perform a more advanced task, as required by the instructions to this project.

The key metric here is the F1 score, which is the harmonic average of the precision and recall of the model, with 0 being the worst and 1 being the best scores. Precision and recall scores are also provided. For each of the eight featuresets, I have provided screen captures of their cross-validation results, and of the results of the naïve Bayes classification. The screen captures show the results from the initial runs only, when the vocabulary size was set at 500; however, the table at the end shows a second run for each featureset, with the vocabulary set at 1,000, along with some additional experimentation on the baseline featureset.

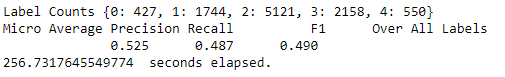
**Generating Featuresets**

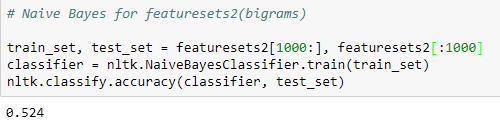
Baseline (bag of words, unigram), vocabulary size 500:



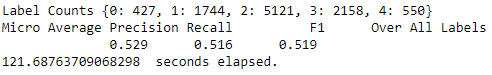


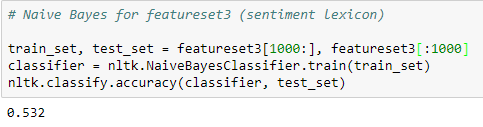
Featureset 2: Bigram measures:



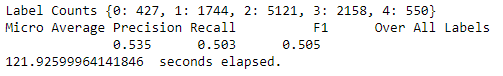


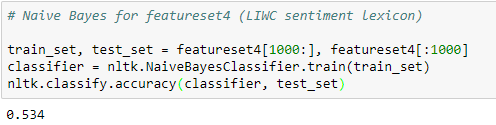
Featureset 3: Sentiment lexicon:



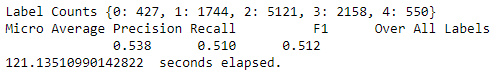


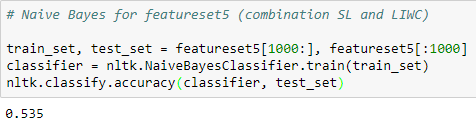
Featureset 4: LIWC sentiment lexicon:



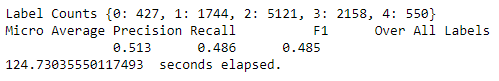


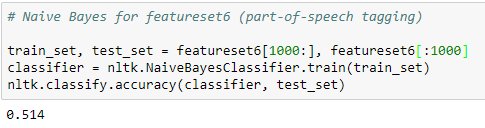
Featureset 5: Combination of SL and LIWC:



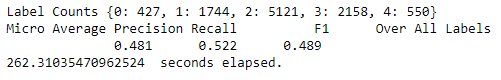


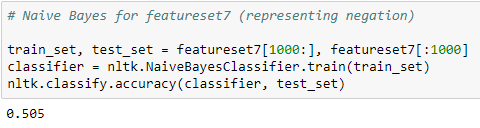
Featureset 6: Part-of-speech tagging:



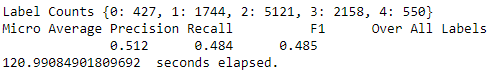


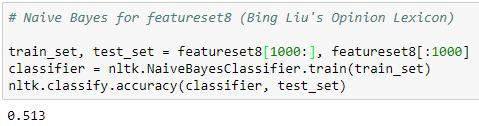
Featureset 7: Representing negation:





Featureset 8: Bing Liu’s Opinion Lexicon:





|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Featureset** | **Vocab size** | **Precision** | **Recall** | **F1** | **NB accuracy** |
| 1. Bag-of-words / unigram (baseline) | 500 | 0.525 | 0.487 | 0.490 | 0.524 |
| 1000 | 0.532 | 0.499 | 0.501 | 0.532 |
| 2. Bigrams | 500 | 0.525 | 0.487 | 0.490 | 0.524 |
| 1000 | 0.532 | 0.499 | 0.501 | 0.532 |
| 3. Using sentiment lexicon (SL) | 500 | 0.529 | 0.516 | 0.519 | 0.532 |
| 1000 | 0.536 | 0.522 | 0.525 | 0.537 |
| 4. Using LIWC sentiment lexicon | 500 | 0.535 | 0.503 | 0.505 | 0.534 |
| 1000 | 0.539 | 0.510 | 0.513 | 0.535 |
| 5. Using a combination of SL and LIWC | 500 | 0.538 | 0.510 | 0.512 | 0.535 |
| 1000 | 0.546 | 0.520 | 0.522 | 0.541 |
| 6. Part-of-speech tagging | 500 | 0.513 | 0.486 | 0.485 | 0.514 |
| 1000 | 0.522 | 0.496 | 0.495 | 0.518 |
| 7. Representing negation | 500 | 0.481 | 0.522 | 0.489 | 0.505 |
| 1000 | 0.510 | 0.526 | 0.508 | 0.492 |
| 8. Using Bing Liu’s Opinion Lexicon | 500 | 0.512 | 0.484 | 0.485 | 0.513 |
| 1000 | 0.518 | 0.492 | 0.492 | 0.512 |
|  |  |  |  |  |  |
| The following are experiments using the baseline featureset only. | | | | | |
| **Featureset** | **Vocab size** | **Precision** | **Recall** | **F1** | **NB accuracy** |
| Baseline, 10 folds | 500 | 0.528 | 0.490 | 0.493 | 0.529 |
| 1000 | 0.540 | 0.509 | 0.510 | 0.538 |
| 1500 | 0.530 | 0.500 | 0.499 | 0.552 |

Legend (“BL” = “baseline”):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| > 0.1 below BL | < 0.1 below BL | = BL | < 0.1 above BL | > 0.1 above BL |

**Analysis**

The results of running a variety of featuresets are mixed. I have made the following observations:

* In nearly every case, increasing the vocabulary size improved the scores, with an average improvement of 0.008. The only two exceptions were with representing negation and with Bing Liu’s Opinion Lexicon, which lowered the naïve Bayes accuracy score by 0.013 and 0.001, respectively.
* Using bigram measures had no discernable effect on any score.
* Using a combination of sentiment lexicon and LIWC provided the largest average improvement in scores. Using a vocabulary size of 500, this featureset improved by 0.017, the largest improvement in this project.
* Part-of-speech tagging and Bing Liu’s lexicon showed a consistent drop in performance for the baseline. The latter had the worst score when compared to the baseline.
* In the final experiment, I used the baseline featureset, but increased the number of folds in the cross-validation to 10. I tried this process with vocabulary sizes of 500, 1,000, and 1,500. In every instance, this resulted in an improvement in scores; however, the model performed better with a vocabulary of 1,000 than it did with 1,500. This suggests there may be a point of diminishing returns for increasing vocabulary size.

**Conclusions**

This classification model benefited from increasing vocabulary size and applying a combination of lexicons. It also showed a slight improvement when negation words were handled appropriately. Additionally, increasing the number of folds in the cross-validation process showed a consistent, and sometimes dramatic, improvement in the performance of the model.