Homework 8

Kyle Walter

# Lie Detection

The challenge can machine learning tell if someone is lying? The data set provided has 92 sample reviews, that evenly balance between 46 lies and 46 true reviews. We will test using the Naïve Bayes Multinomial Model and the Support Vector Machines radial model to see if we can detect the fakes.

To do this testing we utilized Weka and ran the data through these models, testing varying tuning parameters along the way.

There was a column from Sentiment Analysis that is part of another analysis below, for the test of the fakes, this column was removed so as not to influence the model.

For the tuning of the model, the focus was heavily on the tokenization, or the process by which the words or phrases of the model get transformed into numerical values for the model to analyze.

Similar to prior tests this was done using cross validation of 3, whereby the model breaks the training data into thirds and holds 1 of the back for testing and repeats till all of the hold outs are tested. And averages the returned values to give a sense of how well the model will perform when introduced to data it has not seen previously.

## Naïve Bayes Multinomial with text

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Parameter Setting | Overall Accuracy | Precision in Category 1 | Recall in Category 1 | Precision in Category 2 | Recall in Category 2 |
| Base with ngram | 61.9565% | .69 | .435 | .587 | .804 |
| +snowball stemmer | 61.9565% | .69 | .435 | .587 | .804 |
| Lovins Stemmer | 65.2174% | .733 | .478 | .613 | .826 |
| IteratedlovinsStemmer | 63.0435% | .676 | .5 | .603 | .761 |

While I tested other aspect of the model, such as the minimum number of times a word occurred and the removal of stop words. I found the min occurrence of 3 and removal of stops words were producing a consistent result around 51%.

Stop words are common words in a language that provide little meaning out of context but are need grammatically to move the sentence forward. Words like the, of, a, and so on would be examples of these.

In addition, the other factor I found to be make a great increase was using the “ngrams” version of vectorization. What this does is creates counts based on groups of words. Since these responses were written in English where the negation requires an additional word such as “not” in many cases, this was needed to capture the meaning behind some of the variation that could be used to detect fakes.

With these adjustments made, I played around the various stemming techniques as these seemed to have the most impact on getting the accuracy to higher levels than the other attributes. Steming is the process by which the morphological features of a word or removed the stem is what is remaining. An example in English would be the words like love, loving, and loved just becoming “lov”. This gives produces a smaller word set and allow the model to gain more information by certain words rather than not recognizing them as words with similar meaning.

As we can see from the results grid about the results show highest accuracy using the lovinstemming logic and with the exception of the recall on the fakes, the other factors with which we can use to test the ability of the model to preform we’re above board.

## SVM

In addition the Naïve bayes we also tested the support vector machines. For this I utilized Weka’s Filteredclassifer to create the word vectorization that was natively part of the Naïve bayes classifer in the previous part.

Like Naïve Bayes, I found using an “ngrams” classifiers with min 3 occurrences provided the best the results from a base starting point.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Parameter Setting | Overall Accuracy | Precision in Category 1 | Recall in Category 1 | Precision in Category 2 | Recall in Category 2 |
| Base | 52.1739% | .538 | .304 | .515 | .739 |
| Stopwords -Rainbow | 52.1739% | .750 | .065 | .511 | .978 |
| C =.5 | 51.087% | .6 | .65 | .506 | .957 |

As we can see overall the accuracy of the model is barely better than guessing, as the date set was balanced with 46 Fake reviews and 46 True reviews. Adjusting for stop words increases the precision while having mixed effected on the recall. This makes sense when looking at the “ngrams” that affecting the model as many of them have you/yourself and two more words which may contain a stop word. It also points out to weakness of removing stop words, mainly that if they’re measured as a stand alone word, the removal happens, but in the context of other words, stop words are ignored by the code.

Overall we’d choose naïve bayes though neither model has preformed significantly well at detecting the fake reviews.

# Sentiment Analysis

Like the fake classifier, there was also a sentiment classifier that was provided telling us if the review was negative or not. For this section we will look if the NB and SVM algorithms are better equipped to handle classifying the reviews as either positive or negative reviews.

## Naïve Bayes

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Parameter Setting | Overall Accuracy | Precision in Category 1 | Recall in Category 1 | Precision in Category 2 | Recall in Category 2 |
| Base | 80.4348% | .833 | .761 | .780 | .848 |
| Lovins stemmer | 78.2609% | .906 | .630 | .717 | .935 |
| Ngram min 2 | 80.4348% | .938 | .652 | .733 | .957 |

Regarding the performance the model does significantly better in all three variation runs. This makes sense as sentiment analysis looks for positive and negative words. We also the see the models ability to make a decision from training is greatly increased when we require 2 words. As mentioned earlier in English negative expressions often require some two-word combination in English but not always so we get similar accuracies but better precision because the model is more sure of which items are negative.

## SVM

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Parameter Setting | Overall Accuracy | Precision in Category 1 | Recall in Category 1 | Precision in Category 2 | Recall in Category 2 |
| Base | 63.0435% | 1 | .261 | .575 | 1 |
| Lovins stemmer | 63.0435% | .929 | .283 | .577 | .978 |
| Ngram min 2 | 59.7826% | 1 | .196 | .554 | 1 |

For SVM we’re seeing improvement in the model’s ability to predict negative or positive sentiment over the ability to detect fake reviews. However, the gain is marginal. The base setting of vectorization shows an accuracy of 63%. While the precision is quite high the recall on the various passes of the cross fold is quite low. We see a similar effect when stemming is applied, as mentioned before about how English tends to product negative wording, especially in nonacademic contexts. The Ngrams was a bit more surprising, as the accuracy drops. I tried another run with a lower cost, the penalty for incorrect guesses, and the results were identical. It may likely be the point of the two closest points are noise that isn’t easily corrected for in a small data set with as many variables as this one has.

# Conclusion

Overall the models point to the fact it would be much easier to detect negative reviews than whether or not we have fake review. This may point to the reason companies tend not to remove reviews but respond to those that are particularly negative to their model. As for the small data set we were dealing with, the Naïve Bayes out preformed the SVM so if implementing an early version of negative response collection, I would recommend the naïve bayes over the svm.