Nicholas McFadden

Text Classification

Multinomial Naïve Bayes

2/7/2023

Sentiment and Lie Detection Using MNB

Introduction

In the world today technology has completely integrated into our daily lives. It has become so important to how we communicate. This has led to the creation of an incredible amount of data and at a continuously increasing rate. It is estimated that 2.5 quintillion bytes of data are created every day1. This includes the roughly 650 million Tweets per day, 100 billion messages sent on WhatsApp every day2, 333 billion emails sent per day3, and about 4 petabytes of data per day on Facebook. Thus, creating a treasure trove of information.

However, accessing/utilizing this information is not straightforward. Much of it is in the form of text and not necessarily numbers. The text data is often “noisy” or “unclean” meaning it is unstructured and/or needs preprocessing. This is the reason for the advent of text mining and natural language processing, to extract precise meaning from the text provided.

Unfortunately, things like false statements, sarcasm, and sentiment of the text are especially hard to discern but hold the most value. For example, business would find the truthfulness of a statement in a review to be important. A false statement against a product, would not effectively measure product success. Furthermore, sentiment provides another dimension of measurement. For example, a positive review generally indicates customer satisfaction, while the converse is similarly applicable.

**Analysis**

An analysis was created to contrast the effectiveness of a Multinomial Naïve Bayes classification model on lie and sentiment detection. The data for this analysis was a csv file that had sentiment and lie label columns and then the reviews split by their commas. Therefore, the review columns needed to be combined so there were only 3 columns: lie, sentiment, review. A “for” loop was used to iterate through the data, requiring only two splits and appending the values to their respective list. With the data now in a manageable state, the reviews were vectorized in two different ways to view any possible differences: “CountVectorizer” and “TfidVectorizer”. CountVectorizer, is just that it counts the frequency of words. What TF-IDF does is it balances out the term frequency (how often the word appears in the document) with its inverse document frequency (how often the term appears across all documents in the data set). In short, common words are penalized and tries to remove the model’s bias. These are relative frequencies identified as floating-point numbers.

Now there are words that need to be removed because they do not necessarily provide value such as “the”, “a”, etc. They were removed during the vectorization process but there are still numbers and short 3 letter or less words that do not provide as much value/context. These were removed using regular expressions. Now the data is cleaned and able to be viewed in word clouds. The vectorized values are then summed and sorted to extract the top 20 words for each method.

As mentioned, the purpose of this analysis is to measure a MNB model’s performance on the lie and sentiment detection. Each model was created using the cleaned review data and the respective label set for both types of vectorizations. Therefore, a total of 4 models were created. Each model was placed in a confusion matrix and then accuracy scores were calculated.

**Results**

Two word clouds were created one for each vectorizer to see how the vectorizers were ranking all of the cleaned words. As can be seen in Figure 1 and 2, both vectorizers share most of the same top 20 words. However, the ranking/weights of these words were slightly different which can be seen by the size differences in the word cloud.

|  |  |
| --- | --- |
|  |  |
| Figure 1 | |

|  |  |
| --- | --- |
|  |  |
| Figure 2 | |

Then the models are finally run, and their predictions evaluated. Those results are represented in a confusion matrix. Figure 3 & 4 are the confusion matrix for each task using the CountVectorizer. Immediately, it can be seen that the MNB model performed better in sentiment detection than lie detection.

|  |  |
| --- | --- |
| Lie Detection – CV | Sentiment Detection – CV |
|  |  |
| Figure 3 | Figure 4 |

While in Figure 5 & 6, the difference between the detection was not as distinct. The TF-IDF versions were still able to identify sentiment better.

|  |  |
| --- | --- |
| Lie Detection – TF-IDF | Sentiment Detection – TF-IDF |
|  |  |
| Figure 5 | Figure 6 |

With the predictions created and confusion matrices created, accuracy scores can be determined for all iterations (Figure 7). Interestingly, the CountVectorizer was much better than TF-IDF in identifying sentiment, but TF-IDF was slightly better than the CountVectorizer in lie detection.

|  |  |  |
| --- | --- | --- |
| Accuracy Scores | | |
| Lie Detection – CV | 0.5806451612903226 |
| Sentiment Detection – CV | 0.9032258064516129 |
| Lie Detection – TF-IDF | 0.6129032258064516 |
| Sentiment Detection – TF-IDF | 0.7741935483870968 |
| Figure 7 | | |

One would imagine that the TF-IDF vectorization would improve both models, not hinder them in the sentiment case. However, the corpus used was not very large and the resulting demotion of some frequent words could have been detrimental to the model’s ability to identify much difference. The words although common would still be very important and the TF-IDF could be hampering the model’s classification ability.

In general, sentiment detection is an easier task than lie detection. To improve these models’ ability to detect lies, bigrams and negation maybe needed to better understand if someone is making a false statement. At these accuracy rates the models are little better than random chance coin toss. Therefore, it does not seem like the model learned the concept of lie detection from this corpus. However, it does seem like it was able to somewhat understand the sentiment of this corpus’ reviews.

**Conclusion**

Overall sentiment analysis is a task often utilized within text mining. The ability of predict sentiment from a given text is often invaluable for organizations. By integrating a feedback system from customers, an organization can determine preferences for various products. This can then influence future purchasing and stocking strategies. For example, having this knowledge can better prepare a supply chain by ensuring certain regionally successful products are properly distributed to the appropriate regions/facilities.

The added information from lie detection can supplement decision factors. For example, if a lie detection classifier has high predictive qualities, then untruthful reviews can be omitted from successive analysis. This allows descriptive analysis as well as classifier(s) to generalize and increase awareness.

Lie detection though is extremely difficult even for humans to distinguish at times. However, with enough context and representative inputs a level of accuracy can be achieved that will become useful. Thus, utilizing both lie detection and sentiment analysis for reviews, can provide functional information needed for decision making.