Automatic Recognition of Satire in Web Content

Michael Abercrombie (mabercr@stanford.edu)

December 11, 2009

1 Introduction

Irony and satire can be useful weapons in any communicator's rhetorical arsenal. They provide a nuanced means for expressing critical sentiments and for openly exploring divisive subjects. However, the very subtlety that grants these devices their utility also lends to their greatest drawback: implied meanings are often lost on their intended audience. In textual communication this difficulty is magnified by the absence of any non-verbal queues that might imply a non-literal interpretation.

The goal of this project is to utilize machine learning strategies to develop a classifier for recognizing satirical or sarcastic web articles. Such content is, by definition, written to resemble more sincere communication and may be unrecognizable as ironic without sufficient contextual information (Sperber and Wilson, 1981). Therefore, in an attempt to capture broader "context", the proposed classifier will rely not only on original article texts, but also upon user-generated comments associated with each article.

2 Preparation

This project was implemented almost entirely in Python, using the Beautiful Soup library[1] for web interfacing.

2.1 Data Collection

To support the notion of leveraging user-generated comments in article classification, example articles were collected by searching for popular content on a selection of major social bookmarking sites. Initially, in hopes of creating a more representative sample of web content, these articles were to be chosen regardless of source, filtered according to a simple rubric (e.g. all content must be text of a minimum length), and labeled as sincere or otherwise. However, this method was shown to have three major limitations. First, implementing a generalized article extraction algorithm of satisfactory performance proved untenable (at least for this novice to web scripting). Second, evaluating article sincerity during data collection necessarily required human interaction, limiting the speed and accuracy for which articles could be collected. Lastly, it was found that satirical web content made up only a small percentage of texts collected in this way.

Thus, in the interest of time and having a larger, more evenly weighted sample size, several major web content sources, both satirical and otherwise, were identified as appropriate for use in this project, and the article selection algorithm was implemented to choose only articles from these

[1] http://www.crummy.com/software/BeautifulSoup/

sources. This facilitated article text extraction, as the online presentation of the selected articles was limited to a manageable set of formats. Moreover, this eliminated the need for manual classification, as each chosen web content source was known to produce satire either exclusively or not at all. The final data collection script yielded a total of 1854 articles with an average of 642.4 words in the article text and an average of 1053.2 words in associated comments.

2.2 Feature Extraction

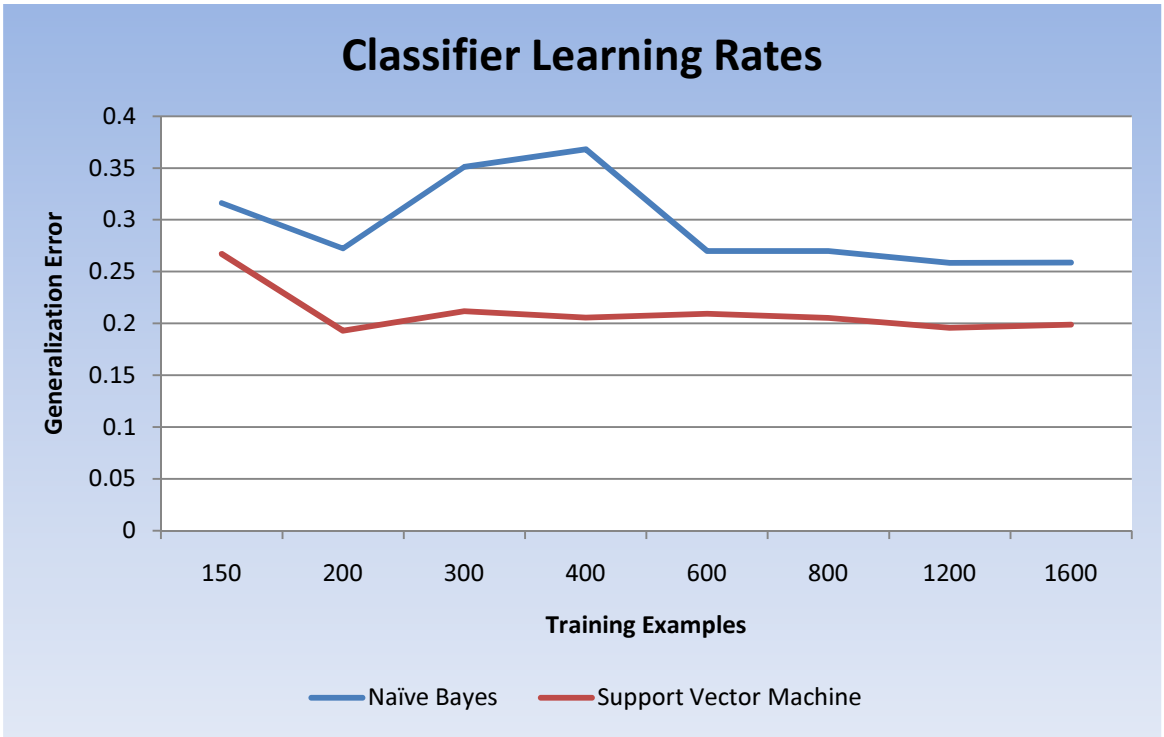
To reduce initial feature set size, the collected example articles and comments were first subjected to stemming via Porter's stemming algorithm. Note that stop words were not removed prior to this preprocessing step. Such vocabulary is often ignored in classification applications as it does not contribute significantly to the meaning of many texts. However, stop words could conceivably reflect the *style* of a written example and, therefore, could be useful indicators of subtle rhetorical properties – namely irony. Thus these words were at least initially afforded consideration for use as features, allowing for them to be removed as part of later feature reduction if needed.

Stemmed words meeting a minimum frequency requirement were selected to form a vocabulary. Each training example was then converted to an array of features, where each feature indicated the presence of a particular vocabulary word at a particular point in the example's text (indexed by word). It should also be noted that the source of each word in the training examples – either article text or user-generated comment – was preserved so that words collected from differing sources could be treated as separate features. This is justified in that some words may be more indicative of satire when used by a commenter than when used by an article's author (and vice versa).

3 Classification

Two classifiers were used in this project: one relying on Naïve Bayes and the other using a Support Vector Machine with a linear kernel. The Naïve Bayes classifier was implemented from scratch in Python while the Support Vector Machine implementation were provided by Thorsten Joachim's SMV Light[2]. The minimum generalization error achieved with the Naïve Bayes classifier was 24.8%, while for the Support Vector Machine based classifier achieved 20.9%. Graphs for the learning rate of each classifier are given on the following page.

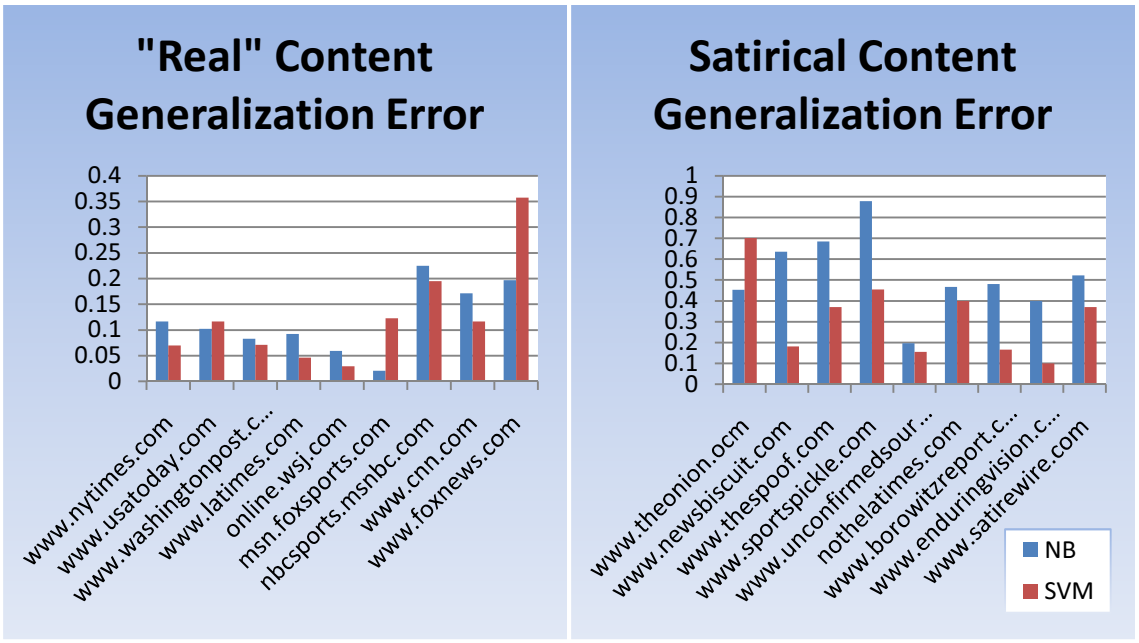
[2] http://svmlight.joachims.org/



3.1 Testing Methods

In testing these classifiers, it is important to remember that the training examples in this project were selected from an artificially limited set of content sources. Because of this, the straightforward approach of using leave-k-out validation where the left-out samples are selected randomly, could yield highly deceptive results. When using such a strategy, the source of any given article reserved for a test set would almost certainly still be represented in the corresponding training set by other articles from the same source. For example, if during one iteration of validation, five articles from "www.theonion.com" were chosen for the test set, there would still be dozens of other "Onion" articles in the training set. A low test error in this case might simply indicate that the classifier has learned to recognize articles from The Onion rather than to recognize satire.

To arrive at an error rate that might reasonably be generalized to content sources beyond those used in this project, a form of leave-k-out validation was used such that, in each iteration, the test set would be comprised of all the articles drawn from a particular content source. Incidentally, this method of testing also provided an insightful breakdown of error rates associated with each source (shown on the next page).



Note that both classification algorithms are heavily biased against classifying articles as satirical. This is relatively unsurprising result given that the data collection phase found more viable content on legitimate news sources than on their satirical counterparts, and both classifiers tend to favor the best represented labels in a data set.

3.2 Feature Reduction

For the Naïve Bayes classifier, two methods of feature reduction were explored. One was to reduce the number of words selected as features in the preprocessing phase. The other was to train the classifier using the entire data set, and then remove features with the lowest mathematical bearing on classification. Surprisingly, the first method was much more effective at reducing the classifier's generalization error. The second method was able to improve generalization error given a large initial vocabulary, but quickly reached (and then retreated from) a high minimum. Combining the two methods did not result in any recognizable improvement.

A similar tuning was performed for the Support Vector Machine classifier by adjusting the tradeoff parameter, C. This was used in place of true feature reduction to reduce the variance of the classifier.

4 Discussion

Initial results with small data sets suggested that a Naïve Bayes classifier might prove very effective for recognizing satire in articles from certain sources, while wholly ineffective with articles from others; however, as the data set – and in particular the set of content sources – was expanded, this source-by-source variance was reduced. The somewhat promising early results are probably due to the distinct similarities and dissimilarities between the particular content sources of the initial data

set. This draws attention to the need for drawing test data from a representative sample of sources – a need which this project has undoubtedly yet to fully satisfy.

One unexpected finding of this project was that splitting individual words into separate features based on whether they were found in the article text or user-generated comments did not improve performance for both classifiers. The choice to use such a feature mapping was based upon the assumption that the language used by those employing satire would be different from that of those responding to satire. This assumption was somewhat legitimized in that changing to the split word mapping resulted in a small boost to support vector machine classification performance; however, the same change caused a sizeable decrease in the effectiveness of the Naïve Bayes classifier. The cost of adding new features to the system apparently outweighed any reward.

Overall, the data from this project does suggest that a classification algorithm could at least assist in recognizing satirical web content. Moreover the leveraging of user-generated comments does seem to be useful, as performance was found to be reduced for both classifiers when comments were ignored. As the successful use of satire relies heavily upon context and subtlety, methods that consider only whether a word was used and not how it is used may ultimately prove incapable of driving a highly effective classifier. Further research may need to explore more advanced language processing methods.

References

[1] Sperber, D., & Wilson, D. (1986). *Relevance: Communication and cognition.* Oxford, England: Basil Blackwell.