Sentiment Classification using Machine Learning Techniques

Pranjal Vachaspati
pranjal@mit.edu

Cathy Wu
cathywu@mit.edu

Abstract

We implement a series of classifiers (Naive Bayes, Maximum Entropy, and SVM) to distinguish positive and negative sentiment in critic and user reviews. We apply various processing methods, including negation tagging, part-of-speech tagging, and position tagging to achieve maximum accuracy. We test our classifiers on an external dataset to see how well they generalize. Finally, we use a majority-voting technique to combine classifiers and achieve accuracy of close to 90% in 3-fold cross-validation, far outperforming Pang's 2002 work [7].

1. Introduction

Sentiment analysis, broadly speaking, is the set of techniques that allows detection of emotional content in text. This has a variety of applications: it is commonly used by trading algorithms to process news articles, as well as by corporations to better respond to consumer service needs. Similar techniques can also be applied to other text analysis problems, like spam filtering.

The source code described in this paper is available at https://github.com/cathywu/Sentiment-Analysis.

2. Previous Work

We set out to replicate Pang's work [7] from 2002 on using classical knowledge-free supervised machine learning techniques to perform sentiment classification. They used the machine learning methods (Naive Bayes, maximum entropy classification, and support vector machines), methods commonly used for topic classification, to explore the difference between and sentiment classification in documents. Pang cited a number of related works, but they mostly pertain to classifying documents on criteria weakly tied to sentiment or using knowledge-based sentiment classification methods. We used a similar dataset, as released by the authors, and made efforts to use the same libraries and pre-processing techniques.

In addition to replicating Pang's work as closely as we could, we extended the work by exploring an additional dataset, additional preprocessing techniques, and combining classifiers. We tested how well classifiers trained on Pang's dataset extended to reviews in another domain. Although Pang limited many of his tests to use only the 16165 most common ngrams, advanced processors have lifted this computational constraint, and so we additionally tested on all ngrams. We used a newer parameter estimation algorithm called Limited-Memory Variable Metric (L-BFGS)[5] for maximum entropy classification. Pang used the Improved Iterative Scaling method. We also implemented and tested the effect of term frequency-inver document frequency (TF-IDF) on classification results.

3. The User Review Domain

For our experiments, we worked with movie reviews. Our data source was Pang's released dataset (http://www.cs.cornell.edu/people/pabo/movie-review-data/) from their 2004 publication. The dataset contains 1000 positive reviews and 1000 negative reviews, each labeled with their true sentiment. The original data source was the Internet Movie Database (IMDb).

Pang applied the bag-of-words method to positive and negative sentiment classification, but the same method can be extended to various other domains, including topic classification. We additionally chose to work with a set of 5000 Yelp reviews, 1000 for each of their five star rating. Yelp is a popular online urban city guide that houses reviews of restaurants, shopping areas, and businesses. Although a movie review and a Yelp review will differ in specialized vocabulary, audience, tone, etc., the ways that people convey sentiment (e.g. I loved it!) may not differ entirely. We wished to explore how training classifiers in one domain might generalize to neighbor domains.

The domain of reviews is experimentally convenient because there are largely available on-line and because reviewers often summarize their overall sentiment with a machine-extractable rating indicator; hence, there was no need for hand-labeling of data.

4. Machine Learning Methods

4.1. The Naive Bayes Classifier

The Naive Bayes classifier[2] is an extremely simple classifier that relies on Bayesian probability and the assumption that feature probabilities are independent of one another. Baye's Rule gives:

$$P(C|F_1, F_2, \ldots, F_n) = \frac{P(C)P(F_1, F_2, \ldots, F_n|C)}{P(F_1, F_2, \ldots, F_n)}$$

Simplifying the numerator gives:

$$P(C)P(F_1, F_2, \ldots, F_n|C)$$

$$= P(C)P(F_1|C)P(F_2, F_3, \ldots, F_n|C, F_1)$$

$$= P(C)P(F_1|C)P(F_2|C, F_1)P(F_3, F_4, \ldots, F_n|C, F_1, F_2)$$

$$\ldots$$

Then, assuming the probabilities are independent gives

$$P(F_i|F_j \ldots F_k) = F(F_i)$$

so

$$P(F_i|C, F_j \ldots F_k) = P(F_i|C)$$

$$P(C|F_1 \ldots F_n) = P(C)[\prod_{i=0}^{n} P(F_i|C)]$$

$P(Fi|C)$ is estimated through plus-one smoothing on a labeled training set, that is:

$$\frac{(1 + count(C, F_i))}{\sum_i count(C_j, F_i))}$$

where $count(C, F_j)$ is the number of times that F_j appears over all training documents in class C.

The class a feature vector belongs to is given by

$$C^* = \arg\max_C P(C|F_1 ... F_n)$$

Taking the logarithm of both sides gives

$$C^* = \arg\max_C (P(C) + \sum_i [F_i(\lg count(C, F_i)$$

$$- \lg(\sum_j countC_j, F_i))])$$

While the Naive Bayes classifier seems very simple, it is observed to have high predictive power; in our tests, it performed competitively with the more sophisticated classifiers we used. The Bayes classifier can also be implemented very efficiently. Its independence assumption means that it does not fall prey to the curse of dimensionality, and its running time is linear in the size of the input.

4.2. The Maximum Entropy Classifier

Maximum Entropy is a general-purpose machine learning technique that provides the least biased estimate possible based on the given information. In other words, "it is maximally noncommittal with regards to missing information" [3]. Importantly, it makes no conditional independence assumption between features, as the Naive Bayes classifier does.

Maximum entropy's estimate of $P(c|d)$ takes the following exponential form:

$$P(c|d) = \frac{1}{Z(d)} \exp(\sum_i (\lambda_{i,c} F_{i,c}(d, c)))$$

The $\lambda_{i,c}$'s are feature-weigh parameters, where a large $\lambda_{i,c}$ means that f_i is considered a strong indicator for class c. We use 30 iterations of the Limited-Memory Variable Metric (L-BFGS) parameter estimation. Pang used the Improved Iterative Scaling (IIS) method, but L-BFGS, a method that was invented after their paper was published, was found to out-perform both IIS and generalized iterative scaling (GIS), yet another parameter estimation method.

We used Zhang Le's (2004) Package Maximum Entropy Modeling Toolkit for Python and C++ [4], with no special configuration.

4.3. The Support Vector Machine Classifier

Support Vector Machines (SVMs) operate by separating points in a d-dimensional space using a (d-1)-dimensional hyperplane, unlike Max-Ent and Naive Bayes classifiers, which use probabilistic measures to classify points. Given a set of training data, the SVM classifier finds a hyperplane with the largest possible margin; that is, it tries finds the hyperplane such that each training point is correctly classified and the hyperplane is as far as possible from the points closest to it. In practice, it is usually not possible to find a hyperplane that separates the classes perfectly, so points are permitted to be inside the margin or on the wrong side of the hyperplane. Any point on or inside the margin is referred to as a support vector, and the hyperplane, given by

$$f(\vec{B}, B_0) = \{\vec{x} | \vec{x}^T \cdot \vec{B} + B_0 = 0\}$$

is selected through a constrained quadratic optimization to minimize

$$\frac{1}{2}|\vec{B}|^2 + C \sum_i \zeta_i$$

given

$$\forall i, \zeta_i \geq 0$$

$$\forall i, y_i(\vec{x}_i^T \cdot \vec{B} + B0) \geq 1 - \zeta_i$$

For this paper, we use the PyML implementation of SVMs [1], which uses the liblinear optimizer to actually

find the separating hyperplane. Of the three classifiers, this was the slowest to train, as it suffers from the curse of dimensionalit

5. Experimental Setup

We used documents from the movie review dataset and ran 3-fold cross validation in a number of test configurations. We ignored case and treated punctuation marks as separate lexical items.

Our testbed supported testing various parameters: frequency vs. presence of features vs. term frequency-inverse document frequency, unigrams vs. bigrams vs. both, number of features, and type of feature tagging. The types of feature tagging were negation, part of speech (POS), and position. We additionally supported training and testing on only adjectives and verbs. We additionally supported the ability to use the full movie dataset as a training set and using the yelp dataset as a test set. Please see the end for the full test results.

6. Results

6.1. Feature Counting Method

There are several ways to construct a probability model for a set of document n-grams. The most obvious is to use feature frequency. The value of a feature in a given document is simply the number of times it appears in that document. Presence, on the other hand, attributes a value of 1 if a feature exists in a document and 0 otherwise.

As a whole (across all other parameters), training on presence rather than frequency performed on average 5.5% better for Naive Bayes, ranging from 0% to 10% improvement, with no particular outliers in other test configurations, from 73.1% accuracy with frequency to 78.5% accuracy with presence. There was no significant difference for SVMs and applying TF-IDF did not provide any improvement from using frequency for either. Both of these comparisons do not apply to Maximum Entropy.

Interestingly, for Naive Bayes, the positive and negative tests performed very differently between presence and frequency tests. Excluding verb tests, which did not exhibit this disparity, positive tests averaged 6.5% worse (up to 12% worse in the case) on presence tests while negative tests averaged 18.9% better (up to 30% better). There was an average aggregate difference of 25.4% between positive and negative results. By comparison, SVMs exhibited an average aggregate difference of 0.7%. These results provide evidence that training on presence rather than frequency yields models with less bias.

6.2. Conditional Independence Assumption

The Bayes classifier depends on a conditional independence assumption, meaning that the model it predicts assumes that the probability of a given word is independent of the other words. Clearly, this assumption does not hold. Nevertheless, the Bayes classifier functions well, in part because the positive and negative correlations between features tend to cancel each other out [9].

We found a huge difference between results of Naive Bayes and Maximum Entropy for positive testing accuracy and negative testing accuracy. Maximum Entropy, which makes no unfounded assumptions about the data, gave very similar results for positive tests and negative tests with a 0.2% difference on average. On the other hand, positive and negative results from Naive Bayes, which assumes conditional independence, varies by 27.5% on average, with the worst cases performing on test configurations using frequency, averaging 40% difference. These disparities suggest evidence that the movie dataset does not satisfy the conditional independence assumption.

6.3. Number of Features

One key decision in a bag-of-words feature set is which words to include. Using more words provides more information, but harms the performance of the classifiers, and words that appear only infrequently in the training data may not present accurate information due to the law of small numbers. We examine results with the entire training data, as well as with only the top 16165 and 2633 unigrams and bigrams.

Using the most frequent unigrams is an extremely simple method of feature selection, and in this case, not a particularly robust one, since feature selection should look for words that identify a given class. Choosing frequent words does not discriminate between the two classes and will select common words like “the” and “it”, which likely are weak sentiment indicators. On the other hand, uncommon words that only appear in a handful or less of reviews will not contribute much to sentiment indication. Pang’s motivation for limiting the number of features was for improve testing performance, but our classifiers and processors were fast enough that this was not particularly noticeable.

On average, limiting the number of features from 16165 to 2633, as in the original Pang paper, caused accuracy to drop by 5.2%, 4.0%, and 2.8% for Naive Bayes, Maximum Entropy, and SVM, respectively. These results indicate that valuable sentiment information was lost in the restriction of features.

However, when restricting from all features down to 16165, the results were a wash. Naive Bayes did vaguely worse, Maximum Entropy remained unchanged, and SVMs did vaguely better. These results suggest that uncommon features do not carry much sentiment information. Additionally, this validated Pang’s use of limited features, as they did not significantly impact the results but satisfied their performance constraints.

6.4. Negation Tagging

In an effort to preserve the potential value of negation information while using dead-simple features, we tagged words between those expressing negation and the next punctuation mark with a postfix "_NOT." This distinguishes sentences like "That movie was very good" and "That movie was not very good." Diverging from Pang, we also added negation tags to bigrams.

Negation tagging did not appear to have a significant effect on the data. For all the classifiers, the results from negation tagged data were almost the same as the results from the raw data. Nevertheless, we used negation tagging for the remainder of the tests, as it did not seem to hurt performance or accuracy.

The ineffectiveness of negation tagging probably comes from a few sources. First, it increases the number of uncommon features, which, as discussed previously, harms effectiveness and cancels out the increase in semantic awareness. Second, the presence of a not does not always indicate negation. Rather, it is often used idiomatically, as in the example fragment "with his distinctive, more often than not ingenious dialogue". Finally, the method of tagging all words up to the next punctuation mark is suspect. Only a few words after the not are actually semantically negated, and these often occur after a comma or other punctuation mark.

6.5. Position Tagging

Reviews are split into a beginning, middle, and end, so to see if one section carries more sentiment than another, we split the reviews into a first quarter, a middle half, and a last quarter and tagged the words in each section.

Position tagging was not helpful. For bigrams, it harmed performance by around 5% in most cases, and for unigrams, it was not helpful. If reviews end up not actually following the model specified or if the model has no bearing on where the relevant data is, position tagging will be harmful because it increases the dimensionality of the input without increasing the information content. We suspect that is the case here.

6.6. Part of Speech Tagging

We appended POS tags to every word using Oliver Mason's Qtag program [6]. This serves as a rough way to disambiguate words that may hold different meanings in different contexts. For example, it would distinguish the different uses of love in "I love this movie" versus "This is a love story." However, it turns out that word disambiguation is a much more complicated problem, as POS says nothing to distinguish between the meaning of cold in "I was a bit cold during the movie" and "The cold murderer chilled my heart."

Part of speech tagging was not very helpful for unigram results; in fact, the NB classifier did slightly worse with parts of speech tagged when using unigrams. However, when using bigrams, the MaxEnt and SVM classifiers did significantly better, achieving 3-4% better accuracy with part of speech tagging when measuring frequency and presence information.

6.7. Adjectives

Intuitively, adjectives like "beautiful", "wonderful", and "great" hold valuable sentiment information, so we trained our classifiers after filtering out only the adjectives within reviews. On average, adjective tests performed about 6% worse than their unfiltered negation-tagged counterparts, with no notable difference between the 3 classifiers. These results suggest that the limited information conveyed in adjectives is not representative of the full review itself.

6.8. Verbs

As in the motivating example for the use of POS tagging, it was in the case of the verb use of "love" ("I love this movie") that conveyed sentimental information, rather than the adjective use of the word. Interestingly, Pang did not include results for training only on verbs. Even more interestingly, despite the motivating example, verbs underperformed all other tests, while still being consistently better than random. The tests ranged from 60% to 67% accuracy, even sometimes doing worse than the 64% accurate human-based classifier from Pang 2002. We suspect this is in part due to the sparsity of features when only using verbs, as there were on average 37.2 verbs and 55.7 adjectives per review.

6.9. Majority Voting

Given a large ensemble of classifiers, an easy way to combine them is with a simple majority voting scheme. This tends to eliminate weaknesses that exist in only one classifier, but can also eliminate strengths that exist in only one classifier. Majority voting in some cases provided a small but significant improvement over the classifiers alone; combining Bayes, MaxEnt, and SVM classifiers over the same data provided a three to four percent boost over the best of the individual classifiers alone.

6.10. Neighboring Domain Data

Mostly out of curiosity, we wanted to see how our test configurations will perform when training on the movie dataset and testing on the Yelp dataset, an external out-of-domain dataset. We preprocessed the Yelp dataset[8] such that it matched the format of the movie dataset and selected 1000 of each of the 1-5 star rating reviews. For evaluation purposes, we scored the accuracy on only 1-star and 5-star reviews, giving our testbed only high-confidence negative

and positive reviews, respectively. The score was simply the average of the two accuracies.

Across the board, the classifiers had a harder time with the Yelp dataset as compared to the movie dataset, performing between 56.0% and 75.2%. The respective lowest and highest performing configurations scored at 67.0% and 84.0% on the movie dataset.

We expected to see worse results, given the difference in vocabulary, subject matter, tone, etc., but all configurations performed better than random. We also saw strong positive trends across all test configurations, classifying reviews with more stars more positively.

References

[1] A. Ben-Hur. Pyml - machine learning in python. http://pyml.sourceforge.net/, 2011.

[2] P. R. Christopher D. Manning and H. Schtze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[3] E. Jaynes. Information theory and statistical mechanics. In *The Physical Review*, volume 106, 1957.

[4] Z. Le. Maximum entropy modeling toolkit for python and c++. http://homepages.inf.ed.ac.uk/lzhang10/maxent_toolkit.html, 2011.

[5] D. C. Liu and J. Nocedal. On the limited memory bfgs method for large scale optimization. In *Mathematical Programming 45*, pages 503–528, 1989.

[6] O. Mason. Qtag. http://phrasys.net/uob/om/software.

[7] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 79–86, 2002.

[8] Yelp. Yelp academic dataset. http://www.yelp.com/academic_dataset.

[9] H. Zhang. The optimality of naive bayes. In *American Association for Artificial Intelligence*, 2004.

Test configurations				Naive Bayes			MaxEnt			SVM		
Domain	Features	# of features	Frequency	+	-	±	+	-	±	+	-	±
No-negation	Unigrams	16165	Frequency	0.94	0.62	0.78	-	-	-	0.82	0.82	0.82
No-negation	Unigrams	16165	Presence	0.87	0.72	0.82	0.85	0.87	0.86	0.85	0.84	0.84
No-negation	Bigrams	16165	Frequency	0.92	0.64	0.78	-	-	-	0.77	0.81	0.79
No-negation	Bigrams	16165	Presence	0.89	0.73	0.81	0.79	0.82	0.81	0.8	0.81	0.80
adjectives	Unigrams	16165	Frequency	0.95	0.52	0.73	-	-	-	0.75	0.77	0.76
default	Bigrams	2633	Frequency	0.91	0.46	0.69	-	-	-	0.74	0.75	0.75
default	Bigrams	16165	Frequency	0.92	0.64	0.78	-	-	-	0.78	0.79	0.78
default	Unigrams	2633	Frequency	0.96	0.5	0.74	-	-	-	0.81	0.79	0.80
default	Unigrams	16165	Frequency	0.93	0.59	0.76	-	-	-	0.82	0.81	0.82
default	Unigrams	maximum	Frequency	0.95	0.49	0.72	-	-	-	0.82	0.81	0.82
partofspeech	Bigrams	16165	Frequency	0.96	0.47	0.71	-	-	-	0.82	0.82	0.82
partofspeech	Unigrams	16165	Frequency	0.96	0.54	0.75	-	-	-	0.82	0.81	0.81
position	Bigrams	16165	Frequency	0.96	0.49	0.73	-	-	-	0.77	0.78	0.78
position	Unigrams	16165	Frequency	0.93	0.58	0.76	-	-	-	0.81	0.82	0.82
verbs	Unigrams	maximum	Frequency	0.8	0.55	0.67	-	-	-	0.61	0.65	0.63
adjectives	Unigrams	16165	Presence	0.93	0.59	0.76	0.79	0.77	0.78	0.75	0.73	0.74
default	Bigrams	2633	Presence	0.86	0.64	0.75	0.75	0.75	0.75	0.73	0.75	0.74
default	Bigrams	16165	Presence	0.89	0.74	0.81	0.81	0.82	0.81	0.78	0.79	0.78
default	Unigrams	2633	Presence	0.84	0.8	0.82	0.84	0.82	0.83	0.78	0.82	0.8
default	Unigrams	16165	Presence	0.87	0.77	0.82	0.84	0.85	0.85	0.83	0.82	0.83
default	Unigrams	maximum	Presence	0.91	0.7	0.81	0.84	0.86	0.85	0.83	0.85	0.84
partofspeech	Bigrams	16165	Presence	0.89	0.73	0.81	0.84	0.84	0.84	0.79	0.82	0.8
partofspeech	Unigrams	16165	Presence	0.86	0.76	0.81	0.85	0.85	0.85	0.84	0.83	0.84
position	Bigrams	16165	Presence	0.87	0.66	0.76	0.82	0.83	0.82	0.73	0.76	0.74
position	Unigrams	16165	Presence	0.86	0.78	0.82	0.84	0.85	0.85	0.80	0.80	0.80
verbs	Unigrams	maximum	Presence	0.80	0.54	0.67	0.65	0.65	0.65	0.64	0.63	0.635
adjectives	Unigrams	16165	TF-IDF	0.82	0.60	0.71	-	-	-	0.79	0.76	0.77
default	Bigrams	2633	TF-IDF	0.92	0.46	0.69	-	-	-	0.76	0.71	0.74
default	Bigrams	16165	TF-IDF	0.90	0.68	0.79	-	-	-	0.83	0.74	0.79
default	Unigrams	2633	TF-IDF	0.85	0.52	0.74	-	-	-	0.81	0.79	0.80
default	Unigrams	16165	TF-IDF	0.88	0.68	0.78	-	-	-	0.83	0.77	0.80
default	Unigrams	maximum	TF-IDF	0.86	0.65	0.76	-	-	-	0.83	0.78	0.81
partofspeech	Bigrams	16165	TF-IDF	0.89	0.67	0.78	-	-	-	0.79	0.74	0.76
partofspeech	Unigrams	16165	TF-IDF	0.89	0.63	0.76	-	-	-	0.81	0.78	0.79
position	Bigrams	16165	TF-IDF	0.89	0.59	0.74	-	-	-	0.79	0.69	0.74
position	Unigrams	16165	TF-IDF	0.91	0.61	0.76	-	-	-	0.81	0.71	0.76
verbs	Unigrams	maximum	TF-IDF	0.64	0.57	0.60	-	-	-	0.62	0.66	0.64

Figure 1. 3-fold cross validation results on movie dataset. Values repesent positive, negative, or overall accuracy.

Test configurations				Naive Bayes					
Domain	Features	# of features	Frequency	*****	****	***	**	*	score
default	Unigrams	16165	Frequency	0.72	0.68	0.53	0.34	0.24	0.74
default	Unigrams	16165	Presence	0.49	0.41	0.24	0.14	0.08	0.71
default	Bigrams	16165	Presence	0.50	0.42	0.26	0.13	0.10	0.70
position	Unigrams	16165	Presence	0.35	0.29	0.14	0.08	0.04	0.65
partofspeech	Unigrams	16165	Presence	0.45	0.37	0.20	0.11	0.06	0.69
adjectives	Unigrams	16165	Presence	0.76	0.73	0.61	0.45	0.36	0.70
verbs	Unigrams	16165	Presence	0.44	0.43	0.41	0.37	0.32	0.56
default	Unigrams	maximum	Presence	0.59	0.55	0.36	0.23	0.15	0.72
position	Unigrams	maximum	Presence	0.54	0.50	0.33	0.22	0.14	0.70
partofspeech	Unigrams	maximum	Presence	0.56	0.52	0.35	0.22	0.14	0.71
adjectives	Unigrams	maximum	Presence	0.76	0.73	0.61	0.45	0.36	0.70
verbs	Unigrams	maximum	Presence	0.44	0.43	0.41	0.37	0.32	0.56

Figure 2. Test results on Yelp dataset with Naive Bayes classifier. Values repesent percent of reviews classified as positive for a given star rating.

Test configurations				MaxEnt					
Domain	Features	# of features	Frequency	*****	****	***	**	*	score
default	Unigrams	16165	Frequency	-	-	-	-	-	-
default	Unigrams	16165	Presence	0.61	0.57	0.39	0.23	0.11	0.75
default	Bigrams	16165	Presence	0.63	0.59	0.45	0.28	0.26	0.68
position	Unigrams	16165	Presence	0.46	0.43	0.28	0.17	0.11	0.67
partofspeech	Unigrams	16165	Presence	0.55	0.50	0.32	0.20	0.10	0.72
adjectives	Unigrams	16165	Presence	0.75	0.72	0.62	0.45	0.37	0.69
verbs	Unigrams	16165	Presence	0.43	0.41	0.38	0.34	0.30	0.56
default	Unigrams	maximum	Presence	0.59	0.54	0.36	0.20	0.11	0.74
position	Unigrams	maximum	Presence	0.44	0.40	0.26	0.15	0.09	0.68
partofspeech	Unigrams	maximum	Presence	0.52	0.47	0.30	0.18	0.09	0.72
adjectives	Unigrams	maximum	Presence	0.75	0.72	0.62	0.45	0.37	0.69
verbs	Unigrams	maximum	Presence	0.43	0.41	0.38	0.34	0.30	0.56

Figure 3. Test results on Yelp dataset with Maximum Entropy classifier. Values repesent percent of reviews classified as positive for a given star rating.

Test configurations				SVM					
Domain	Features	# of features	Frequency	*****	****	***	**	*	score
default	Unigrams	16165	Frequency	0.78	0.76	0.62	0.42	0.30	0.74
default	Unigrams	16165	Presence	0.58	0.54	0.38	0.25	0.14	0.72
default	Bigrams	16165	Presence	0.62	0.58	0.48	0.30	0.29	0.67
position	Unigrams	16165	Presence	0.42	0.39	0.27	0.39	0.42	0.50
partofspeech	Unigrams	16165	Presence	0.52	0.48	0.31	0.21	0.01	0.75
adjectives	Unigrams	16165	Presence	0.71	0.71	0.61	0.46	0.37	0.67
verbs	Unigrams	16165	Presence	0.45	0.45	0.42	0.38	0.32	0.57
default	Unigrams	maximum	Presence	-	-	-	-	-	-
position	Unigrams	maximum	Presence	-	-	-	-	-	-
partofspeech	Unigrams	maximum	Presence	-	-	-	-	-	-
adjectives	Unigrams	maximum	Presence	0.71	0.71	0.61	0.46	0.37	0.67
verbs	Unigrams	maximum	Presence	0.45	0.45	0.42	0.38	0.32	0.57

Figure 4. Test results on Yelp dataset with SVM classifier. Values repesent percent of reviews classified as positive for a given star rating.