Assessing Opinion and Usefulness of Product Reviews

Brian Percival, Ali Motamedi

Abstract

In this project we examined the possibility of using supervised learning techniques to classify the opinion of online product reviews that are not accompanied by a quantitative opinion measure. The classifiers were trained using amazon.com customer reviews. The test results showed that a combination of a heuristic feature selection algorithm and a support vector machine (SVM) classifier trained with 1200 samples can classify the tone of product review written for the same product type with 80% confidence. Our results also suggest that a classifier trained on reviews of one product type can predict the tone of reviews of a random product with 70% accuracy.

Data Collection

Source of training data. The amazon.com website is a rich source for customer-written product reviews. The on-line store has thousands of products under many different product categories. Each product may have hundreds of reviews. As a result, the website offers thousands of reviews for many products. Each review is accompanied by the author's rating for the product being reviewed, which provides raw data for determining the author's intended opinion in the review. Further, each review can be identified as helpful or non-helpful to other consumers, yielding a score for the helpfulness of the review.

We selected two product categories on which to train opinion classifiers: digital cameras and photography books. Both categories offered over 500 products with one product review or more. Using a perl script, we crawled through search results listing the products within the specified category in order by average customer rating. This allowed us to be sure to extract reviews from all products that had been reviewed at least once. To avoid brand or product names from biasing the classifier, a maximum of only 10 reviews were extracted for each product. The first 10 reviews listed by amazon.com were the reviews extracted, which appeared to be the 10 most recently written reviews. The HTML for each review was converted to an XML representation to allow for multiple iterations of feature extraction. The text of the reviews was then stemmed according to Porter's stemming algorithm [1], and each review converted to a feature row vector in a training or test matrix.

Class Definition. Initially, we intended to classify the reviews into three categories: positive, neutral and negative, with the following raw rating to classification mapping. Each possible pair of classes were to be used to train an SVM to vote for one of two classes, and the class that received a majority vote would be the output of the overall classifier.

Customer rating	Class	Camera Examples	Book Examples
1.0-2.0	Negative	1652	1560
3.0	Neutral	683	1007
4.0-5.0	Positive	5067	10587
	TOTAL	7429	13246

Table 1: Review classes and numbers of examples of each.

Source of test data. We were able to obtain a large set of review data, enabling us to randomly select up to 900 positive and 900 negative reviews, leaving the remaining reviews to be used for verification. No training reviews were ever used for verification on the same classifier they were used to train.

We also manually saved and scored "reviews" retrieved through online search of newsgroups on google. We searched with queries such as "digital camera opinion" and "photography book opinion", and extracted all messages that appeared to be expressing an opinion. These were roughly 10% of all results returned. We each used our personal impression of the nature of the opinion being expressed in the review to be used as the expected output of the classifier. Because of the manual nature of this process, we had many fewer newsgroup reviews for verification: 31 book reviews, 10 camera reviews and 16 "random" reviews, which were found by searching for "product opinion."

Feature Selection

Features. We considered three types of features to classify the reviews: single-word features, punctuation, and numeric features

Intuitively, the negative reviews would be expected to contain more features of negative meaning such as *poorly, disappointed, worst etc*. and positive reviews would have more inherently positive words such as: *good, impressive, and superb*. There were many words in all caps formats. These words were considered as an emphasized version of the original word, thus they are counted more than one when they have occurred in a review. For punctuations we consider the count of '?', '!' and '%' to be a separate feature of each review. The total number of digits in a review can be used to assess its usefulness. The number of numeric characters is another feature.

Feature Selection Algorithm. We trained our classifier using the SVM method. Since the complexity of SVM algorithm is proportional to the feature size, we chose to limit the total number of features to guarantee convergence in a reasonable time. To achieve this goal, the total number of features were chosen to be at most 2000, this value achieves a balanced tradeoff between efficiency and accuracy of the classifier.

Table 2 shows the total number of unique single-word features in a set of training samples with different sizes.

#of reviews	1300	2000
# of unique features	12000	16500

Table 2: The total number of unique features before filtering.

To achieve goal of reducing the number of features below 2000, we first used a stemming algorithm [1] to condense words derived from the same stem word to the same feature, as mentioned above. Further, we used a heuristic method to rank the features according to their utility and kept the 10% of the most useful features. These heuristics are based on the following observations:

- The features that occur in only a tiny fraction of all reviews are not present in enough examples to be useful for classification. In order to remove these less frequent features, we required that each feature's of occurrence in all documents be at least 0.01%.
- Features with uniform distribution across all classes convey no information about the opinion expressed in a review. Many of these words fall into the following categories:
    - Common words related to English grammar structure such as: *is*, *a*, *to*, *the*
    - General/neutral/factual terms: *camera*, *book*

Value function. To remove the words identified in the second observation, we defined a heuristic value function. This value function is a measure of the "discriminability" of each feature, or a measure of how disproportionately present it is in one class of reviews over another. The value function should return zero for a feature uniformly distributed across categories. It should also return a relatively high value for features with high probability of presence in one category versus the others. In order to calculate this function we need to find the distribution of each feature in each category defined as follows:

$$P_j^i = \frac{\text{The percentage of occurences of feature i in category j}}{\text{The sum of percentages of occurence of feature i in all categories}} \tag{1}$$

The following heuristic function is then calculated to rank the features:

$$V^i = \sum_{\substack{j,k \\ j \neq k}} (P_j^i - P_k^i)^2 \tag{2}$$

If a feature has a uniform distribution across categories this function return a value close to zero; on the other hand, if a feature shows a strong presence in one of the categories compare to the others, this function returns a value between 1 and 2. Table 3 shows the most valuable features according to the defined value function, and their corresponding frequencies.

Word	# Pos	#Neut	#Neg	Value Function
crap	9	1	54	1.60151382472893
junk	21	3	137	1.58732743327792
Lemon	5	1	22	1.32953753919606
thottam	77	1	1	1.28600886762435
evolt	8	10	1	1.26940833555542
jameson	73	1	1	1.25634768111289
honor	4	1	17	1.23427153389859
cancel	6	1	20	1.23420776534806
ds	63	1	1	1.1704487668608
gateway	5	2	26	1.1631992801242
void	2	1	12	1.14886524435676
trash	9	1	21	1.14458346783872
corrupt	5	2	24	1.11471206837143
refund	5	5	45	1.05253213405523
receive	24	1	33	1.0374101770151
pleasantly	50	1	1	1.02514549688275

Table 3 : The most useful features ranked by the their value.

As the table shows, the feature selection method chooses the words that are mostly indicative of certain tone in a review. For example, in the top of the list the words like *crap* and *junk* are most useful in classifying a review as negative and words like *pleasantly* are useful in classifying a review as positive.

Training Method

We used a MATLAB implementation of Platt's SMO [2] algorithm to calculate the SVM parameters *[w,b]* for a linear classifier. In the first experiment we trained three different classifiers with 1000 training samples for each. Table 4 shows the misclassification rate for these classifiers.

	Positive vs. Negative	Positive vs. Neutral	Neural vs. Negative
Error	0.0505	0.0492	0.4231

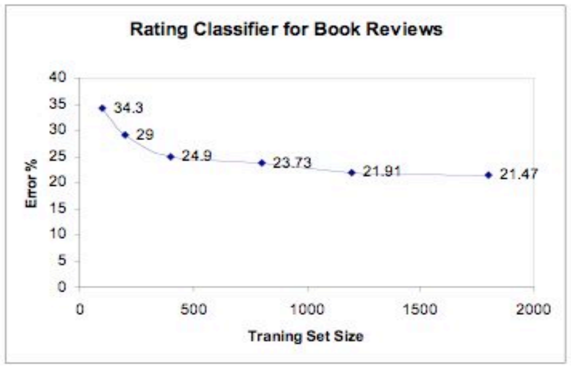
Table 4: The error performance of each classifier after training with 1000 samples

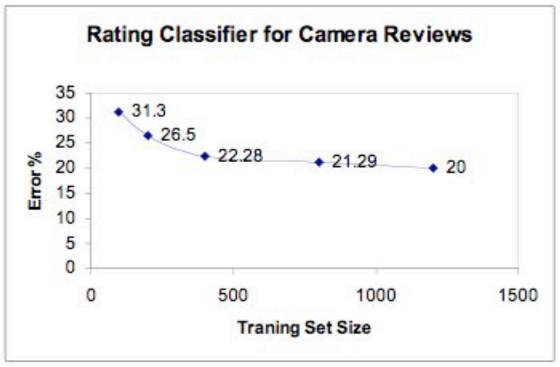
As the results indicate, the classifiers cannot differentiate between the negative and neutral classes. On the other hand, it can successfully classify a positive review from the other two types. To resolve this issue we decided to combine the neural and negative classes into a single class.

After this modification, we trained a single SVM to classify between positive and negative reviews. We performed two experiments: in the first experiment we examined the misclassification rate as a function of the training set size. Two SVM were separately trained by 100, 200, 400, 800, and 1200 samples from digital camera and book reviews. We then tested the classifiers on a number of random reviews of the same type that were not included in the training set. In order to have a symmetric error probability of positive and negative reviews we decided to have equal number of training samples of each category. In the second experiment we planned to test if a classifier trained with one product category can estimate the opinion of other product reviews. In this experiment the classifiers were trained with digital camera reviews and tested on book review and vice versa.

Results

Error vs. training set size on same review type. Figure 1 shows the error during verification on the test set for various training set sizes. In each case, all reviews in the corpus not used for training were used in the verification step. The training set sizes used were 100, 200, 400, 800, and 1200. In each case, 50% of the training set were positive reviews and 50% were negative reviews. In both cases, the error rate appears to asymptotically approach a value near 20%, with little decrease in error from 1200 to 1800 in book reviews, and from 800 to 1200 in digital camera reviews.





(a) (b)

Figure 1: Error vs. training set size for a) Book Reviews and b) Digital Camera Reviews

Error for reviews from different sources. Table 5 shows the errors obtained when a classifier for one product type is used to classify the reviews for the other product type. In both cases, the entire review corpus for each product type was used in the test set. For testing the book classifier with digital camera test data, there were 7429 test reviews. For testing the digital camera classifier with book review test data, there were 13,246 test reviews.

In Table 5, the error rates for positive test reviews and negative test reviews are reported separately, to reflect our desire to equalize the error between positive and negative reviews. The overall error rate in each experiment is also presented.

Training Data	Testing camera reviews		Testing book reviews	
	Positive reviews	**Negative** reviews	**Positive** reviews	**Negative** reviews
Camera reviews	21%	17%	29%	37%
Overall	20%		31%	
Book reviews	37%	25%	22%	23%
Overall	34%		22%	

Table 5: Errors for each cross-training experiment. First two rows present accuracy of the digital camera-trained classifier. Second two rows present accuracy of the book-trained classifier. Errors are shown on the overall test set as well as for the positive and negative sub-sets.

Table 6 presents similar data for the product reviews found in newsgroups. For these experiments, fewer test reviews were available: 31 book reviews, 10 digital camera reviews, and 16 random reviews. Here again, errors are higher than the review classifiers testing reviews of the same product type, but errors are near the errors obtained on the cross-training experiments of Table 5. Though the sample sizes are small, these preliminary results suggest that a single general classifier is feasible that will obtain error rates less than 30%.

Training Data	Book review from newsgroups		Camera reviews from newsgroups		Random product reviews from newsgroups	
	Detecting **positive** reviews	Detecting **negative** reviews	Detecting **positive** reviews	Detecting **negative** reviews	Detecting **positive** reviews	Detecting **negative** reviews
Camera reviews	22%	63%	17%	75%	22%	43%
Overall	32%		40%		31%	

Table 6: Errors for cross-training experiment using newsgroup postings as "reviews."

Conclusions

With the feature set we selected, our results suggest that 1200 training samples are sufficient to achieve 20-22% error rate when trying to classify reviews on the same product type. The results also suggest that roughly 30% error rate can be achieved with a single general classifier using the features obtained through the above methods. Some further work that might improve the error rate would be to include bi- and tri-grams of words as features, to capture meaning in such phrases as "not bad" or "never appropriate, " which are lost in a unigram feature set as in this work. In addition, some work might be done to investigate ways to detect reviews that refer to more than one product. Some reviews, for example, compare one product to another, which might appear to the classifier to present conflicting opinions within the same review.

References

[1] http://tartarus.org/~martin/PorterStemmer/
[2] J. Platt, *Fast Training of Support Vector Machines using Sequential Minimal Optimization*, 2000. (also: http://research.microsoft.com/users/jplatt/smo.html)