Sentiment Regression: Using Real-Valued Scores to Summarize Overall Document Sentiment

Adam Drake, Eric Ringger, Dan Ventura
Computer Science Department
Brigham Young University
3361 TMCB PO Box 26576
Provo, UT 84602-6576

Abstract

In this paper, we consider a sentiment regression problem: summarizing the overall sentiment of a review with a real-valued score. Empirical results on a set of labeled reviews show that real-valued sentiment modeling is feasible, as several algorithms improve upon baseline performance. We also analyze performance as the granularity of the classification problem moves from two-class (positive vs. negative) towards infinite-class (real-valued).

1. Introduction

Sentiment classification is the problem of classifying the opinion or feeling of written text. It has many potential applications including systems for automatic product recommendation, "flame" detection in online forums, assigning ratings to written reviews, organizing written surveys by satisfaction level, email filtering, and organizing/summarizing reviews of products by feature.

Previous work in sentiment analysis has considered classification scenarios involving just a few sentiment categories. In some applications, this coarse-grained view of sentiment may be sufficient. In other situations, however, such a coarse-grained analysis may be unacceptable. Furthermore, even in cases where a coarse-grained analysis is acceptable, more fine-grained sentiment distinctions, if possible, would generally be preferred.

In this paper, we examine the ability of learning algorithms to make precise, fine-grained assessments of overall sentiment. Specifically, we consider the problem of summarizing the overall sentiment of a review by labeling it with a real-valued score. In doing so, we introduce a real-world data set of reviews labeled with scores on a 91 point scale (1.0 to 10.0 in increments of 0.1) and compare the performance of several machine learning algorithms on the task of predicting the score given by the author.

In addition, we compare the performance of the algorithms as the granularity of the sentiment categorization moves from two-class (positive vs. negative) towards infinite-class (real-valued). As the granularity becomes increasingly fine, it seems reasonable to expect that a regression approach to sentiment analysis will eventually be more appropriate than a classification approach. Consequently, we compare the performance of both regression- and classification-based algorithms as the number of sentiment categories increases.

2. Related Work

Early work in classifying the overall sentiment of reviews as positive or negative includes the work of Turney [9] and Pang, Lee, and Vaithyanathan [7]. Turney used the mutual information between phrases and specific semantic words to find positive and negative clauses, which were used in turn to determine overall document sentiment, while Pang, Lee, and Vaithyanathan applied machine learning algorithms to the task. Pang and Lee [6] later extended their work to more fine-grained 3-class (positive, negative, or neutral) and 4-class (x out of 4 stars) scenarios, and they introduced a method for allowing standard classification algorithms to make use of the natural ordering of sentiment classes. Bikel and Sorensen [1] presented results of applying an averaged perceptron with a word subsequence kernel to user reviews from Amazon.com (on a 5 point scale), finding that it can accurately distinguish between reviews that are above and below a chosen score.

In other related work, Yu and Hatzivassiloglou [12] have presented work on distinguishing between opinions and facts at both the sentence and document level. Dave, Lawrence, and Pennock [2] and Hu and Liu [3] have presented results of work on extracting feature-specific sentiments from reviews of a single product and presenting

summaries (by feature) of the results. Meanwhile, Wilson, Wiebe, and Hwa [10] presented experimental results on a problem related to sentiment classification: determining the strength of an opinion.

3. Real-Valued Sentiment Analysis

Performing sentiment analysis on a real-valued scale can be viewed as a generalization of the typical sentiment classification scenario. Many of the issues in sentiment classification, such as differences in word usage and meaning between authors, become even more pronounced as the desired precision increases. For example, consider the difficulty of accurately inferring a real-valued score from a review when one author might describe a product given a score of 80 out of 100 as "excellent" while a more demanding author may reserve that term for products with scores above 95.

To test the ability of learning algorithms to make fine-grained sentiment distinctions, a set of video game reviews was collected from GameSpot.com. Each game review is labeled with a score between 1.0 and 10.0, rounded to one decimal point. (Since scores are rounded to one decimal point there is a finite set of 91 possible scores; nevertheless, a real-valued perspective seems appropriate.)

All reviews from the months of June 2005 through December 2006 were collected, resulting in a total of 1,822 reviews (roughly 96 per month). The average review length is approximately 1,300 words, but the length varies significantly, with the smallest review containing only 93 words and the largest containing 4,638. There are 31 different authors represented in the data, although some contributed more reviews than others (several authors contributed only a few reviews, while just over half of the reviews came from the seven most common authors).

4. Feature Selection

In the experiments described in the following sections, there were typically around 30,000 unique words in the training data. However, the learning algorithms generally performed better when the vocabulary was reduced to a smaller subset, V, of those words.

In order to determine which words should be considered, we used a word-score correlation metric. The correlation of a word w with the scores of a set of reviews R, denoted $c(w, R)$, was defined by the following:

$$c(w,R) = \frac{1}{|R|} \sum_{r \in R} \left\{ I(w,r) \cdot \left(S(r) - \frac{1}{|R|} \sum_{r' \in R} S(r') \right) \right\}$$

where $S(r)$ is the real-valued score associated with review r and $I(w, r)$ is a function that outputs 1 if r contains word w and outputs -1 otherwise.

Table 1. The top 10 positively and negatively correlated words, according to the word-score correlation metric.

Positive		Negative	
0.514	great	-0.251	dull
0.369	quite	-0.236	generic
0.353	excellent	-0.199	decent
0.309	new	-0.196	repetitive
0.301	experience	-0.195	ugly
0.300	definitely	-0.187	boring
0.291	best	-0.183	bland
0.290	expect	-0.170	poor
0.290	year	-0.168	terrible
0.285	unique	-0.166	poorly

Note that the $\frac{1}{|R|} \sum_{r' \in R} S(r')$ term is the average review score. Intuitively, if a word is positively correlated with review scores then it would tend to appear in documents with above average scores and be absent from reviews with below average scores. Similarly, if a word is negatively correlated with review scores then it would tend to appear in documents with below average scores and be absent from reviews with above average scores.

To see how this applies in the correlation metric defined above, notice that if a word w appears in a review r and r's score is above average, then both $I(w, r)$ and $(S(r) - \frac{1}{|R|} \sum_{r' \in R} S(r'))$ are positive, and the correlation goes up. If w does not appear in review r and r's score is below average, then both terms are negative, and again the correlation goes up. Meanwhile, in the other two cases (when w is not in r and r's score is above average and when w is in r and r's score is below average), the terms have different signs and the correlation drops.

This metric reveals how much a word's presence/absence tends to cause a review's score to deviate from the mean on average. A large positive value indicates that the word tends to occur in reviews with above average scores and be absent from reviews with below average scores, while a large negative value indicates the opposite. A value near 0 indicates that the word's presence does not tend to influence the score significantly in either a positive or negative direction. This metric implicitly tends to remove words that occur too rarely or too frequently to be useful for learning.

Table 1 shows the top 10 positively and negatively correlated words over the entire set of reviews from June 2005 through December 2006.

It is interesting to note in Table 1 that some of the most

correlated features, particularly on the positive side, do not carry obviously positive sentiment. However, they appear to be used so much more often in phrases of positive sentiment that there is a significant positive correlation between their usage and the score of the review.

Since both positive and negative correlations are useful for learning, the words added to the vocabulary were those that maximized the absolute value of $c(w, R)$. This has the effect of selecting the words whose presence/absence causes the score to deviate from the mean the most on average.

The number of vocabulary words used by each learning algorithm is a parameter that is tuned by validation along with any other algorithm-specific parameters. (The validation process is described later in the Results section.) Using the set V of vocabulary words, each review r is converted into a Boolean vector $\vec{x} = \{x_1, x_2, ..., x_{|V|}\}$ in which x_i is true if and only if the i^{th} vocabulary word appears in r. All of the learning algorithms presented in the following section used these Boolean vectors as input features. Thus, the algorithms learn to predict the score of a review based solely on presence/absence of words in the review.

5. Learning Algorithms

Four learning methods, two classification-based and two regression-based, were used in the sentiment experiments: a Naive Bayes classification algorithm, a linear regression algorithm, and classification and regression support vector machine (SVM) algorithms.

5.1. Naive Bayes

The Naive Bayes algorithm treats each possible score as a different class. Given a review to classify, it will predict the class c that is most likely given the feature vector $\vec{x}$, under the assumption that the input features are conditionally independent of each other given c:

$$\begin{aligned}\operatorname*{argmax}_c P(c|\vec{x}) &= \operatorname*{argmax}_c \frac{P(c)P(\vec{x}|c)}{P(\vec{x})} \\ &= \operatorname*{argmax}_c P(c)P(\vec{x}|c) \\ &= \operatorname*{argmax}_c P(c)\prod_i P(x_i|c)\end{aligned}$$

The probabilities $P(c)$ and $P(x_i|c)$ are estimated from counts in the training data. In order to smooth the probability distribution, the counts used to estimate $P(x_i|c)$ in our experiments were incremented by a small value δ, which was set to the value that gave the best result in the validation process described in the next section.

5.2. Linear Regression

The linear regression algorithm attempts to learn a function f that maps input vectors to scores. It represents f by a linear combination of the input features:

$$f(\vec{x}) = w_0 + \sum_i w_i x_i$$

We used a forward step-wise approach to set the weights and perform feature selection. Initially, w_0 was set to the mean review score. Then, until the desired number of input features had been added, input features were added incrementally. At each step, the feature that would reduce squared error most if added was selected. When added, its weight was set such that squared error would be minimized, given the features and weights already added. The number of input features to include was determined by validation. (Note that since the linear regression algorithm performed its own feature selection, it did not use the word-score correlation metric to choose vocabulary words.)

5.3. SVM

The two SVM algorithms are based on the classification and regression variants of Joachim's SVM$^{\text{light}}$ [4]. In the experiments reported below, the default settings of SVM$^{\text{light}}$ were used.

The regression SVM, like linear regression, learned a function that mapped the Boolean feature vectors to scores. The classification SVM, on the other hand, like Naive Bayes, treated each possible score as a unique class. In order to use the SVM in a multi-class scenario, n binary classifiers were created, each used to distinguish one of the n classes from the others. Test instances were classified by choosing the classifier that reported the largest positive distance from its decision surface.

6. Results

The experiments on the GameSpot data were conducted as follows. The reviews of the six months from July 2006 through December 2006 were used as test data, with each month tested separately. When testing on a particular month, the algorithms were given the previous 12 months of reviews as training data. This resulted in training sets of roughly 1,000 labeled reviews.

The parameters of the algorithms, including the number of vocabulary words to use, were set by using the previous month as a validation set. Specifically, the parameter settings that gave the best results in month i (with months $i-12$ through $i-1$ used as training data) were used as the parameter settings when testing on month $i+1$.

Table 2. Average squared error on the real-valued sentiment prediction task. All algorithms outperform the baseline on average, while the SVM Regression algorithm has the lowest error overall.

Test Set	# Reviews	Baseline	Linear Regr	Naive Bayes	SVM-Class	SVM-Regr
Jul 2006	53	3.07	1.49	3.19	3.31	1.33
Aug 2006	50	2.60	1.10	1.69	1.83	0.87
Sept 2006	89	1.63	0.89	1.49	2.18	0.84
Oct 2006	118	2.29	1.45	1.75	1.66	1.45
Nov 2006	157	2.50	1.11	1.80	1.81	0.99
Dec 2006	124	2.32	1.40	1.93	1.75	0.92
Average		2.35	1.24	1.89	1.96	1.06

Classification accuracy, the percentage of correctly classified examples, is an effective performance metric in a 2-class sentiment classification scenario. However, as the sentiment spectrum is subdivided into more categories, classification accuracy becomes less meaningful (and less realistic). It becomes more important to predict a sentiment that is "close" to the true sentiment. Thus, the performance metric we use to compare the algorithms is mean squared error, or the average squared distance between the score given by an author and the score predicted by a learning algorithm:

$$\frac{1}{R}\sum_{r\in R}(score(r) - prediction(r))^2$$

This metric requires that the sentiment categories be mapped to real numbers. In our case, the reviews are already labeled with scores, although one could imagine any sentiment categorization being mapped to reasonable real values. Note that when there are two classes/values, there is a direct relationship between squared error and classification accuracy, as lower squared error always implies higher classification accuracy, and vice versa.

To provide a point of reference on the effectiveness of the algorithms, a simple baseline algorithm was also tested. The baseline algorithm uses the mean score of the training data as its prediction on future data.

6.1. Real-Valued Sentiment Prediction

Table 2 shows the average squared error of each of the algorithms on the six months of test data. The average error is a weighted average, with each month weighted by the number of examples that month (i.e., it is the per-example average over the six month time period.) The classification algorithms (Naive Bayes and SVM Classification) treat each possible score as a unique class. Since the review scores are rounded to one decimal point in the 1.0-10.0 range, there are potentially 91 classes; however, the algorithms only consider classes observed during training, and many of the possible scores do not occur in every training set. (On average, there were 75 unique scores per training set.) The regression algorithms (Linear Regression and SVM Regression) treat the scores as real-valued outputs of an unknown function. When testing, the real-valued predictions of the regression algorithms are rounded to the nearest valid score.

Table 2 reveals that each of the algorithms outperformed the simple baseline. The best performing algorithm was the regression-based SVM, which had an average squared error of 1.06, compared to the baseline squared error of 2.35. In terms of the absolute error, or the average distance between the true and predicted score, the SVM Regression and baseline algorithms erred by 0.76 and 1.21 points, respectively, on average. (If the distribution of scores were uniformly spread across the 1.0 to 10.0 range, we would expect the baseline algorithm to be off by about 2.25 on average. However, the distribution of scores for these reviews is somewhat concentrated around a mean value of about 7, so baseline performance is better.) Thus, while non-uniformity in the distribution allows the baseline algorithm's predictions to be off by only 1.21 points on the 10 point scale, the SVM algorithm's predictions were nearly half a point closer on average.

Given that the algorithms are making predictions on the basis of word presence alone, this result is encouraging, and it suggests that learning to make accurate predictions on a fine-grained sentiment scale is feasible. We suspect that there is still room for improvement on this sentiment analysis task, and we expect that the application and development of more sophisticated techniques will lead to improved results.

6.2. Classification vs. Regression

Of the four algorithms tested on the GameSpot reviews, the classification algorithms performed the worst. This is not surprising, given that the regression algorithms naturally

incorporate the concept that nearby sentiment values are close, while to the classification algorithms there is no similar concept of closeness between classes (although a method for explicitly incorporating a measure of closeness between sentiment classes was successfully applied by Pang & Lee [6] in 3- and 4-class scenarios). Furthermore, the ability of classification algorithms to make precise sentiment distinctions is limited by the fact that their precision is limited to the number of classes they consider, and as the number of classes increases the number of examples of each class becomes small.

Figure 1 demonstrates the effectiveness of the algorithms as the granularity of the sentiment spectrum moves from two classes towards the full [1.0...10.0] range. In this experiment, the 10 point scale was subdivided into regions of equal size, and review scores were set to the midpoint of their region. Thus, for example, in the two class case, the [1.0...10.0] range was divided into the regions [1.0...5.5] and (5.5...10.0], with scores in each region set to 3.25 and 7.75, respectively. (Reviews with scores on the boundary between two regions were arbitrarily assigned to the lower region.) The figure shows algorithm performance relative to the baseline algorithm. Specifically, the figure plots each algorithm's squared error divided by the baseline squared error. The actual squared errors of the algorithms are shown in Table 3.

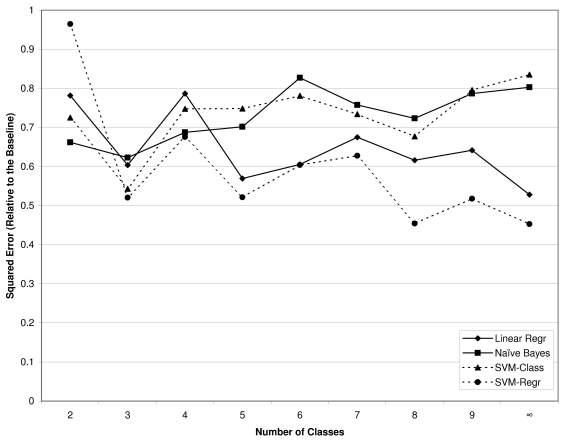
Interestingly, in the two-class case, the two best results came from the classification algorithms. In the three- and four-class scenarios, results were mixed. Beyond four classes, the regression algorithms were always superior. Also noteworthy is the fact that the SVM regression algorithm was the top performer in all experiments with more than two classes (although in the 6-class scenario it was matched by linear regression). These results suggest that unless the number of sentiment categories will be quite small, a regression approach will likely be best.

7. Conclusion

In this paper, we have considered a real-valued approach to sentiment analysis, and compared the performance of several learning algorithms at the task of assigning a real-valued score to a review. Empirical results suggest that learning to accurately evaluate sentiment on such a fine-grained scale is possible. Using a simple approach based on the presence and absence of words, the SVM regression algorithm reduced the squared error of the baseline algorithm by more than half. In absolute terms, its predictions were off by an average of 0.76 points (compared to a baseline of 1.21 points) on the 1.0-10.0 scale.

We expect that this result can be improved as more sophisticated sentiment analysis techniques are applied. Possible areas for improvement include accounting for contextual changes in the sentiment of a word (e.g., "not good" vs. "good") [11], identifying subjective portions of the reviews and applying the algorithm on just those portions [5], or identifying the reviewer's sentiment with respect to specific aspects of the product [8].

Figure 1. Average squared error, relative to the baseline, as the number of sentiment classes increases. Although the classification algorithms (Naive Bayes and SVM Classification) initially perform better than the regression algorithms (Linear and SVM Regression), the regression algorithms perform better as the number of sentiment classes grows.



As expected, the results also suggest that regression approaches will outperform classification approaches as the number of sentiment classes approaches a real-valued scale. In fact, our experiments showed that the regression and classification algorithms performed similarly when there were three or four classes, while beyond four classes the regression algorithms always performed better.

References

[1] D. Bikel and J. Sorensen. If We Want Your Opinion. In *Proceedings of the International Conference on Semantic Computing*, pages 493–500, 2007.

[2] K. Dave, S. Lawrence, and D. Pennock. Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews. In *Proceedings of the 12th International World Wide Web Conference*, pages 519–528, 2003.

[3] M. Hu and B. Liu. Mining and Summarizing Customer Reviews. In *Proceedings of the tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 168–177, 2004.

Table 3. Average squared error for increasingly fine-grained sentiment categorization. Although the classification algorithms (Naive Bayes and SVM Classification) initially perform better than the regression algorithms (Linear and SVM Regression), the regression algorithms perform better as the number of sentiment classes grows.

# Classes	Baseline	Linear Regr	Naive Bayes	SVM-Class	SVM-Regr
2	4.87	3.80	3.22	3.53	4.69
3	4.77	2.88	2.97	2.59	2.48
4	2.85	2.24	1.96	2.13	1.93
5	3.31	1.89	2.32	2.48	1.73
6	2.56	1.55	2.12	2.00	1.55
7	2.48	1.68	1.88	1.82	1.56
8	2.63	1.62	1.91	1.78	1.20
9	2.44	1.56	1.92	1.94	1.26
∞	2.35	1.24	1.89	1.96	1.06

[4] T. Joachims. Making Large-Scale SVM Learning Practical. In *Advances in Kernel Methods - Support Vector Learning*, pages 168–177. MIT-Press, 1999.

[5] B. Pang and L. Lee. A Sentimental Education: Sentiment Analysis using Subjectivity Summarization Based on Minimum Cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 271–278, 2004.

[6] B. Pang and L. Lee. Seeing Stars: Exploiting Class Relationships for Sentiment Categorization with Respect to Rating Scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 115–124, 2005.

[7] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs Up? Sentiment Classification using Machine Learning Techniques. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 79–86, 2002.

[8] A. Popescu and O. Etzioni. Extracting Product Features and Opinions from Reviews. In *Proceedings of the Human Language Technology Conference and the Conference on Empirical Methods in Natural Language Processing*, pages 339–346, 2005.

[9] P. Turney. Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 417–424, 2002.

[10] T. Wilson, J. Wiebe, and P. Hoffmann. Just How Mad Are You? Finding Strong and Weak Opinion Clauses. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence (AAAI '04)*, pages 761–769, 2004.

[11] T. Wilson, J. Wiebe, and P. Hoffmann. Recognizing Contextual Polarity in Phrase-Level Sentiment Analysis. In *Proceedings of the Human Language Technology Conference and the Conference on Empirical Methods in Natural Language Processing*, pages 347–354, 2005.

[12] H. Yu and V. Hatzivassiloglou. Towards Answering Opinion Questions: Separating Facts from Opinions and Identifying the Polarity of Opinion Sentences. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 129–136, 2003.