CIS 520 Final Project

Sentiment Analysis for Yelp

Team Autoboost
Gareth Cross, Pimkhuan Hannanta-Anan, Chao Qu

December 10, 2013

1 Introduction

This project aims at building a learning algorithm for sentiment classification of the Yelp review-rating system. Given user reviews of businesses listed on Yelp, the final algorithm must produce an accurate prediction of satisfaction score (star rating) on a per-review basis. The feature space of the data consists of the English vocabulary that appears in the entire review dataset. A great number of words are not informative; including such terms decreases performance and may even impair the accuracy of the learned model. A method for feature selection is therefore essential to this problem, in order to exclude the non-informative words, while retaining the highly indicative ones. We selected Bi-Normal Separation (BNS) as our main feature selection method, as it was found to yield best performance for this dataset. Text classification based strictly on unigrams can be misleading. Phrase-based learning is crucial, in order to capture more context and enable disambiguation of some features. We therefore constructed another feature space from bigrams and used it to train a subset of our base models. The viable hypothesis space is large in text classification problems. The use of a single learning model is unlikely to perform well, and combining multiple non-redundant base methods is adviseable. Prior literature suggest that three methods - Multinomial Naive Bayes (NB), Logistic Regression (LR), and Support Vector Machines (SVM) - show best performance in document classification problems. [3] Accordingly, we chose to implement these methods as base models and then combined them using several ensemble methods, including bagging, averaging, and ridge regression.

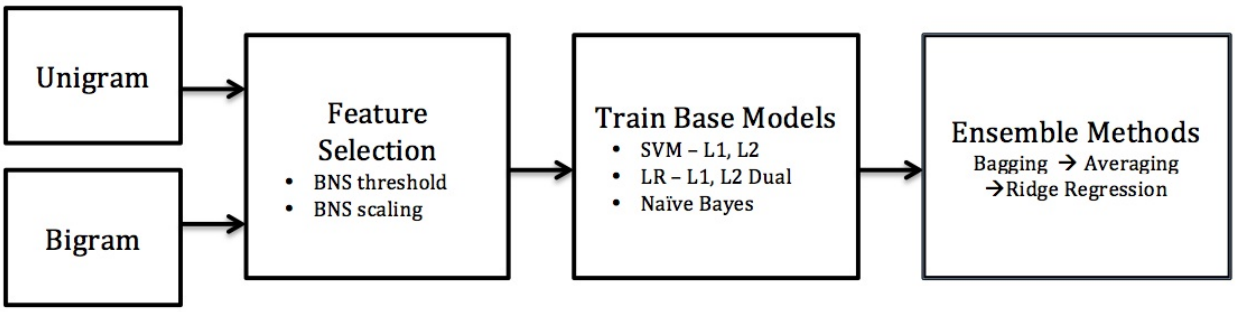

Figure 1: Workflow of the learning algorithm for predicting Yelp business ratings

Figure 1 illustrates the workflow of our learning algorithm, consisting of four major steps. First, two feature spaces (unigram and bigram) are constructed out of the words and sentences contained in the reviews. Next, a BNS feature selection is applied to select out the non-predictive features and distribute proper weights to each informative feature. The weighed features are used to train each individual base model to yield a set of weak classifiers, which are subsequently combined using a bagging method, resulting in a classifier with better performance. Finally, all classifiers are combined with ridge regression in order to yield the final ensemble model.

2 Data Analysis and Feature Selection

2.1 The Yelp Data

The Yelp dataset consists of 5-star ratings on 25,000 labeled training data and 5,000 unlabeled quiz data - used for leaderboard ranking. We also have another 10,000 unseen data points reserved for the final test. The feature space is described by a bag-of-words model with 56,835 different terms. In addition, metdata is provided for every observation - including the full review text.

We first divide the entire training data into an 80% training, 20% validation split. We pick model parameters based on 5-fold cross-validation RMSE using the larger portion. Once the ideal parameters have been identified, the model is trained again using all of the 80% portion, then tested on the 20% validation set. If the changes pass the validation test, we train our model once more (using the same parameters) on the entire training data before submission. It was found that this technique provided a highly effective estimate of the final quiz RMSE.

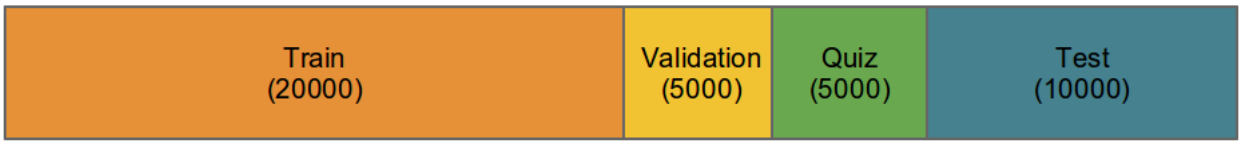

Figure 2: The Yelp dataset

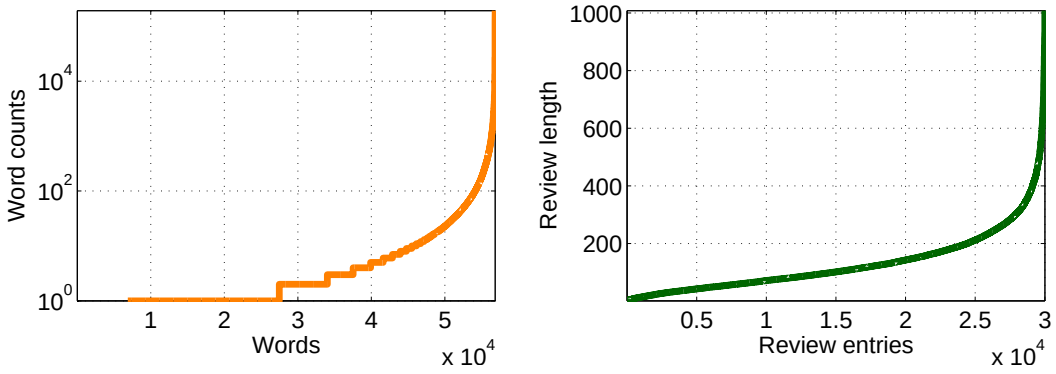

Figure 3: a). Word vs Word counts b). Review entires vs Review length

2.2 Bi-Normal Separation

Selecting informative features and culling useless features generally leads to much better runtime performance and lower RMSE. Two feature selection methods were examined: information gain (IG) and Bi-Normal separation (BNS). IG is well documented, and measures the change in entropy when the feature is retained. BNS measures the separation between the sample true positive rate and the sample false positive rate as described by the equation below [1]:

$$\text{BNS} = |F^{-1}(\text{tpr}) - F^{-1}(\text{fpr})|$$

Where:

- F^{-1} is the inverse CDF of the standard normal distribution
- tpr is a sample true positive rate
- fpr is a sample false positive rate

We applied these two feature selection algorithms separately to the unigram feature space - the ten most predictive features are plotted in figure 4. When comparing the two algorithms, it should be apparent that the words considered as predictive were quite distinct. The BNS favours negatively correlated words (words that appear in a large number of negative reviews). IG, on the other hand, tends to select positive terms. The non-predictive words (as determined by both algorithms) convey little sentimental meaning, and appear in positive and negative reviews in approximately equal proprotion. These words are clustered along the diagonal, and tend to be rejected by both selection methods.

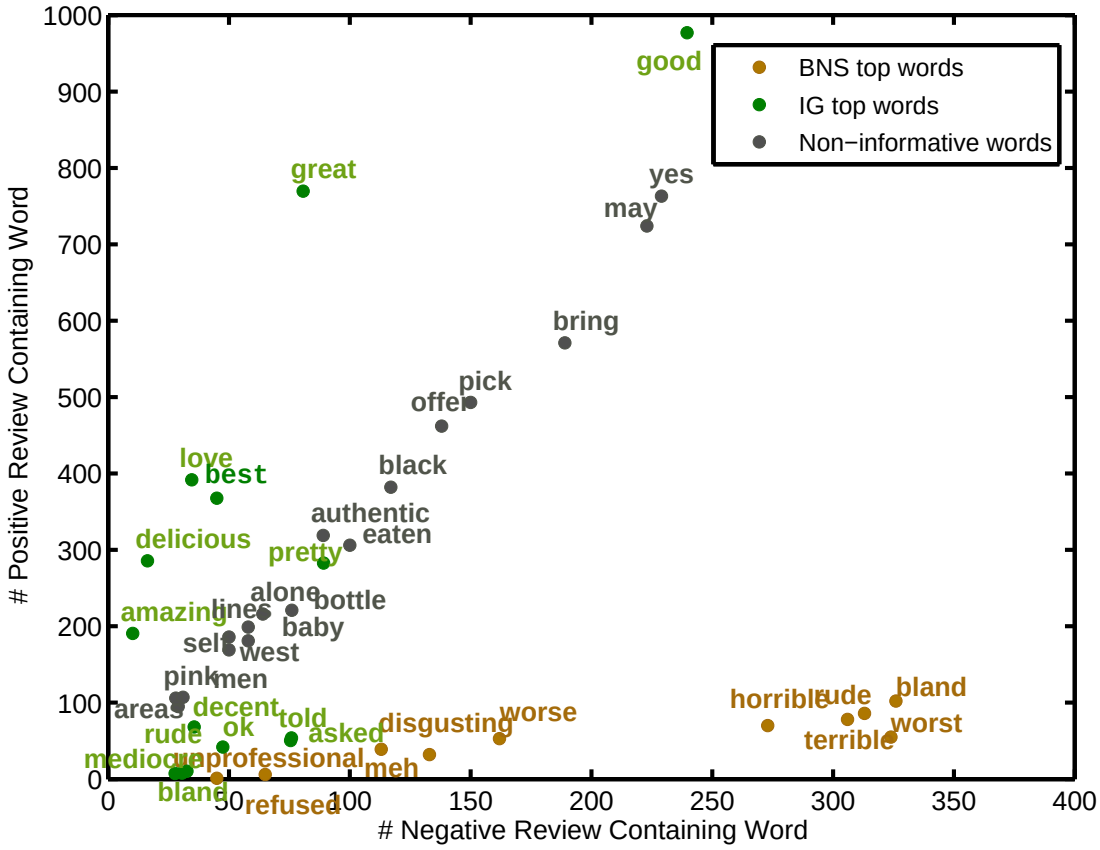

Figure 4: Examples of predictive and non-predictive words determined by BNS and IG

Because BNS and IG favor very distinctive groups of words, it was important to assess which would perform better on the Yelp dataset. The two feature selectors were implemented and applied to SVM, LR, and NB classifiers. It was found that BNS outperforms the IG algorithm handily in all classifiers, scoring lower RMSE by as much as 0.3. We inferred that negative terms offer a greater level of discriminatory power, and selected BNS for all subsequent feature selection.

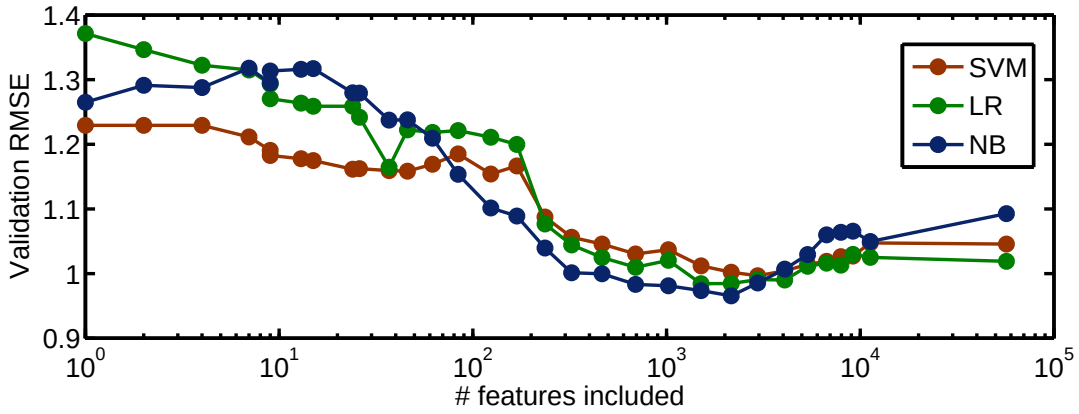

Figure 5: The effects of number of features on the RMSE

Only those features with BNS higher than a given threshold were included. As illustrated in figure 5, the number of features included has a significant impact on the prediction RMSE. Having excess or insufficient number of features can greatly impair the prediction. Referring to the graph, the optimal number of features is approximately 12,000 for all classifiers, which corresponds to the BNS value of 0.01. With this threshold, the cross-validation RMSE is improved by 0.02-0.05, as compared to the model trained on all the otirinal features.

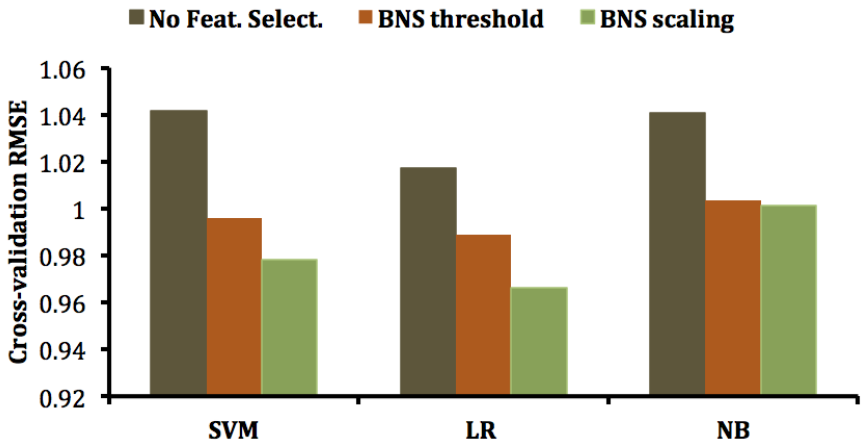

Figure 6: The impact of BNS feature selection on RMSE scores

We next examined whether weighing features could further improve the accuracy of the model. After thresholding away the non-predictive features, we scaled the rest according to their BNS values and applied the scaled features to SVM, LR, and NB. The BNS scaling appears to further reduce the RMSE by 0.02 in SVM and LR, while no or very small effect on RMSE is observed in NB. The improvement on BNS feature scaling is expected since some features are more predictive than others. Weighting relevant features more heavily is particularly important in text classification, as it naturally supresses uninformative terms that appear in high frequency.

3 Base Models and Performance Analysis

Cross-validation RMSE								
Bagging	n	Feat. Selection	SVM(l1rl2l)	SVM(l2rl1l)	SVM(l2rl2l)	LR (l1)	LR(l2)	NB
None	1	No Feat.	1.0891	1.1114	1.0633	1.0034	1.0173	1.0408
		BNS thresh.	1.1329	1.1629	1.0872	0.9830	0.9887	1.0035
		BNS scaling	1.0460	1.0140	0.9479	0.9894	0.9662	1.0015
Vote	1	BNS scaling	0.9912	0.9932	1.0046	0.9806	0.9657	0.9965
Average	1	BNS scaling	0.8518	0.8407	0.8604	0.8277	0.8489	0.8889
	2	BNS scaling	n/a	0.8840	0.8808	0.8277	0.8489	0.8889

Table 1: Base model performance

3.1 Logistic Regression (LR)

Logistic regression (LR) is one of the more powerful classifiers for text categorization. We implemented two classes of LR including LR with L1 regularization and dual LR with L2 regularization. The objective functions the two LR optimize are provided below:

$$L1 : argmin_w \sum |w_j| + C \sum log(1 + exp(-y_i w^T x_i))$$

$$L2, dual : argmin_\alpha \; 0.5(\alpha^T Q \alpha) + \sum \alpha_i log(\alpha_i) + \sum (C - \alpha_i) log(C - \alpha_i) - constant$$
$$s.t. \; 0 <= alpha_i <= C,$$

C represents a cost value determining the weight of the loss function in relative to regularization term. This C value was tuned such that it minimized the objective function, while prevented over-fitting. To find the optimal C value, we ran a 10-fold cross-validation on the training dataset and select the C value that minimizes the cross-validation RMSE score. The optimal cost values for the L1 regularization and L2 regularization with dual view was calculated to be 5.01 and 7.94, respectively.

We next sought to examine the relative performance of two LR methods. Shown in table 1, both LR methods perform equally well on the original feature space and thresholded features. Dual LR with L2 regularization, however, outperforms the other on the BNS weighed features. No improvement with feature scaling in the LR with L1 regularization could be due to redundancy in feature selection of L1 regularization. By scaling features with BNS scores, a number of features carry very small values, and are likely to be forced to zero with L1 regularization. Some informative features are, therefore, lost in this L1 optimization resulting in slight compromise in the LR (L1) performance. Dual LR with L2 regularization, on the other hand, performed bettered with weighted features. This can be expected since duality takes into account of the difference among both features and examples opposed to the primal view that focuses only on the features. Because some reviews are more informative than others, weighing each example differently in addition to the features should lead to better performance.

3.2 Naive Bayes (NB)

NB is another method that has been shown to work well with text classification. The key parameters in NB are the type of prior probability of the label and type of feature distribution. We chose an empirical prior, and implemented the multinomial distribution described by [2]. By comparison, normal distribution appeared to perform very poorly. This could possibly be explained by non-normally distributed nature of word features. With short documents like Yelp reviews, each term is likely to show up only once in each review. The feature values are therefore primarily 0 or 1, and therefore map poorly to a Gaussian distribution. On the other hand, NB with multinomial distribution performs very well.

NB performance also improve markedly when implemented on BNS reduced feature space. Shown in table 1, the RMSE was reduced by 0.04. However, NB was consistently outperformed by SVM and LR in term of classification RMSE. This could be explained by the assumption (implied by NB) that the features are conditionally independent. Since words are evidently correlated, this assumption does not hold.

3.3 Support Vector Machine (SVM)

Support vector Machine (SVM) has shown to be effective for document classification. SVM focuses on maximizing the distance of the observations between classes as opposed to generating probabilistic models as in LR and NB. We implemented three classes of SVM including L1-regularized L2-loss (L1RL2L) SVM, L2-regularized L1-loss (L2RL1L) SVM, and L2-regularized L2-loss (L2RL2L) SVM. To obtain optimal models, we ran a 10-fold cross-validation to find the best cost values for all the models. The optimized cost values are calculated to be 0.3162, 1.1788, and 0.6105, respectively. With the optimal cost values, the three classes perform relatively similar with validation rmse scores range around 1 when implemented with BNS scaled features (see 1). We also tested the SVM models with different kernels including linear, polynomial, sigmoid, and intersection kernels. Among all kernels, the SVM with the intersection kernel appears to perform best with rmse score improved by 0.01. The improvement with the instersection kernel could be expected since this type of kernel has shown to work best with sparse data especially in text classification. With the optimized parameters, SVM still does not perform as well as the logistic regression. However, it is still advantageous to include SVM to the final ensemble model as it provides unique optimization algorithm compliment to the other two models.

3.4 Models with Additional Features

3.4.1 Bigrams

The bigram feature space consists of all the pairs of words that occur sequentially in the review texts. Leveraging bigrams requires first extracting all of such pairs from the metadata in order to form a bag of bigrams space, where each feature is a unique bigram. This first step was performed using binary trees. Review texts are first converted to lowercase and stripped of any non-alphanumeric characters. The resulting bigrams are appended to an alphabetically-ordered binary tree, where each node records every observation of its bigram. An example stump assembled from part of a review is displayed in figure 7.

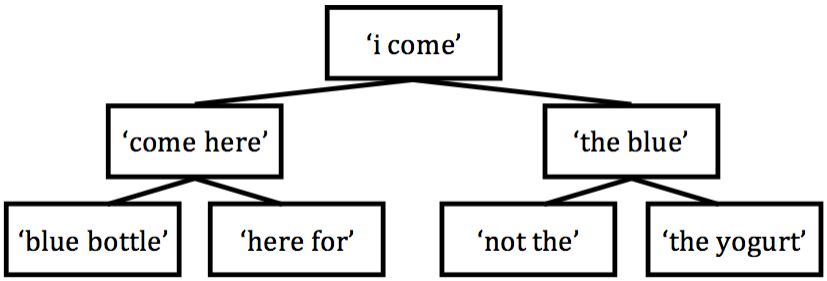

Figure 7: Bigram binary tree stump

By traversing the tree from left to right, one can construct a matrix corresponding to the bigram feature space. An alphabetized list of vocabulary is obtained in the same manner. While the total number of unique bigrams in the training set is considerable ($> 700,000$), the sparsity of the resulting observation matrix remains high. 71.6 percents of the bigrams appear only once in the entire training metadata. As in the normal unigram space, BNS is employed to measure the usefulness of each bigram.

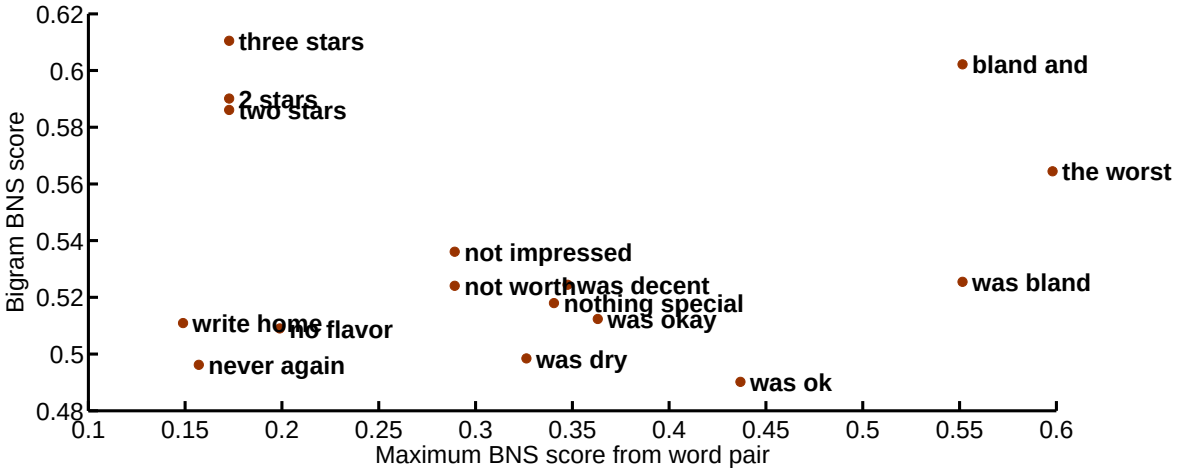

Figure 8: Examples of bigram words

As in the case of unigrams, negative bigrams tend to offer greater discriminatory power (as measured by BNS) than positive bigrams. In figure 8, several bigrams have been plotted such that the Y axis represents the bigram BNS score, while the X axis represents the greater of the two individual corresponding unigram BNS scores. This plot illustrates conceptually how bigrams add value to the final solution. Features that carry less value individually can be boosted when paired together. For example, the word pairs "two stars" and "write home" become are much more effective discriminants than their independent parts.

A total of four models were trained on bigram data: two SVMs, one LR, and one multinomial NB. In the case of the first three, the cost parameter was optimized via 5-fold cross-validation before being verified on a holdout set. The significant size of the bigram feature space poses a computational challenge, which we resolve by aggressively thresholding features by BNS score, prior to training. Figure 9 demonstrates the holdout-validation performance of all four models as a function of the BNS threshold employed.

In order to simplify training, all four models were trained on the same reduced feature space. The threshold selected to minimize their collective errors was 0.07, resulting in approximately 26,500 bigrams.

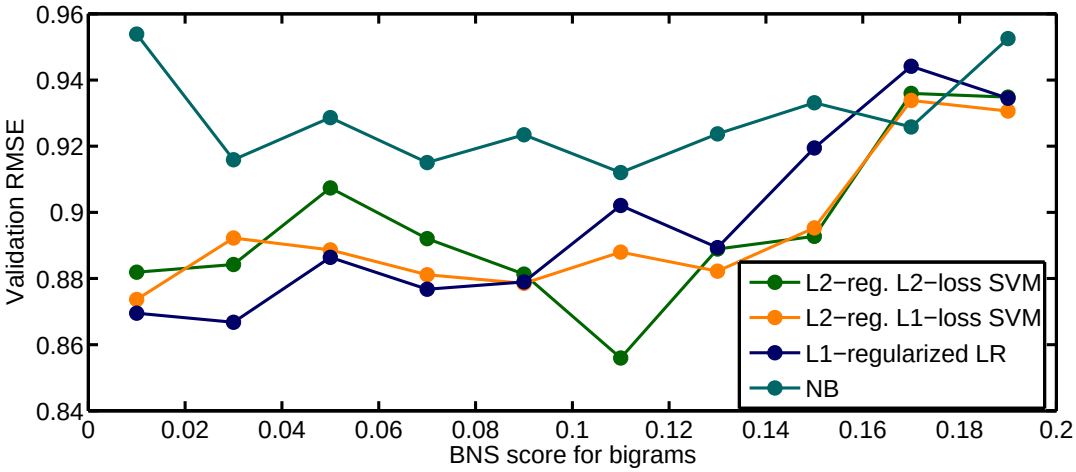

Figure 9: Selection of BNS threshold for bigrams

3.4.2 Other Additional Features

Though we have base models that perform fairly well on the review contents, it was hypothesized that the models could be improved if we included features from the metadata. To achieve that, we incorporated average rating of each business into the prediction and ran a regression to determine the proper weights on the predicted rate and the average business rate. The results, however, showed that the information on the average business rate worsened the learning performance. This could possibly be explained by a few reasons. First, there are a number of businesses with a small number of reviews making the average rates of such businesses highly unreliable. Weighting the prediction with the average values, as a result, often impairs the final prediction as opposed to improving it. In addition, the standard deviations on the rating of some business are large, causing the average to become less trustworthy. Other data, including the "cool", "funny" and "useful" labels are extremely sparse, and yield no discernible improvement.

4 Ensemble Method

Previously, we optimized all the parameters to obtain the best individual classifiers possible. However, the accuracy obtained from each classifier is still quite far from desired. Ensemble algorithms are therefore essential for combining these classifiers to achieve a better learning model. The first chosen method is bagging, which works extremely well with noisy data such as is found in sentiment classification. Bagging trains a learning algorithm with a different subset drawn randomly from the original training examples. [4] The algorithm is run several times on different training subsets to yield multiple hypotheses, which are then combined through voting or averaging to yield a single hypothesis. The key factors that affect the performance of the final classifier are the number of subsets (number of bags) used to train an algorithm and the method employed to combine the hypotheses obtain from these different bags.

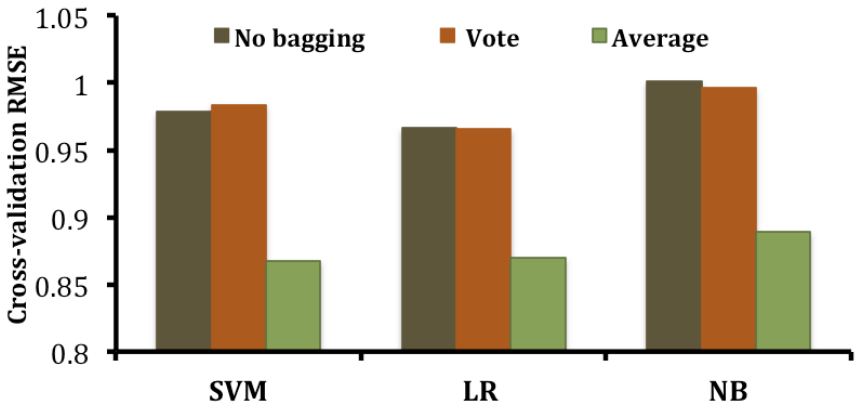

Figure 10: The effects of different bagging modes ('vote' vs 'average') on the RMSE

We first examined the method for combining the hypotheses by comparing between the best RMSE obtained from voting and averaging methods. The bagged classifiers obtained by averaging perform significantly better, lowering the RMSE score by at least 0.1, whereas the voted bagged classifier does not improve any of the models (figure 10). We next set out to find the optimal number of bags resulting in the highest prediction accuracy. SVM, LR, and NB were trained with different numbers of bags, and the corresponding cross-validation RMSEs were plotted in figure 10. The RMSE decreases with increased numbers of bags, and begins to level off as the number of bags approaches 10. We selected 10 as the optimal number of bags to train all the base models. Once the parameters were optimized, nine base models including four classes of SVM (L2RL2L, L2RL1L, L2RL2L-bigram, L2RL1L-bigram), three classes of LR (L1, L2-dual, L1-bigram), and two classes of NB (NB and NB-bigram) were trained with bagging in average mode to yield nine high performance bagged learning algorithms. To construct the final ensemble model, we ran a ridge regression to determine a proper weight for each base algorithm. The ridge regression coefficient was selected as $\lambda = 55000$ using cross-validation. With this final ensemble model, we are able to achieve a quiz RMSE score of 0.75

- demonstrating a considerable improvement over the performance of a single classifier. 12 illustrates the performance of the ensemble as a function of the number of models included.

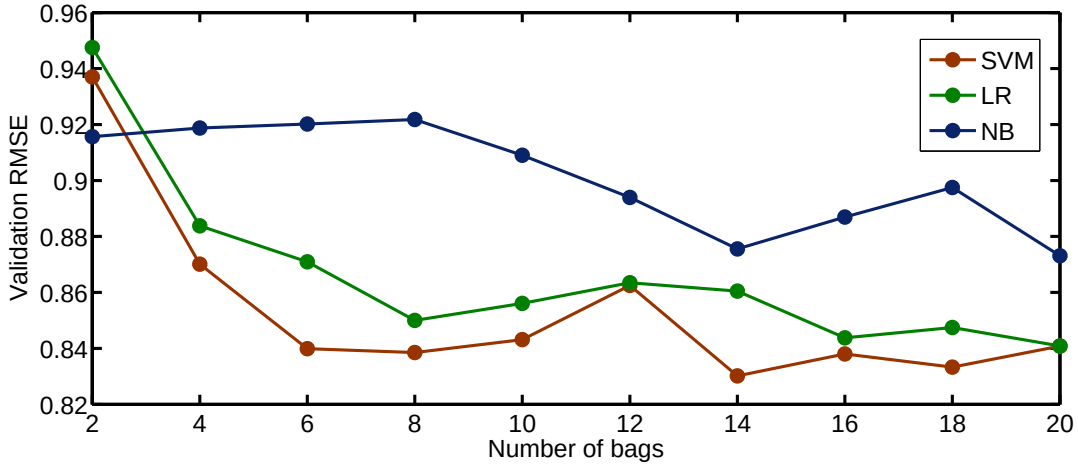

Figure 11: Effect of bag count on model performance

Bagging improves the learning models by overcoming a statistical problem. The amount of training data is too small compared to the size of the hypothesis space. By training the learning algorithm with multiple subsets of data, we obtain different hypotheses with similar accuracy. Averaging over the predictions from all hypotheses allows us to reduce the risk of choosing the wrong classifier. Because all hypotheses perform equivalently well (approximately), they should be weighted equally as opposed to giving bias towards the majority - hence averaging outperforms voting.

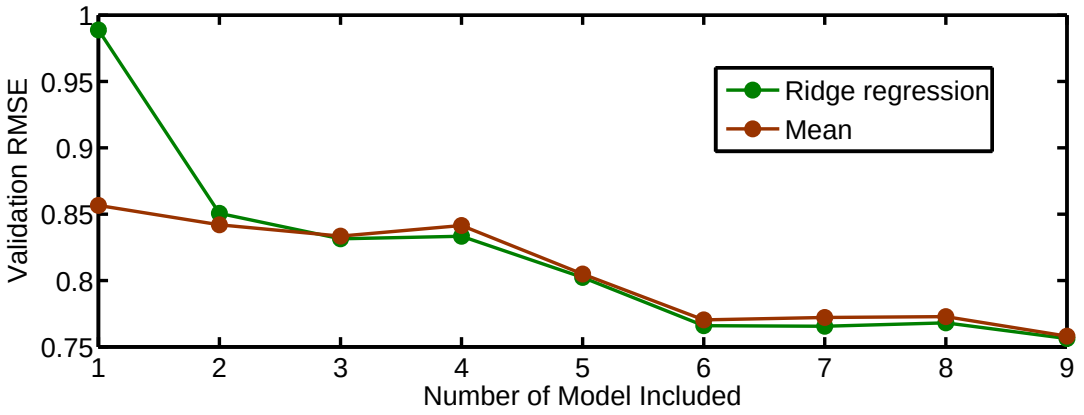

Figure 12: Effect of model inclusion on performance

5 References

References

[1] George Forman. BNS feature scaling: An improved representation over TF-IDF for SVM text classification. Technical report, HP Laboratories, 2007.

[2] Ashraf M. Kibriya, Eibe Frank, Bernhard Pfahringer, and Geoffrey Holmes. Multinomial naive bayes for text categorization revisited. Technical report, Department of Computer Science, University of Waikato, New Zealand, 2008.

[3] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. In *IN PROCEEDINGS OF EMNLP*, pages 79–86, 2002.

[4] Zhi-Hua Zhou. Ensemble Learning. Technical report, National Key Laboratory for Novel Software Technology, Nanjing University, 2006.