CS 536: Final Project Report: Yelp Rating Prediction

Jia Song
Department of Statistics and Biostatistics
RUID: 146006458
NetID: js1582
jia.songplus@rutgers.edu

Yichen Zhang
Department of Statistics and Biostatistics
RUID: 140009836
NetID: yz501
yz501@scarletmail.rutgers.edu

1. INTRODUCTION

Yelp is a very useful app that lots of people use everyday to find a reliable business. However, time is resticted and people rarely have enough time to read the text reviews. Most of the time, the star rating is the main standard when choosing from many similar business. In this project, our goal is to minimize the error in our prediction for Yelp star rating based on text reviews and some potential other features thus to get an idea of how people give the star rating. Three methods are implemented in our experiments: Naive Bayes, SVM and Bigram. The challenge in this experiment is the trade-off between the accuracy and space, time complexity of the model. To make better training and prediciton, feature selection is implemented for improving the accuracy as well as reducing the complexity of the model. Our final model is a SVM-based model adopting feature selection (reduced to 6011 features from 56835) and bigram (3486 features added), and with model-training iterations of 20 times. This model can improve RMSE of the basic SVM model by an average of 0.044, improve from an average of 0.992 to an average of 0.948.

Keywords

Yelp; rating prediction; bag of words; feature selection; SVM; bigram

2. BACKGROUND

A lot of efforts have been invested in information extraction from web[1], opinion mining[2] and review mining[3]. In these literature, Dave et al. developed a system to find review's tags and associated sentiment score with them. Lee et al. introduced a system that users were able to add tags with a negative/positive sentiment to a review[4]. Turney[5] et al. and Pang[6] et al. analyzed review texts using machine learning algorithms and n-gram techniques to determine the sentiment orientation of the phrases. Our work use the idea of bigram but we didnot use sentiment analysis.

Yatani et al.[7] and Huang et al.[8] designed different interfaces for Yelp that show top frequent adjectives used to describe a business. The authors did not provide any evidence to show whether using adjectives are more effective than other words. It is also not clear whether using sentimental scores have more advantages over raw reviews' text.

Students in University of California, Irvine did a project on predicting yelp rating[9] based these ideas. They use top K frequent words or top K adjective frequent words and compare these two feature models with the basic "bag of words" models, found that linear regression always give the best result, for the basic "bag of words" model, they can get RMSE of 0.6014 and by using top K frequent words as features, RMSE is 0.6488; by using top K frequent adjective

[1]C.-H. Chang, M. Kayed, M.R. Girgis and K. F. Shaalan, "A Survey of Web Information Extraction Systems", IEEE Trans Knowl Data Eng, vol. 18, no. 10, pp. 1141-1428, OCt. 2006

[2]B. Pang and L. Lee, "Opinion Mining and Sentiment Analysis", Found Trends Inf Retr, vol. 2, no. 1-2, pp. 1-135, Jan. 2008

[3]A.-M Popescu and O. Etzioni, "Extracting product features and opinions from reviews", Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, Stroudsburg, PA, USA, 2005, pp.339-346

[4]K. Dave, S. Lawrence and D. M. Pennock, "Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews" Proceedings of the 12th international conference on World Wide Web 2003, pp. 519-528

[5]P. D. Tuckey "Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews", Proceedings of the 40th Annual Metting on Assocition for Computational Linguistics, Stroudsburg, PA, USA, 2002, pp. 417-424

[6]B. Pang, L. Lee and S. Vaithyanathan, "Thumbs up?: sentiment classification using machine learning techniques", Proceedings of the ACL-02 conference on Empirical methods in natural language processing- Volumn10, Stroudsburg, PA, 2002, pp. 79-86.

[7]K. Yatani, M. Novati, A. Trusty and K. N. Truong "Review Spotlight: a user interface for summarizing user-generated reviews using adjective-noun word pairs" Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, New York, NY, USA, 2011, pp. 1541-1550

[8]J. Huang, O. Etzioni, L.Zettlemoyer, K.Clark and C. Lee, "RevMiner: an extractive interface for navigating reviews on a smartphone," Proceedings of the 25th annual ACM symposium on user interface software and technology, New York, NY, USA, 2012, pp. 3-12

[9]M. Fan, M. Khademi, "Predicting a business' Star in yelp from its reviews' text alone"

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

words, RMSE is 0.6052. They conclude that the adjective words are the top important features in predicting. However, we can find that although top K frequent adjective words work better than normal top K frequent words but they didn't improve at all compared with the basic model, which means this method is kind of useless. Besides linear regression, they also tried SVM and decision trees, all these give a basic "bag of words" RMSE around 0.7, similar to linear regression, top K frequent words will increase the RMSE and top K adjective words will decrease compared with normal top K frequent, but still higher than the basic "bag of words" model. The low RMSE they got (0.6) may occur for the sub-dataset they chose, since the for the basic Naive Bayes model, their RMSE is just around 0.7 while for our dataset the basic Naive Bayes model without any modification is more than 1.

Besides the above mentioned literature, there has been little published or public research on Yelp's data to our knowledge. So in this paper, we proposed our own feature selection method and tested these ideas. Based on the current result, we proposed a new model which we believe will potentially work better and give lower RMSE.

3. DATASET

We used Yelp Dataset Challenge [10]which is available on yelp's website. This dataset contains $11,537$ businesses, $8,282$ check-in sets, $43,873$ users, and $229,907$ reviews. In this huge dataset, $25,000$ restaurants were randomly selected. And the review part and rating part was used at first. Others are proposed to be used in our future work. We summarized 56835 bag of words features and calculated the count of words for review. Thus, dataset was transformed to "Counts" and "Label". Where the "Label" is the star rating: 1, 2, 3, 4 or 5. The form of "Counts" is $25,000 \times 56835$ sparse matrix. The form of "Label" is $25,000 \times 1$ list. Then the dataset was randomly partioned into two parts, one contains 20000 items as training and one contains 5000 items as test.

4. METHODS

4.1 Metric

The metric we used to evaluate the performance is root mean squared error (RMSE).

$$RMSE = \sqrt{\frac{1}{N}\sum_{1}^{N}(y_i - \hat{y_i})^2}$$

where $\hat{y_i}$ is the predicted star rating by the models we proposed and y_i is the real star rating in test dataset.

4.2 Data Processing (feature selection)

Feature Selection Method is chosen for reducing the dimensions of training set data. Combined with SVM, it can improve the performance greatly, especially when the number of features are much greater than the number of samples. Moreover, it tremendously reduces the time and space complexity of the model while not decreasing the accuracy at the same time. Therefore, feature selection is necessary for the establishment of the models. Three steps are done in our experiment. Even for Naive Bayes, feature selection can reduce time complexity and decrease RMSE values, although not obvious as SVM.

4.2.1 Remove StopWords

We did a lot of research to find a list of 65 stop words, including a,an,the,but etc, and counts of the corresponding columns are changed to 0 and will be deleted in later reduce sparsity step. These words are deleted since they provide no information. Including them in the "bag of words" sparse matrix will only increase the model running time.

4.2.2 Stemming

Stems of 56835 words are found by using $nltk.stem.porter()$ (python package) and words with the same stem are combined as the same vocab. The dimension is reduced from 56835 to 38011. Stemming should work for Naive Bayes, however it should also be noted that since SVM itself can expand original features to infinite feature dimensions to increase the prediction accuracy. Stemming may not work well for SVM. But it at least decrease the memory required for saving dataset.

4.2.3 Reduce Sparsity

There are words that may misspell or appear in only a small portion of the documents, which provide little information. Therefore, we chose to remove the words that appear less than 0.05% of the total documents in SVM. The dimension is reduced from 38011 to 6518. The reason we choose 0.05% but not a larger number to do further reduction is due to the fact that SVM works better in higher dimensions compared with some other methods.

4.3 Naive Bayes

Multinomial Naive Bayes was used in this model. The python package $sklearn.naive_bayes.MultinomialNB()$ was utilized for building this model.

4.4 BiGram (add additional features)

Since unigram algorithm ignores the words order, we decided to implement bigram algorithm to take phrase into consideration. We add more than 3000 additional features extracted from raw dataset by implementing bigram. Since there are 56835 original features, trying to extract all bigrams will introduce about 170000 additional features which will cost very long time and greatly increase the computation load. So what we did was to use the negative words: not, not a, not very, not a very, not too, nothing as the first part of bigram and search through all the texts to append the potential second word and add it to our bigram vocabulary. This finally introduced 3486 additional features. These additional features were added to the selected features, and used in different algrithms for classification.

4.5 SVM

The reason we chose SVM as our base model is that SVM works efficiently in high dimensions as well as when features are larger than samples like our datasets.

4.5.1 Find Best c

c is a regularized parameter of SVM. SVM with increasing c will select more features. SVM with decreasing c will make the model sparse. Cross validation is used to find the best c of SVM.

[10]Available online at https://www.yelp.com/dataset_challenge/dataset

4.5.2 Feature Selection

One of the disadvantages for the SVM is that it may give poor performance when the number of features is much greater than the number of samples. As described in 4.2, dimension of dataset is reduced.

4.5.3 Average Model

Since the reviews behave randomly, to reduce the fact that some training data may mislead the predictions, each time we randomly select a subset of data to train the model and repeat the process several times. After that, we do the average analysis (find majority vote) on the predictions made by all the models trained in the previous steps. This randomization process and average analysis eliminate the noisy to some extent, and thus, reduce the prediction error. To be precisely, we implement the algorithm as follows: In each iteration, the whole dataset is randomly divided into five parts, three parts or four parts of them are used to generate one model. All the models are stored to generate a class of results in further predictions. We then use the most voted predicion to be the final result. Based on the general performance of the model, different numbers of models are used in different implementations to balance the final performance and model memories. Finally, 1-50 models are trained in SVM.

4.5.4 with BiGram

We add more than 3000 additional features extracted from raw dataset by implementing Bigram. Since there are 56835 original features, trying to extract all bigrams will introduce about 170000 additional features which will cost very long time and greatly increase the computation load. So what we did was to use the negative words: not, not a, not very, not a very, not too, nothing as the first part of bigram and search through all the texts to append the potential second word and add it to our bigram vocabulary. This finally introduced 3486 additional features and final dimension was increased to $10,004$. These additional features were added to the selected features, and used in svm for classification.

4.5.5 Repeatability

To check if the model is repeatable, we randomly splited the dataset to training and test data 25 different times(with bigram), 5 times(without bigram) and did the average svm model with corresponding dataset. We calculated the mean and standard deviation of each repeat times.

5. RESULTS

5.1 Naive Bayes

From table 1, mean RMSE of dataset after stemming step was reduced to 1.0550 from 1.0691 of original dataset. And the error bars (standard deviation) show in the figure 1. The first point is original dataset, the second point is dataset without stop words and third one is dataset with stem. However, Naive Bayes is more suitable for small data set. But when it comes to large dataset and large dimension, there are methods that can work better than Naive Bayes, which we will reach deeper in our project as below.

5.2 SVM

Table 1: RMSE of Naive Bayes with Different Dataset

Dataset	RMSE	sd
Original	1.0691	0.0176
W/o stopwords	1.0601	0.0092
W/ stem	1.0550	0.0105

Table 2: Cross Validation of SVM

C value	RMSE
0.005	1.012
0.008	0.9746
0.01	0.9694
0.05	0.9702
0.10	0.9972
0.50	1.0720
1.00	1.0937
1.20	1.0987
1.50	1.1143
2.00	1.1209

5.2.1 Find Best c

We have tried different c values in the range from 0.005 to 2.00. The results are shown in figure 2 and table 2, when c is 0.01 or 0.05, the RMSE reaches the lowest. When c = 0.01, RMSE is 0.9694. When c = 0.05, RMSE is 0.9702. Therefore, we chose c = 0.01 and c = 0.05 as candidates for building further model.

5.2.2 Feature Selection

One of the disadvantages for the SVM is that it may give poor performance when the number of features is much greater than the number of samples. As mentioned in the method part, we reduced the number of features to 6518 by feature selection. Using $c = 0.05$ as an example, the error does not change too much because of reducing sparsity and adding bigram features according to table 3 and figure 3. In figure 3, the leftmost point is mean RMSE of original datset, the middle point is mean RMSE of dataset with reducing sparsity and right one is dataset with bigram. So, we decided not to limit our model to the traditional model and do more steps to improve the performance.

5.2.3 Average Model

To further improve the behavior of the mode, we used average model which was described in method section. In figure 4, red lines are SVM with averaging model. RMSE of $c = 0.01$ (start and plus line) is apparently higher than RMSE of $c = 0.05$ (circle and triangle line). So c= 0.05 are sued in further analysis. When $c = 0.05$, subset = 0.6 or 0.8 of training data in each iteration perform very similar. Also, after 20 steps, triangle and circle lines trend to be stable. So we decided that 20 iterations is a good place to stop to get the average model.

5.2.4 with BiGram

As the steps described in method part, we haved added bigram features and tried both the basci SVM model and the average SVM model. In table 3, RMSE of basic SVM with bigram was reduced to 0.968 from 0.970. In figure 4, average model of svm with bigram is shown in blue lines.

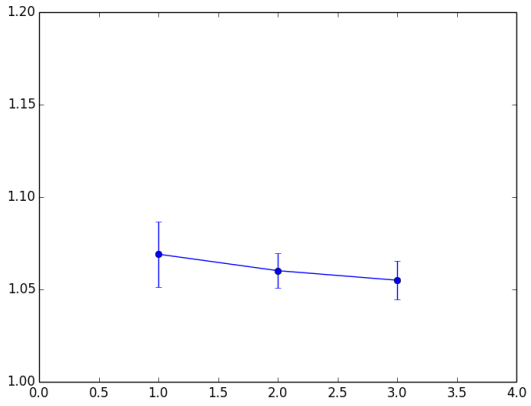

Figure 1: Naive Bayes with Different Dataset

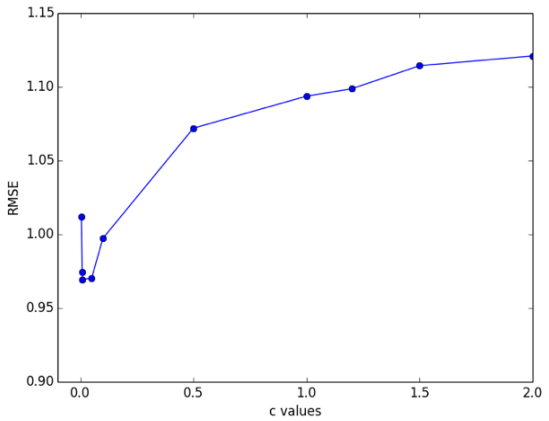

Figure 2: Cross Validation of SVM

Table 3: RMSE of SVM with Different Dataset

Dataset	RMSE (c=0.05)	sd
Original	0.98877	0.0098
W/ reduce sparsity	0.9925	0.0148
W/ BiGram	0.9911	0.0136

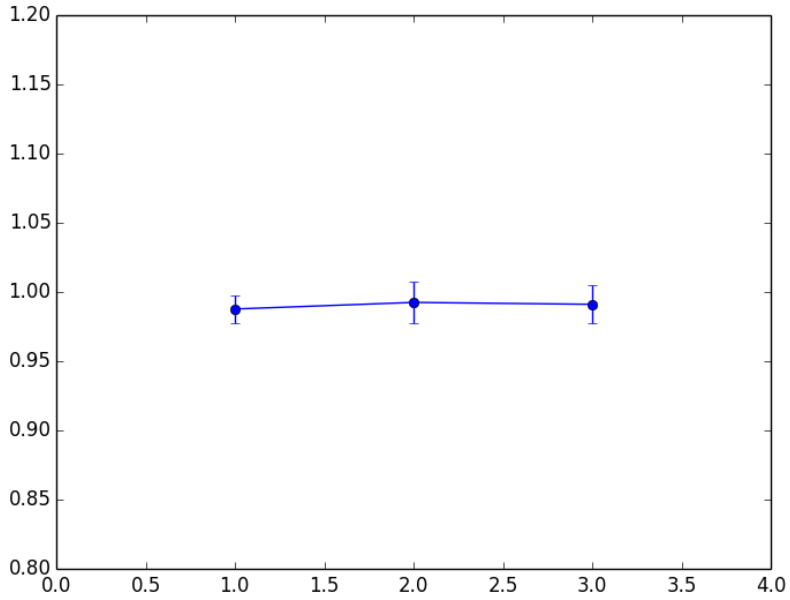

Figure 3: SVM with Different Dataset

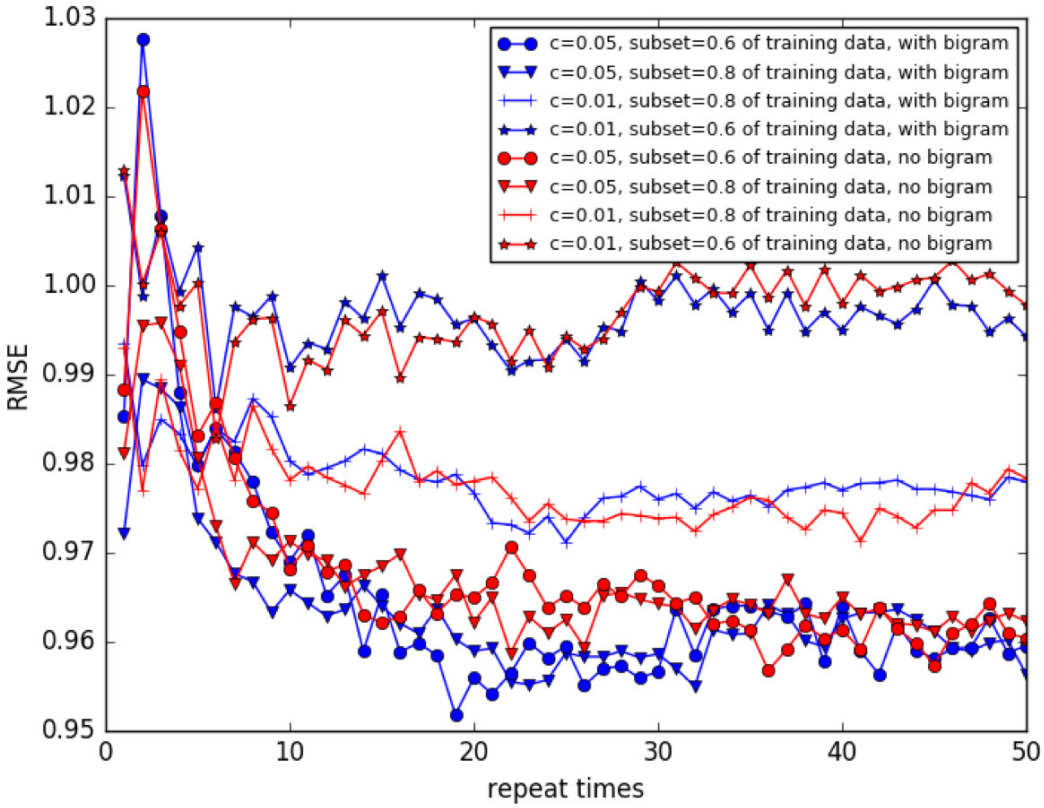

Figure 4: SVM with Average Model

Consistently, RMSE of svm with bigram ($c = 0.05$) is lower than svm with bigram ($c = 0.01$). When $c = 0.05$, not like figure 3, RMSE of svm with bigram (red lines) is obviously lower than svm without bigram (blue lines). When repeat time reaches 20, the RMSE is the lowest. The lowest RMSE reaches 0.9518. Combine all results in figure 4, we decide our model to be: average SVM with bigram feature; $c = 0.05$; 20 iterations, 0.6 of trainingset used in one iteration.

5.2.5 Model Repeatability

Since in figure 4, the results are from just one dataset. To make sure the model performs consistently, we selected svm ($c = 0.05$) average model (subset 0.6 of training data) to evaluate how bigram and the number of repeated times affect the model. Like the description in method, we did average SVM model with and without bigram using 5 different spltting of training and test data. In figure 5, the red line is without bigram and the blue line is with bigram. The circles are the mean RMSE of each repeat time of average model and the error bars are the standard deviation of RMSE for each repeat time. Similar to figure 4, mean RMSE reaches the lowest at around 20 repeated times and the model with bigram is performing better than that without bigram. Even though the error bars look not very small, the error bar at the start point is also not small. Thus, the value of RMSE highly depends on the dataset, but the overall trend is that the RMSE drops with more repeat times and 20 times are enough for getting the best accuracy.

To clearly make sure if bigram improves the model performance, we calculated the mean and standard deviation of (RMSE without BiGram - RMSE with BiGram) for each repeat time (figure 6). The redline is no change with BiGram. If the point is above the redline, it indicates that RMSE of model with BiGram is lower than RMSE model without BiGram. In figure 6, most of the points are above the redline, meaning that BiGram did improve the performance of the model and the error bars show that the improvement is stable.

To show that svm average model with 20 repeated times do improve the performance of the model compared to traditional svm model, figure 7 shows the change of the RMSE in 25 different splitting of the whole $25,000$ data. The y axis is the RMSE differences between 20 repeated time average model and the basic SVM model. The decrease of RMSE is at least 0.02. The highest decrease reaches 0.08. The average improvement is 0.044 (from an average RMSE of 0.992 to an average of 0.948) and the standard deviation is 0.014. There is no negative values, this indicates that the average model with 20 repeat times largely and stably improves the performance.

6. CONCLUSION

By removing meaningless stopwords combine words with same stems and delete the words that appeared really few times in the whole dataset, we successfully reduce the dimension from more than around $60,000$ to just around $6,000$, this should greatly reduce the computational complexity for most algorithms. Then, to make the model more accurate, we added more than $3,000$ new bigram features. Naive Bayes model benefit from these tricks and improves a little every time we modify the features. But the improvement is not obvious. Then the feature tricks combined together with the trick of average model gave very satisfied improvement of the basic SVM model, the RMSE of basic SVM model was reduced by 0.044 unit. Which means our strategy definitely works.

7. FUTURE WORK

For the current model we have for now, there's some small aspect that may potentially improve the accuracy by a little. For example: Normalization: for the Naive Bayes, it's not necessary. But for the SVM, it may perform better if we normalize the data. Another small aspect may be that we can extract more data point from the original huge dataset. More training data means lower chance of overfitting and higher chance of finding the optimal model. However, this will definitely be accomanied by longer training time. Which is a problem of trade-off.

Besides these small aspect, we think we should try some other models beyond the current ones we have. In all the models we have for now, only pure "text features" are used. Same thing happend to all previous research in this field to our knowledge. No other kinds of features have been studied. However, the star rating is really personal and subjective opinion. Even with very similar text reviews, star ratings can be really different. Just as figure 8 shows: these are two reviews selected from Yelp website, two users wrote about "Providence", a restaurant in LA area. Both users seem to be very pleased with their experience because they described it with multiple strongly positive words such as "perfection", "must go", "great treat", "tasted great", etc. However, the first reviewer gave five stars and the second user gave only three stars. So ratings prediction based on its reviews text alone is not enough. Here we discuss some features that will influence the ratings below.

7.1 Possbile Features that will Influence Star Ratings

7.1.1 User History

This information was included in the Yelp dataset. It gives the average star ratings that a particular user have given in the past. Just as described above, even with similar text reviews, some users personally tends to give higher star ratings while others tends to give lower star ratings. This really depends on personality and can be reflected in this users average ratings. So this feature should be included in the model to reflect this aspect.

7.1.2 Geometric Distance

Restaurants tastes and services are designed mostly to satisfy local people. As an example, one person travel from Texas to New York who dine in a restaurant in New York may find the taste and services very different from his "common sense". So even if he or she thinks the restaurant is high quality, it's still very possible that a lower star rating will be given due to this cultural difference. This effect should increase with increased value of the geometric distance between the location of the restaurant and the user's residential location. Yelp dataset has the city and state of every user where it create the account and the GPS location of every restaurant. Thus geometric distance can be calculated and should be used in the model.

7.1.3 Consumption Level

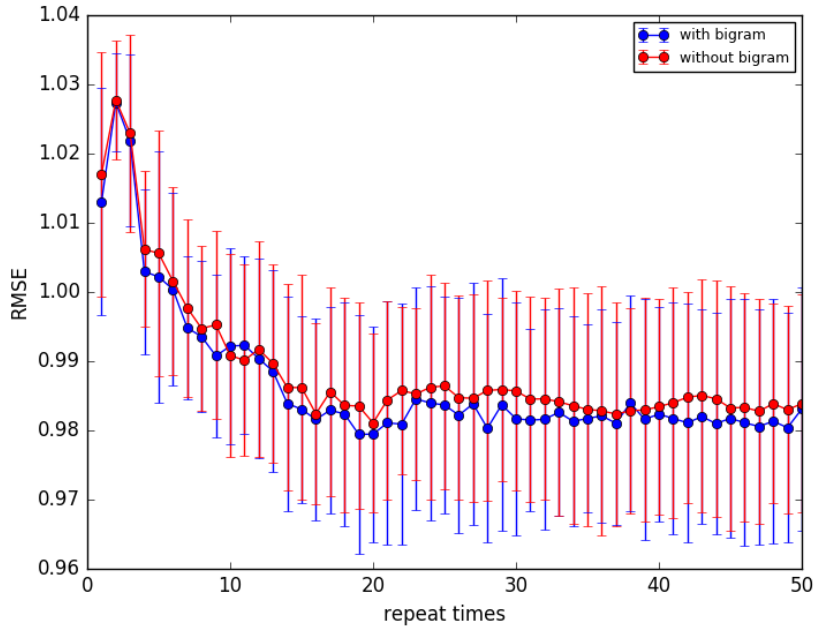

Figure 5: Average SVM Model with Bar

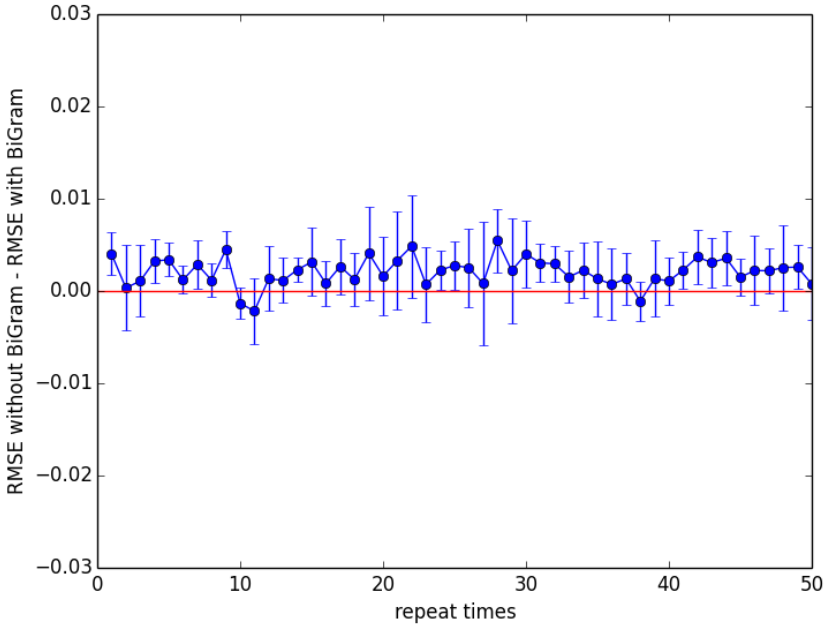

Figure 6: How BiGram Improves the Model's Performance

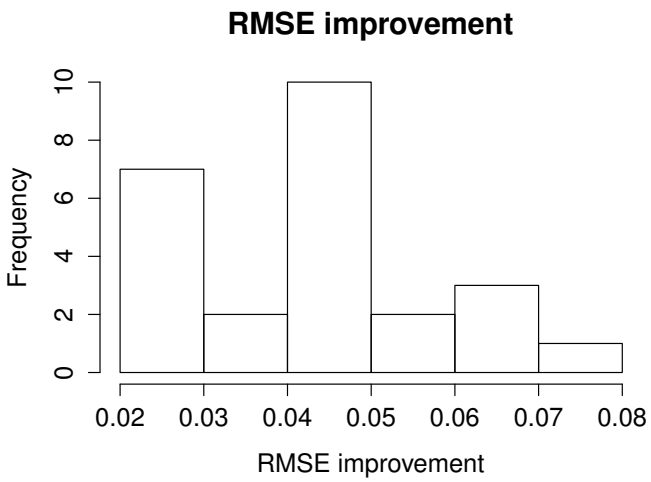

Figure 7: Improvement of RMSE with Average SVM Model

Figure 8: Example of Yelp Review

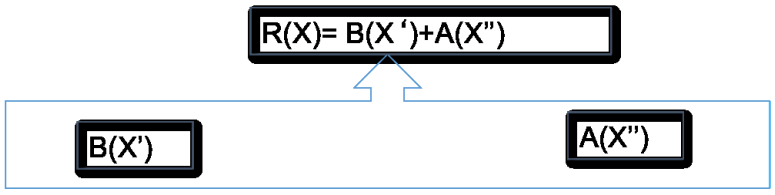

Figure 9: Proposed New Model

Yelp has a 4-level standard to illustrate the average spending of a restaurant, with "$", "$$", "$$$","$$$$" as symbols respectively. A "$$$$" might be a michelin 3 star restaurant with an average of $300 per spending per person. A "$$" might just be a very small ramen restaurant with just an average of $30 spending per person. Rationally speaking, we should expect 10 times tastier or more expensive food and better service from the michelin 3 star. However, the fact is that when people give star ratings, they seems to be too greatly satisfied with the overwhelmed serive and rare high quality food serviced by the high level restaurant thus to easily give out 5 stars. But when the problems comes to a normal restaurant, just with a little problem with the food or the service, four stars, three stars or even lower rating appeared. But these small defects for a cheap restaurant should be within the range of acceptance, but people tends to ignore this when give out ratings.

So this is a "must" to be taken into consideration when dealing with prediction.

7.2 Proposed Model

The idea of the new model we proposed came from a house price prediction model published by Yann et al. in 2007[11]. They build a mixed model which contains of two parts: the first parts is a linear part, the feature used is the basic house feature like number of bathrooms, number of bedrooms, w/ or w/o pool. This part gives a basic price of this house. And the second part of the model is a non-parametric model, the features include the GPS location of the house and the rating of the high school in the area. This part gives addition values of the house. These two parts are combined together to predict the final value of the house.

The feature structure we proposed is very similar to that of the Yann's housing price model. So we proposed a similar model. As shown in figure 9, this new model contains two parts. First part is function $B(x')$. The function used here is just what we have already done. A Naive Bayes model, SVM model or averaged SVM model. This part gives a basic rating b based on the features x' vector: which is just the "text features", modified "bag of words" and bigrams.

The second part is a nonparametric part, with x'' vector as features. These features should include user history, geometric distance and consumption level. This part gives additional adjustment a to the basic prediction.

For a sample X, let $N(X)$ denote the set of indices of K training samples that are closest to X (according to some pre-determined similarity measure) for some K. Then the output a of the function A(x") is given by

$$a = A(x'') = \sum_{j \in N(X)} Ker(X, X_j) \cdot a_j$$

The kernel function $Ker(X, X_j)$ is defined as

$$Ker(X, X_j) = \frac{e^{-q||X - X_j||^2}}{\sum_{k \in N(X) e^{-q||X - X_k||^2}}}$$

where q is a constant.

Then the result of these two parts are combined together to give the final prediction, which is $a + b$.

[11]S. Chopra, T. Thampy, J. Leahy, A. Caplin and Y. LeCun "Discovering the hidden structure of house prices with non-parametric latent manifold model", Proc. Knowledge Discovery in Databases (KDD'07), 2007

For the training process, use cross validation to split the training set into two parts and treat one part as training, one part as validation. We can fix first part, train the second part until lowest RMSE on validation set can be reached; then fix this second part, train the first part until lowest RMSE on validation set can be reached. Repeat this process until converge.

With this mixed algorithm new model, we think more accurate predictions can be got with the introduction of the proposed reasonable new features.