Review Rating Predictions on Yelp Dataset

Bo Zhang
b4zhang@ucsd.edu

Xuanyu Wu
xuw057@ucsd.edu

Abstract

Reviews data are everywhere on the Internet and a lot of data mining and text mining can be done to discover hidden information like user's sentiment. To predict ratings based on review texts and other users' opinions about this review, Logistic Regression, Ensemble Learning, Latent Factor Model and Bi-LSTM are tested and compared in terms of accuracy and Mean Squared Error (MSE). Using a subset of Yelp dataset, the best accuracy achieved is 0.66 and the best MSE is 0.82.

1.Dataset Introduction

The Yelp dataset we choose to use is a subset of Yelp's business, reviews, and users data. It was collected by the Yelp Dataset Challenge for people to analyze and discover what's hidden in the data. There are many datasets available which focus on different aspects of Yelp's business data, including check-ins, reviews, tips, and so on. For this assignment, we focus primarily on the review data.

The whole dataset contains millions of reviews data. Subject to the computing power limits, 50,000 reviews are randomly selected. For each record, there are 'review_id', 'user_id' and 'business_id' which uniquely identify each review, user, and business. Review texts are included in 'text' column, with a star rating ranging from 1 to 5 to illustrate the user's opinion to this business. Interestingly, the dataset also contains columns 'useful', 'funny' and 'cool' which describe the numbers of people who think the review is useful, funny or cool, which can potentially be helpful for the prediction.

1.1 Ratings

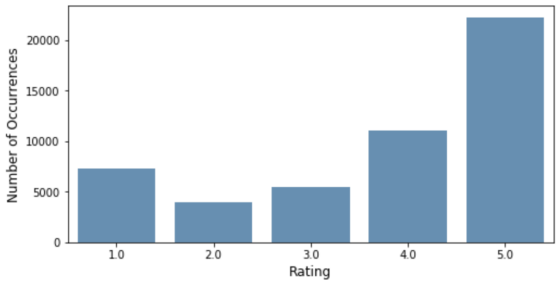

Figure 1.1: Rating Distribution

For randomly selected 50,000 reviews, 44.5% of them are 5-star and 22.12% are 4-star, so most of the reviews are really positive. 2-star reviews are the fewest, which makes sense because users tend to give reviews when they are really satisfied or unsatisfied, leading to a bowl-shaped distribution. The mean is 3.74 and median is 4.

1.2 Review Texts

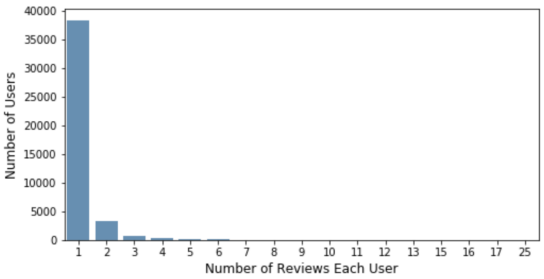

Figure 1.2.1: Distribution of Number of Reviews for Each User

76.7% of users in the dataset have only written one review and 6% of users have written

two reviews. Therefore, the history data of each user is limited, which might constrain the performance of models that rely on hidden attributes of users.

Figure 1.2.2: Word Cloud for Review Texts of 5-star Reviews (left) and 1-star Reviews (right)

After preprocessing the texts by lowercasing, removing punctuation and stopwords and stemming, two word clouds are generated based on review texts from 5-star and 1-star reviews. It can be seen that for 5-star reviews, positive words like 'good' and 'place' are prevalent, and for 1-star reviews, words like 'order', 'time' and 'go' appeared a lot of times, which are neutral words without sentiments. Therefore, words from 5-star reviews are more sentimental while 1-star reviews are more like telling stories of their user experience.

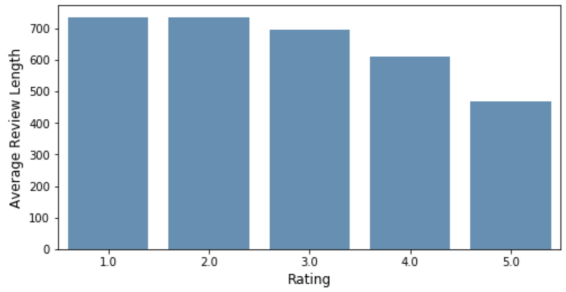

Figure 1.2.3: Average Review Length by Rating:

For reviews with different ratings, the average length of review texts differs a lot. 5-star reviews only have texts of average length 470, compared with an average of 736 for 1-star reviews. People tend to give short reviews if they are happy and long reviews otherwise. It can be explained by the fact that people like sharing their detailed unhappy experiences to express the frustration. This corresponds to the common words found in the word cloud for 1-star reviews.

1.3 Regression Plots

For each review in the dataset, we have "funny", "useful", "cool" features indicating the number of such tags received by each review. Linear regression plots of the data conditional on the number of stars each review gives are plotted to explore possible relations among the number of "funny", "useful", "cool" tags received given different stars from each review. For example, in Figure 1.3.1, given different star ratings, we generated the linear regression plot of the number of "useful" tags received by this review against the "cool" tags. In the figure, all slopes are positive, indicating that there is a positive relationship between the number of "useful" tags and "cool" tags, which is reasonable as they are both positive comments for a review. Most interestingly, we can see that the slope is steepest if the stars given are 1 and less if the stars given are 2. The slopes of star 3,4,5 are almost stacked over each other with even less steep slopes. Since all regression plots go through the origin point on the coordinate system, by incorporating the ratio of "useful" over "cool" (i.e. slope) as a new feature, we assume the model can perform better in distinguishing 1 star and 2 star ratings from others.

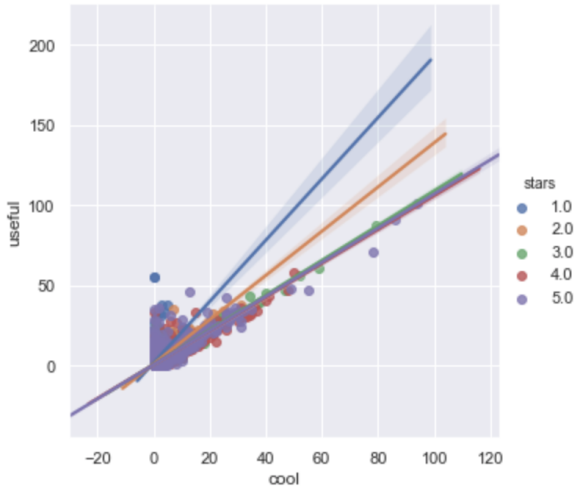

Figure 1.3.1 Regression Plot of "useful" vs "cool"

Similarly, we also perform the regression plot of the number of “funny” tags over “cool” tags in Figure 1.3.2. In the figure, we could see the general relation between the magnitude of the slope and the number of stars in the review. Generally, the higher the stars given, the smaller the slopes of “funny” over “cool”, except we have the slopes almost stacked up over each other for star 3 and star 5 reviews. Thus, the ratio of “funny” over “cool” (i.e. slope) is expected to provide extra information in distinguishing the stars of each review.

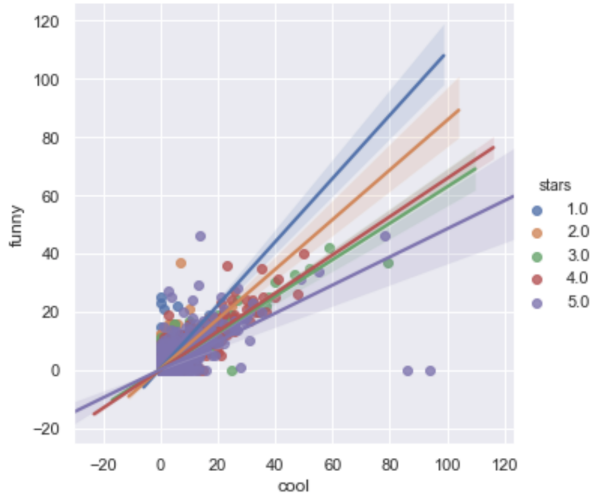

Figure 1.3.2 Regression Plot of “funny” vs “cool”

We incorporated these two new features via stacking in Section 3.3.

2. Prediction Task

We aim to predict the rating for each review based on features present in the dataset, including reviews texts, number of votes for ‘useful’, ‘cool’ and ‘funny’, and user id and business id. This task can be used in multiple settings. First of all, the prediction gives a uniform standard for ratings based on review texts while users have different standards when giving ratings. Due to different attributes of users, they might rate the item differently even though they have the same level of satisfaction. Our prediction task can help generate ratings from the review texts, which is a more reliable representation of user experience than the rating star given by the user. Also, fake online reviews can be detected when the rating star does not match our prediction from its review text.

2.1 Evaluation Metrics

There are two different ways to evaluate the quality of our predictions, which are prediction accuracy and Mean Squared Error (MSE). These two metrics focus on different aspects of the prediction. Accuracy measures how the model performs in terms of predicting exactly the true ratings and treats ratings as categorical values. While MSE measures how close the predictions and true ratings are and treats ratings as continuous values.

The baselines we choose are naive and straightforward. To maximize accuracy, we predict the mode of ratings in the dataset, which is 5. To minimize the Mean Squared Error, the average rating of the dataset is used because it is a central value of all ratings. The dataset is randomly splitted into training set, validation set, and test set with a ratio of 8:1:1. After training models on the training set, we evaluate them on the validation set to find the best hyperparameters and lastly report the scores on the test set.

2.2 Feature Engineering

Multiple preprocessing techniques are used for the prediction task in different models. To incorporate review texts into models, we clean the texts by lowercasing, removing punctuation and stopwords and stemming. The most common words are extracted as a dictionary and then for each text, we calculate the tf-idf of each common word in the text to convert the text to a vector.

Besides baseline, the first model is Logistic Regression, which is a common classifier that can be applied for multi-class prediction. After preprocessing the text data, we performed bag-of-words with tf-idf score for the most frequent 5,000 words of the entire dataset to generate a highly sparse matrix to be the feature representation of the text data. And such sparse

matrix is then treated as input for training the Logistic Regression model.

The second model is a stacking of Logistic Regressions in order to incorporate derived dense features along with the sparse matrix representation of texts. The two new dense features are “funny”/ “cool” and “useful”/ “cool” ratios as introduced in Section 1.3. If any one of “funny”, “cool” or “useful” features are 0, rendering the ratios invalid, we replace it with the mean value. Also, since Logistic Regression in its heart is a Linear Model, it assumes numeric features to center around 0 and have a variance in the same order. Thus, we perform standard scaling over the numeric values in case an uncommonly large numeric feature dominating the L2 regularizer function in Logistic Regression.

The third model we propose is the Latent Factor Model. It can be used because for each review, user id and business id are indicated. Latent factors for each user and business can therefore be learned from the dataset. The model learns a global constant, a bias term and a vector representation for each user and each business. To build the model, all unique users and businesses are stored so that we can iterate over the training set and change the parameters for each user and business.

The last complicated model is Bi-LSTM, which is a special kind of Recurrent Neural Networks. This model is appropriate because review texts are essential in this task and Bi-LSTM deals with natural language pretty well. We preprocess the text and find the embeddings for each text by incorporating word vectors given by a state-of-the-art model GloVe. The vector therefore contains the features for the text it is representing.

3. Model

3.1 Baseline

The very naive baseline model that we choose to use is the mode of stars in training dataset for accuracy metric and the mean of stars in training dataset for Mean Squared Error (MSE) metric. Since the mode of stars, which is 5, is the most frequent rating among all reviews in the dataset, the accuracy ($\frac{\#\ of\ times\ predicted\ correctly}{total\ \#\ of\ predictions\ made}$) will be higher than predicting any other star rating. The result of this baseline model for accuracy metric on the test dataset is 45.3%. By using the mean as the baseline for MSE metric, we are setting the bottomline with the MSE ($\frac{1}{n} * \sum_{i=1}^{n} (y_i - \hat{y})^2$) equal to Variance ($\frac{1}{n} * \sum_{i=1}^{n} (y_i - \bar{y})^2$). By predicting the mean value, we have the MSE of 2.064.

3.2 Logistic Regression

The reason for using Logistic Regression is that it can predict the probability of the class rather than only gives the class prediction. This can be helpful when we apply the stacking strategy in Section 3.3 to incorporate other features along with the sparse matrix representation of text data. More importantly, using one-vs-rest schemes in Logistic Regression can resolve the problem of “multi-class” in this prediction task as we have 5 classes. However, imbalance issues may arise by one-vs-rest scheme. Using this scheme, we train 5 binary Logistic Regression to determine whether the new example belongs to the class or not and predict the class with highest confidence. When the binary classifier is “5 stars” vs “the rest”, the dataset would be balanced as about half of the review data have 5 stars. However, if the binary classifier is “2 stars” vs “the rest”, it would be a highly imbalanced dataset as there are only 7.91% reviews that have 2 stars rating. Another drawback of this method is its low scalability. Since the method requires a lot of computing power in generating the bag-of-word feature representation of text data in predicting, the model may not be able to cope well under expanded load of dataset. Also, in order to prevent overfitting, we perform cross validation with different regularization strength in Logistic Regression.

The resulting MSE values are shown in Figure 3.2.1.

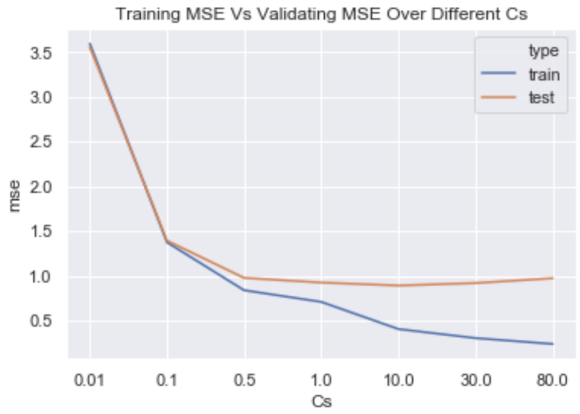

Figure 3.2.1 Training and validating MSE values over different penalizing strength. The larger the C, the weaker the penalizing strength.

As we can see from Figure 3.2.1, the performance of Logistic Regression is the best on the validation dataset if we have C equal to 10. With this configuration, the performance on test dataset is 0.8832 in MSE and 0.634 in accuracy.

3.3 Ensembling Via Stacking

As we've introduced in Section 1.3, "funny"/"cool" and "useful"/"cool" ratios can provide extra information for our prediction task. Thus, we seek ensembling via stacking as our method of incorporating these new features along with the sparse matrix representation of the text data. In this method, we first use the Logistic Regression in Section 3.2 to generate probabilities of belonging to each class with only sparse matrix representation of text data. Then, we combine the output with our new features to train another Logistic Regression model. To prevent overfitting, we perform cross validation with different regularization strength in the same way as Section 3.2. The resulting MSE values are shown in Table 3.3.1.

Cs	Training MSE	Validation MSE
0.01	0.4148	0.8916
0.1	0.3758	0.8774
0.5	0.3731	0.8706
1	0.3735	0.8710
10	0.3725	0.8710
30	0.3728	0.8692
80	0.3728	0.8692

Table 3.2.2 Training MSE and Validation MSE with Ensembling Via Stacking over different penalizing constant. The larger the C, the weaker the penalizing strength.

As we can see from Figure 3.3.1, the performance is the best on the validation dataset if we have C equal to 30. With this configuration, the performance on test dataset is 0.8258 in MSE and 63.58% in accuracy. We can see that the improvement in MSE is really decent from 0.8832 to 0.8258. However, the improvement in Accuracy is relatively small that it only increases from 0.634 to 0.636. One reason for this limited improvement is that a lot of (75.4%) the reviews have received either 0 "funny" tags, 0 "cool" tags or 0 "useful" tags, rendering the derived ratio features invalid.

3.4 Latent Factor Model

The Latent Factor Model is implemented from scratch and optimized using L-BFGS-B algorithm provided in scipy package. After testing different hyperparameters, the best hyperparameters found are vector size of 100, regularizer lambda of 0.0001, and learning rate of 0.01. The resulting MSE is 1.82, which is not a great improvement from the baseline. The problem might be that the number of reviews for each user is really small. As shown in Section 1.2, 76.7% of users have written only one review. Therefore, there is not much data for the model to learn vector representation of each user. This idea is verified when we resample from the entire Yelp dataset by only selecting reviews whose users have written at least 10 reviews, and the MSE decreases to 1.04. Another issue is that the model tends to overfit the training set, when the regularizer is 0.0001, training MSE is 0.83 while testing MSE is 1.82, and when the regularizer is

0.00001, training MSE is 0.1 while testing MSE is 1.92. Scalability is another problem because of the iterative nature of the training process, which makes it expensive for large-scale datasets.

3.5 GloVe and Bi-LSTM

The GloVe model learns the word vectors by looking at the context each word appears and is trained on Wikipedia and Common Crawl [5] so that the vector encodes some form of meaning of the word. The Long Short Term Memory networks preserves information from inputs that have already passed through it using the hidden state. In addition, Bidirectional LSTM runs inputs in two ways, one from the past and one from the future. Therefore, the sentiment of reviews can be inferred from context information and ratings can be predicted. The model is not prone to overfit as the validation accuracy does not decrease even with a large amount of data. So the training process can continue until the validation accuracy stops increasing. However, it might take much longer to converge as the dataset size grows because the computational complexity of training a neural network is high.

We tested this model using Keras and chose adam optimizer to optimize it. An embedding layer is followed by a Bi-LSTM layer with 0.2 dropout rate and 100 neurons. After that, a dense layer with ReLU activation function is applied and lastly, a output layer with softmax activation function is used to get the final prediction. After training on 15 epochs, the prediction accuracy is 0.657.

4. Related Literature

In 2006, Netflix Inc. announced the Netflix Grand Prize competition to predict the rating that the user would give to the film. In the end, the winning team [1, 2] largely introduced the latent factor based method in collaborative filtering in order to reach the matrix factorization of users and items. They studied a similar prediction task as ours with a method heavily relying on hidden factors of users and items. Even though we use a different dataset from Netflix, Yelp and Netflix datasets share similar characteristics that pose challenges to the prediction task. In both datasets, large portions of users only appeared once, leaving it limited space for studying the users' history. Also, both datasets have highly nonrandom patterns. That is, the amount of observed data for some active user can be even three orders of magnitude more than those inactive users who appeared less frequently in the dataset.

Other recent works related to review rating predictions largely relate to sentiment analysis by extracting sentiment features from the review text. The method in [3] tackles this problem by introducing "bag-of-opinions" representation for feature extraction where opinion is defined to a combination of a root word, modifier words and negation words. The review's rating is then predicted by aggregating the scores of all opinions in the review and combining it with a domain-dependent unigram model.

Additionally, [4] studied the same Yelp dataset by treating it as a multi-class classification problem. In [4], it experimented sixteen prediction models by combining four feature extraction methods with four machine learning algorithms, which is a very thorough exploration. In our method, we use tf-idf score in bag-of-words for feature representation, which is absent in the methods used in [4]. However, the best model in [4] is a "Unigram" & "Bigram" feature representation trained on Logistic Regression with 67% Accuracy and 0.78 MSE, which is slightly better than our "tf-idf" scheme. Also, due to computing power limit, we are unable to train on the entire dataset like what [4] did.

5. Results and Conclusions

Based on several models tested, on test dataset, ensemble learning through stacking achieves the best MSE of 0.83 and decent prediction accuracy of 0.64 while Bi-LSTM model achieves the highest accuracy of 0.67. These are huge improvements from the baseline

model, which has MSE of 2.06 and accuracy of 0.45.

To further improve our model in Section 3.3, we can experiment with the "Unigram" & "Bigram" rather than "tf-idf" score representation as discussed in [4] in the future.

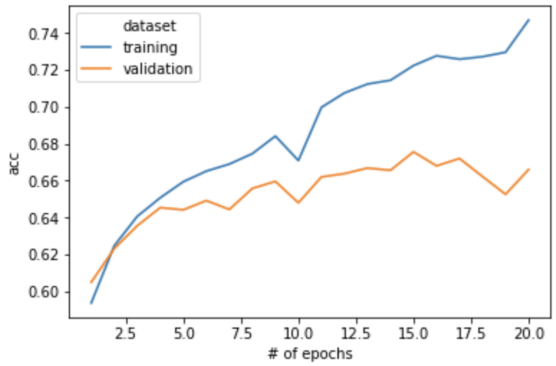

Figure 5.1: Training and Validation Accuracy for Different Passes of Epochs

Passing in 20 epochs of training data and recording the training and validation accuracies after each epoch, we can see that the Bi-LSTM model keeps increasing validation accuracy with more data for the first 15 epochs and then overfit the training set after 15 epochs. The highest validation accuracy of 0.676 occurs after 15 epochs, with test accuracy of 0.667. We can see from the plot that even when the model overfits on the training set, its performance on validation accuracy does not decrease but fluctuates around a certain level.

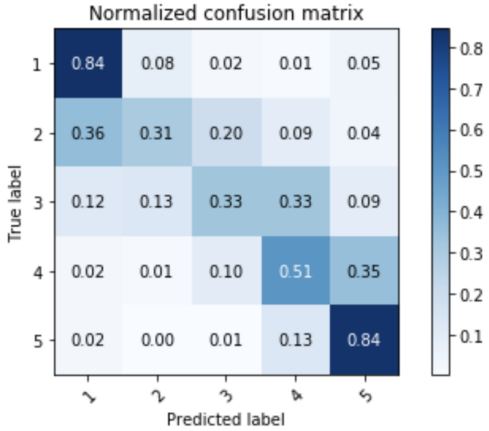

Figure 5.2: Normalized Confusion Matrix for Bi-LSTM

From the normalized confusion matrix for predictions based on Bi-LSTM, we can see the prediction accuracy for 1-star and 5-star reviews are the highest, which can be explained by their extreme sentiments. Many 2-star reviews are predicted as 1-star and many 4-star reviews are predicted as 5-star. This is reasonable because sometimes users do not give extreme ratings because they are preservative, even though they've expressed their polarized experience in the review text.

In conclusion, among these models, the Latent Factor Model does not perform well in terms of both accuracy and MSE. The most likely reason is the limited information for each user due to the fact that 76.7% of the users have only one review. Logistic Regression does a decent job and Ensemble Learning via Stacking lowers the MSE to the optimal result. Bi-LSTM achieves the highest prediction accuracy due to its informative representation of words and texts.

Citations

[1].
R. Bell, and Y. Koren. Lessons from the Netflix Prize Challenge. ACM SIGKDD Explorations Newsletter, vol. 9, no. 2, pp. 75. 2007.

[2].
R. Bell, Y. Koren and C. Volinsky. The BellKor Solution to the Netflix Prize. 2009.

[3].
L. Qu, G. Ifrim, and G. Weikum. The bag-of-opinions method for review rating prediction from sparse text patterns. In Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010),.pages 913–921. 2010

[4].
Nabiha Asghar. 2016. Yelp Dataset Challenge: Review Rating Prediction. CoRR abs/1605.05362, 2016.

[5].
Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014, GloVe: Global Vectors for Word Representation.

[6].
A. Ahmed, M. Aly, J. Gonzalez, S. Narayanamurthy, and A. Smola. Scalable inference in latent variable models. In Web Search and Data Mining, New York, NY, USA, 2012.