Comparison of Sentiment Analysis Based Models and Latent Factor Models for Rating Prediction of Yelp Reviews

Alexander Marinov, Rongze Yuan, Vincent Tsui

Abstract—**Review analysis and predicting has always been an important aspect of building recommendation systems. We made an attempt to build a model to predict the rating of a review based on its text content, reviewer, and business. This paper compares the performance of several models and explores the most useful features for prediction.**

I. Introduction

YELP is a relatively recent corporation that began as a startup in San Francisco in 2004 [6]. It hosts Yelp.com and the Yelp mobile app, which is a free-to-use application that publishes crowd-sourced reviews about businesses. In recent years, Yelp has also expanded its business to provide online reservation service. Yelp was a pioneer in its own right, capitalizing on being one of the first to provide a platform for everyday people to rate and share reviews about local businesses, back when reviews were largely dominated by critics. Yelp has rapidly since its creation, and in recent years, has even expanded overseas to Europe and Asia. Yelp has publicly made available a portion of its dataset for educational purposes, and it is the dataset we will be working with for this assignment. In each review, the customer gives a rating ranging from one to five stars and a text review that summarizes their experience with the business. For review platforms like Yelp, recommending businesses to a user's liking makes up a significant part of their business model. With such a large user base, analyzing each user's preference and building an accurate recommending system can be both challenging and profitable. For this assignment, the objective is trying to predict the star rating a user will give to a business based on the review. The predictive models we chose for this dataset tries to make the best possible prediction, evaluated by the metric of root mean squared error (RMSE).

II. Dataset

The dataset we chose to use is from Yelps dataset challenge 2018 [7]. It is a subset of the entire Yelp dataset made publicly available for educational purposes. We obtained our dataset straight from the yelp website at https://www.yelp.com/dataset. It contains information about local businesses in over 10 metropolitan areas spanning at least 4 countries. We chose this dataset because of has very detailed information about users and businesses. Another advantage is that Yelp reviews are much more straightforward compared to our original choice of Steam review dataset, which contains a lot of subtle or community-specific language and short reviews and thus significantly increased the difficulty of sentiment analysis. The Yelp dataset contains six files, containing information about businesses, users, reviews, photos, checkins as well as tips. We mainly worked with three of the files containing business, user and review information.

The main fields of the json files we used are shown as following:

yelp_academic_dataset_business.json

- **Business** - The hashed business ID
- **Location** - Data such as neighborhood's name, the exact street address, and the exact geographical location are given

yelp_academic_dataset_user.json

- **User** - The hashed user ID
- **Review Count** - Number of reviews written

yelp_academic_dataset_review.json

- **Review** - The hashed review ID
- **User** - The hashed user ID of writer
- **Business** - The business ID being reviewed
- **Stars** - The star rating given
- **Text** - The actual review

III. Exploratory Analysis

Before diving into the main predictive task, we did some exploratory analysis on the dataset. The dataset files are all of considerably large size because they contain data dating back to 2004, when the Yelp website was first launched.

The followings are some simple statistics of the dataset:
Total number of users: 1 518 169
Total number of businesses: 188 593
Total number of reviews: 5 996 993

From the statistics collected on the number of ratings in the dataset, we can observe that there is a skew on both extreme sides. The majority of reviews are 5-star and 4-star ratings, followed by 1-star ratings. On the other hand, the number of 2-star reviews is the lowest. This is most likely because people tend to review things only if they feel very strongly about them whether that is strongly positive or strongly negative. Given the skew of the data, a good approach would be to take as many reviews from the 1-star, 3-star, 4-star, and 5-star ratings as the 2-star ones so that we create an equal representation for the model to learn to predict all 5.

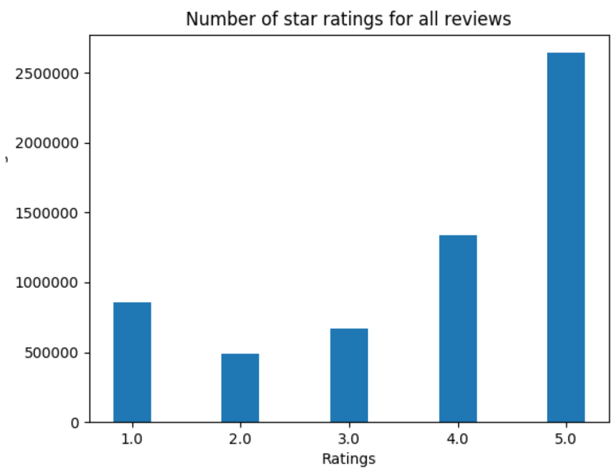

Fig. 1.

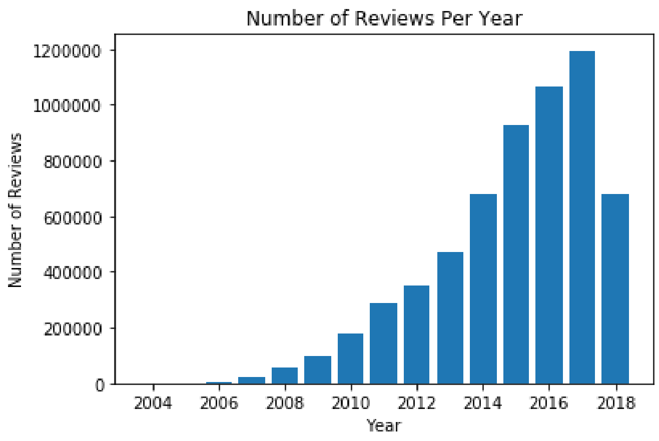

Fig. 2.

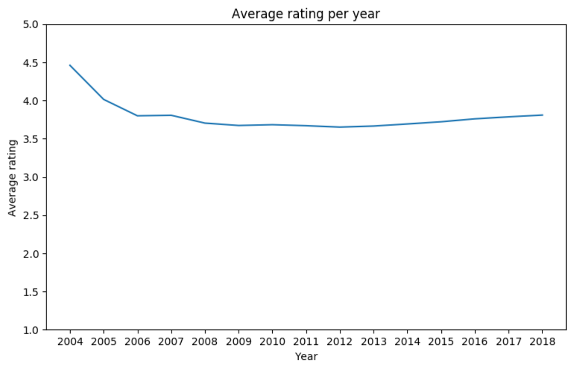

Fig. 3.

In Figure 3, we observe the average ratings on businesses and how they change each year from the first reviews in 2004, to the latest reviews in 2018. At first glance, it might seem that businesses are a lot higher rated on average in the earlier years, however this is because there are very few reviews from 2004-2008 in comparison to later years. In particular, 2004 has only 13 reviews, and 2004-2008 have less reviews together than just 2009 alone. This explains the spike in the ratings in those earlier years, which should not be used as an assumption for a higher rating bias in a generalized model, since it is just a side effect of the sparse data. On the other hand, there is a trend for the average rating to slightly increase from 2014 to 2018, which is a far more representative portion of the dataset, since each of these years contains a large amount that is a roughly similar size each year. Thus we might infer that people in the later years tend to rate things higher, and possibly add a small bias reflecting that in the model.

IV. Predictive Task

The predictive task that we aim to study on this dataset is to predict the rating a user will give to a business based on different features that we select in various models. Since the dataset was too big to practically work with, we subsampled the dataset in various ways that show how different models perform better in different situations. In the end, we evaluated the models on the same subsample of the dataset so that we can objectively compare their performance.

The main measure we used to evaluate the performance of the different models was RMSE. While rating is a discrete variable in this particular dataset, as the users are allowed to only submit an integer rating between 1 and 5, we evaluate the performance of the models as a continuous variable through RMSE, since in the context of recommender systems we care more about the overall feeling of the user towards the business so that we can make recommendations preferable to the user. In that context, it doesnt matter that we predict the exact rating the user gives, but rather how close we predict to that rating. Thats why we believe RMSE was an appropriate measure to evaluate the performance of the models on.

A. Baseline Model

As a baseline to evaluate more complicated models, we used a simple SVM system that predicts rating based on the number of appearance of popular words. We used this SVM system as a trivial predictor, since SVMs typically perform better on predicting and distinguishing clearly separable categories of data. Thus we suspected that such a model would not do as well on predicting continuous ordinal data, such as ratings, as other models.

The two models we tested against the baseline were a linear regression model and a latent-factor model. For the linear regression, we used also used word count of popular words as features, which is the same as the SVM system. For the latent-factor model, we experimented with 50, 10 and 5 number of factors, and added user bias, business bias, and a global mean on top. The methodology of the three models is described in more detail in the following section.

V. Methodology

A. *Linear SVM Model (Baseline)*

The linear SVM model is designed based on the assumption that certain words are more related to a particular rating than others due to their attached sentiment. In other words, reviews that contain words with a positive connotation is more likely to be a high rating and vice versa. Thus, a linear SVM system was built based on the words included in the review in an attempt to classify the reviews to the right categories, which are specific ratings in this case.

We first extracted the 2000 most frequently used words from 100,000 randomly selected reviews, where all review texts are lowercase and stemmed before counting word appearances. We then use the word counts of the 2000 popular words in a review as features and train the five SVM models using the same 100,000 reviews. In actual prediction, each SVM model produces a confidence score for a specific rating and the rating with the highest confidence score is chosen to be the final prediction of the SVM system.

Since linear SVM takes a long time to train, we had to limit the number of max iterations (which is default to be 1000) each linear model could run. For the same reason, we were also unable to train the SVM system with different penalty parameter values. Therefore, we trained the SVM system with penalty parameter c = 1.0 with four different maximum iteration values: 30, 50, 100, and 500. To produce the final RMSE value, we chose the system that generated the lowest RMSE on a validation set of size 25,000, and ran that system on a test set of the same size as the validation set. All the results are listed in the results section.

B. *Linear Regression Model*

Similar to the linear SVM model, the linear regression model explored the predictive task from a text sentiment analysis standpoint. The linear regression model used the same features as the linear SVM models: the word count in the review text of the 2000 most frequently appeared word. In fact, the linear regression model is simpler in implementation than the SVM model, as the same features were used, but only one model was trained instead of a system of five models.

After the training on 100,000 reviews is done, the linear regression model was ran on a validation set. We trained five linear regression models with regularization strength alpha ranging from 0.1 to 100. After that, the regression model with the lowest RMSE on the validation set was chosen and ran on a test set in order to obtain the final RMSE value.

C. *Latent-Factor Model*

Based on the exploratory analysis done earlier, we decided a latent-factor model would be the most appropriate for the task for this dataset. In particular, the vast number of users with a lot of reviews motivated this, as latent-factor models tend to do better when more information about previous ratings and reviews of users is available.

To verify this hypothesis, we tested the latent-factor model on three subsamples of the dataset: first one containing all the reviews by users who have written 50+ reviews, second one containing all the reviews by users who have written 150+ reviews, and a third one containing all the reviews by users who have written 300+ reviews. The size of the three subsamples was 2 124 569 reviews, 1 090 508 reviews and 597 435 reviews respectively. In order to obtain the rating matrix to be optimized, we first did one pass through all users and stored the user ID of all the users that have above a certain number of reviews written, then we iterated through all the reviews and stored the user ID, business ID, and rating of a review only for the reviews where the user ID matched one of the users IDs that were stored from the procedure described above.

The particular matrix factorization model we used, was the SVD algorithm with added biases as proposed by Koren, Bell and Volinsky in their paper Matrix Factorization Techniques For Recommender Systems [3]. In particular, the exact objective we optimized is described by:

$$r_{ui} = \mu + b_i + b_u + q_i^T p_u$$

Where μ represents the global offset, b_i and b_u represent the business and user biases, and q_i^T and p_u represent the learnt latent features for the businesses and users respectively. The optimization we used for the objective function was stochastic gradient descent. For each of the three subsamples of the data, we ran the model with 50 factors, 10 factors and 5 factors, and with 10, 20 and 40 iterations of gradient descent on each. A learning rate of 0.005 and a regularization of 0.02 were used for all the runs. To tune the optimal parameters, we split each of the subsamples into 80% training set and 20% validation set. While the reviews for training and validation that were picked were random, this was done with a seed so that the results can be objectively compared over runs. The best parameters were then used to test the RMSE on the same test set as the SVM system and the linear regression model, so that we could quantitatively compare the performance of each on a random sample of the dataset.

VI. Literature

We obtained our dataset from the official yelp dataset challenge website for 2018. We found similar datasets available on Kaggle for previous years' dataset challenge, which have almost the same format, but with different subsets of data. The dataset we are working with is also very similar to the Amazon dataset and the Beer dataset provided from earlier in the course, with item attributes, text reviews, and rating scores. Since the dataset we used is part of a well-known dataset challenge, there are many other projects done by other students in various educational institutes using the same dataset. While these projects span across a wide range of different interesting objectives, many of them have done something similar to our objective to predict the star rating a user would give to a business.

Many of the other projects included a Naive Bayes model as one of their models which they tested. While Naive Bayes is a simple and useful model, we decided not to include a model for it because we believed it would perform quite poorly on predicting continuous ordinal data, such as ratings. Also, since we decided to use RMSE as our performance measurement and not accuracy, the integer difference between ratings would result in a large RMSE. Our intuitions were correct based on another projects results. Indeed, as seen in [5], the Naive Bayes model was the worst performing model in terms of test error, and in [1], the Naive Bayes model was the second worst performing model, ranking only above perceptrons.

Many projects received decent results with logistic regression, but with very different feature vectors. In [5] and [4], the feature vector was created by extracting concrete features such as number of fans, reviews, average rating, etc. from the user, business, and review data in the dataset. In [1], the feature vector was created by converting the review text into features by association with monograms, bigrams, etc. These different methods achieved varying degrees of success, but the one thing in common is that the models using these methods had relatively high performances in each respective project.

Matrix factorization models such as the more developed ones described in Koren, Bell and Volinskys paper Matrix Factorization Techniques For Recommender Systems [3], include implicit features and temporal dynamics, which can help give more accurate predictions than just the global average for new users that havent explicitly rated anything before. Other state-of-the-art models in this field employ deep learning methods along with collaborative filtering methods such as matrix factorization, to give even more accurate predictions [2]. Both of these show promising results for datasets which not only include users that we have a lot of information on, but varied datasets which include users we dont have a lot of explicit information about. Thats something that our latent-factor model struggled with.

Our own conclusions do coincide with the conclusions from existing works. While latent factor model is good for robust datasets with a lot of existing information, it performs quite poorly on varied datasets. In comparison, the well-established logistic regression method performs quite well for rating prediction, as shown in its relatively good performance compared with other models in our own project as well as other existing works.

VII. Results

A. *Linear SVM (Baseline)*

The SVM was trained on 100 000 randomly sampled reviews, and validated on another 25 000 randomly sampled reviews. The results on the initial runs of the linear SVM are outlined below. Due to time constraints, we were only able to run the SVM model on one regularization coefficient C=1. The best RMSE is bolded.

TABLE I
Linear SVM (Baseline)

Max Iterations	RMSE on Training Set	RMSE on Validation Set
30	1.4986	1.5170
50	1.3208	**1.3665**
100	1.4765	1.5101
500	1.5107	1.5372

B. *Latent-factor model*

The latent-factor model was ran on three different subsamples of the data set based on the review count of users as described in the methodology section. The results of these are outlined below with the best RMSE from each sample bolded. The final RMSE on the the same training and validation set as the other two models is in the final results table at the bottom of the section.

Subsample 1: (All reviews by users who have 50 or more reviews posted)

Factors	SGD Iterations	RMSE
50	10	1.0707
50	20	1.0697
50	40	1.1014
10	10	1.0682
10	20	1.0628
10	40	1.0828
5	10	1.0679
5	20	**1.0616**
5	40	1.0742

Subsample 2: (All reviews by users who have 150 or more reviews posted)

Factors	SGD Iterations	RMSE
50	10	1.0006
50	20	0.9991
50	40	1.0272
10	10	0.9980
10	20	0.9929
10	40	1.0099
5	10	0.9977
5	20	**0.9919**
5	40	1.0030

Subsample 3: (All reviews by users who have 300 or more reviews posted)

Factors	SGD Iterations	RMSE
50	10	0.9593
50	20	0.9570
50	40	0.9824
10	10	0.9569
10	20	0.9515
10	40	0.9653
5	10	0.9567
5	20	**0.9508**
5	40	0.9596

For all three subsamples, we found that 5 factors and 20 iterations of stochastic gradient descent yielded the best results on the validation set. This is probably due to the relatively low L2 regularization penalty and the dense subsamples, which is causing the addition of more factors to overfit on the training set. Thus a relatively low number of factors acts as a natural L1 regularization by taking into account only a few factors with higher importance.

C. *Linear Regression model*

The initial runs on the linear regression model are outlined below. The model was trained on the same random sample of 100 000 reviews as the SVM and latent-factor models, and validated on the same random sample of 25 000 reviews as well.

Regularization Strength	Training Set RMSE	Validation Set RMSE
.01	1.0008	1.0417
.1	1.0008	1.0417
1.0	1.0008	1.0416
10	1.0008	1.0412
100	1.0015	**1.0381**

The final results on the same training and validation set ran on the three models is outline as above. The RMSE is ran on a test set of 25,000 randomly selected reviews.

SVM (Baseline)	1.3602
Linear Regression	**1.0367**
Latent-factor	1.4629

VIII. Conclusion

In conclusion, the result produced by the linear SVM system was not entirely trivial. Upon close examination of the predictions, it succeeded to determine the sentiment of the review text but it was unable to improve its RMSE. The relatively high RMSE could be due to the reason that certain words had a strong influence on the prediction. For example, if a 2 or 3-star review contains the word suck, the SVM system will likely be most confident in predicting 1 star. In other words, the SVM model was able to predict the general tone of the review, but it is unable to give an accurate numerical prediction or it is likely to be biased towards the extreme.

Linear regression model used the same features as the SVM system but produced much better results due to its nature as a regressor, which is much more suitable to make a numerical prediction on continuous data in contrast to SVM, which is a classifier, suitable for categorical prediction.

There are some ways that can potentially improve RMSE on the linear models, including both the SVM and the regression models. For example, increasing the number of features, e.g. using more words from the dictionary, is likely to increase the accuracy. Using bigram instead of unigrams can be another way to improve accuracy as a bigram contains reveals some context of the word. In short, there is a lot of potential in exploiting the text for rating prediction.

On the other hand, the latent factor model explore the user preference to a business without using any features. Our results show that when we have enough information on users and businesses from previous reviews, the latent-factor model outperforms all other models. We can see that when we sample reviews with users on which we have more information, we get increasingly better RMSE on the validation set. For the set of reviews, where each user had 300 or more reviews written, we were able to achieve a RMSE of 0.9508. However, in the case when we have a small dataset such as the one we trained and tested all the three models on in the final results, the latent-factor model performs a lot more poorly, and is beaten even by the baseline SVM. This is a good example of the well-known cold start problem that latent-factor models face. For users and businesses that we have no previous information on, our model can only predict the global mean. It is likely that for such a small sample of 100 000 reviews we did not have enough information on a lot of the users and businesses when it was tested. These results werent too far of from what we expected, since we implemented a very early stage latent-factor model. New and improved state-of-the-art have found various ways to combat the cold start problem for more accurate rating predictions than just the global average rating when it comes to predicting ratings for new users who havent provided any explicit feedback yet. This is further discussed in the Literature section.

References

[1] Nabiha Asgharm. *Yelp Dataset Challenge: Review Rating Prediction.* https://arxiv.org/pdf/1605.05362.pdf.

[2] Wenwen Ye. Yongfeng Zhang. Wayne Xin Zhao. Xu Chen. Zheng Qin. *A Collaborative Neural Model for Rating Prediction by Leveraging User Reviews and Product Images.* http://yongfeng.me/attach/airs17-chen1.pdf

[3] Yehuda Koren. Robert Bell. Chris Volinsky. *Matrix Factorization Techniques for Recommender Systems.* https://datajobs.com/data-science-repo/Recommender-Systems-[Netflix].pdf

[4] Yifei Feng. Zhengli Sun. *Yelp User Rating Prediction.* http://cs229.stanford.edu/proj2014/Yifei%20Feng,%20Zhengli%20Sun,%20Yelp%20User%20Rating%20Prediction.pdf.

[5] Yiwen Guo. Anran Lu. Zeyu Wang. *Predicting Restaurants Rating And Popularity Based On Yelp Dataset*. http://cs229.stanford.edu/proj2017/final-reports/5244334.pdf.

[6] ”Yelp”, Yelp, 2018. [Online]. Available: https://www.yelp.com/.

[7] ”Yelp Dataset”, Yelp.com, 2018. [Online]. Available: https://www.yelp.com/dataset.