Rating Predictions of Google Local Reviews

Raj Sunku, Joseph Teh, Sanjith Devineni

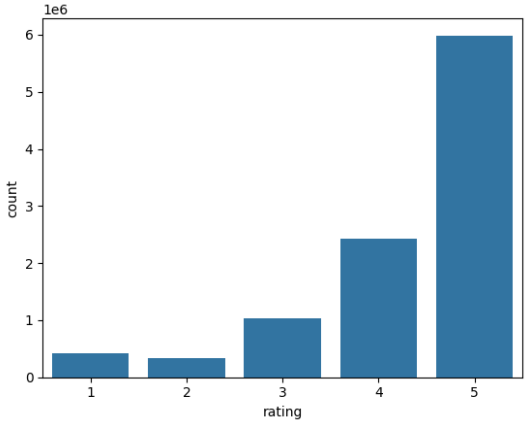
1. INTRODUCTION

Dataset: Google Local Data (2021) on the state of Washington.

This dataset contains google reviews of businesses from Washington State, which was specifically collected from Google Maps. To be specific, we are using the 10-core subset of the entire dataset, meaning that each of the remaining users and items in the subset have 10 reviews each. Each review in this dataset has a rating and time posted, as well as possibly having review text and a response from the business. Since each review also has an associated business with it, there is also the metadata for each business included in the dataset, which describes the average ratings and number of reviews for each business. There are a total of 10, 192, 020 reviews in this dataset, with a total of 121, 304 businesses being reviewed.

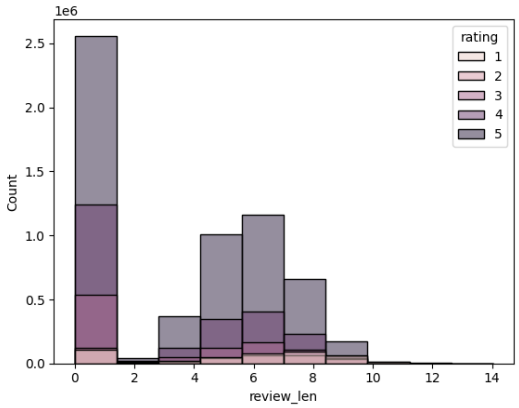
1.1 EXPLORATORY ANALYSIS

1.1.1 Number of reviews for each category of ratings:



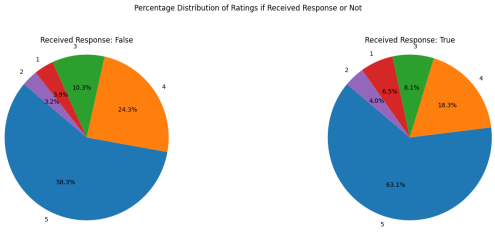
Based on the above graph, it can be seen that the graph is heavily skewed left, meaning that a majority of reviews given are either 4 or 5 and the rest are given as 1, 2, or 3. Thus, it is important to keep in mind that reviews are not evenly spread among the ratings and we will have to account for that in our model.

1.1.2 Number of reviews based on transformed review_len: (log_2(review_len + 1)):



Based on the histogram, we can see that most of the reviews have smaller review text, and among the reviews that have review text, there lies a somewhat even distribution with most reviews lying around 6 in transformed length. We can also see that there seems to be only a little correlation between the length of the review text and the rating it receives.

1.1.3 Percentage of Reviews based on whether a response was given:



The pie charts above illustrate how the presence of a response could affect the ratings given by the users. Based on these charts, it seems like reviews receive responses from businesses if they are

extreme, which could explain why there is a higher percentage of 5-star and 1-star reviews in the second pie chart.

1.2 Interesting Findings

Looking at the skew of the reviews, it is interesting that an overwhelming majority of the reviews are positive (4 or 5). It seems that people often do not give negative reviews compared to positive reviews. Additionally, we discovered that a large number of reviews do not actually contain text. And the proportions of ratings to text are similar to the overall proportion of ratings of the entire dataset. It does not seem like the length of text in a review plays a large role in the rating given. Lastly, comparing the proportions of reviews that did and did not receive a response from the business, it is noticeable that ratings of 1 and 5 receive slightly more responses from businesses given the proportions of reviews that do not receive responses.

2. PREDICTIVE TASK

Given a user and business, predict the rating that the user would give to that business using relevant features. Predicting a rating as a function of the features such as review_len, resp, avg_rating, and num_of_reviews.

The features that are being used to predict the rating are:

Review_len: contains the text of the review that the user wrote for the business with the transformation function applied from before
Resp: whether a business gave a response or not
Avg_rating: the average rating of the business
Num_of_reviews: the number of reviews for each business

Obtaining review_len came from extracting the length of the 'text' section of the review. This length was then transformed by adding 1 and taking the log_2 of that length.

Obtaining resp came from whether or not the 'resp' section of the review had any information. If there was any information present, then this would be set to true, and false otherwise.
Obtaining avg_rating came from the metadata of the dataset. Using the gmap_id (the id of the business) we can obtain the correct avg_rating for that business.
Obtaining the num_of_reviews uses the same process as avg_rating. Since this information is in the metadata, we use the gmap_id to find the correct num_of_reviews for each business.

2.1 Baselines to Compare

The baseline that is used for comparison purposes is a model that just predicts the average rating for every user and business pair. The MSE (Mean Squared Error) that this model produced for the test set is 1.100.

2.2 Validity of Model

To evaluate the model and assess its validity, we will take a subset of the data to be our test set, and run our model on the test set. Our validity can be assessed using the MSE (Mean Squared Error) between the actual ratings and the model's predicted ratings on the test set, which would determine our model's accuracy.

3. MODEL

To create our model, we first extracted three different sections from our dataset after shuffling it with a random seed of 0. The first 100,000 data points from that were used as a training dataset, the next 25,000 were used for validation, and the next 100,000 were used for testing. The training dataset was used to develop our model, with the validation being used along the way to test it along the way to implement early stopping. Finally, the test dataset is

what we used at the end to test how our model performed. These three datasets will be essential to creating our models and identifying how effective they actually are.

3.1 Iterative Model

We first decided to create a basic iterative model that can be used to warm start any future models we made.

This is the iterative procedure that we used to determine the general alpha value, and the beta values for each of the features:

L_{feat} describes a list of tuples, where each tuple contains the values for each feature (except the current feature) for all of the data points whose value for the current feature is the same as *feat*. For example, if we are looking at review lengths currently, L_0 would contain tuples that represent the features and ratings for all of the data points with review text length of 0.

$$\alpha = \frac{\Sigma_{data \in train}(R_{data} - \Sigma_{feats \in data}(\beta_{feat}))}{N_{train}}$$

$$\beta_{feat} = \frac{\Sigma_{feats \in L_{feat}}(R_{(feat,\, feats)} - (\alpha + \Sigma_{feat \in feats}(\beta_{feat})))}{\lambda + |L_{feat}|}$$

Using the global training rating average as the starting value of alpha, and setting the betas to 0 initially, we ran this iterative model until it stopped improving its MSE on the validation dataset. From just this model, we were able to get an MSE of 0.8731 on the test dataset, dropping it significantly from the baseline.

3.2 Latent Factor Model

Now that we had good starting alpha and betas from the iterative model, we decided to use them to warm start different Tensorflow Latent Factor models with unique hyperparameters. For the Latent Factor models, we decided to have a unique lambda for each of the different features that we could tune, as well as tuning the K alongside them. Utilizing GridSearch hyperparameter tuning with numerous different potential values for each parameter, we ran each model with the similar early stopping technique from before. Due to warm starting our models, we were able to relatively quickly run the GridSearch algorithm and tune the hyperparameters thoroughly. After that tuning, we were able to lower the MSE to 0.847 on the test dataset, which was the lowest we eventually got. Although it was an improvement from our previous iterative model, it was not that significant of an improvement.

The final model ran for only 4 iterations, had a K of 3, a lambda of 0.0000015 for review_len, 0.0001 for resp_received, 0.00015 for avg_rating, and 0.00002 for num_ratings.

4. LITERATURE

This dataset contains information gathered from Google Maps and respective businesses. There are many different types of datasets that are similar to this one as it is mainly based on reviews of customers who want to share their opinion on some product or service.

4.1 Similar Past DataSets

Google Local Reviews (2018) and the Maps dataset API from Google. These are similar Google datasets that have been used for a lot of literature.

4.2 Methods Employed

As this media is so common today, there are many different ways that people have gone about analyzing them. A popular method that has stood out from others is sentiment analysis which aims to study and understand subjective data[6.3]. This method analyzes the words that may be present in a review by removing insignificant language and then categorizing words as maybe positive or negative on some scale. In doing so, methods to create better services or advertisements have come about where models are more accurately able to predict what people would like and what they may dislike. In being able to predict market trends, businesses are able to develop or improve products that people

may need or want more than other products. These models go more in depth to analyze not only the length of the text review, but also the language used and how it affects the overall review. Thus, these models may see better results than our model.

5. RESULTS/CONCLUSION

5.1 Performance Comparisons

Model	MSE
Baseline	1.1
Iterative	0.8731
Latent Factor	0.847

Overall, from the above table, we were able to determine that the Latent Factor model performed the best on our dataset. However, there are a lot of caveats to this. Firstly, we did not tune the iterative model that much, and only really used it as a baseline for the Latent Factor model. Because of this, the Latent Factor model at minimum had to be as good as our iterative model.

In addition, we were able to learn how to take a lot of effort to continue to lower MSEs and there's diminishing returns for it. Lowering our MSE from the baseline was relatively easy using our untuned iterative model, but getting even an improvement from the Latent Factor model proved to be very difficult.

5.2 Feature Representations

When we initially made the models, we decided not to use the metadata and only used the review length and resp_received as the features. This resulted in only minor improvements from the baseline. However, when we started using the metadata and included the two additional features, it improved the iterative model's performance drastically. This would indicate that the average rating of the business and the number of reviews they got affect the ratings a user would give them significantly.

5.3 Interpretation of Parameters

Since our K value of 3 was relatively low, this means that a simpler Latent Factor model was more effective. This could be because we warm started our model and it didn't have to be improved that much. The lambdas don't really give that much useful information other than when our model stops. Since we had a low number of iterations, it means that it wasn't changed that much from the iterative model.

5.4 Conclusion

Overall, we were able to create models that could predict the ratings of user reviews decently well compared to the baseline average rating model. This could be because we utilized both the iterative and Latent Factor models to implement this, however we also chose useful features that could be used to build these models.

6. CITATIONS

6.1 UCTopic: Unsupervised Contrastive Learning for Phrase Representations and Topic Mining

Jiacheng Li, Jingbo Shang, Julian McAuley
Annual Meeting of the Association for Computational Linguistics (ACL), 2022
https://aclanthology.org/2022.acl-long.426.pdf

6.2 Personalized Showcases: Generating Multi-Modal Explanations for Recommendations

An Yan, Zhankui He, Jiacheng Li, Tianyang Zhang, Julian Mcauley
The 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), 2023
https://arxiv.org/pdf/2207.00422.pdf

6.3 Sentiment Analysis of Review Datasets Using Naive Bayes and K-NN Classifier

Lopamudra Dey, Sanjay Chakraborty, Anuraag Biswas, Beepa Bose, Sweta Tiwari

International Journal of Information Engineering and Electronic Business (IJIEEB)

https://doi.org/10.5815/ijieeb.2016.04.07