

Deep Learning Rating Predictions for D.C. & Rhode Island Google Businesses

1 Dataset & EDA

The dataset that we have decided to investigate is “Google Local Data (2021)”, found within the selection of Recommender Systems and Personalization Datasets recommended by Professor Mcauley. The link to the dataset is the following:

https://datarepo.eng.ucsd.edu/mcauley_group/gdrive/googlelocal/#sample-review

To be specific, we have decided to explore both the reviews and business metadata within the District of Columbia area in order to train and test our model, as well as use the Rhode Island area in order to verify that overfitting has not occurred. We chose these smaller datasets as to both contain the scope of this paper, as well as give ourselves a more concise dataset to work with.

Our initial steps were to first transform the json.gz files into read/workable dataframes and drop columns unnecessary for our work, such as picture data, reviewer names, business descriptions, and business urls. Following that we joined both the reviews and metadata dataframes into singular dataframes on the ‘gmap_id’ column, which itself can be used to identify any individual business. We do this in order to attach the business metadata to each review. In total, the District of Columbia dataset contains slightly over 1.8 million rows whilst the Rhode Island dataset contains roughly over 1.7 million rows. Both datasets contain 14 columns.

When taking a look at the data, we noticed some interesting things about our future prediction statistic, ‘rating’. Figure 1 shows that the ratings

within the District of Columbia dataset are skewed rather high, such that the mean rating is 4.31, and the median is a flat 5.0, the maximum rating possible. People within this dataset tend to be rather positive in their ratings, compared to the overall mean rating as of July 2022, which is 4.11^[3].

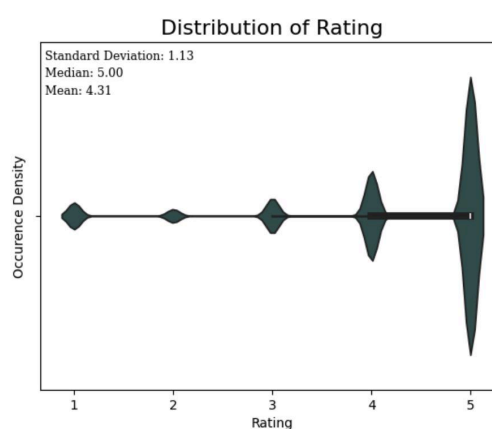


Figure 1. A graph showing the distribution of the ‘rating’ column

An important part of these reviews are the actual review texts themselves, and we wanted to make sure we represented that aspect within our model, in the form of text sentiment. In our data, reviews are stored in the ‘text’ column, and rows without review text are instead filled with numpy nan values. To compute text sentiment we used the TextBlob library, a library that aims to provide access to common text-processing operations for Natural Language Processing. We ran the sentiment function on all of the reviews, which returns two values, the sentiment polarity, ranging from -1 to 1, with -1 being a very negative text, and 1 being a very positive text,

and the subjectivity value, ranging from 0 to 1, with 0 being a very objective text, and 1 being a very subjective text.

Our findings from this sentiment analysis, shown in figure 2 below, is that a user leaving a review is typically associated with better than neutral sentiment, meaning that it is more likely a user will leave a review after a positive experience than a negative experience. The average sentiment associated with a review is .38, a generally positive sentiment, while only around 10% of reviews are associated with a sentiment lower than 0.

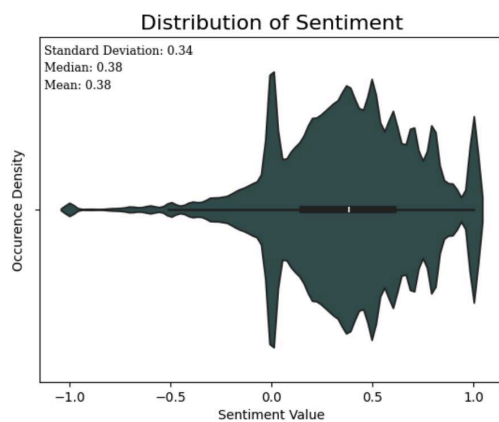


Figure 2. A graph showing the distribution of the 'sentiment' column

All in all, our general initial findings on our dataset is that people within the dataset are generally positive in comparison to the national average, and that reviews in the dataset are generally associated with a positive sentiment.

2 Prediction

The predictive task that we want to approach is **predicting the rating**, either 1, 2, 3, 4, or 5, given the associated review and business data. In particular, we want to create a model that includes the numerical aspects of a particular

review, in order to ensure objectivity. The following section will be used to describe the numerical columns that will ultimately be utilized in our model:

“unix_time”: an integer of unix-time representation, or the milliseconds that have passed since 1 January 1970. Came with the dataset.

“sentiment”: a float of the sentiment polarity for the associated “text” review that was derived from running TextBlob sentiment analysis on the associated review “text”.

“subjectivity”: a float of the subjectivity value for the associated “text” review that was derived from running TextBlob sentiment analysis on the associated review “text”.

“avg_rating”: a float of the average rating for the business the associated review is referring to. Came with the dataset.

“num_of_reviews”: an integer of the total amount of ratings the business the associated review is referring to has. Came with the dataset.

For baseline models to compare to our actual model we will be using a simple model that estimates the mean rating for every prediction rating as well as a linear regression model. We will evaluate the performance of these models based on their minimization of our loss objective, which is mean squared error. We will be using MSE in order to penalize worse errors magnitudes more than predictions that are only slightly off.

3 Model Optimization

The following section will cover our baseline model performance, before continuing onto a description of our actual model.

Mean-assuming Model: The MSE for the DC dataset, the dataset on which all of the models mentioned are trained on is 1.28, and the MSE for the verification set, the Rhode Island dataset, is 1.30

Linear Regression Model: The MSE for the DC dataset is 0.93 and the MSE for the RI dataset is 0.97

The model that we ultimately developed is a deep learning model built with TensorFlow and the Keras 3 API. We utilized a deep learning model for this problem as when we attempted to use more traditional modeling techniques they failed to capture the complex relationships between the features. We trained it on the numerical columns of the dataset that we mentioned before, being: “unix_time”, “sentiment”, “subjectivity”, “avg_rating”, and “num_of_reviews”. Figure 3 below shows the compilation details of the model and figure 4 shows the model architecture:

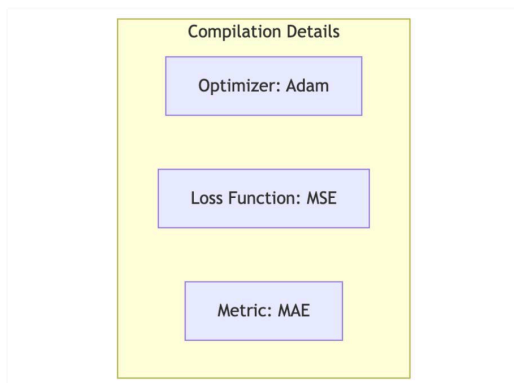


Figure 3. A diagram of the model's compilation details



Figure 4. A diagram of the model architecture

When creating the model we ran into several issues that had to be overcome, namely overfitting and compilation optimization. To fix the overfitting, we optimized the epoch amount such that the training set still had good performance whilst not sacrificing validation and test set performance. For the compilation optimization, we debated using huber loss over MSE as our loss function, but since our data does not, and cannot contain significant outliers, as the ratings must fit between 1 and 5, and MSE performs better on datasets without outliers, we decided to move forward with MSE for our model. This decision was also corroborated by testing.

The performance for our actual model was a significant improvement over both baseline models; the MSE for the DC area was 0.82 and the MSE for the RI area was slightly larger at 0.85. Remember that since the training set itself is derived only from the DC set that means the model performs similarly well on unseen data, in this case, the RI set.

The model shows a roughly 36% improvement over the baseline mean-assuming model and an around 12% increase in performance over the baseline linear regression model. Both of these statistics are taking into consideration both performance on the test set, derived from the DC set, and performance on unseen data, in this paper the RI set.

Our model results in better performances run on the same data, and comparing the linear regression model to our final model, on the same features. However taking into consideration

some of the downsides of deep learning, there perhaps may be some complications trying to replicate these results with a smaller dataset.

4 Literature Review

Concerning the dataset, past works have used the data in order to determine the importance of specific words in the reviews. Previously the Journal of Multimedia Information (Shin et al.) used a tf-idf algorithm in order to find the importance of a word in relation to all of the reviews.

In this model they labeled all of the positive and negative comments based on how many stars the user rated with 1 representing 3 stars and above, and 0 representing below 3 stars. Our model utilizes a more complex approach to text sentiment analysis by using TextBlob in order to process the text, in which we get two features representing text polarity and text subjectivity.

The current state-of-the-art methods concern using contrastive learning in order to perform analysis on the Google Review Dataset. ACL (Li et al.) used unsupervised contrastive learning in order to perform topic-mining. Instead of using standard natural language processing, they used context in order to better differentiate important topics. They applied unsupervised contrastive learning based on positive pairs from phrase-oriented assumptions and used a cluster assigned negative sampling method in order to pull semantically close neighbors together and push apart non-neighbors. Another use of unsupervised contrastive learning is through SIGIR (Yan et. al) using this learning in order to create personalized showcases for each user. They made a recommender system which gives customized images with appropriate text. The role that contrastive learning played was that it helped to improve diversity and visual alignment of the generated text.

What remains similar about the conclusions of existing works and our own work was the role that polarity plays into all of the models. The core of all the models involve using polarity to either generate or procure results. Whether the models were used to determine different things, in the end they all relied on some way to determine polarity of the reviews.

5 Conclusion

Overall, our model heavily outperformed both baselines models, in both MSE & MAE, meaning that it makes predictions much more accurately and with much higher precision than the baselines. We ran a feature gradient analysis in order to observe which of the features were most impactful to our model. The numerical values can be interpreted as their correlation to the resulting prediction. For example, if the feature gradient value is 0.32, a 1.0 increase in that particular (standardized) feature value is correlated with a 0.32 **increase** in the final prediction all other things being equal. Conversely, a -0.32 feature gradient suggests that a 1.0 increase in that particular (standardized) feature value is correlated with a 0.32 **decrease** in the final prediction all other things being equal. The results are as follows:

- “unix_time”: -0.0056
- “sentiment”: 0.1317
- “subjectivity”: -0.6554
- “avg_rating”: 0.3049
- “num_of_reviews”: -0.0356

Overall, the feature gradients are as you would expect. The time of the review has a small but rather negligible effect on the prediction, whilst things like sentiment have a positive correlation with rating prediction. Average rating also has high correlation with the predicted rating, which makes sense as businesses with higher quality and higher consistency are likely to receive

better ratings. The number of reviews a business has surprisingly has a small negative correlation but the big surprise of our analysis, and ultimately the most important feature, is the subjectivity of the review text. At -0.6554 it has a very significant negative correlation with the rating prediction, meaning that a higher subjectivity value correlates to a lower rating prediction. When diving deeper into the subjectivity column, we found the reason why this may be. For rows with no review text, the "text" column is imputed with a np.nan value, and therefore the related polarity and subjectivity values are both zero. When running some further computations, we found that the average rating for reviews with no text is 4.4, higher than the average rating for reviews that contain a written review. This makes sense in the context of the gradient, as it took in many reviews that contained no written portion, and ultimately learned to penalize rows that contained written text.

Overall, we are happy with the performance of our model and its ability to minimize MSE compared to our baseline models. In future testing we'd want to eliminate all bias for reviews without written text but even still our model is able to accurately predict ratings based on only the numerical and objective parts of a review.

References

- [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
- [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
- [3] “The State of Google Reviews.” *SOCi*, www.soci.ai/insights/state-of-google-reviews/. Accessed 30 Nov. 2024.
- [4] **Analysis on Review Data of Restaurants in Google Maps through Text Mining: Focusing on Sentiment Analysis**
Bee Shin, Sohee Ryu, Yongjun Kim, Dongwhan Kim
Journal of Multimedia Information System, vol. 9, no. 1, 2022