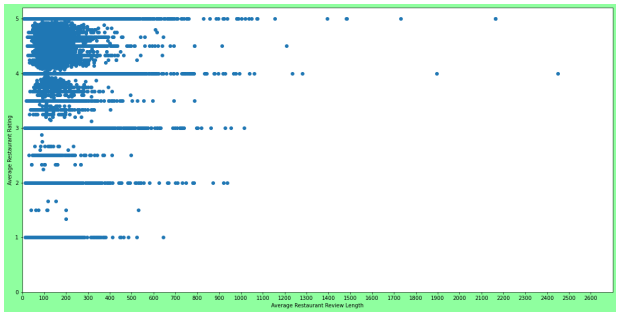
Predicting Average Restaurant Ratings From User Reviews

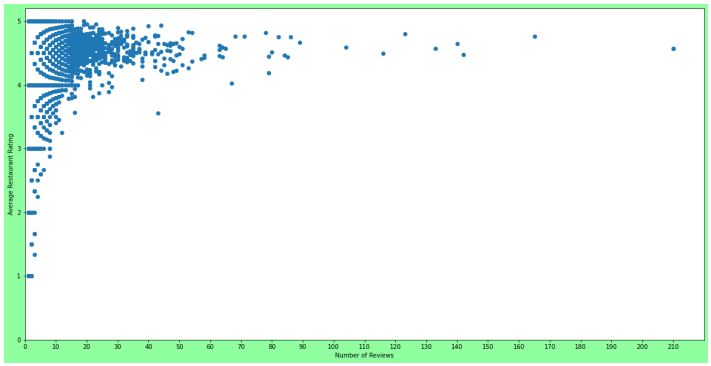
PART 1- INTRODUCTION

We have chosen to use 'Google Restaurants' which is a large dataset consisting of user reviews of restaurants. Each entry consists of a user ID of the user making the review and an item ID of the restaurant being reviewed labeled as 'business_id.' An entry also contains the review text that the user said about the restaurant, a rating out of five that the user gave of the restaurant, a list of pictures represented by unique identifiers that the user has uploaded, and a list of the user's previous reviews represented by pairs of the concatenation of the user's user ID and the restaurant's ID in the format userID_businessID, and the review text containing what the user said about that restaurant. The dataset has been split into training, validation, and test sets with the training set containing 87,013 reviews, the validation set containing 10,860 reviews, and the test set containing 11,015 reviews, totaling 108,888 reviews. The dataset also includes 30,000 restaurants, 37,000 users, and 203,000 images. This is a sizable amount of data allowing us to collect a lot of information and make accurate predictions by using models we have studied in class. After conducting exploratory data analysis on the dataset, we have found that the average rating a restaurant receives per review is 4.4653 out of 5 and the median rating is 5 which is relatively high. Upon further analysis, we found that the data is severely positively skewed with the training data containing 54665 ratings of 5, 22333 ratings of 4, 6918 ratings of 3, 2027 ratings of 2, and only 1070 ratings of 1. Although this may present a challenge because our data is somewhat biased, this may open opportunities for emphasizing our predictive task to deal with how accurately we can predict negative reviews. The length of a review ranges from 0 to 3456 characters and averages at around 134 characters. By comparing the average length of each user's review texts with that user's average rating out of all the ratings they have given out, we can determine a slight positive correlation shown below.

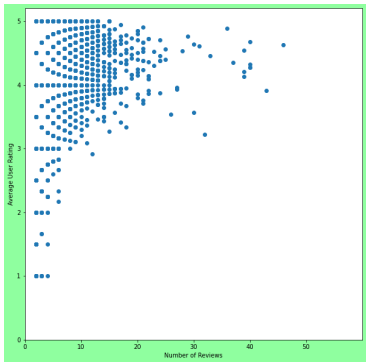
Although you can clearly identify the skew, you can also see that the longer a user's average review is, the higher they tend to rate restaurants. Similarly, the higher a restaurant's average review is, the higher they tend to be rated, as shown below.



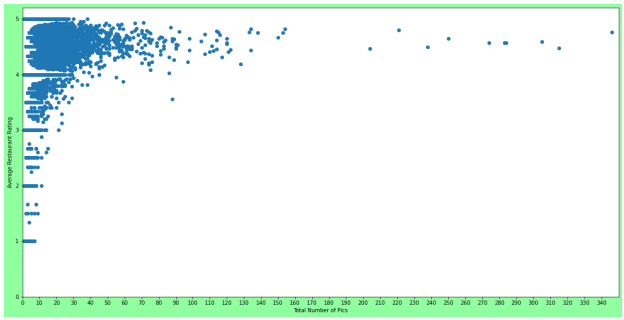
These associations are likely attributed to the fact that people would not want to take time out to write a long review for a restaurant they dislike. Most of the time, if a restaurant leaves a lasting impression on users, they will not mind writing a detailed and extensive review but if a user dislikes a restaurant, they will not bother writing much about it if they write anything at all which can also explain why our data is so positively skewed. Another feature we can extract from this data is the popularity of a restaurant determined by how many reviews there are of it. The more popular a restaurant is, the higher it tends to be rated on average shown below.

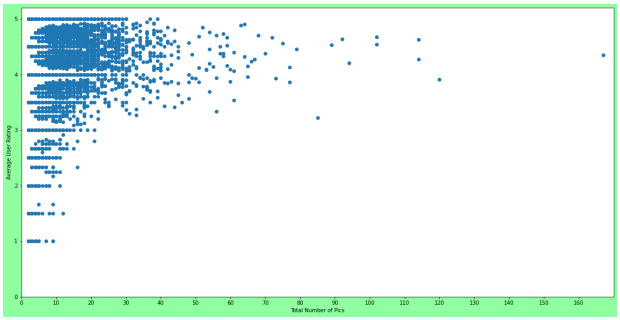


Likewise, the more reviews that a user writes, the higher they tend to rate restaurants on average shown below.



Lastly, we found a positive association between the total number of pictures that a restaurant has, with the average rating they are given, and the total number of pictures a user has uploaded with the average rating they give out which are both shown below with the former on the left and the latter on the right side. This can also be attributed to the same point raised earlier about users not wanting to put in the time and effort to upload pictures of restaurants they dislike.





PART 2 - PREDICTIVE TASK

Predictive Task

Using this data, we intend on predicting the mean rating of restaurants using past individual user review data. We'll accomplish this by extracting features from the review text and user histories. As the dataset contains an evident skew favoring higher review ratings, we plan on emphasizing the model's ability to predict negative ratings.

Model Evaluation

When evaluating our models, we can either treat the task as a continuous regression problem or as a classification problem wherein mean ratings fall within 10 ordinal categories spanning from the (1, 1.5) range (worst rating) to the (4.5, 5) range. For this project, we evaluated models under each category.

The skewed distribution of reviews ratings does present minor challenges when evaluating models. As the majority of individual ratings are *5*—the max possible evaluation—and the overwhelming majority of ratings are *4* or above, we have to consider how our models may default to these high ratings.

This is a common prediction issue across fields, in which the rarity of certain values leads to the model achieving a high score by ignoring and neglecting the extreme values. For example, if a certain disease affects 1 in 1000 individuals, one can always achieve a model that predicts the occurrence of this disease at 99.9% accuracy by simply never predicting a positive value.

Whether we use mean squared error (continuous model) or accuracy (classification model), this issue persists. One solution would be to undersample the majority classes and oversample the minority classes such that the dataset is balanced. With the classes balanced, the model would no longer default to a particular common rating. Though, this method would affect the model's performance on actual data with the same rating distribution as the population.

Alternatively, we considered employing evaluation metrics to favor the model's ability to predict all classes and provide a more nuanced perspective of the model's performance. Classifier metrics such as F1 score and AUC achieve this by evaluating the specifics of the confusion matrix (i.e. true positive rate and false positive rate opposed to just accuracy).

However, after initially assessing the models using the basic metrics of accuracy, MSE, and R^2, we deemed that these metrics would be suitable for our task as our models were successfully producing accurate estimates across the entire rating range. R^2, in particular, enabled us to evaluate our models relative to our baseline—rendering it a more comprehensible gauge for our model's performance.

To assess these models, we used the test division of the dataset, while training the models on a separate training dataset.

Baseline

Our baseline classification model simply selected the default class (4, 4.5) which was accurate about 38% of the time. Our baseline regression model involved always guessing the global mean restaurant rating, producing an MSE around 0.65.

Features Engineering

We identified positive correlations between the following potential features with our outcome variable, review rating:

- Review length
- Number of restaurant reviews
- Number of pictures taken by a user

Accordingly, we engineered features using those variables from the base dataset. To account for the skewed distribution for each of these three variables, we applied a log transformation to convert these feature sets into roughly normal distributions.

We also utilized NLTK's sentiment analyzer to evaluate the sentiment of each user review. The method provides a sentiment score ranging from -1 (negative) to 1 (positive). We calculated the

mean sentiment score by restaurant to manufacture the fourth feature used by our optimal model. We found that the correlation coefficient between the mean sentiment score and mean rating for each restaurant was 0.789, indicating that these features are strongly correlated.

Various other features such as rating of similar restaurants (as determined by common customers) did not contribute significantly to the model and encouraged overfitting.

In addition, we later incorporated a bag of words model into our feature vector in hopes of improving our results. This worked by using one bag of words matrix that combined all reviews for each restaurant in the dataset using the sparse matrix feature in Scipy in order to allow for more efficient computations. While we initially aimed to combine the bag of words matrix with our preexisting features, this proved to be impossible due to computational limitations.

Models

We experimented with a variety of models for this task. For our classification attempts, we designed a logistic regression-based classifier model, an XGBoost classifier, and a Naive Bayes model. For our models producing continuous outputs, we built a ridge regression model, a KNN regressor, an XGBoost regressor, and a couple of bag-of-words model variations.

We also applied post-processing to each model, restricting the output between 1 and 5 as no true value should fall outside this range.

PART 3 - MODELS

We tried the different models bolded below in order to try and predict the average rating of a restaurant based on the user reviews:

Ridge Regression Model:

First, we chose to use a regression model that calculated the average rating of a restaurant given all reviews that have been made of it instead of using a classification model that calculated the rating one user gave it given their review because we wanted to investigate using analysis of text to make our predictions and the former would give us more data (in the form of text) allowing us to explore our interests in this area more and make more accurate predictions.

We initially experimented with a ridge regression model in order to manage potential overfitting issues. The model attained a training MSE and test MSE of 0.214 as well as an R^2 of 0.671.

XGBoost Regressor:

We chose the regressor provided by the XGBoost library to do this because it is one of the leading libraries in regression because of its efficiency, flexibility, and portability. This was important for us because we were working with different operating systems in different environments. It also integrates well with Scikit which is a library that we have become accustomed to as students of CSE 158. It has also been used extensively to help students win Kaggle competitions which often have similar datasets and predictive tasks to ours.

XGBoost is a unique algorithm that implements a base model of decision trees in conjunction with gradient boosting—the creation of an ensemble of weak models that are trained to correct the flaws and errors of each other. The sum of these models when paired together can produce fairly accurate results as the effect of individual flaws found within each submodel are diluted and minimized when these models are merged into an ensemble. Decision tree models (such as the model at the core of XGBoost) also can provide acceptable results as long as the tendency to overfit is moderated. We encountered some issues regarding overfitting, but after tuning several regularization parameters, the model performed quite well. The regressor achieved a training MSE of 0.199, test MSE of 0.204, and a test R^2 of 0.686. The test MSE of 0.204 is relatively low, as it indicates that on average, the difference between the predicted and actual average ratings for each restaurant only differ by about 4% ($\sqrt{MSE}/5$). In order to evaluate the effectiveness of the features used, we then ran this model with the bag of words, which combined all reviews for each restaurant, being the only feature. This regressor achieved a training MSE of 0.211, a test MSE of 0.233, and a test R^2 of 0.641, which was slightly worse, indicating to us that simpler features worked better. While we would've liked to combine the original feature vector with the bag of words, this unfortunately wasn't feasible due to computational limitations.

In addition, to find the best hyperparameter for the XGBoost model, we used GridSearchCV, which allowed us to test numerous hyperparameter combinations. Specifically, we tested our model on 64 different hyperparameter combinations using 5-fold cross-validation for a total of 320 different tests. This did not seem to meaningfully impact our results as the best fit model gave us very similar results to our initial hyperparameters, which were arbitrarily selected.

N-grams Regressor:

Another model we considered was a ridge regressor that had a feature matrix of n-grams with n up to 5. We combined this with other features such as the number of reviews made for a restaurant, the average length of review, the sum of the lengths of all reviews, the total number of pictures uploaded, and the average number of pictures uploaded per review. This performed extremely well on the training set having an MSE of around 0.002 and an r-squared score of around 0.997. However, we found that this model was subject to overfitting because the MSE increased to around 0.569 and the r-squared score decreased drastically to around 0.131.

Logistic Regression-Based Classifier:

Our investigation of classification models began with experimentation using a logistic regression-based classifier. The model required significant regularization to prevent overfitting, but eventually outputted a comparable accuracy to our better classifier models. The model realized a training accuracy of 0.538 and a test accuracy of 0.533.

Naive Bayes:

We also experimented with a naive bayes model which performed fairly poorly in comparison to its counterparts. The model appeared to underfit as both its training and test MSE were significantly lower than other models, despite altering and tweaking various hyperparameters. This outcome is unsurprising as underfitting is a known tendency of naive bayes models. The model's training accuracy was 0.513 and test accuracy was 0.518.

XGBoost Classifier:

We also constructed a XGBoost classifier model, which evaluated restaurant ratings divided into ten categories using the same feature set. We experienced noticeable overfitting issues initially with our XGBoost classifier despite the model still performing the best among our various classification models. The classifier scored a training accuracy of 0.585 and a test accuracy of 0.565. With an eye on improving this model, we increased the number of classes to 40 (using 0.1 increments), and experienced a drastic reduction in accuracy, with the training accuracy being 0.212 and the test accuracy being 0.211, indicating that just over 1 in 5 data points were correctly classified.

KNN Classifier:

We also tried using a KNN classifier. Comparatively to our other models, we found that the KNN Classifier when using a feature vector including review_length, num_pictures, and sentiment, did not perform as well. For our KNN classifier, we tried different values for the number of neighbors (n) and while we noticed a slight improvement in terms of MSE, accuracy, and R^2 values with increasing the number of neighbors, and while all our KNN models had an MSE better than the baseline MSE of 1.1521630026617435, it still did not perform better than any of the other models we implemented. It can be seen that in spite of the fact that we tried many different n values, it did not perform better than any of our other models.

More specifically, for the different n values, we found...

n = 3:
- Mean Squared Error (MSE): 0.8574433650596738
- Accuracy: 0.07439830577440965
- R^2 Value: 0.25579682468644127

n = 5:
- Mean Squared Error (MSE): 0.800681694817055
- Accuracy: 0.13567197151673638
- R^2 Value: 0.3050621370697474

n = 7:
- Mean Squared Error (MSE): 0.7779242426278703
- Accuracy: 0.16023841772289626
- R^2 Value: 0.32481407506516136

n = 10:
- Mean Squared Error (MSE): 0.7779242426278703
- Accuracy: 0.16023841772289626
- R^2 Value: 0.32481407506516136

n = 12:
- Mean Squared Error (MSE): 0.751996733083978
- Accuracy: 0.18822690971992373
- R^2 Value: 0.34731740964889146

n = 15:
- Mean Squared Error (MSE): 0.7448017778335876
- Accuracy: 0.1959937932728394
- R^2 Value: 0.35356214692457943

n = 20:
- Mean Squared Error (MSE): 0.7368259623449164
- Accuracy: 0.20460360778651399
- R^2 Value: 0.36048461837197476

n = 30:
- Mean Squared Error (MSE): 0.7283907058550392
- Accuracy: 0.2137093843502108
- R^2 Value: 0.36780585370967434

PART 4 - LITERATURE REVIEW

As mentioned, our goal in this study is to build a model to predict average ratings for restaurants based on all written reviews for that restaurant from the “Google Restaurants” dataset (Yan et al., 2022). As described, this dataset contains various information about restaurants and user’s reviews for those restaurants most importantly including the user ID of a reviewer, their overall restaurant rating, and their written review of the restaurant. Previously, this dataset was used in the study “Personalized Showcases: Generating Multi-Modal Explanations for Recommendations" (Yan et al., 2022) in which the researchers leveraged the data in order to develop a personalized cross-modal contrastive learning framework to enhance explanations for restaurant recommendations with images along with text. The main model building methods that were used in this study were using CLIP - a state-of-the-art pre-trained cross-modal retrieval model for multi-modal encoding, Determinantal Point Process (DPP) for image selection modeling, and GPT-2 for multi-modal decoding. Another study, “Sentiment Analysis of Review Datasets using Naïve Bayes’ and K-NN Classifier” (Day et al., 2916) used a similar type of dataset as the one used in our study in order to evaluate the performance for sentiment classification in terms of accuracy, precision and recall. The dataset used in this study included 10,000 reviews that were crawled from a movie review website which consisted of written movie reviews from users, similar to that of the reviews written in the Google Restaurants dataset. The main model building methods that were used in this study were sentiment analysis, naive bayes, and K-Nearest-Neighbors. The results of this study demonstrated that Naïve Bayes‘ approach gave above 80% accuracy and outperformed K-NN approach. A third study, “Prediction of COVID-19 Possibilities using KNN Classification Algorithm“ (Research Square, n.d.), utilized KNN classification, logistic regression, decision tree, support vector machine, and cross validation techniques in order to predict COVID-19 possibilities. In this study, the KNN classification algorithm performed the best. Although this wasn’t a recommendation or rating based task, it was a predictive task with a dataset structure similar to the one used in our study. A fourth study, “Machine learning using the extreme gradient boosting (XGBoost) algorithm predicts 5-day delta of SOFA score at ICU admission in COVID-19 patients” (Science Direct, 2021) used XGBoost and logistic regression algorithms to assess risk stratification for critically ill COVID-19 patients. The results of this study demonstrated that the XGBoost model was more effective than the classical method of logistic regression in identifying critically ill COVID-19 patients admitted to the ICU whose clinical condition was likely to worsen or improve. Hence, in this study, the XGBoost model proved to be the best. Based on the previous studies described here, we can see that the main state-of-the-art methods currently used for review-type/predictive task data sets are KNN classification, logistic regression, naive bayes, and XGBoost. Similar to the primary methods that are currently used, we also used a naive bayes model, regression models, and some classifier models as shown above in order to predict average ratings for restaurants based on all written reviews for that restaurant. However, for KNN, we tried KNN regression rather than KNN classification. From our results we found that similar to the past studies, our XGBoost models, for both regressor and classifier performed the best. However,

unlike the past studies, our KNN model did not perform well. Overall, for the purposes of our study, we found that the XGBoost regressor algorithm worked the best with a test set MSE of approximately 0.204.

PART 5 - CONCLUSION

Our most effective model by far turned out to be the XGBoost model with the following features: review count, log-transformed review count, image count, and sentiment score. We also ran this model with a bag of words matrix being the only feature used and found that the latter model produced a slightly lower MSE (~0.21 as opposed to ~0.23). Beyond the bag of words and sentiment scores, including features such as transformed number of words and the image count also seemed to increase our accuracy. It's likely that reviewers who felt more strongly about their rating would leave a longer review or include more images. Based on these results, we also believe that users who had a particularly negative experience would be more prone to use more text or images. The simpler feature set (as opposed to the bag of words and bag of n-grams models) also contributed to the prevention of overfitting, which is another reason why we believe that we achieved a slightly better MSE with the simple features as opposed to the bag of words model. As one increases the feature set and the number of predictors approaches the number of observations, the model becomes increasingly susceptible to overfitting. By using the NLTK sentiment analyzer, we captured the general purpose of the bag of words models (to gauge the opinion of the review text) without compiling an extensive list of features. While this process may result in the loss of some textual nuance, the avoidance of overfitting proved to be imperative. These conclusions align with outputted metrics discussed in part 3.

In order to further curtail overfitting with our XGBoost model, we applied max depth, max leaves, and gamma regularization parameters. The max depth parameter limits the height of each tree within the XGBoost ensemble while the max leaves parameter specifies the maximum number of leaves for each node. Gamma defines the cost of introducing another leaf, balancing the trees within the ensemble. The gamma parameter becomes particularly relevant once confining tree heights as it is an "across trees" parameter (as opposed to a "within trees" parameter which only affects each individual tree). Therefore, the gamma parameter ensures that each additional leaf is purposeful rather than simply attaining the maximum depth and maximum height within each tree in which case the model would likely overfit.

Overall, we found that regression models gave us more desirable results than classification models. The highest test accuracy among all our classification models was 56.5%. It makes sense that classification models would not be suitable for our task, given that the average ratings among restaurants is a continuous variable and splitting it up into classes mostly involves setting arbitrary dividing lines. Among our regression models, the XGBoost regressor performed optimally as the simplified feature set provided the best balance between model complexity and

overfitting prevention. Overall, with this model, we were able to predict the average rating within around 4% of the ground truth, demonstrating that our model was successful. Although there's always room for improvement, our model serves as a good starting point for this task and we're satisfied with the results.

Sources

Brownlee, J (2021). A Gentle Introduction XGBoost for Applied Machine Learning. https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/

Dey, L., Chakraborty, S., Biswas, A., Bose, B., & Tiwari, S. (2016, October 31). Sentiment analysis of review datasets using naive Bayes and k-nn Classifier. arXiv.org. https://arxiv.org/abs/1610.09982v1

Jonathan Montomoli et al (2021). Machine learning using the extreme gradient boosting (XGBoost) algorithm predicts 5-day delta of SOFA score at ICU admission in COVID-19 patients. National Library of Medicine. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8531027/.

Prediction of covid-19 possibilities using KNN classification algorithm. https://assets.researchsquare.com/. (n.d.). https://assets.researchsquare.com/files/rs-70985/v2_stamped.pdf

Okamura, Scott. "GRIDSEARCHCV for Beginners." Medium, Towards Data Science, 30 Dec. 2020, towardsdatascience.com/gridsearchcv-for-beginners-db48a90114ee.

Munir, Samira. "Basic Binary Sentiment Analysis Using NLTK." Medium, Towards Data Science, 27 Mar. 2019, towardsdatascience.com/basic-binary-sentiment-analysis-using-nltk-c94ba17ae386.

Yan, A., He, Z., Li, J., Zhang, T., & McAuley, J. (2022). Personalized showcases: Generating multi-modal explanations for ... cseweb.ucsd.edu. https://arxiv.org/pdf/2207.00422.pdf