CS 273P Final Project Report

McManigal, Connor
mcmanigc@uci.edu

Politewicz, Peyton
peyton.p@uci.edu

Ng, Harold
haroldn1@uci.edu

Abstract

Social media is ubiquitous: it's nearly unavoidable in one's digital life, and user interactions contribute behind-the-scenes to search engine rankings and recommender optimization. Our team is using data provided to the UCI Machine Learning Library by the paper, "Textual Taste Buds: A Profound Exploration of Emotion Identification in Food Recipes through BERT and AttBiRNN Models". This data concerns 18,000 reviews of food recipes, and while the original paper was using these to compare sentiment analysis models, we attack a different angle: predicting the algorithmic 'weight' of recipe review comments. We attempt to reverse-engineer or at least partially infer why the hosting website assigns higher algorithmic scores to some comments and not others, a crucial metric for determining display order of user-submitted content, and thus what an average user of a given website might see. Can NLP, particularly sentiment and objectivity analysis, help us isolate comments that an algorithm would boost to the top of the chain? Does adding this layer of data to the set help us predict algorithmic score of comments at all? To answer these questions, we incorporate tools from two popular libraries: VADER and TextBlob. These libraries provide polarity and subjectivity scores for each review, allowing us to leverage additional textual information.

1 Problem Statement

Content reviews play a pivotal role in shaping user engagement and satisfaction on social media platforms. Accurate prediction of algorithmic weight assigned to user comments enables platforms to prioritize relevant and high-quality content, thereby enhancing user experience. Our task involves predicting the best score associated with a recipe review, however, one of the primary challenges we face is the scarcity of data.

2 Methodology

To tackle this problem, we explored two machine learning algorithms: Multilayer Perceptron (MLP) Regressor and Gradient Boosting Regressor. These models were trained on a dataset comprising recipe reviews, their associated features, augmented features, and sentiment scores. We carefully selected hyperparameters and fine-tuned the models to optimize their performance.

3 Data Ingestion and Preprocessing

The dataset contains approximately 18,000 review comments and 15 features for each.

- Descriptive features include recipe details (number, code, name), username of the poster, unique IDs for the user and comment, a posting timestamp, the user's submitted recipe rating on a zero-to-five scale(stars), and the text body of the comment itself.

Preprint. Under review.

- Quantitative features directly used in later analysis and prediction include reply count, thumbs up and thumbs down the comment has received, a user reputation score from the time of posting which indicates the quality of their previous contributions, and 'best_score', the algorithmic score given to the comment.
- 'best_score' is the response variable we concern ourselves with in our project. Its baseline value is 100, and demonstrates notable 'clumps' at arbitrary values near every 50-point increment (i.e. 193). Beyond 300, these disappear. Some descriptive statistics of best score include a minimum of 0 and a maximum of 946, a mean of approximately 153 and a median of 100.

3.1 Notes on best_score

This algorithm-boosting score metric has some interesting characteristics which give the prospect of predicting its value some depth. First, let's look at some histograms of best_score's value distributions.

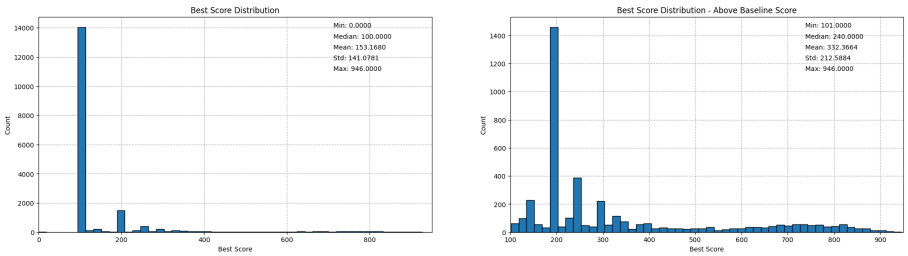


Figure 1: Histograms of best_score. The left includes all records. The right only includes records that are above the baseline score (i.e. best_score > 100).

We can see, immediately, that approximately 14,000 of the records are at that baseline score - 100. Very few are below it (indicating negligibly few comments are ever de-boosted). This indicates that there is a tremendous amount of 'noise' in terms of what comments are going to be shown to a prospective user. Removing comments at or below baseline, we still see some massive peaks coming around those arbitrary clumping points mentioned above, but this is a much more reasonable spread across the 4,000 remaining comments.

From here, let's take a look at what a reasonable person might immediately identify as a good predictor for an algorithm's internal scoring metrics: upvotes. Surely, if many other users like a post, shouldn't it be shown it to others?

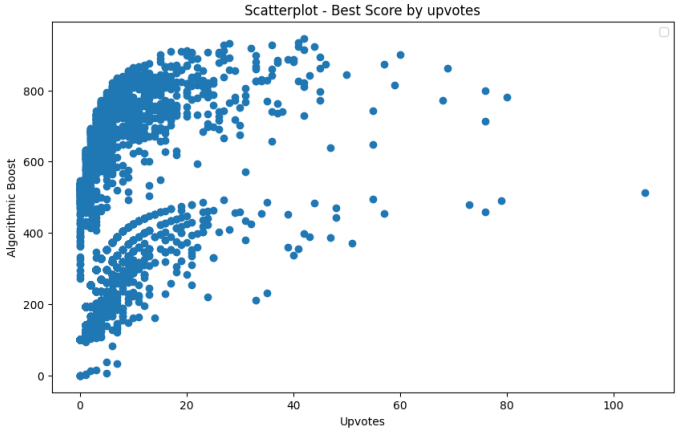


Figure 2: Scatterplot of best_score against upvotes.

We can see there is some positive trend here, but this feather-shaped scatterplot reveals some additional nuance. There is a wide margin for best_score at every level of user feedback. We're seeing comments

with absolutely no interactions that have huge algorithmic boost scores. Our work is directed toward unearthing those additional layers, and testing whether or not data augmentation and NLP can help break through some of them.

3.2 Feature Engineering

To conduct our work we synthesized a number of features from the given data, as well as conducting semantic analysis on the comment text itself.

1. **Feature Extraction**: We extracted relevant features including 'votescore', 'thread_index', and 'review_length'. Votescore captures the overall popularity of a review. This was calculated by subtracting the number of thumbs down votes from thumbs up votes. A higher votescore suggests a more positively received review, while a negative score indicates a less favorable response from users. Thread_index represents the sequential ordering of reviews within each recipe thread, thus providing temporal progression of reviews and their relative positioning in the thread, allowing us to assess the search-engine optimization conventional wisdom of 'does the early bird catch the worm?' with algorithmic attention. Review_length quantifies the number of characters within a given review. Longer reviews may contain more detailed information or opinions.
2. **Sentiment Analysis**: We utilized VADER and TextBlob libraries to compute polarity (positive/negative sentiment) and subjectivity (degree of objectivity) scores for each review. In particular, VADER's polarity score is well suited for social media analysis because it can capture nuanced sentiment and slang. TextBlob offers a polarity score, similar to VADER, but also a subjectivity score which aims to capture the degree to which text expresses opinions or emotions, classifying text as subjective or objective.

3.3 Clustering

There were a number of attempts to use unsupervised learning to unearth user clusters, based on indications that clustering was an effective tool for connecting similar users together per one of the speakers at the MDS seminar series last fall. However, this data proved particularly resilient to analysis by both Gaussian Mixture Models and K-Means clustering methods, largely because of the effect we see in the feather-shaped plot in figure 2. At almost any metric, there is a large 'window' of potential best_score results. The clusters in almost every comparison echoed this, demonstrating clear 'vertical slices' at different levels of score. However, one chart in particular was helpful for general insight purposes.

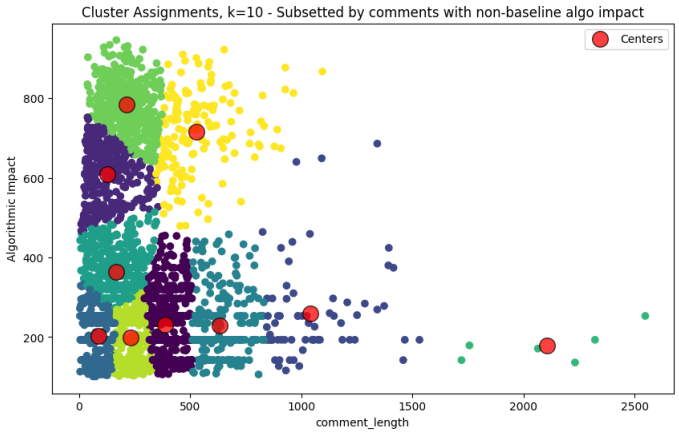
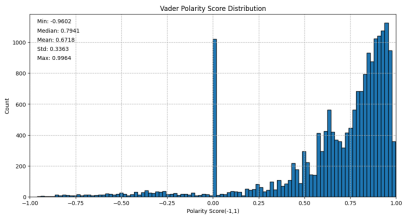


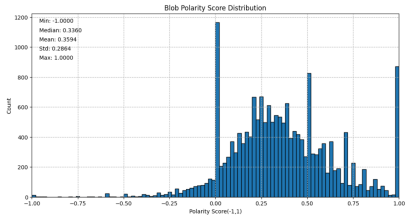
Figure 3: Clustering applied to best_score charted against comment length.

Here, we can see that the optimal comment (for algorithmic purposes) maintains some principle of brevity. Looking at the clustering results, there appears to be a penalty for excessively long comments.

3.4 Sentiment



(a) Vader polarity score histogram



(b) Blob polarity score histogram

Figure 4: Polarity score histograms

Above are histograms illustrating the distribution of polarity scores. In our exploration of the polarity scores, we uncover intriguing patterns that shed light on user sentiments within recipe reviews. Here are the key characteristics revealed by the graph:

- Positive Skew: The distribution of polarity scores exhibits a pronounced skew toward positive sentiment. Most reviews fall within the range of 0.75 to 1.0 on the polarity scale. This suggests that the majority of users express favorable opinions when discussing recipes. The high median score of 0.7941 and mean score of 0.6718 reinforce this positivity trend.
- Neutral Gap: Interestingly, there is a conspicuous gap in scores around the neutral mark (score of 0). Few reviews express neutral sentiments, indicating that users tend to lean either positively or negatively. The absence of neutrality implies that recipe discussions evoke distinct emotions, with users rarely sitting on the fence.
- Sparse Negative Scores: Negative polarity scores are scarce. Users are less likely to leave outright negative feedback. This aligns with the observation that people generally prefer to highlight positive aspects rather than dwell on shortcomings. Outliers: Extreme negative scores (outliers) exist, but they are infrequent. The minimum score of -0.9602 represents these rare instances of strong negativity.
- NLP Considerations: Surprisingly, the correlation between polarity scores and our response variable (best_score) is weak. Despite leveraging sentiment analysis, we find that polarity alone doesn't strongly influence the review outcome. This challenges our initial assumption that NLP would significantly impact user preference.

In summary, while polarity scores provide valuable insights, their direct impact on recipe review scores remains subtle. As we delve deeper into our algorithms, exploring neural nets and other advanced techniques, we may unearth hidden connections that go beyond surface-level sentiment analysis.

3.5 Final Insights from Exploratory Data Analysis

With all of these factors in place, we use a correlation plot to see what we might expect to prove most potent for predicting our response variable, best_score. Its relevant cells are highlighted in the following figure.

Here, we see that our strongest indicators of algorithmic boost relate to the interactions a comment generated. In this case, having your comment down voted has just as much impact on its position in the algorithm as generating a reply. Also notable is that the polarity and subjectivity scores demonstrate a very weak correlation to our response variable. Even if the algorithm doesn't use NLP in its processing explicitly, we expected to see some clear relationship here that gave texture to user preference - do people prefer subjective conversations, speaking about how good or delicious a recipe was, or objective feedback, like comments that offer adjustments to ingredient portions, nutritional assessments, or similar recipes summarized in the discussion chain? This plot indicates NLP might not actually suit our purposes as well as we thought - but it's worth testing, especially with neural nets that might be able to detect patterns beyond this elementary layer of analysis.

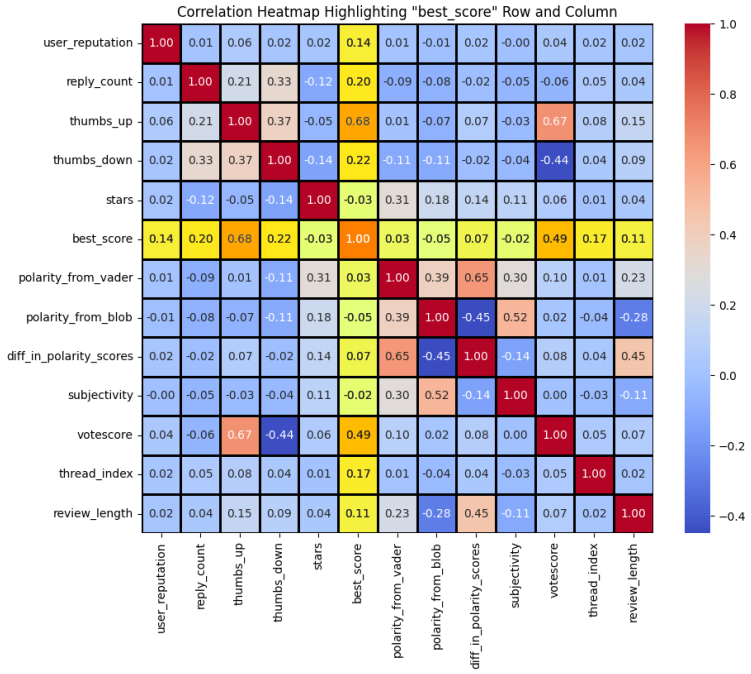


Figure 5: Correlation heatmap with cells relevant to the response variable highlighted.

4 Model Selection and Training

1. **Algorithm Selection**: We chose two algorithms to tackle our problem, Multi Layered Perceptron and Gradient Boosting Regressors, due to their flexibility and ability to capture complex relationships. MLP Regressor, a type of neural network, can often automatically identify hierarchical representations of features, thus potentially capturing intricate non linear patterns within the data. GB Regressor, a type of ensemble, is robust to overfitting since it builds upon many weak learners, which leads to strong predictions that iteratively improve model performance.
2. **Training**: The models were trained on a labeled dataset, where the target variable was the best_score; a discrete quantitative value reflecting the recipe website's algorithmic score assigned to the review. We trained two sets of models that differed in training sets. The first set of MLP and GB Regressors was trained on important features provided in the original dataset, including 'user_repuation', 'reply_count', 'thumbs_up', and 'thumbs_down'. For these plain models, we trained the MLP and GB Regressors using an identical 80/20 split of the data without stratification. The second set of MLP and GB Regressors was trained on the original features listed above in conjunction with our augmented data: 'polarity_from_vader', 'polarity_from_blob', 'subjectivity', 'votescore', 'thread_index', and 'review_length'. For the augmented data models, we stratified the training data by our VADER polarity score with a custom metric to ensure a balanced representation of positive, neutral, and negative sentiments, with identical 80/20 splits for the MLP and GB Regressors. This was done with the hope of enhancing the robustness and generalization capabilities of these models. For the MLP Regressors, we used scikit-learn pipelines to standardize features (through StandardScalar) to ensure consistent preprocessing and to facilitate streamlined integration with cross-validation and grid search. We did not use pipelines for the GB Regressors, because these models are inhernetly robust to feature scaling due to their ensemble nature.
3. **Hyperparameter Tuning and Cross-Validation**: We utilized scikit-learn's GridSearchCV to simultaneously train, select hyperparameters, and cross validate our models. We opted for grid search despite its clunkiness and slow performance because of its thoroughness, as it exhaustively searches the parameter grid. This library works by taking in an instance of the MLP Regressor pipeline or GB Regressor, a grid of hyperparameters to be searched upon, the number of folds for cross validation, and a desired scoring metric. For our MLP Regressors,

we used identical but separate parameter grids that included a ReLU activation function, an 'adam' adaptive gradient solver, a mixture of hidden layer sizes((100,), (100,50), (50,50)), and several regularization alphas (0.0001, 0.001, 0.01). In regard to our GB Regressors, we used parameter grids that included a few number of estimators (100, 200, 300) and several learning rate coefficients (0.01, 0.1, 0.2). During cross-validation, the training data was subdivided further across 5 folds, resulting in a total of 45 fits across all our algorithms (9 candidates for each). This means that the 80% of the data designated for training was further divided into 60/20 splits across each fold with 60% to train and 20% to validate each fold.

4. **Evaluation Metrics**: We assessed model performance using the following metrics: Mean Absolute Error (MAE), Mean Squared Error (MSE). We primarily used MAE as our error metric because it is less sensitive to outliers. We recorded the average cross-validation MAE scores across our candidate models and have plotted them below.

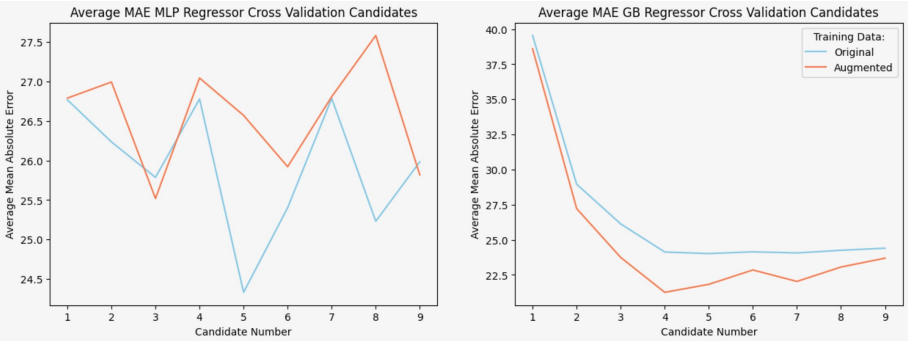


Figure 6: Graphs showing the results of cross validation for both methods - MLP and GB.

The cross-validation candidates for both models are presented above. The GB Regressor clearly demonstrates an intuitive solution at its elbow, whereas the shifting hyperparameters for the MLP regressor follow a less distinct pattern.

5 Results

Results				
Model	Layers	Estimators	Learning Rate	MAE
MLP: Orig.	100,50	N/A	0.001	25.286
MLP: Aug.	50,50	N/A	0.0001	**22.672**
GB: Orig.	N/A	200	0.1	24.618
GB: Aug.	N/A	100	0.1	**21.446**

5.1 Model Comparison

The GB Regressor, when combined with sentiment features and data augmentation, outperformed other approaches(MAE of 21.446). Its ability to handle complex relationships and adapt to the augmented data contributed to its success. The MLP Regressor, while competitive, fell slightly short with an MAE of 22.672.

The success of the Gradient Boosting Regressor (GB Regressor) can be attributed to several key factors. Firstly, its inherent capability to capture complex relationships within the data proved invaluable, especially when combined with the sentiment features extracted using VADER and TextBlob. The GB Regressor excels in handling non-linear relationships and intricate patterns present in the review data, allowing it to effectively model the nuances of recipe feedback. Moreover, the incorporation of data augmentation techniques further bolstered its performance by providing a richer and more diverse training dataset. By synthesizing additional instances from the existing data, the model gained exposure to a wider range of scenarios, enhancing its robustness and generalization capabilities.

Furthermore, the ensemble nature of the Gradient Boosting algorithm played a significant role in its success. By aggregating the predictions of multiple weak learners, each iteratively improving

upon the residuals of its predecessors, the GB Regressor was able to refine its predictions iteratively, leading to more accurate estimations of recipe review scores. This ensemble approach not only mitigates overfitting but also ensures that the model adapts dynamically to the complexities present in the data, resulting in superior predictive performance.

Conversely, while the Multilayer Perceptron (MLP) Regressor demonstrated competitive performance in this task, it fell slightly short compared to the GB Regressor. One reason for this discrepancy could be the relatively simpler architecture of the MLP model compared to the ensemble-based approach of Gradient Boosting. MLPs are powerful models capable of learning complex relationships, but they may struggle to generalize as effectively as ensemble methods, particularly in scenarios with limited training data or high-dimensional feature spaces. Additionally, the MLP Regressor may have been more susceptible to overfitting, especially in the absence of extensive regularization techniques or data augmentation strategies.

Despite its slightly lower performance, the MLP Regressor still yielded promising results, indicating its potential utility in similar prediction tasks. Its neural network architecture enables it to capture intricate patterns in the data and adapt to various input modalities, making it a versatile choice for predictive modeling tasks. However, for the specific objective of predicting recipe review scores with enhanced accuracy and robustness, the Gradient Boosting Regressor emerged as the superior choice, leveraging its ensemble framework, adaptability to augmented data, and ability to capture complex relationships to achieve exemplary performance.

6 Conclusion

Our hypothesis that sentiment scores would enhance predictive power was confirmed. By leveraging sentiment information, we achieved better performance metrics, demonstrating the importance of considering both textual content and sentiment in recommendation systems.

While our study successfully validated the efficacy of sentiment scores in augmenting predictive power for recipe review scores, it's crucial to acknowledge certain limitations that could be addressed in future research endeavors. One notable limitation is the reliance on existing sentiment analysis libraries like VADER and TextBlob, which may not capture the nuances of culinary language and context-specific sentiments as effectively as domain-specific sentiment analysis models. Developing or fine-tuning sentiment analysis models tailored specifically for recipe reviews could potentially yield even more accurate sentiment scores, thereby further enhancing predictive performance.

Moreover, while augmented features played a significant role in improving model performance, exploring additional features beyond textual content and sentiment could provide valuable insights and potentially boost predictive accuracy. For instance, incorporating metadata such as recipe category, cooking time, ingredient lists, or user demographics could offer supplementary information that enriches the predictive capabilities of the model. Additionally, exploring advanced text representation techniques like word embeddings or contextualized embeddings could capture more nuanced semantic relationships within the review text, potentially leading to more nuanced and accurate predictions.

Furthermore, the evaluation of our models was limited to traditional error metrics such as Mean Absolute Error and Mean Squared Error. While these metrics provide valuable insights into model performance, incorporating additional evaluation measures such as ranking-based metrics or user satisfaction metrics could offer a more holistic assessment of recommendation system performance in real-world scenarios.

In conclusion, while our study demonstrates the efficacy of sentiment scores and data augmentation in enhancing predictive power for recipe review scores and identifies the GB Regressor as the top performer in this task, there remains ample opportunity for further exploration and refinement. Future research endeavors could focus on refining sentiment analysis techniques, incorporating additional features, exploring advanced text representation methods, and employing comprehensive evaluation strategies to continue advancing recipe recommendation systems.

7 Statement of collaboration

- Connor McManigal
  - Exploratory data analysis - code
  - Feature Engineering - code
  - Feature extraction and sentiment analysis - write up
  - Model selection, training, and stratification - code
  - Model selection and training - write up
  - Cross validation, CV plots, and testing - code
- Harold Ng
  - part of Abstract
  - problem statement
  - methodology
  - Sentiment
  - part of Model Selection and Training
  - Model Comparison
  - Conclusion
- Peyton Politewicz
  - Problem statement and hypothesis construction - writeup/ideation
  - Exploratory data analysis - code
  - Correlation analysis - code
  - GMM and K-Means clustering - code
  - Filtering and further analysis of best score - code
  - Model results extraction - code
  - Abstract, data preprocessing, model selection and training - writeup

8 References

[1] Amir Ali, Stanisław Matuszewski, Jacek Czupyt, Usman Ahmad. (2023). Textual Taste Buds: A Profound Exploration of Emotion Identification in Food Recipes through BERT and AttBiRNN Models. International Journal of Novel Research and Development. Volume 8. ISSN: 2456-4184.

[2] Connor Mcmanigal, Peyton Politewicz, Harold Ng. (2024). Recipe_Review_ML. Retrieved from https://github.com/connormcmanigal/Recipe_Review_ML