Predicting User Ratings for Recipes Utilizing Machine Learning Models

ABSTRACT

To enhance personalized recommendations on recipe websites and platforms, it's important to be able to predict user ratings for recipes. This study uses various machine learning models to predict recipe ratings, considering variables such as preparation time, number of ingredients, and the number of steps. These structured features are combined with text data from the user reviews using language processing techniques, such as TF-IDF, to turn categorical text data to quantifiable statistical information. Multiple predictive models were implemented to test their efficiency at capturing user preferences, such as Logistic Regression, Multinomial Naive Bayes, Random Forest Classifier, and XGBoost.

In order to optimize model performance, evaluation metrics such as accuracy, precision, recall, F1-score, mean absolute error (MAE), and mean square error (MSE) and GridSearchCV hyperparameter tuning were used. Our results demonstrate that incorporating textual features from reviews greatly improves predictive accuracy, which shows a stronger correlation between user-generated content and rating prediction ability. These findings may not represent the potential of including diverse data sources to develop comprehensive recipe recommendation systems. Future research may be necessary to explore additional factors, such as user demographics or social interactions, to enhance the model's predictive capabilities further.

INTRODUCTION

Online recipe platforms have transformed the culinary experience, allowing people to easily discover and share ideas and experiences with one another. Platforms with millions of recipes and corresponding reviews offer an extensive dataset that help understand user behaviors and preferences. A key challenge for such platforms is the ability to predict user ratings for recipes. An accurate rating prediction can significantly enhance recommendations for each user, helping them find the best recipes for their dietary needs and personal tastes.

This study examines the issue of recipe rating predictions using a combination of variables, such as preparation time, cooking steps, and number of ingredients, alongside user inputted textual data. We hope to use these diverse data sources to build models that capture the explicit and nuanced correlations influencing user ratings.

Machine learning models demonstrate great efficiency in similar systems for streaming platforms or e-commerce, but recipe rating prediction carries unique challenges due to the different build of recipes and the personalization of user preferences. For this study, we implemented various predictive models, including Logistic Regression, Multinomial Naive Bayes, Random Forest, and XGBoost to identify the most effective predictor for this task.
The main goals of this study are to:

1. Investigate the correlation between different variables and the predicted rating
2. Compare the performance rates of different models for predicting recipe ratings
3. Optimize the best-performing model through hyperparameter tuning

The results from this study implicate improvements for recipe recommendation systems, making them more personalized and user oriented. By using the vast data available to us on recipe platforms, we hope to contribute to the improvement of recipe recommendation systems.

DATA PRE-PROCESSING

For this study, we used a dataset composed of both user reviews and recipe information. This dataset includes variables such as number of ingredients, preparation time, and number of steps. It also includes unstructured data such as user reviews, recipe descriptions, and tags.

The raw data that was given was split into five files that held recipe data, review data, training, testing, and validation data sets. The original training data had a much higher percentage of five star ratings compared to the other data sets, so we decided to combine and re-split the training, validation, and testing data sets.

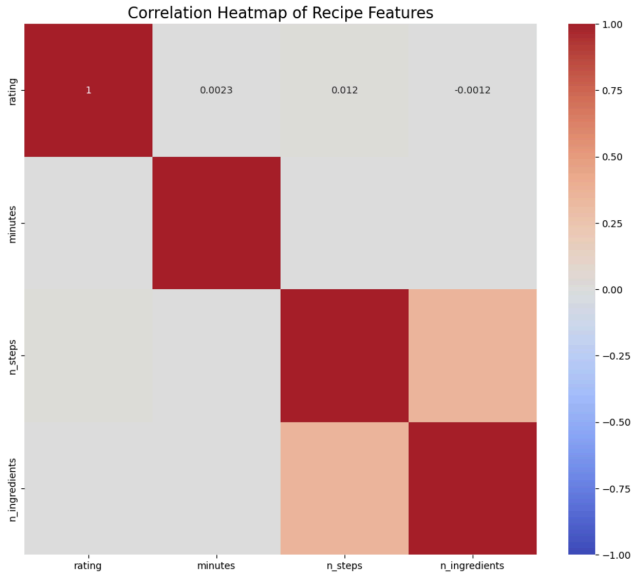
As the files were merged to match a review to the recipe, several rows were lost because of unmatched user_id, recipe_id, rating, or date and other rows had to be removed because they did not have a review text associated. Many recipes also had a null description, but these data entries were kept because we did not evaluate based on the description column, leaving us with nearly 720,000 data entries.

In addition, a portion of the user review data had a rating review of 0, which largely showed positive user input, which implies that the user intended to leave a positive rating, but forgot to provide one. To simplify the prediction task, we decided to remove all the reviews with a dataset rating of 0. By removing all the uncertain data points, we avoid all the unnecessary noise and improve the interpretability of the dataset.

On top of that, removing all the data points with a rating of 0 allows us to see the problem as a classification model with discrete labels: 1, 2, 3, 4, and 5. This decision allows the problem to be more tractable while ensuring that the correlation between review content and the corresponding labels are consistent.

DATA EXPLORATION

For the exploratory data analysis (EDA), we hope to understand the patterns in user behavior, understand the distribution of ratings, and identify potential challenges in tuning predictive models.



From this correlation heatmap, we can see that there is no correlation between the rating and these features, which is why for the rest of our analysis we chose to analyze the text inside the user review data and used language processing techniques.

A key observation in the EDA was the obvious skew in the user ratings. A vast majority of the ratings were concentrated around 4s and 5s, with much less reviews with a rating of 1, 2, or 3.

The pattern demonstrates that users tended to review recipes more positively. There may be a few factors that induce this trend.

1. Users tend to rate recipes that they have tried, which usually implies that the user found the recipe appealing, resulting in a selection bias in the dataset.

2. Other studies have shown that positive experiences are more likely to induce users to engage in reviews, compared to neutral or negative experiences.
3. Due to the creative and subjective nature of cooking, recipes may induce higher ratings even when recipes are adjusted to serve each user's unique purposes.

The skew in the data distribution has major implications for model training. Classification models trained to predict recipe ratings are now biased to predict more favorably for all recipes, which leads to lower predictive accuracy for underrepresented ratings, such as 1, 2, or 3. It is important to address this imbalance by improving model predictions across all datasets through oversampling minority data, using advanced loss functions, or class weighting.

Significance of Insights

It is important to understand the patterns in our data so that we can shape our models appropriately. The skew in the ratings and the decision to exclude ratings of 0 influence the ratings of the machine learning models, data preprocessing techniques, and evaluation metrics. These insights also highlight the importance of taking into account user behavior and biases when working with real-world datasets.

Accounting for these factors, we hope to build models that perform well in both predicting recipe ratings and also ensuring robust predictions across the entire rating class range. This balanced approach is crucial for improving user experience on recipe platforms.

MODEL SELECTION

The baseline model we chose was to predict the average rating of the training set. While simplistic, this baseline offers a meaningful benchmark, particularly given the skewness of the dataset toward ratings of 4 and 5. A model must outperform this baseline significantly to demonstrate added value.

We evaluated four models for the recipe rating prediction task: Logistic Regression, Multinomial Naive Bayes, Random Forest, and XGBoost. The primary feature used to train these models was the **review text** provided by users, processed through TF-IDF vectorization to extract meaningful textual patterns. The tables below summarize their performance on the testing dataset, comparing Mean Absolute Error (MAE), Mean Squared Error (MSE), and classification metrics such as accuracy, precision, recall, and F1 score.

Classifier	Testing MAE	Testing MSE
Average Rating Baseline	0.48	0.42
Logistic Regression	**0.22**	**0.28**
Multinomial Naive Bayes	0.31	0.51
Random Forest	0.29	0.45
XGBoost	0.89	0.97

Logistic Regression achieved the lowest MAE and MSE, significantly outperforming the baseline and other models. Multinomial Naive Bayes and Random Forest also outperformed the baseline, but to a lesser degree. XGBoost performed the worst on both metrics, with an MAE of 0.89 and an MSE of 0.97.

The tables below show the different performance on the training set and testing set:

Classifier	Training Accuracy	Precision	Recall	F1 Score
Logistic Regression	82%	79%	82%	79%
Multinomial Naive Bayes	76%	71%	76%	66%
Random Forest	**99.8%**	**99.8%**	**99.8%**	**99.8%**
XGBoost	15%	3.4%	15%	5.5%

Classifier	Testing Accuracy	Precision	Recall	F1 Score
Logistic Regression	**80%**	**77%**	**80%**	**77%**
Multinomial Naive Bayes	76%	67%	76%	66%
Random Forest	77%	72%	77%	69%
XGBoost	15%	3.7%	15%	5.8%

RESULTS

Logistic Regression consistently delivered the best performance across all metrics. By leveraging the review text as the main feature, this model effectively captured patterns in user sentiment and tone that correlate with ratings. Its simplicity, coupled with TF-IDF vectorization for text preprocessing, allowed it to generalize well to unseen data.

Strengths for Logistic Regression is its performance on both training and testing datasets and relatively low computational complexity. Weaknesses is that it can struggle with non-linear relationships that could be present in more nuanced review text.

Multinomial Naive Bayes performed moderately well, as some strengths for this model is its efficient and interpretable, particularly well-suited for text classification tasks. Weaknesses is its inability to capture complex patterns in textual data that can reduce its overall effectiveness.

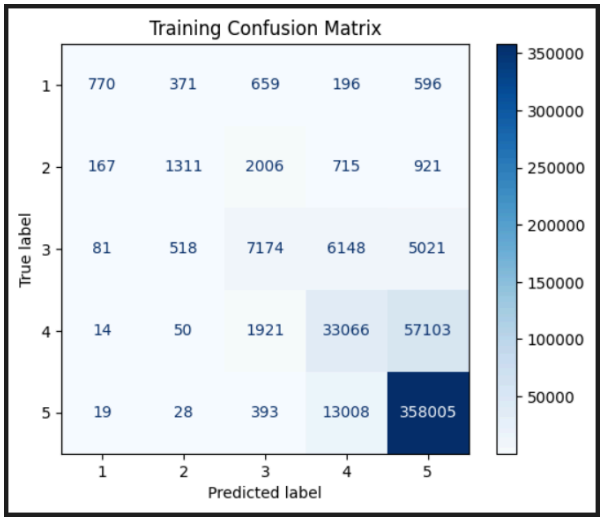
Random Forest achieved high performance on the training set but exhibited a noticeable drop in metrics on the testing dataset, indicating overfitting. While it effectively learned intricate patterns from the text, it struggled to generalize due to the sparsity and high dimensionality of TF-IDF features.

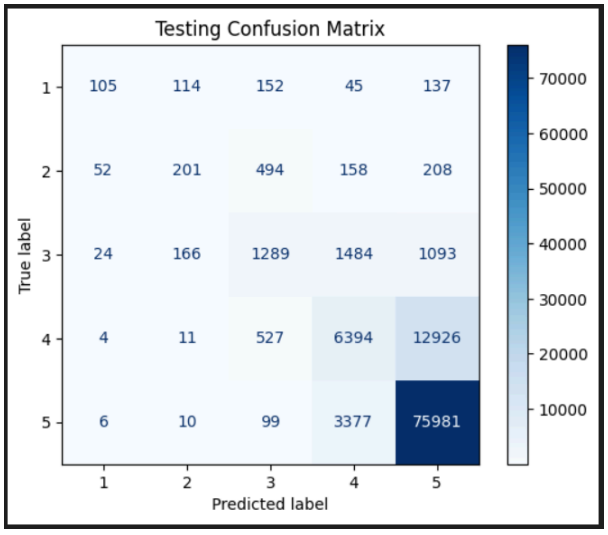
Strengths for this model is its ability to capture non-linear relationships and handling complex interactions in the text. And weaknesses are prone to overfitting on high-dimensional text data and computationally expensive.

Despite its reputation for strong performance in structured data tasks, XGBoost performed poorly in this study. Its inability to handle the high dimensionality and sparsity of TF-IDF features without extensive tuning likely contributed to its poor metrics.

Its strengths are effectiveness for structured data when optimized and handles non-linear relationships well. Weaknesses are sensitive to hyperparameters and preprocessing. In this case, it struggled to adapt to the nature of textual data represented through TF-IDF

In the following confusion matrices, we can see that the model is accurately predicting most of the recipe ratings. There is some variation, where the predictor may predict +1 or -1 from the actual rating, but it generally predicts in the correct range. However, there is a large number of reviews with a true rating of 1, but the predicted label is 5 or 4. This trend is seen in the training data as well as the testing data.





HYPERPARAMETER TUNING

To optimize the Logistic Regression model, we conducted hyperparameter tuning with grid search and cross-validation, focusing on both the Logistic Regression parameters and the TfidfVectorizer settings. The goal was to find a combination that maximizes predictive performance by effectively extracting information from the review text and ensuring the model generalizes well to unseen data. Below, we outline the key parameters tuned and their significance.

The ‘C’ parameter in Logistic Regression controls the strength of regularization, which helps prevent overfitting by penalizing large coefficients. It is the inverse of the regularization strength, meaning smaller values of C impose stronger regularization.
Values Explored: [0.01, 0.1, 1, 10, 100]
Impact: Balances model complexity and generalization ability.

The ‘penalty’ type specifies the norm used for regularization. Common options include l1 (Lasso) and l2 (Ridge), which encourage sparsity and small coefficients, respectively.
Values Explored: ['l1', 'l2']
Impact: Determines how coefficients are constrained to handle high-dimensional features effectively.

The ‘solver’ determines the optimization algorithm used to train the model. Some solvers, such as liblinear, work well with small datasets or l1 penalty, while others, like saga or lbfgs, are suitable for larger datasets and support both l1 and l2.
Values Explored: ['liblinear', 'saga', 'lbfgs']
Impact: Ensures compatibility with the chosen penalty and improves training efficiency.

The max_features parameter in Tfidf limits the number of features (unique words) included in the feature space, retaining only the most informative ones based on term frequency.
Values Explored: [500, 1000, 5000, None]
Impact: Reduces noise and simplifies the model without sacrificing essential textual information.

Results of Hyperparameter Tuning:
We performed grid search over the combinations of the above parameters using 3-fold cross-validation. The best configuration identified was:
C: 1, Penalty: l2, Solver: saga, max_features: None

This combination of parameters led to an increase in our performance metrics as seen below:

Logistic Regression	Training Accuracy	Precision	Recall	F1 Score
Pre-tuning	79.93%	76.84%	79.93%	77.37%
Post-tuning	79.97%	76.88%	79.97%	77.37%

LITERATURE REVIEW

The dataset we used to tune and test our model is sourced from Food.com (formerly GeniusKitchen). It contains over 231,000 recipes and 1.1 million user reviews with detailed information about each recipe, such as its name, description, ingredients, preparation steps, nutritional facts, and tags or categories. Originally, the data was crawled for a study that generated personalized recipes from historical user preference. A model was trained to consider a user’s previously consumed recipes, recipe name, and incomplete recipe ingredients to generate a recipe that the user might be interested in.

The Food.com dataset has also been used for various studies, aimed at understanding culinary preferences, user responses, and recipe personalization. Beyond the original authors’ use of the data, here are three other studies that were conducted using this dataset.

1. Jimmy Nguyen et al.[1] uses the dataset to predict food recipe ratings based on features such as preparation steps and ingredients used. Their study aimed to optimize prediction accuracy through regression techniques and feature engineering. Of all the studies conducted on this dataset, this one is most similar to the study we are doing.
2. Gowshitha A.[2] explored the classification of recipe ratings using machine learning techniques, comparing various models to determine the most effective one. Her study emphasizes preprocessing steps such as ingredient vectorization and feature reduction.
3. Ramesh K et al.[3] uses the data set to build a machine learning model approach to predict food recipe ratings. They use advanced feature extraction techniques and ensemble learning techniques to calibrate model performances.

For our own study, we explored the dataset using state-of-the-art machine learning techniques, including logistic regression, XGBoost, random forest, and multinomial naive bayes. These models each have unique strengths and weaknesses, so by testing each one of them,

1 https://github.com/jimmy-nguyen-data-science/Predict-Food-Recipe-Ratings/tree/main

2 https://medium.com/@gowshitha123/food-recipe-rating-classification-using-machine-learning-a28a233828ce

3 https://www.researchgate.net/publication/351505321_An_Intelligent_Approach_for_Food_Recipe_Rating_Prediction_Using_Machine_Learning

we can find the best balance of simplicity, interpretability, and predictive power:

When we compared our study to others', we found that they were similarly aligned in identifying key predictors of recipe ratings (user input was most correlated to recipe rating). Akin to Jimmy Nguyen et al.'s findings, our results also demonstrated that a logistical regression model proved to be most accurate for predicting the recipe rating. However, we found that our accuracy was higher than their model. We tuned our model to return an accuracy of about 0.8, which was better than their accuracy of 0.7. Our finding also echoed those of Ramesh K. et al., affirming the importance of preprocessing the data and extracting important features such as the review text using Tfidf Vectorization.

CONCLUSION

Our study explored the task of predicting recipe ratings using review text as the primary feature, evaluating multiple models: Logistic Regression, Multinomial Naive Bayes, Random Forest, and XGBoost. Among these, Logistic Regression emerged as the best-performing model, significantly surpassing both the baseline and other classifiers in terms of accuracy, precision, recall, and F1-score.

Performance Comparison

Baseline Model: Predicting the average rating of the training set provided a simple yet effective benchmark. However, while it had a low MSE (0.42), it lacked predictive granularity, failing to account for individual review nuances.

Logistic Regression: This model excelled due to its ability to effectively leverage textual features represented through a tuned TF-IDF vectorizer. Its testing accuracy (80%) and F1-score (77%) were the highest among all models. The balance between simplicity and interpretability made it a robust choice for this task.

Multinomial Naive Bayes: While lightweight and efficient, this model struggled to capture the subtle complexities in text due to its reliance on word independence assumptions. Consequently, it achieved lower accuracy (76%) and F1-score (66%).

Random Forest: Despite its high training performance (near 100% across all metrics), Random Forest overfit the data and generalized poorly, with a testing accuracy of 77%. Its reliance on numerous weak learners made it less effective in capturing the nuances of textual data.

XGBoost: Surprisingly, XGBoost performed the worst, with a testing accuracy of just 15%. This failure likely stemmed from the limited size and feature sparsity of the dataset, which are not optimal conditions for gradient-boosting algorithms.

Feature Representations

The primary focus of this study was on review text as the key feature for predicting recipe ratings, with text data being transformed into numerical representations using the TF-IDF (Term Frequency-Inverse Document Frequency) vectorizer. TF-IDF proved to be the most effective feature representation, capturing the relative importance of words and phrases within the reviews while filtering out common but less informative terms.

In contrast, during our EDA, recipe features such as n_steps, n_ingredients, and minutes were found to be weak predictors of ratings. These features primarily describe the structure or preparation of a recipe rather than its qualitative outcome or user satisfaction.

Recipes with a higher number of steps or ingredients did not consistently correspond to better or worse ratings, as these attributes can be subjective. Some users may prefer simplicity, while others might value complexity. The preparation time showed no significant correlation with ratings, as users' preferences for quick or elaborate recipes vary widely.

The insufficiency of recipe features underscores the importance of using the review text, which directly reflects user opinions, as the primary feature for this task. By leveraging textual data, our models could analyze patterns in language that better aligned with users' expressed satisfaction or dissatisfaction.

In conclusion, the TF-IDF representation of review text was instrumental in achieving high predictive performance, while recipe-level features provided minimal additional value for predicting ratings. This aligns with the intuition that subjective factors, as expressed in textual reviews, are more indicative of user sentiment than objective recipe characteristics.

Takeaways

The success of Logistic Regression stems from its simplicity, robustness, and ability to scale well with textual features represented by TF-IDF. Unlike Random Forest or XGBoost, it did not overfit the sparse, high-dimensional data. The interpretability of its parameters also allowed us to fine-tune it effectively.

This study demonstrates that Logistic Regression, when paired with appropriately tuned textual features, offers a highly effective and interpretable approach for predicting recipe ratings. The model's superior performance underscores the importance of selecting features and algorithms that align with the dataset's characteristics. While more complex models like Random Forest and XGBoost have theoretical advantages, they require more structured data or extensive tuning to outperform simpler alternatives like Logistic Regression.

REFERENCES

Nguyen, J. (n.d.). Predicting food recipe ratings using machine learning techniques. GitHub. Retrieved December 2, 2024, from https://github.com/jimmy-nguyen-data-science/Predict-Food-Recipe-Ratings/tree/main

Gowshitha, R. (2021). Food recipe rating classification using machine learning. Medium. Retrieved December 2, 2024, from https://medium.com/@gowshitha123/food-recipe-rating-classification-using-machine-learning-a28a233828ce

Ramesh, K., & others. (2022). An intelligent approach for food recipe rating prediction using machine learning. ResearchGate. Retrieved December 2, 2024, from https://www.researchgate.net/publication/351505321_An_Intelligent_Approach_for_Food_Recipe_Rating_Prediction_Using_Machine_Learning