Culinary Insights: A Journey into Recipe Rating Predictions

Dhruv Agarwal Mithil Chaudhary Sudhansh Peddabomma
UC San Diego
{*d5agarwal, michaudhary, speddabomma*}*@ucsd.edu*

Abstract

Predictive modeling in today's era is multifaceted, integrating a variety of advanced machine learning techniques to enhance accuracy while maintaining robustness across diverse domains. This report delves into the realm of predictive modeling, employing a range of methodologies to analyze a dataset comprising recipe descriptions and user reviews. Aligned with the current trend, we explore the relationship between recipe attributes and user ratings, utilizing models such as Linear Regression, Logistic Regression, Random Forest Classifier, Latent Factor and Collaborative Filtering models. Through careful analysis of the data and feature engineering, we uncover correlations, deepening our understanding of the factors contributing to a recipe's success. Additionally, the report describes the significance of features and provides a comprehensive overview of the results obtained by different models.

1 Introduction

Rating prediction holds a pivotal role in diverse domains, offering invaluable insights into user preferences and behaviors. Whether in e-commerce, entertainment, or culinary realms, the ability to forecast user ratings fosters enhanced decision-making and personalized experiences. Predicting ratings not only empowers businesses to tailor their offerings to individual tastes but also facilitates the creation of recommendation systems that elevate user satisfaction. By delving into the factors influencing ratings, one gains not only a comprehensive understanding of user preferences but also the potential to optimize content, improve user engagement, and foster a more enriching experience overall. In the context of our culinary analysis, rating prediction unveils the intricacies of recipe success, offering a lens through which we can explore the elements that elevate a culinary creation to higher acclaim. To do so, we consider a dataset covering 18 years of recipe uploads and user interactions on Food.com (formerly GeniusKitchen). In the context of our dataset and Food.com, users can interact with a recipe by rating a recipe on a categorical scale of 1 to 5 and submitting a text review. Upon analysis of the data, we performed outlier removal to improve the quality of the data and reduce noise, making it easier to interpret the trends present. As the dataset includes sufficient information about users and recipe, regression-based approaches were initially utilized for prediction to aid relevant feature selection. Additionally, latent-factor models, collaborative filtering approaches and ensemble models were also employed to perform a comparative analysis.

2 Literature Review

Review rating prediction is one of the most common tasks for making recommendations. Researchers in the past have explored various methods to solve this problem, from classical machine learning methods to using deep learning and natural language processing.

2.1 Similar Datasets

The dataset for this analysis is taken from Food.com which is used to generate personalized recipes based on user interactions [14]. A similar dataset is a restaurant dataset, by a popular food delivery company JustEat in the United Kingdom. Researchers in [1] explored the use of topic membership and sentiment analysis on the user reviews to predict the user ratings for food. Amazon Fine Food dataset [15] is another such which provides user reviews for food. The dataset has been explored extensively in research community for analysing user reviews and making recommendation system models.

Rating prediction is typically done on Amazon and Yelp datasets, and papers such as [2], [17] and [5] analyze different models for rating prediction of restaurants.

2.2 Other Methods

Researchers have explored various methods for predicting user ratings for products. The work in [3] uses Multimodal Deep Learning based approaches for sentiment analysis and used textual features for sentiment classification and predicting user ratings. They tested their approach on popular datasets like IMDB movie dataset and Yelp dataset and established the efficacy of deep learning methods on

rating predictions. Such text mining methods are extensively explored in [13] using the Yelp reviews dataset. On the other hand, the work in [10] takes a very different approach to predict user ratings for movies using behavior over time. They proposed an improved TimeFly algorithm to resolve the problem in fluctuation of user's preferences with respect to the time and improve the recommendation results.

In contrast to feature-based methods mentioned above, some works explore interaction based models. These approaches rely on historical data to make a prediction rather than the features of the user and item. In particular, [7] uses Collaborative Filtering, and discusses the underlying assumptions and implications. On a similar note, the work in [6] explores the applications of Latent Factor Models on the Amazon and Yelp datasets. These approaches are also referred to as matrix factorization methods in the literature.

The work in [11] explores the combination of both the types of features mentioned above, wherein text-based features and interactions are combined to predict ratings. In our work, we analyse the performance of both feature-based as well as similarity based models.

2.3 State-of-the-Art methods

The state-of-the-art methods for rating prediction include Matrix Factorization [9], Bayesian probabilistic tensor factorization [16], and Neural Collaborative Filtering [8]. Factorization machines are also a new model class which combine the advantages of Support Vector Machines (SVMs) with factorization models [18]. They combine linear and second-order feature interactions. Transfer learning is also one of the techniques used to leverage knowledge learned from one recommendation domain to improve performance in a different domain. For example, transfer learning can be utilized to provide knowledge transfer in biased data and unbiased data in recommender systems[12].

3 Exploratory Data Analysis

The dataset chosen for this report comprises 231,637 recipes and 1,132,367 reviews, representing 18 years of data crawled from Food.com. It was originally used in [14] for generating recipes based on historical user preferences, which involved taking input in the form of a recipe name and a few primary ingredients and therefore generate plausible recipes given the ones previously consumed by the user, which is a much more complicated problem statement than ours. The dataset consists of 7 files:

- Three interaction splits designated for training, validation, and testing, respectively.
- Two files containing tokenized recipe text metadata, processed via the GPT subword tokenizer.
- Two raw files providing unaltered descriptions of the recipes and user reviews in their original state.

For the purpose of this analysis, we chose to work with the raw files to understand the data from the ground up and create splits as well as features according to our requirement.

3.1 Dataset Schema

The two raw files are:

1. **`RAW_recipes.csv`**: Contains a detailed description of the recipes, including
    - Recipe name and ID
    - Contributor ID
    - Tags highlighting the recipe
    - Date of submission
    - Nutritional information containing 6 values such as calories, fat, sugar etc.
    - Steps to prepare the recipe and a short description
    - Ingredients for the recipe
    - Attributes: minutes for preparation, number of ingredients, number of steps
2. **`RAW_interactions.csv`**: Contains complete review text data including
    - User ID
    - Recipe ID
    - Rating
    - Date of submission
    - Review text

3.2 Analysis

Before hopping on to training models for our predictive task, we analyzed the data in depth to gain a comprehensive understanding of its patterns and trends and facilitate the identification of outliers. Outliers can significantly impact statistical analyses and modeling, so detecting and addressing them is crucial. We started with analyzing the distribution of the number of ingredients over all recipes in Figure 1.

The distribution of the number of recipes across varying counts of ingredients and steps appears to follow a normal pattern, aligning with realistic expectations in real-world

Figure 1.

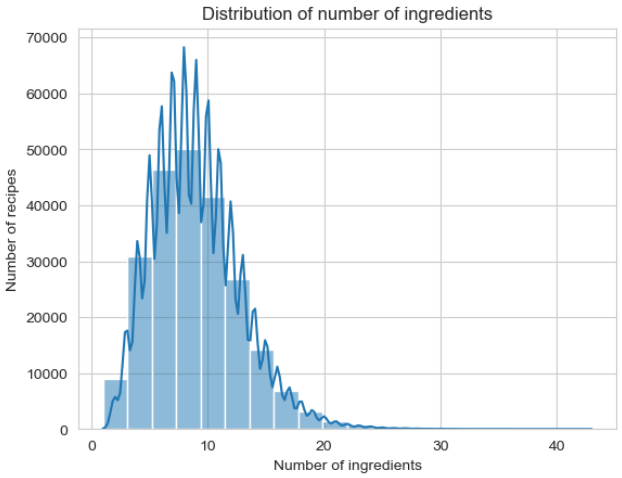
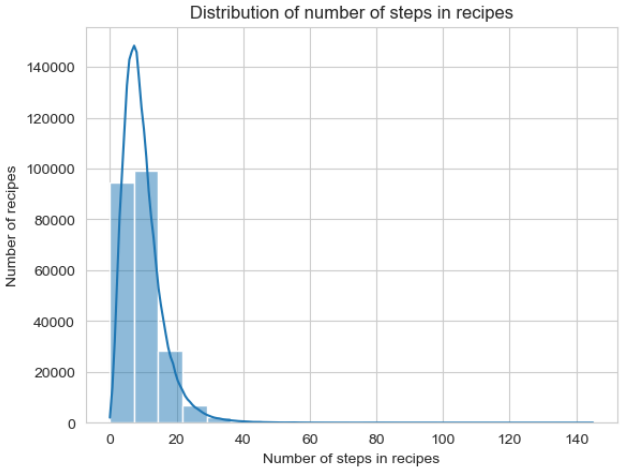


Figure 2.



scenarios. Figure 2 indicates that majority of the recipes have less than 20 steps.

The distribution of recipe against the time taken to cook them is plotted in Figure 3. It is evident from the graph that there are many anomalous results. Upon finding the range of values for the time taken, we observed that the minimum values and maximum values are 0 and $2.147e^9$ respectively. This observation hinted about the presence of possible outliers in the data. Hence, it was imperative to cleanse our data of these values, which was achieved by filtering out values below bottom 1% and above top 99%. Post filtering, the time distribution appears to be more consistent with what would be expected from a real-world scenario. (Figure 4)

A similar trend is observed with nutrient values. The Figure 5 shows the box plots for each nutrient, clearly showing that outliers are still present in the data. The same method to remove the outliers was applied to the six distinct nutrition values. After removing the outliers for each nutrient, we are left with total of 178,504 recipes which is about 77% the original size of the dataset. The box plots and distribution for each nutrients are plotted in Figure 6 and Figure 7 respectively.

As we are modeling a predictive task over the ratings of recipes, we analyzed the distribution of number of recipes that are in the dataset for each rating class (1, 2, 3, 4, 5). The Figure 8 shows the distribution. It is observed that the data is highly skewed towards classes and 4 and 8, with 88.5% of ratings belonging to those classes and the rest 11.5% be-

Figure 3.

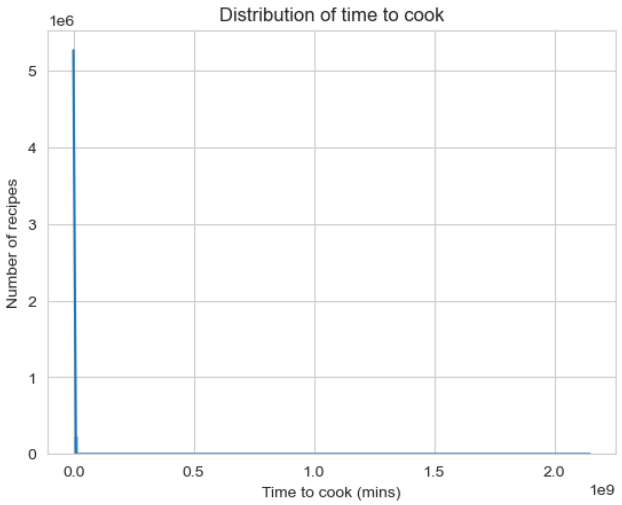
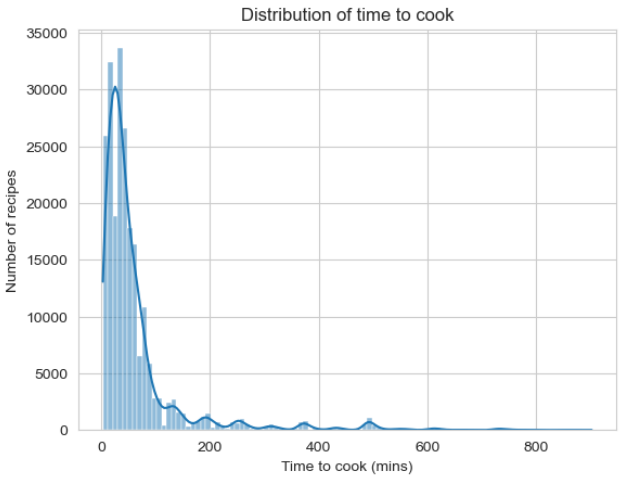


Figure 4.



longing to classes 0, 1, 2 and 3, with the least number of reviews giving a 1 rating.

Figure 5.

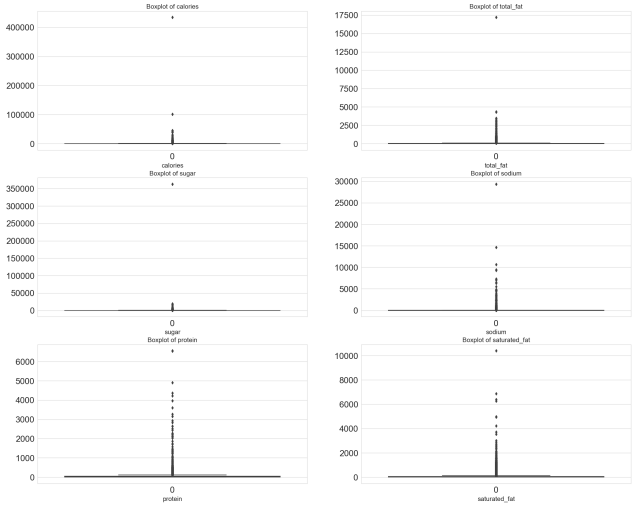
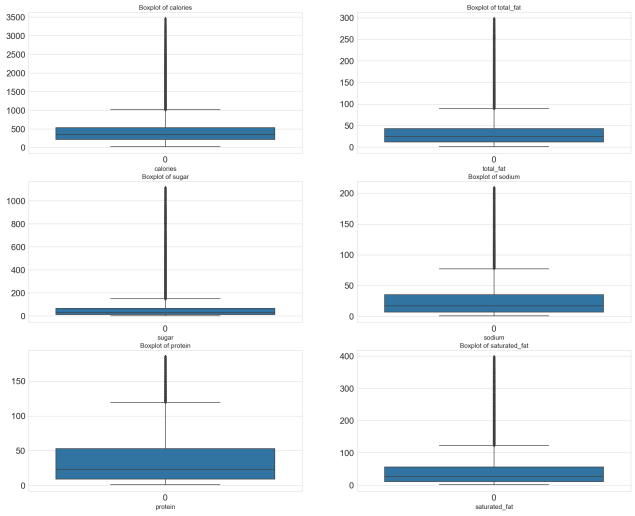


Figure 6.



We also studied the statistics for the number of reviews that different number of users had given. The statistics are present in the table 3.2. An interesting insight is that one user gave 5523 reviews.

Similar statistics for the number of recipes are presented in table 3.2

Figure 7.

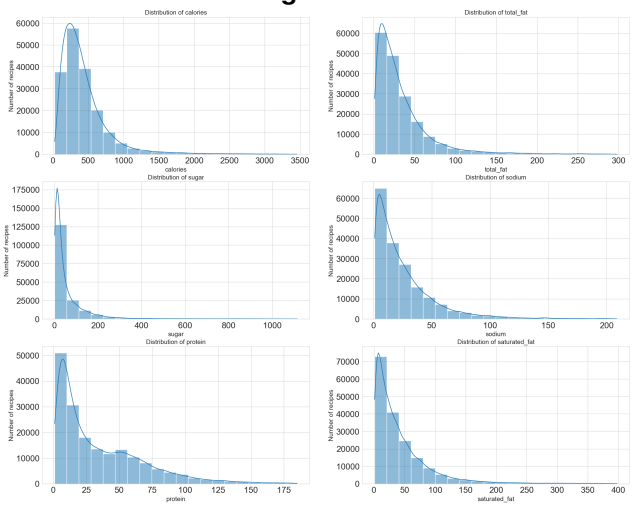
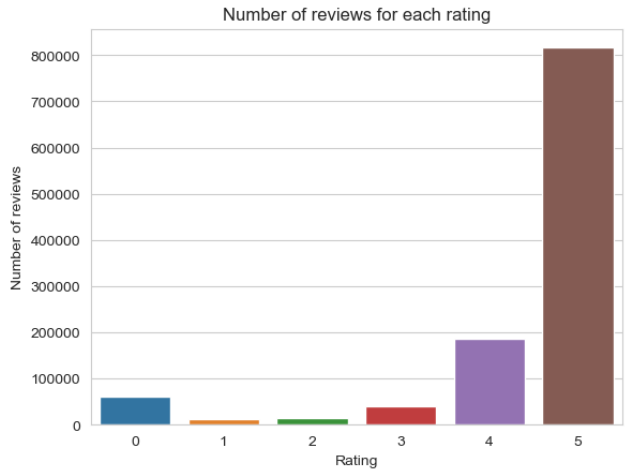


Figure 8.



Statistic	Value
Count	182262.000000
Mean	4.607982
Standard Deviation	39.552143
Minimum	1.000000
25% Percentile	1.000000
50% Percentile	1.000000
75% Percentile	2.000000
Maximum	5523.000000

Statistic	Value
Count	168764.000000
Mean	4.976535
Standard Deviation	18.451831
Minimum	1.000000
25% Percentile	1.000000
50% Percentile	2.000000
75% Percentile	4.000000
Maximum	1609.000000

4 Data Preprocessing

4.1 Textual Features

Consider the textual features in the dataset - recipe name, recipe description, associated tags, ingredients, steps and review text. It is essential to remove punctuation from these fields and convert all characters into lower case to ensure that variations of words are not represented separately. For example, the text "spicy", "Spicy" and "spicy," should correspond to the same features. Furthermore, some reviews in the dataset only contain text ".." and erroneous punctuation marks without any alphabets. These reviews do not contribute any useful information to ur models, and they are eliminated in the punctuation removal step.

Subsequently, we *stem* each word using the *Porter Stemmer* from the `nltk` library. This step normalizes the variations of English words, such as converting the words "cooking" and "cooked" to "cook", which helps us obtain words in their root or base form.

4.2 Filtering the dataset

Following the above textual feature extraction, some features can be empty if the text has ambiguous values. For example, if a recipe contains name or description with only punctuation symbols, it must be removed. The above steps will completely void such recipes and/or reviews, hence we filter the dataset to remove the records with empty feature values.

Finally, after filtering the recipe dataset, we clean the reviews dataset by removing reviews without any corresponding recipes. After this step, the number of recipes reduce to 168,764 and the number of reviews drop to 839,860.

4.3 Numerical Features

The numerical attributes in the dataset - nutrition values, time to cook, number of steps and number of ingredients - need to be normalized before they are used in the models as features. Without normalization, the predictions of the model can get heavily influenced by high magnitude numerical values while ignoring the lower ones. With the intention of understanding these effects further, we trained a regression model without normalizing the numerical features. The accuracy of the model was unsatisfactory and upon analyzing the trained weights in the model, we noticed that 90% of the model's weights corresponded to the nutrition features.

Before any feature engineering, a snapshot of a data sample looks like:


```
{
    "name": "['all','in',...,'chili']",
    "minutes": 130,
    "contributor_id": 196586,
    "submitted": "2005-02-25",
    "tags": "['timetomak',..., 'prepar']",
    "n_steps": 6,
    "steps": "['brown',...,'chees']",
    "description": "['thi',..., 'origin']",
    "ingredients": "['ground',..., 'chees']",
    "n_ingredients": 13
}
```


After all the processing mentioned above, a sample entry in the recipes dataset looks is shown in 4.3


```
{
    "name": "['all','in',...,'chili']",
    "minutes": 0.144,
    "contributor_id": 196586,
    "submitted": "2005-02-25",
    "tags": "['timetomak',..., 'prepar']",
    "n_steps": 0.207,
    "steps": "['brown',...,'chees']",
    "description": "['thi',..., 'origin']",
    "ingredients": "['ground',..., 'chees']",
    "n_ingredients": 0.302,
    "nutrition_0": 0.0717259046,
    "nutrition_1": 0.0707070707,
    "nutrition_2": 0.0278526505,
    "nutrition_3": 0.2259615385,
    "nutrition_4": 0.2065217391,
    "nutrition_5": 0.0654911839,
    "nutrition_6": 0.0264900662
}
```


4.4 SMOTE

As discovered in the data analysis, it was found that the dataset has a skewed distribution over the rating values with most of the recipes being rated either 4 or 5. Training a predictive model on such data was found to be problematic because the model was easily achieving a low loss value by simply predicting all the values as the major class, also commonly known as overfitting. One way to deal with such datasets is to use oversampling methods. Oversampling methods create a synthetic dataset out of existing samples

such that all the classes have a comparable number of samples, yielding a balanced dataset.

We used SMOTE (Synthetic Minority Oversampling Technique) [4] to oversample minority class sample. The reason for choosing SMOTE over other oversampling methods was that it doesn't duplicate any samples and takes into account to the relative distance in the feature space of samples to synthesize new samples. After using SMOTE on our dataset, we increased the dataset size by 76.7%. This massive increase in the dataset size is attributed to the extent of skewness that is present in the dataset

5 Predictive Task

In the initial stages of our analysis, we considered predicting the time required for recipe preparation based on ingredient composition and preparation steps. However, we later redirected our efforts towards predicting ratings. Given a user and recipe pair, we explored the task of predicting a rating using different models. This decision was influenced by the recognition that predicting preparation time, a metric already provided alongside recipes, may not yield significant practical value. This predictive task can further be used for recommending recipes to a user given some of the user's preferences and predicted ratings for a subset of recipes deemed similar to the user's taste.

5.1 Evaluation Metrics

We use the following metrics to evaluate the performance of the models

1. Mean-Squared Error

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(x_i - y_i)^2$$

2. Mean-Absolute Error

$$MAE = \frac{1}{N}\sum_{i=1}^{N}(x_i - y_i)^2$$

3. Accuracy

$$Accuracy = \frac{TP + TN}{TP + TF + FP + FN}$$

4. F1-Score

$$F1 - Score = \frac{2 * Precision * Recall}{Precision + Recall}$$

Accuracy may not provide a meaningful evaluation in our case due to the class imbalance, hence we chose to evaluate our models on F1 score as well because it takes both precision and recall into account.

5.2 Baselines

As a foundation for our predictive task of estimating ratings for user-recipe pairs, we established three baseline models for comparison.

- Predict the global mean. Accuracy = 16.90, MSE = 1.72%
- Predicting the user average. Accuracy = 54.74, MSE = 1.57%
- Predicting the recipe average. Accuracy = 49.80, MSE = 1.83%

Predicting the global mean rating serves as a straightforward starting point, highlighting the challenge of surpassing a simple average estimation.. The next two baselines that we incorporated provide a more personalized approach, one predicting the user average rating whereas the other predicting the recipe average rating. These baseline models provide essential benchmarks for evaluating the performance of the models we implement further on.

6 Feature Engineering

Feature engineering is a crucial step in the machine learning pipeline, where the goal is to transform raw data into a more informative representation that facilitates the learning process of models. During the exploratory data analysis, we corrected for statistical anomalies by removing the outliers. It is equally important to carefully craft features to improve the overall predictive performance of machine learning models.

6.1 Word2Vec

The textual features intuitively can greatly contribute to a user's rating. Facilitating their usage in predictive models requires us to obtain a numerical representation using *Word Embeddings*.

We use the `Word2Vec` library to train embeddings on the dataset. It takes the textual features as input and produces the word vectors as output. It does so, by constructing a vocabulary from the training text data and learning their vector representation. For example, we can use these vector representations to find tags that are similar to each other. The vector embeddings associate the word "salad" with words like "side dish", "healthy", and "sauce".

6.2 Averages

In a user-item dataset, it is typically seen that some users tend to rate higher/lower than the average and similarly such

a bias exists in the items as well. These biases cannot be estimated only using the recipe features because they are implicitly present for each user independent of these features.

Therefore, we also add the normalized user averages and recipe averages as features in the predictive models.

6.3 Features

The final set of features are listed below -

- Recipe name, description, tags, ingredients and steps embeddings
- Averages for the query user and item
- Normalized numerical values - nutrition, time to cook, number of ingredients, and number of steps
- One hot encoding vector for the weekday of the review

7 Models

7.1 Feature based models

We trained the following feature based predictive models on the dataset

- Linear Regression with L2 Loss: We trained a linear regression model using MSE as the loss function on the rating values. For prediction we rounded of the values to nearest integer for evaluating against the ground truth.
- Linear Regression with L1 Loss: Motivated from the fact that L1 loss is more robust to outliers, we trained the same linear regression model as above with the L1 loss.
- Logistic Regression: Since the dataset has rating values as categorical and not continuous, we tried fitting a logistic regression model as well.
- Logistic Regression with balanced class weights: As discussed in section 3.2, our dataset has a skewed distribution for rating values. Since a model trained on such a dataset can be biased towards the majority class, we trained the same logistic regression model with balanced class weights.
- Random Forest Classifier: Gravitating towards more complex models in order to capture more complex relations in the data, we trained a random forest classifier.
- Random Forest Classifier with balanced weights: We trained the same random forest classifier as above with balanced class weights to counter the class imbalance problem in the dataset as we did for logistic regression.
- XGBoost Classifier: XGBoost has been very successful in similar tasks, and given that our model performs worse on minority class samples, boosting the performance on those samples was a prudent choice. Therefore we tried our prediction modeling with XGBoost classifier

7.2 Interaction based models

The models included here predict ratings merely on the basis on interactions between users and recipes.

- Collaborative Filtering based on the following similarity functions:
  - Jaccard Similarity
  - Cosine Similarity
  - Pearson Similarity
- Latent Factor Model (Bias-Only)

8 Results

The results from our experiments are summarized in the tables 1 and 8. As mentioned previously, we analyse MSE, MAE, Accuracy and F1-scores for each rating value. The models are trained on 80% of the data chosen randomly. The remaining 20% of the data is chosen as the test set.

Model	MSE	MAE	Accuracy (%)
Ridge Regression	1.50	0.76	52.57
LASSO	**1.46**	0.79	51.17
Logistic Regression	1.93	**0.64**	**65.94**
Logistic Regression balanced	2.01	0.88	47.82
Random Forest	2.02	0.70	62.14
Random Forest balanced	2.07	0.70	62.50
Latent Factor	1.66	0.68	57.61
Jaccard Similarity	1.91	0.79	50.03
Cosine Similarity	1.90	0.79	50.02
Pearson Similarity	1.92	0.79	48.17

Table 1. Performance Metrics

As expected, Regression and LASSO models have the least values of MSE. The weakness of the regression models is highlighted in the accuracy on the test set. These functions aim to reduce the MSE, rather than predict the correct

classes for rating, which can be seen in the accuracy metric. The classification model implemented using Logistic Regression gives the highest accuracy. However, the per-class accuracy of this model is sub-par because of the skewed data distribution. To address this, we train a logistic regressor with balanced class weights option. On balancing the class weights, the accuracies of the minority classes are improved but the total accuracy reduces.

After observing the skewed distribution of the data, we hypothesised that the Random Forest model may work better on the dataset. Random Forest model is better than logistic regression in terms of overfitting, easily scalable to a large number of features and works well on imbalanced data. We get high accuracies using the random forests which come close-second to the classification model.

Interactions based approaches (Jaccard, Pearson, and Cosine similarities) do not perform well on this dataset. This is due to the skewness of user-interactions in the dataset. As summarized in tables 3.2 and 3.2, the average number of interactions per recipe and user are very low. Empirically, out of the 182,262 users, there are 133,458 users who gave only 1 review. Similarly, out of the 168,764 recipes, there are 66,851 recipes with only one review. These low-number of interactions significantly affect the performance of these models. This highlights the weakness of such interaction based models.

These effects are also seen in the latent-factors models. It can be understood this way - the β_u's of highly active users are minimized better as compared to the inactive users. This can bring down the total accuracy, while keeping the accuracy for these high users/items high.

Model	$\leq$ 3-stars	4-stars	5-stars
Regression	0.196	0.310	0.693
LASSO	0.072	**0.319**	0.671
Classification	0.127	0.202	**0.807**
Classification balanced	**0.315**	0.281	0.666
Random Forest	0.182	0.285	0.779
Random Forest balanced	0.172	0.274	0.781
Latent Factor	0.206	0.314	0.740
Jaccard Similarity	0.175	0.300	0.665
Cosine Similarity	0.168	0.302	0.665
Pearson Similarity	0.173	0.295	0.646

Table 2. F1-scores

The F1-scores provide a better insight into the per-class performance of each model. As expected, the F1-scores for the balanced methods for classification and random forest are better than their unbalanced counterparts. The balanced logistic regression model has the highest F1-score for the undersampled classes (have rating $\leq$3). Since the 5-stars class has the highest population, all models have a high F1-score for this class.

8.1 Feature Importance

We studied the feature importance for regression and classification models to get an insight on what feature are most relevant for our predictive task.

For balanced random forest classifier, we obtain the following feature importance. We show the feature importance of only random forest classifier because a similar trend was obtained for all the other models as well.

Feature	Order of Importance
Tags	0.064
Ingredients	0.062
Name	0.062
Description	0.062
Steps	0.063
Nutrition	0.073
Number of Steps	0.009
Number of Ingredients	0.008
Minutes	0.011
Recipe average	0.099
User average	0.384
Weekday	0.102

Table 3. Feature Importance in Random Forest balanced)

As seen above, the user averages contribute heavily towards predicting the rating. The next important feature is the recipe's average rating. Other than that, the features such as recipe steps, nutrition values and tags all contribute to the decisions as expected.

9 Conclusion

In conclusion, data imbalance heavily affects the performance of feature-based and interaction-based models alike. The dataset used for the analysis has high skewness in interactions as well as ratings.

Amongst the models we tested, the logistic regression model has the highest accuracy followed by the random forest models. However, the F1-scores for the classes are the highest when the logistic regression has balanced class weights.

After analyzing the features, we conclude that the recipe features such as nutrition and tags are of high importance after the user and recipe average rating values.

10 Future Work

In our work, we have not explored the text reviews to a great extent. Past research has demonstrated the importance of sentiment analysis in user review prediction and its absence in our predictive modeling is one of our approach's weaknesses. Our work can be extended by incorporating sentiment analysis using advanced natural language processing and deep learning. These features can greatly improve the performance of the feature-based models.

Another interesting progression in the future could be that of exploring the landscape of neural collaborative filtering for predictionss.

References

[1] Justeat restaurant data. 2021.

[2] N. Asghar. Yelp dataset challenge: Review rating prediction, 2016.

[3] S. K. Banbhrani, B. Xu, H. Lin, and D. K. Sajnani. Spider taylor-choa: Optimized deep learning based sentiment classification for review rating prediction. *Applied Sciences*, 2022.

[4] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: Synthetic minority oversampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, June 2002.

[5] Y. Chen and F. Xia. Restaurants' rating prediction using yelp dataset. In *2020 IEEE International Conference on Advances in Electrical Engineering and Computer Applications(AEECA)*, pages 113–117, 2020.

[6] Z. Cheng, Y. Ding, L. Zhu, and M. Kankanhalli. Aspect-aware latent factor model: Rating prediction with ratings and reviews. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web - WWW '18*, WWW '18. ACM Press, 2018.

[7] M. D. Ekstrand, J. T. Riedl, and J. A. Konstan. Collaborative filtering recommender systems. *Foundations and Trends® in Human–Computer Interaction*, 4(2):81–173, 2011.

[8] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, pages 173–182, 2017.

[9] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[10] R. A. Kumar, R. Deepika, K. S. Kumar, and P. Dinesh. Recommender system : An automated movie rating prediction using an improved timefly algorithm (itfa). 2021.

[11] C.-H. Lai and C.-Y. Hsu. Rating prediction based on combination of review mining and user preference analysis. *Information Systems*, 99:101742, 2021.

[12] Z. Lin, D. Liu, W. Pan, and Z. Ming. Transfer learning in collaborative recommendation for bias reduction. In *Proceedings of the 15th ACM Conference on Recommender Systems*, RecSys '21, page 736–740, New York, NY, USA, 2021. Association for Computing Machinery.

[13] S. Liu. Sentiment analysis of yelp reviews: A comparison of techniques and models, 2020.

[14] B. P. Majumder, S. Li, J. Ni, and J. McAuley. Generating personalized recipes from historical user preferences. In K. Inui, J. Jiang, V. Ng, and X. Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5976–5982, Hong Kong, China, Nov. 2019. Association for Computational Linguistics.

[15] J. McAuley and J. Leskovec. From amateurs to connoisseurs: Modeling the evolution of user expertise through online reviews, 2013.

[16] H. Morise, S. Oyama, and M. Kurihara. Bayesian probabilistic tensor factorization for recommendation and rating aggregation with multicriteria evaluation data. *Expert Systems with Applications*, 131:1–8, 2019.

[17] A. Rafay, M. Suleman, and A. Alim. Robust review rating prediction model based on machine and deep learning: Yelp dataset. In *2020 International Conference on Emerging Trends in Smart Technologies (ICETST)*, pages 8138–8143, 2020.

[18] S. Rendle. Factorization machines. In *2010 IEEE International Conference on Data Mining*, pages 995–1000, 2010.