Recipe for Success: Accurately Predicting Recipe Rating

Abstract

The Food.com dataset, encompassing 231,637 recipes and 1,132,367 reviews, provides a rich source for exploring culinary preferences and user behaviors. In this study, the primary focus lies in uncovering the intricate factors influencing recipe popularity and ratings. The dataset includes essential components such as recipe details, user interactions, and reviews, with each interaction comprising crucial elements like the user, recipe, a free-form text review, and a numerical rating. The central challenge addressed is predicting the rating a user assigns to a recipe, akin to the Rating Prediction problem prevalent in recommendation systems. This problem is approached by implementing various models, each leveraging different features, including user interactions, review texts, and temporal factors to forecast recipe ratings. The journey begins with the implementation of a baseline Similarity-based Rating Prediction model, progressing to more robust Latent-Factor models. Text-based methodologies, utilizing models like Bag-of-Words and Term-Frequency Inverse Document Frequency (TF-IDF), are explored for rating predictions. The performance of each model is meticulously analyzed, leading to the observation that the latent factor model outperforms their counterparts in capturing the nuances of recipe ratings within this diverse and extensive dataset.

Keywords: Recommendation Systems, Food Recipes, Textual Features, Latent Factor Models

Introduction

The food recipe recommendation domain exhibits distinctive characteristics that shape the nature of the recommendation problem. Food recipe recommendations contribute significantly to the online culinary landscape, constituting a substantial portion of user engagement with recipe platforms. In the realm of food datasets, the interaction history between users and recipes introduces inherent challenges characterized by a considerable degree of sparsity. This sparsity poses challenges for the application of conventional collaborative filtering techniques, rendering them resource-intensive and inefficient.

The problem of food recipe rating prediction involves creating a model that can predict the likely rating a user would assign to a particular food recipe. This task is rooted in the context of platforms, such as Food.com, where users can submit and review recipes. The goal is to develop a predictive model that can anticipate the user's satisfaction or approval level with a given recipe based on various features associated with both the user and the recipe itself. Specifically, the study seeks to investigate the impact of incorporating textual, item-related, and temporal features on the model's performance, deviating from traditional collaborative models. The models developed in this project will be strategically tailored to predict user ratings for specific recipes. Evaluation metrics such as Mean Squared Error is employed to compare the performance of the proposed models against traditional collaborative filtering approaches. This exploration aims to advance our understanding of how external features contribute to the effectiveness of recommender systems in the context of food datasets.

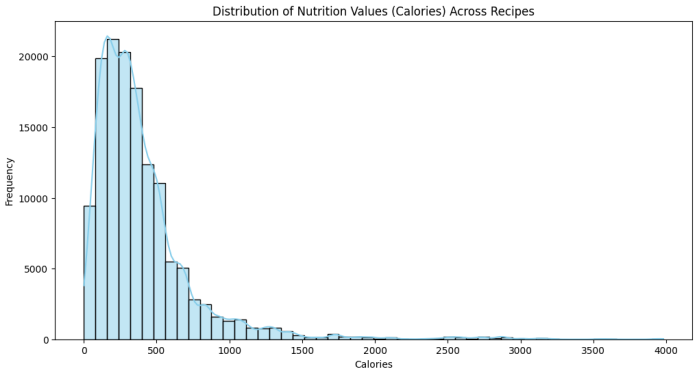
1. Exploratory Analysis

Exploratory analysis on raw data is conducted, focusing on the Recipes and Interactions components. We begin by importing necessary python packages, including the use of dataframes and setting a random seed. Initial preprocessing steps involve separating nutritional values into distinct columns and converting date columns to datetime format.

user_id	38094
recipe_id	40893
date	2003-02-17
rating	4
review	Great with a salad. Cooked on top of stove for...

The analysis delves into recipe statistics, such as the distribution of recipe minutes and the contributor ID patterns. Outliers, particularly recipes with excessively high minutes, are removed. Nutrition-related outliers are

addressed, filtering out recipes with excessive nutrition calories. The exploration extends to analyzing user interactions, considering review counts and average ratings per user. Thresholds are established to filter out recipes and users, addressing cold start issues. We filter out recipes with less than 20 reviews and users who have submitted less than 15 reviews. The resulting dataset is relatively denser with 3217 reviews, 3275 users, and 138216 interactions.



name	arriba baked winter squash mexican style
id	137739
minutes	55
contributor_id	47892
submitted	2005-09-16
tags	['60-minutes-or-less', 'time-to-make', 'course...
nutrition	[51.5, 0.0, 13.0, 0.0, 2.0, 0.0, 4.0]
n_steps	11
steps	['make a choice and proceed with recipe', 'dep...
description	autumn is my favorite time of year to cook! th...
ingredients	['winter squash', 'mexican seasoning', 'mixed ...
n_ingredients	7

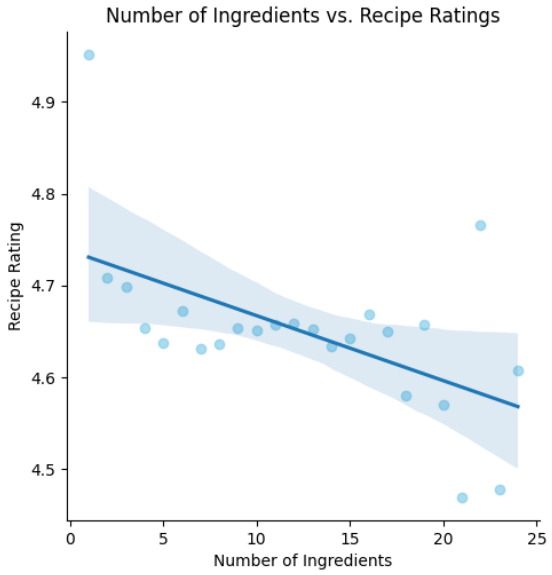
Insights are derived from contributors, and the analysis explores whether recipes created by top contributors tend to receive higher ratings. Correlations between recipe features and ratings, such as minutes, steps, and ingredients, are visualized. User patterns, including consistently high or low ratings, are identified, and the distribution of user ratings is visualized.

The exploratory analysis provides a comprehensive understanding of the dataset, uncovering patterns, correlations, and potential factors influencing recipe ratings. These insights guide the subsequent design and selection of predictive models for the rating prediction task.

The observation indicates a predominant occurrence of positive ratings, with a significant count of 5-star ratings (816,364), while lower ratings gradually decrease. Notably, there are instances of 0-star ratings in the dataset.

rating	count
5	816364
4	187360
3	40855
2	14123
1	12818
0	60847

We identified the recipe with the highest number of reviews and the top 5 most reviewed recipes. The recipe with the ID 27208, titled "To Die For Crock Pot Roast," stood out with 545 reviews. Upon further inspection, this recipe features a simple crock pot roast with beef, brown gravy mix, and Italian dressing mix, praised for its amazing flavor and minimal seasoning requirements.

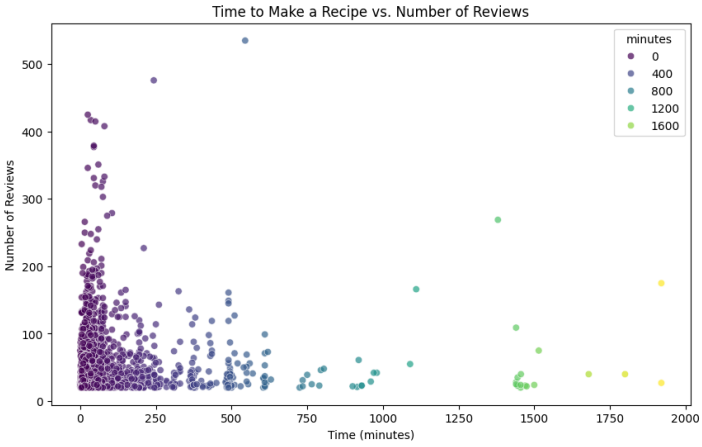


The top 5 most reviewed recipes cover a range of dishes such as "Creamy Cajun Chicken Pasta," "Crock Pot Chicken with Black Beans Cream Cheese," "Kittencal's Italian Melt in Your Mouth Meatballs," and "Whatever

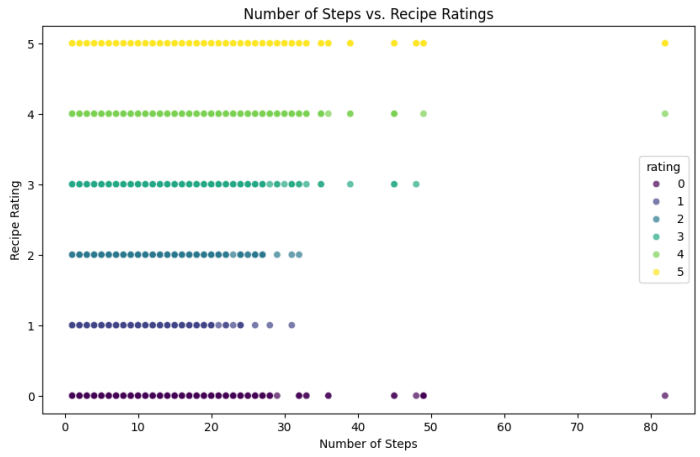
Floats Your Boat Brownies," showcasing diverse culinary preferences and high user engagement.

Analyzing user contributions in the dataset reveals that the most active contributor, identified by ID 89831, has an impressive average of 8.29 recipes contributed. The collective appreciation for these recipes is evident in the dataset's high average rating of 4.65, with a substantial 3,217 recipes achieving a perfect 5.0 rating, highlighting the positive reception.

The average time required to make a recipe is 123.11 minutes, but the median time is only 40.00 minutes. This suggests that there is a long tail of recipes that take a very long time to make. The plot of time to make a recipe vs. number of reviews shows a lot of scatter, with no clear trend.



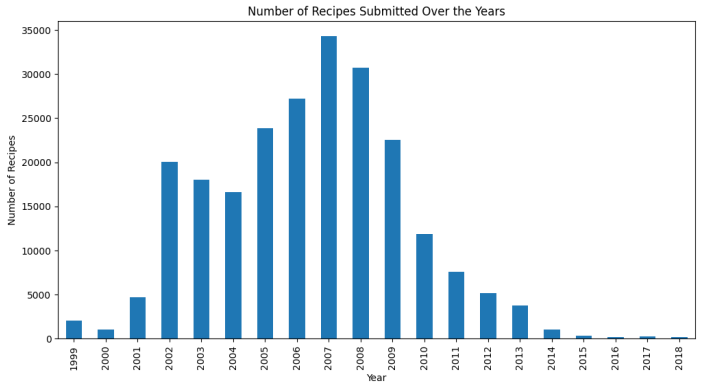
The average number of steps in a recipe in the Food.com dataset is 9.77. This suggests that most recipes are relatively simple to follow, with only a few steps required



to complete them.

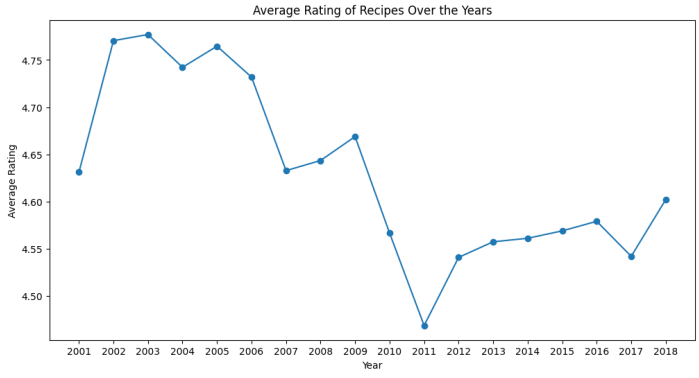
The distribution of the number of steps in recipes is skewed to the right, with more recipes having a smaller number of steps and fewer recipes having a larger number of steps.

The number of submitted recipes on Food.com has increased steadily over the years between 1999 and 2008. It's likely that the significant increase in 2002 was due to the internet boom. The waning number of recipes submitted after 2009 matches our observation on Google Trend about the search ranking of the website.



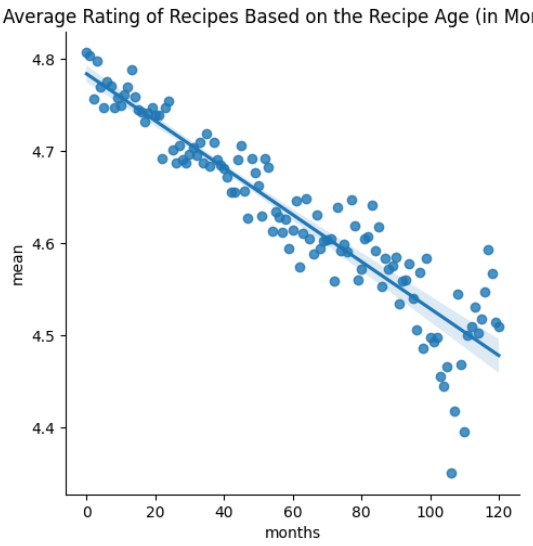
The average rating of recipes on Food.com has remained relatively constant over time. The line graph shows that the average rating has fluctuated slightly from year to year, but it has not shown any clear upward or downward trend. This suggests that the overall quality of recipes on Food.com has remained stable over time. People are still submitting high-quality recipes to the website, and other users are still rating them highly.

The average length of a recipe description is 195.58 characters, and the correlation between recipe length and popularity is 0.13. One possible explanation for the weak



positive correlation between recipe length and popularity is that longer recipes tend to be more detailed and informative. Another possible explanation for the correlation is that longer recipes tend to be more complex.

Temporal trends were analyzed, focusing on the timing of reviews, including specific dates, years, and months. The analysis identified a correlation between the time elapsed after a recipe is posted and the ratings it receives. The graph included demonstrates a clear pattern: reviews submitted a significant time after the recipe's initial posting generally exhibit lower average ratings.



The review text word cloud is a collage of most prominent words from the review text. We can see adjectives such as “good”, “easy”, “loved”, “yummy” that represent high quality recipes. Most of the words are positive because our dataset is skewed towards highly-rated reviews.

The ingredient word cloud contains common cooking ingredients, but it is more specific to American and European cuisine. The largest words are "garlic", "olive oil", "salt", and "black pepper". These are all basic ingredients that are used in a variety of dishes, such as pasta, chicken, and fish.

2. Predictive Task

A suitable predictive task for this dataset involves building a recipe rating prediction model. The evaluation metric for this task would be Mean Squared Error (MSE), measuring the average squared difference between predicted and actual ratings. For relevant baselines, three models can be considered:

- *Global Average Predictor:* Always predicting the overall average rating. (Observed MSE: 0.77894)
- *Product Average Predictor:* Predicting ratings using the average rating for each recipe. (Observed MSE: 0.77508)
- *User Average Predictor:* Predicting ratings using the average ratings given by a user. (Observed MSE: 0.71451)

To assess the model's predictions, the MSE scores of these baselines would be compared to the MSE of the developed model. Feature engineering would include incorporating relevant information such as recipe details (e.g., ingredients, cooking time), user history (e.g., previous ratings, interaction patterns), and potentially external factors. Data preprocessing steps involved handling extreme values, processing compound features, and normalizing numerical features wherever necessary. Additional details regarding the selection and processing of features are elucidated in sections pertinent to their application.

3. Models

In this section, we explain the models used, the rationale behind their selection, their advantages, and their limitations.

3.1. Similarity-Based Models

Our initial approach to the predictive task involved using a traditional collaborative filtering algorithm (CF). CF utilizes interactions among similar groups to predict object preferences. We considered both user-based and item-based approaches. For user-based predictions, we analyzed the similarity of other users' evaluations for the recipe in question, while for recipe-based predictions, we assessed the similarity of other recipes evaluated by the same user.

The decision to adopt a similarity-based approach was driven by the belief that the behavior of the target user can be predicted by studying the past behavior of a group of similar users. This rationale extends to predicting the behavior of a target recipe based on evaluations of a group of similar recipes. Commonly used similarity measures include Jaccard similarity, Cosine similarity, and Pearson similarity.

3.1.1 Jaccard Similarity

Jaccard similarity measures the intersection over the union of sets, quantifying the similarity between two sets by comparing the common elements to the total distinct elements.

$$\text{Jaccard}(i, j) = \frac{|U_i \cap U_j|}{|U_i \cup U_j|}.$$

where Ui and Uj represent the set of users who rated the items i and j respectively.

3.1.2 Cosine Similarity

Cosine similarity evaluates the cosine of the angle between two vectors, providing a measure of similarity irrespective of the magnitude. It is commonly used in collaborative filtering to assess the similarity between user or item vectors.

$$\text{Cosine Similarity}(u, v) = \frac{R_u \cdot R_v}{|R_u| \cdot |R_v|}.$$

where Ru and Rv represent the set of users who rated the items u and v respectively.

3.1.3 Pearson Similarity

Pearson similarity measures the linear correlation between two variables, indicating how well their relationship can be represented by a straight line. In collaborative filtering, it is employed to assess the similarity in preferences between users or items.

3.1.4 Ratings Prediction Function

We use similarities between users and similarities between items for rating prediction.

$$r(u, i) = \overline{R}_i + \frac{\sum_{j \in I_u \setminus \{i\}} \left(R_{u,j} - \overline{R}_j\right) \cdot sim(i, j)}{\sum_{j \in I_u \setminus \{i\}} sim(i, j)}$$

Similarity-based methods are often straightforward to implement and provide clear explanations for their recommendations. However, their effectiveness hinges on the selection of an appropriate similarity measure. Moreover, the prevalent issue of data sparsity can lead to the cold-start problem, where making predictions for new users or items with limited historical data becomes challenging. Additionally, computing similarities for every new user or item can be computationally expensive. Furthermore, as the number of users and items grows, the model's scalability becomes a significant concern.

3.1.5 Temporal Factors

To further improve our similarity-based models, we incorporated temporal factors to our models. Specifically, we calculated the number of days between two reviews. Reviews that are further in the past should have less weight. To model this, we use the function below as the temporal factor.

$$f(days) = 1 / (floor(days / \lambda) + 1)$$

For two reviews that are made on the same day, the temporal factor equals to 1.

3.2. Latent Factor Models

The next group of models that we tried are latent factor models. These models are particularly adept at identifying underlying patterns and relationships, which are not

immediately apparent. Central to this approach is the concept of matrix factorization, a method where the interaction matrix is decomposed into two low-rank matrices, each representing the users' and items' latent factors that influence the interactions (i.e. ratings).

3.2.1 Bias-only model

The simplest model is a bias-only model, which can be defined as

$$f(u,i) = \alpha + \beta_u + \beta_i$$

where α is the overall average rating and β_u and β_i are the effects of the users and items (recipes) respectively.

The parameters are found by solving the minimization problem:

$$\arg\min_{\alpha,\beta} \underbrace{\sum_{u,i}(\alpha + \beta_u + \beta_i - R_{u,i})^2}_{\text{error}} + \underbrace{\lambda\left[\sum_u \beta_u^2 + \sum_i \beta_i^2\right]}_{\text{regularizer}}$$

where λ is the regularization constant.

3.2.2 Latent Factor Model

For the latent factor model, we use the SVD algorithm that is popularized by Simon Funk [2] in the Netflix Prize competition. The model is defined as follow:

$$f(u,i) = \alpha + \beta_u + \beta_i + \gamma_u \cdot \gamma_i$$

In addition to the parameters that we have seen in a bias-only model, we have u and i which represents the latent factors for the users and the items (recipes) respectively. Both are low-dimensional matrices with dimension U x K and K x I, where K is the number of latent factors and U, I are the number of users and items respectively.

The parameters are found by solving the minimization problem:

$$\arg\min_{\alpha,\beta,\gamma} \underbrace{\sum_{u,i}(\alpha+\beta_u+\beta_i+\gamma_u\cdot\gamma_i-R_{u,i})^2}_{\text{error}} + \underbrace{\lambda\left[\sum_u \beta_u^2 + \sum_i \beta_i^2 + \sum_i \|\gamma_i\|_2^2 + \sum_u \|\gamma_u\|_2^2\right]}_{\text{regularizer}}$$

We experimented using different value of K and λ to find the model with the lowest error.

We employed the Surprise package for our model's implementation. To compare models, we utilized 5-fold cross-validation, eliminating the need for a separate validation set. This method aids in avoiding overfitting to the training data while also optimizing the training procedure. Once the optimal hyperparameters were selected, we retrained the model using the entire training dataset.

3.2.3 SVD++

SVD++ [3] builds upon the foundation laid by SVD, offering a more sophisticated model that integrates both explicit and implicit feedback. Traditional SVD primarily utilizes explicit data, such as user ratings. In contrast, SVD++ amalgamates this with implicit feedback.

3.3. Text-Based Models

We utilized the textual features focusing on user reviews and recipe metadata. Specifically, the proposed approach involves:

a. Rating Regression with Text Input:

- Generate textual vector embeddings from user text reviews.
- Employ these embeddings as features for training the regressor, facilitating the prediction of ratings based on new review text.

b. Recommendation Using Textual Compatibility Model:

- Combine the recipe name, description and ingredient list as a feature to assess recipe similarity.
- Derive ratings through a combination of similarity scores and predefined heuristics.

The project incorporates textual features to adapt content filtering-based recommendations. The dataset and metadata provide textual elements such as reviews, recipe name, description and ingredient list. The text features are comprehensive to be overlooked, and therefore we intend to use text as features to create regressor and text-similarity based rating predictors.

A text pre-processing approach is adopted for both implementations. Any unnecessary elements like stopwords are eliminated. All words are converted into lowercase to avoid redundancy.

3.3.1 Bag of Words (BoW)

Bag of Words (BoW) converts a document into a set of words, disregarding grammar and word order. Each word's frequency is counted, creating a "bag" of words, forming a sparse vector that represents the document's content. We selected the 10,000 most popular words to form our feature vector while not overwhelming the memory.

3.3.2 TF-IDF Vectorization

Term Frequency-Inverse Term Frequency (TF-IDF) evaluates how important a word is to a document in a cluster of documents. It is obtained by multiplying the frequency of a word in a document and the inverse document frequency of the word across a series of documents.

$$tf - idf(w, d, D) = tf(w, d) * log_2(\frac{|D|}{1 + df(w, D)})$$

Python's NLTK and regular expression libraries are employed for the text processing tasks. Scikit-learn's TfidfVectorizer function is used for creating text representations.

3.3.3 Using text embeddings

The text features are vectorized into TF-IDF and BOW Vectors for training our model. We created a predictor with following the form $y = \Theta x + b$ where x are the vectorized text features, and y is the rating assigned to a particular item.

The scikit-learn module is used to train the regression model, and create word vectors. A 90:10 split is employed for training and testing.

3.3.4 Textual Compatibility Based Recommendation

The model's conceptual framework aligns with the principles outlined in Section 3.1.4. However, in a departure from conventional approaches reliant on interaction data for similarity computation, our model computes similarity based on key attributes such as the recipe name, description, and ingredients list of two items. This alternative methodology is integrated into the existing model by passing the computed similarities through the same analytical function. This divergence in similarity computation strategy is designed to explore the potential influence of intrinsic recipe characteristics on the model's performance, enhancing its ability to discern patterns and relationships beyond user interactions.

$$r\left(u,i\right) = \overline{R}_i + \frac{\sum_{j \in I_u \setminus \{i\}} \left(R_{u,j} - \overline{R}_j\right) \cdot sim\left(i,j\right)}{\sum_{j \in I_u \setminus \{i\}} sim\left(i,j\right)}$$

4. Literature Review

4.1 Problem

Our daily interactions with food and culinary practices play a pivotal role in shaping our health, habits, and cultural traditions. Recognizing this connection, the design of computational systems has progressively turned its attention to these everyday activities. This evolving field, known as food computing, encapsulates a diverse range of computational approaches dedicated to unraveling the complexities inherent in food-related data [6]. The exploration of recipe data has been a prominent focus within the realm of food computing. Numerous studies examine user-recipe interactions and preferences, relying on collaborative-filtering methods based on historical user data to predict taste in new recipes [7,8,9].

4.2 Dataset

The foundation of our research rests upon the Food.com dataset [1], a comprehensive repository encompassing 231,637 recipes and 1,132,367 reviews, spanning the years 1999 to 2018. This rich dataset, extensively utilized by numerous researchers, has been instrumental in addressing various challenges within the food recommendation domain. For instance, predictive models have been developed to anticipate recipe ingredients based on the title and cooking instructions [24]. Moreover, the dataset has been harnessed as a recipe generator, fostering creativity and innovation in culinary creations [25]. In response to critical issues such as the water crisis, researchers [23] have employed the dataset to recommend recipes with minimal water consumption. Notably, it has been utilized to suggest suitable alternatives for unavailable ingredients [26]. Additionally, the Food.com dataset has proven valuable in diverse rating recommendation tasks [4,14]. Its

versatility and scope make it a pivotal resource for exploring multifaceted aspects of food-related research.

4.3 Similar Datasets

Numerous food-based datasets akin to ours have been explored in the literature. In [10], the authors curated data from a German-based recipe website. Web scraping techniques were employed to gather data from Epicurious.com in [27,5], while [6] reported data collection from diverse platforms, including Yummly, Meishijie, foodspotting, and Allrecipes. Notably, [7,14] utilized the Kaggle dataset, foodRecSys-VI, featuring images absent in our dataset. This Kaggle dataset comprises recipes and ratings sourced from AllRecipes.com, a widely used platform for food recipes with accompanying ratings and reviews. [8] introduced a dataset obtained from Yummly, while [11] adopted a user survey approach to collect preference data and ratings. The literature thus offers a spectrum of datasets with varying sources and collection methodologies, enriching the landscape of culinary datasets for research and analysis.

4.4 State-of-the-art models

Recent advancements in recipe recommendation systems have introduced innovative approaches, particularly characterized by a notable emphasis on graph-based methodologies. This paradigm, as exemplified in the works of [4], strategically leverages structured user-recipe interaction data, introducing two pivotal enhancements. Firstly, the incorporation of user-recipe review embeddings, derived from a sentence-based transformer model, adeptly captures nuanced preferences embedded in textual reviews. Secondly, the integration of healthy recipe features, aligned with international standards, serves to enhance the model's depth and predictive capabilities.

In complementary research, as detailed in [5], an alternative avenue is explored with a meticulous focus on the analysis of recipe reviews. Employing distributional methods such as Information Gain and Bi-Normal Separation, this study rigorously selects features and evaluates their performance. The investigation extends to contrasting distributionally selected features with linguistically motivated ones within one-layer and two-layer frameworks for comprehensive analysis.

Recipe ratings emerge as a fundamental facet within the purview of food computing, significantly impacting food-based recommendation systems. Notably, [10] underscores the relevance of user preferences in rating studies, with a dedicated focus on comprehending user likes and dislikes through diverse recipe recommendation models. Concurrently, [11] endeavors to enhance user suggestions by parsing user history data and considering recipe ingredients.

In the realm of neural network-based approaches, [12] introduces a regression-based neural network mapping ingredient lists to recipe ratings. Iteratively altering ingredients to refine recipes, the study pushes the boundaries of recipe improvement. Furthermore, recent works incorporate both rating data and healthy nutritional features into food recommendation systems [9,13].

The significance of recipe social networks, such as Food.com, AllRecipes, and Yummly, is pivotal for rating-related tasks due to their exhaustive registers of user ratings, reviews, and interactions [6]. Text reviews from recipe social networks are recognized for their rich insights into culinary culture and user preferences, contributing to predicting dish popularity [16,17].

Finally, graph-based approaches in food computing, as proposed by [18,19], demonstrate their prowess in representing recipe objects as structured data. Leveraging these structured representations, [20,21] craft sophisticated recipe recommendation algorithms, with the added ability to predict ratings based on food ingredients [9]. The versatility of graph-based approaches extends to explainable food recommendation systems, as seen in [22], where historical data interprets top-k recommendations for users, utilizing user interactions and recipes from Food.com to construct a comprehensive recommendation system.

4.5 Conclusions from existing work

In contrast to predominant trends in the existing literature, our project diverges by employing collaborative filtering methodologies, a departure from prevalent graph-based and deep learning algorithms. Our distinctive approach involves a nuanced exploration of various similarity matrices within the collaborative filtering framework. Furthermore, in the realm of textual modeling, we introduce a novel dimension by incorporating features such as recipe description, ingredients, and title. While drawing

inspiration from prior works, particularly [24,25], our adaptation focuses on predicting ratings rather than conventional recommendation tasks. The outcomes of our study align closely with the established range observed in existing literature, contributing a unique perspective to the landscape of recommendation system research.

5. Results and Conclusions

5.1 Similarity-Based Models

For the similarity-based model, we calculate ratings using two distinct prediction functions and two different measures of similarity. The resulting MSEs on the test set are shown in the table below.

Similarity Measure	**User-based**	**Item-based**
Jaccard	0.73156	0.73136
Cosine	0.72764	0.73008

We also incorporated the temporal factors to the similarity based models. The resulting MSEs on the test set are shown in the table below.

Similarity Measure	**User-based**	**Item-based**
Jaccard	0.75315	0.72984
Cosine	0.72769	0.74892

5.2 Latent Factor Models

We used the surprise python package to implement the latent factor models. To ensure the reproducibility and comparability of the result, we set a seed value for both the model and the cross validation method that we use.

A grid search was employed to find the best set of hyperparameters. Specifically, we searched for the number of latent factors and regularization factors. After choosing the best set of hyperparameters, the model was re-trained using the full training set.

5.2.1 Bias-Only Model

Below is the test MSE value for different regularization factors

Regularization Constant	**Training MSE**
0.00001	0.68822
0.0001	0.68821
0.001	0.68819
0.01	0.68797
0.1	**0.6863**
1	0.69077
10	0.73273
100	0.74686

The training MSE is minimized at 0.6863 when $\lambda = 0.1$. On the test set, the test MSE is 0.70877. For a slight improvement, we re-trained the model on the whole training set. The final test MSE is 0.70329.

5.2.2 Latent Factor Models

Below is the test MSE value for different regularization factors

	Latent Factors			
Regularization Constant	**1**	**2**	**4**	**8**
0.00001	0.68864	0.68896	0.68956	0.69037
0.0001	0.68864	0.68896	0.68955	0.69036
0.001	0.68861	0.68892	0.6895	0.69028
0.01	0.68831	0.68857	0.68904	0.68959
0.1	**0.68641**	0.68646	0.68658	0.68648
1	0.69078	0.69078	0.69078	0.69077
10	0.73273	0.73273	0.73273	0.73273
100	0.74686	0.74686	0.74686	0.74686

	Latent Factors			
Regularization Constant	**16**	**32**	**64**	**128**
0.00001	0.69244	0.69487	0.70113	0.7091
0.0001	0.69243	0.69485	0.7011	0.70905

0.001	0.69231	0.69466	0.70079	0.7086
0.01	0.69126	0.69305	0.69813	0.70466
0.1	0.68692	0.68702	0.68806	0.68933
1	0.69079	0.69078	0.69079	0.69083
10	0.73273	0.73273	0.73273	0.73273
100	0.74686	0.74686	0.74686	0.74686

The training MSE is minimized at 0.68641 when $K = 1$ and $\lambda = 0.1$. On the test set, the test MSE is 0.70873. For a slight improvement, we re-trained the model on the whole training set. The final test MSE is 0.70329.

5.2.3 SVD++

Below is the test MSE value for different regularization factors

	Latent Factors			
Regularization Constant	**1**	**2**	**4**	**8**
0.00001	0.6973	0.70044	0.70864	0.7203
0.0001	0.69714	0.70018	0.70824	0.71955
0.001	0.69603	0.69834	0.70532	0.71423
0.01	0.69276	0.69374	0.69669	0.70023
0.1	**0.68866**	0.68874	0.68888	0.68876
1	0.69085	0.69085	0.69085	0.69085
10	0.73281	0.73281	0.73281	0.73281
100	0.7469	0.7469	0.7469	0.7469

	Latent Factors			
Regularization Constant	**16**	**32**	**64**	**128**
0.00001	0.73778	0.7505	0.76235	0.76792
0.0001	0.73655	0.74891	0.76042	0.76574
0.001	0.72787	0.73781	0.74754	0.75174
0.01	0.70635	0.71085	0.71852	0.72298
0.1	0.68928	0.68913	0.69017	0.69088
1	0.69085	0.69085	0.69085	0.69087
10	0.73281	0.73281	0.73281	0.73281
100	0.7469	0.7469	0.7469	0.7469

The training MSE is minimized at 0.68866 when $K = 1$ and $\lambda = 0.1$. On the test set, the test MSE is 0.71083. For a slight improvement, we re-trained the model on the whole training set. The final test MSE is 0.70477.

5.3 Text-Based Models

Text Features are adopted to conduct two very distinct predictive tasks.

1. Regression: The regression models perform well when compared to the baseline model. The review text does a great job of capturing the user sentiment and correlating it to the user rating.
2. Textual Compatibility based Recommendation: The model fails to capture any compatibility between recipes. This proves that using recipes metadata to compute similarity is not necessarily an ideal solution, at least in the food domain.

Model	**Test MSE**
Regression using BoW	0.60
Regression using TF-Idf	0.6234
Textual Compatibility based Recommendation	0.7388

It is important to note that the work does not intend to compare the regression task with the recommendation-based approach for rating prediction. The regression tasks make use of review text which consists of text features that the user has already written and is expected to present a better score.

5.4. Conclusion

Our research explored various models designed to predict user ratings, taking into account user interactions and characteristics of the recipes. We observed differing degrees of effectiveness across these models. Here are our key findings:

1. We observed that the accuracy of both user-based and item-based models in similarity-based approaches is comparable. This similarity in performance is attributed to the balanced number of users and recipes in our dataset, along with a comparable level of interaction between them (each user reviews at least 15 recipes and vice versa).

Additionally, incorporating temporal elements can enhance model accuracy in certain scenarios. Notably, the item-based model using Jaccard similarity showed a notable improvement, reducing the Mean Squared Error (MSE) to 0.72984.

2. Regarding latent-factor models, the bias-only model demonstrated better performance in our dataset compared to other latent factor variants, likely due to the sparsity of our dataset. Unfortunately, for our dataset, the SVD++ algorithm does not provide us with a better accuracy compared to the original SVD algorithm.

After fine-tuning the hyper-parameters, all our latent factor models surpassed both the baseline and similarity-based models in accuracy. This can be attributed to the fact that latent-factor models generally perform better on sparse datasets.

3. Models that integrate text from reviews significantly outperform other models. However, their practicality is limited as review texts are typically submitted with the user rating, making this approach more akin to sentiment analysis. Despite this limitation, these models provide a valuable benchmark for the potential effectiveness of our predictive models.

We think that there are additional opportunities to integrate temporal factors into the various models that we experimented with. One strategy could involve applying temporal bias to latent factor models, potentially accounting for transient, burst-like user behaviors. This aspect remains an area for future enhancement.

6. References

[1] Generating Personalized Recipes from Historical User Preferences. Bodhisattwa Prasad Majumder, Shuyang Li, Jianmo Ni, Julian McAuley. EMNLP, 2019

[2] S. Funk, 2006. [Online]. Available: http://sifter.org/~simon/journal/20061211.html

[3] Yehuda Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. 2008. URL: https://people.engr.tamu.edu/huangrh/Spring16/papers_course/matrix_factorization.pdf.

[4] Morales-Garzón, Andrea, et al. "How Tasty Is This Dish? Studying User-Recipe Interactions with a Rating Prediction Algorithm and Graph Neural Networks." *International Conference on Flexible Query Answering Systems*. Cham: Springer Nature Switzerland, 2023.

[5] Liu, C., Guo, C., Dakota, D., Rajagopalan, S., Li, W., Kübler, S., & Yu, N. (2014, August). "My Curiosity was Satisfied, but not in a Good Way": Predicting User Ratings for Online Recipes. In *Proceedings of the Second Workshop on Natural Language Processing for Social Media (SocialNLP)* (pp. 12-21).

[6] Min, Weiqing, et al. "A survey on food computing." *ACM Computing Surveys (CSUR)* 52.5 (2019): 1-36.

[7] Chavan, Pallavi, Brian Thoms, and Jason Isaacs. "A recommender system for healthy food choices: building a hybrid model for recipe recommendations using big data sets." (2021).

[8] Cueto, Paula Fermín, Meeke Roet, and Agnieszka Słowik. "Completing partial recipes using item-based collaborative filtering to recommend ingredients." *arXiv preprint arXiv:1907.12380* (2019).

[9] Rostami, Mehrdad, et al. "A novel healthy and time-aware food recommender system using attributed community detection." *Expert Systems with Applications* 221 (2023): 119719.

[10] Harvey, Morgan, Bernd Ludwig, and David Elsweiler. "You are what you eat: Learning user tastes for rating prediction." *String Processing and Information Retrieval: 20th International Symposium, SPIRE 2013, Jerusalem, Israel, October 7-9, 2013, Proceedings 20*. Springer International Publishing, 2013.

[11] Freyne, Jill, and Shlomo Berkovsky. "Intelligent food planning: personalized recipe recommendation." *Proceedings of the 15th international conference on Intelligent user interfaces*. 2010.

[12] Russo, Abigail, Brynne Hurst, and Trey Weber. "TastifyNet: Leveraging adversarial examples for generating improved recipes."

[13] Khan, Mansura A., et al. "Personalized, health-aware recipe recommendation: an ensemble topic modeling based approach." *arXiv preprint arXiv:1908.00148* (2019).

[14] Vani, K., and K. Latha Maheswari. "Novel Nutritional Recipe Recommendation." *Journal of Information Technology* 5.1 (2023): 1-12.

[15] Gona, Sai Nikhil Rao, and Himamsu Marellapudi. "Suggestion and invention of recipes using bi-directional lstms-based frameworks." *SN Applied Sciences* 3 (2021): 1-17.

[16] Teng, Chun-Yuen, Yu-Ru Lin, and Lada A. Adamic. "Recipe recommendation using ingredient networks." *Proceedings of the 4th annual ACM web science conference*. 2012.

[17] Mao, Xudong, Yanghui Rao, and Qing Li. "Recipe popularity prediction based on the analysis of social reviews." *2013 International Joint Conference on Awareness Science and Technology & Ubi-Media Computing (iCAST 2013 & UMEDIA 2013)*. IEEE, 2013.
[18] Adaji, Ifeoma, et al. "Personality based recipe recommendation using recipe network graphs." *Social Computing and Social Media. Technologies and Analytics: 10th International Conference, SCSM 2018, Held as Part of HCI International 2018, Las Vegas, NV, USA, July 15-20, 2018, Proceedings, Part II 10*. Springer International Publishing, 2018.
[19] Gharibi, Mohamed, Arun Zachariah, and Praveen Rao. "Foodkg: A tool to enrich knowledge graphs using machine learning techniques." *Frontiers in big Data* 3 (2020): 12.
[20] Forouzandeh, Saman, Mehrdad Rostami, and Razieh Sheikhpour. "Hfrs-Han: Health-Aware Food Recommendation System Based on the Heterogeneous Attention Network." *Razieh, Hfrs-Han: Health-Aware Food Recommendation System Based on the Heterogeneous Attention Network* (2023).
[21] Song, Yaguang, Xiaoshan Yang, and Changsheng Xu. "Self-supervised calorie-aware heterogeneous graph networks for food recommendation." *ACM Transactions on Multimedia Computing, Communications and Applications* 19.1s (2023): 1-23.
[22] Yera, Raciel, Ahmad A. Alzahrani, and Luis Martínez. "Exploring post-hoc agnostic models for explainable cooking recipe recommendations." *Knowledge-Based Systems* 251 (2022): 109216.
[23] Gallo, Ignazio, et al. "Food recommendations for reducing water footprint." *Sustainability* 14.7 (2022): 3833.
[24] H. Lee, Helena, et al. "RecipeGPT: Generative pre-training based cooking recipe generation and evaluation system." *Companion Proceedings of the Web Conference 2020*. 2020.
[25] Bień, Michał, et al. "RecipeNLG: A cooking recipes dataset for semi-structured text generation." *Proceedings of the 13th International Conference on Natural Language Generation*. 2020.
[26] Shirai, Sola S., et al. "Identifying ingredient substitutions using a knowledge graph of food." *Frontiers in Artificial Intelligence* 3 (2021): 621766.
[27] Khan, Ismam Hussain, Md Habib Ullah Khan, and Md Mamun Howlader. "An Intelligent Approach for Food Recipe Rating Prediction Using Machine Learning." *2021 1st International Conference on Artificial Intelligence and Data Analytics (CAIDA)*. IEEE, 2021.