CSE 158 Assignment 2

Recipe Predictor

Introduction

Our goal for this assignment is to analyze a dataset of recipes and user ratings and build a binary classification model that predicts whether or not a user has interacted with a recipe. We want to build the most accurate model possible, hoping to incorporate several techniques from class to do so. This write up will summarize our entire process, starting from choosing and pre-processing our dataset, to the build of our final model. We will go over specifics at each process, and we will conclude with the final overall performance of our model.

Dataset Description

For our dataset, we chose to use a Food.com dataset collected by Shuyang Li and Bodhisattwa Prasad Majumder on Kaggle which comprises of multiple CSV files: PP_recipes.csv, PP_users.csv, RAW_interactions.csv, RAW_recipes.csv, ingr_map.pkl, interactions_test.csv, interactions_train.csv, and interactions_validation.csv. For our analysis we specifically chose to focus on three of these CSV files: RAW_recipes.csv, RAW_interactions.csv, and interactions_train.csv. The dataset includes ~83,782 recipes (RAW_recipes.csv) described by 12 attributes, such as the recipe name, unique ID, preparation time, contributor ID, submission date, associated tags, nutritional information, number of steps, instructions, ingredients, and a brief description of each recipe. The dataset also contains ~731,927 user interactions (RAW_interactions.csv) , including ratings, reviews, and dates for the recipes. These two datasets were merged using the recipe ID to pair recipes with their corresponding user feedback, enabling a comprehensive exploratory data analysis. Our predictive model leverages the interactions_train.csv file from the Food.com dataset. This file contains user-recipe interaction data, structured to provide valuable insights into user preferences and behavior. The dataset comprises interactions where each record details the user's engagement with a specific recipe. Each record in the dataset represents a unique engagement between a user and a recipe, with key attributes including user_id , recipe_id, date , rating, u and i. This dataset is substantial, containing 731,927 entries, making it suitable for robust predictive modeling.

RAW_recipes.csv:

Variable	Description
`name`	Name of Recipe
`recipe_id`	Unique identifier for the recipe
`minutes`	Time required to prepare the recipe (in minutes)
`contributor_id`	ID of the user who submitted the recipe
`submitted`	Submission date (in YYYY-MM-DD format)
`tags`	Associated tags for the recipe
`nutrition`	Nutritional information in the format [calories (#), total fat (PDV), sugar (PDV), sodium (PDV), protein (PDV), saturated fat (PDV), carbohydrates (PDV)], where PDV stands for "Percentage Daily Value"
`n_steps`	Number of steps in the recipe
`steps`	Detailed instructions for preparation of recipe
`description`	Brief description of the recipe
`ingredients`	List of ingredients used
`n_ingredients`	Number of ingredients required

Figure 1.1

RAW_interactions.CSV:

Variable	Description
`user_id`	ID of the user who posted the review
`recipe_id`	ID of the reviewed recipe
`date`	Date of the review (in YYYY-MM-DD format)
`rating`	Star rating given (1 to 5)
`review`	Textual review of the recipe

Figure 1.2

interactions_train.csv:

Variable	Description
`user_id`	ID of the user who posted the review
`recipe_id`	ID of the reviewed recipe
`date`	Date of the review (in YYYY-MM-DD format)
`rating`	Star rating given (1 to 5)
`u`	User ID, mapped to contiguous integers from 0
`i`	Recipe ID, mapped to contiguous integers from 0

Figure 1.3

Exploratory Data Analysis

Our exploratory data analysis begins with examining basic statistics using the .describe() function. Preparation times (minutes) show a wide range with some extreme outliers, while the number of steps (n_steps) and ingredients (n_ingredients) follow more reasonable distributions centered around 7-8 steps and 9 ingredients, respectively. Outliers, particularly in minutes, are identified and removed using the Interquartile Range (IQR) method. This involves calculating the IQR and filtering data outside the range defined by 1.5 times the IQR from the 25th and 75th percentiles.

The basic statistics of the quantitative variables in RAW_recipes.csv are as follows[1]:

	minutes	n_steps	n_ingredients
Mean	36.613	8.592	8.535
Standard Deviation	25.364	4.279	3.283
Minimum	0	0	1
Q1	20	5	6
Median	30	8	8
Q3	50	11	11
Maximum	132	21	18

Figure 2.1

[1] The dataset had to be preprocessed, as there were outliers in all of these columns

Visualizations provide deeper insights into the data. Therefore, we created histograms to illustrate the distributions of minutes, n_steps, and n_ingredients before and after cleaning. Removing the outliers shows a distribution relatively close to the Normal distribution, albeit skewed to the right.

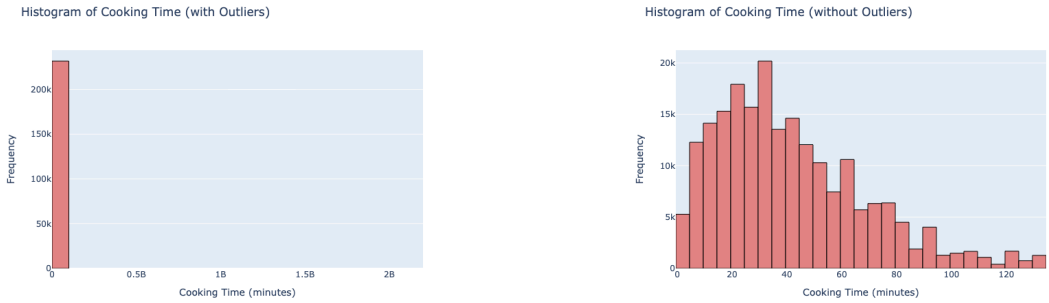

Figure 2.2

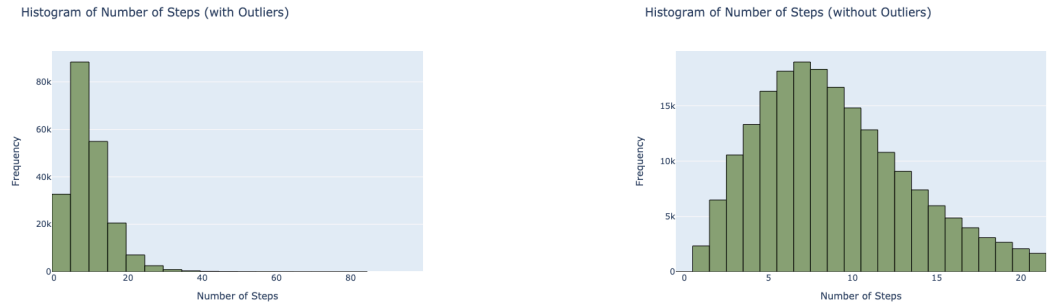

Figure 2.3

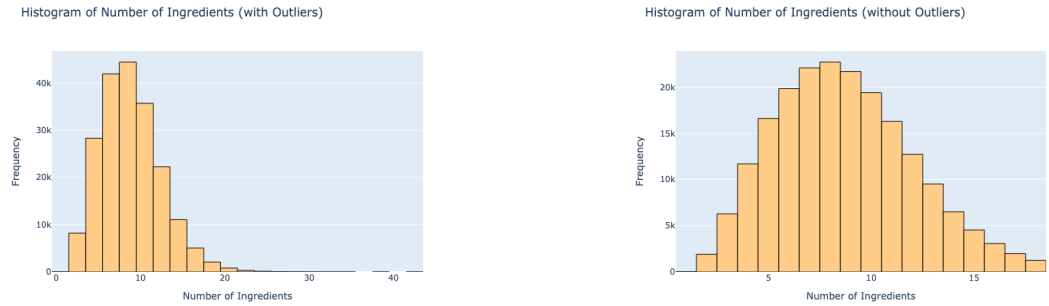

Figure 2.4

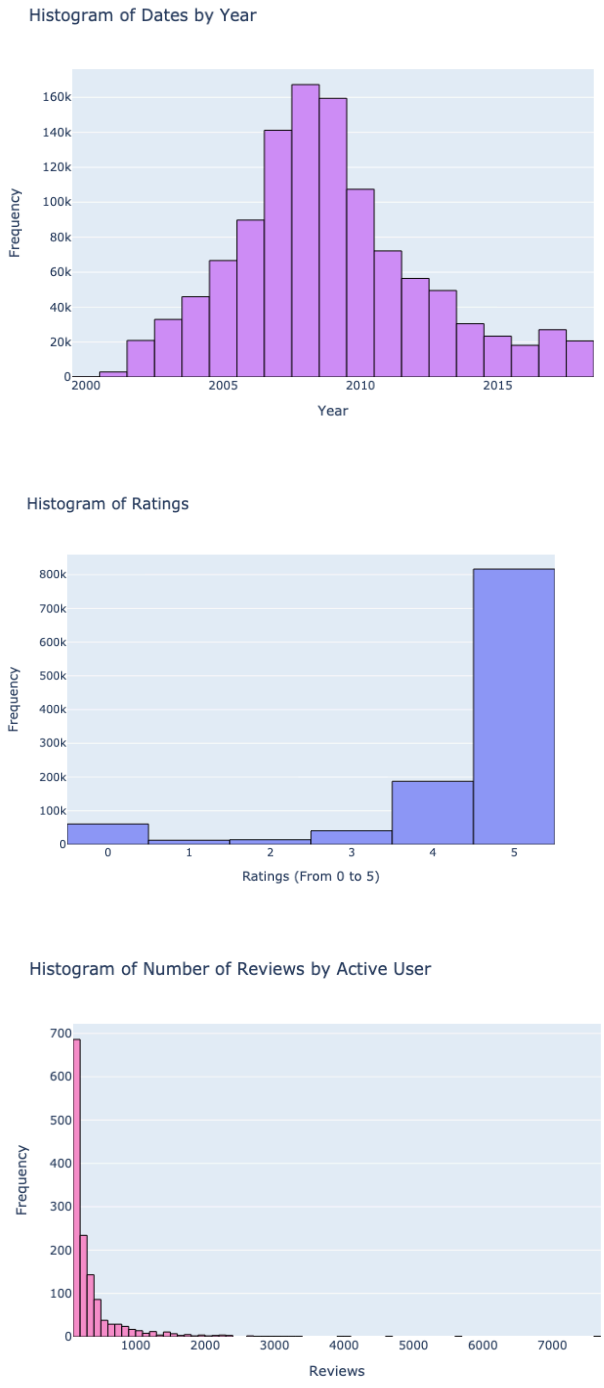

Figure 2.5

In Figure 2.5, the distributions show some clear takeaways from the RAW_interactions.csv dataset.

First, most interactions are collected on or around the year 2008, showing a slightly normal distribution once again.

The ratings distribution shows that most people tend to vote along the extremes, with the least ratings being 1 and 2.

Furthermore, the distribution of reviews by users shows that most people post less than 10 reviews, with a few outliers that post thousands of reviews.

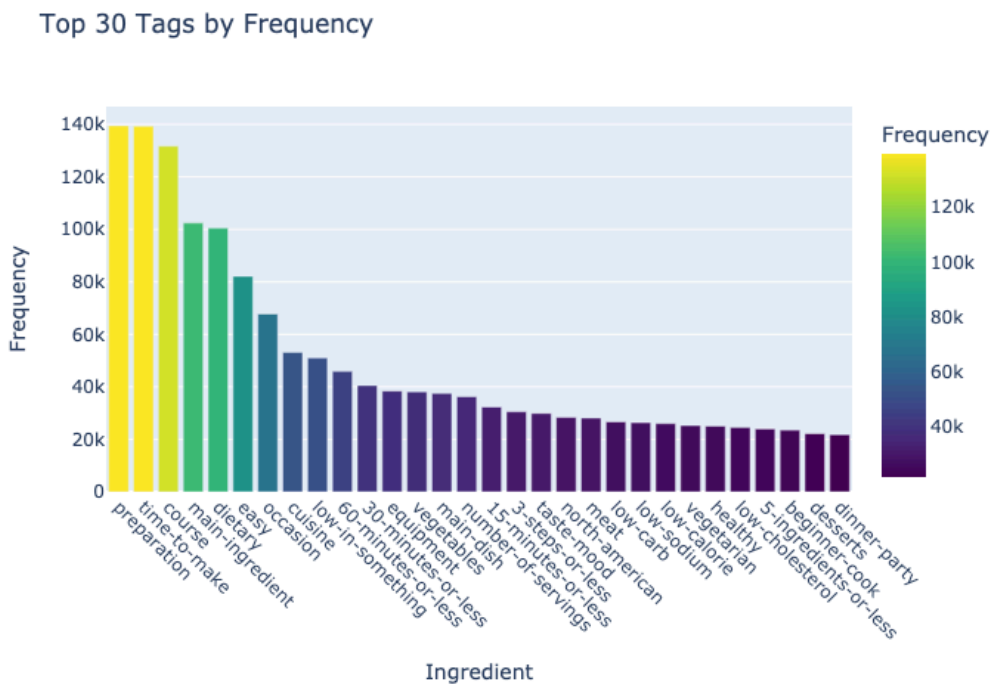

Figure 2.6

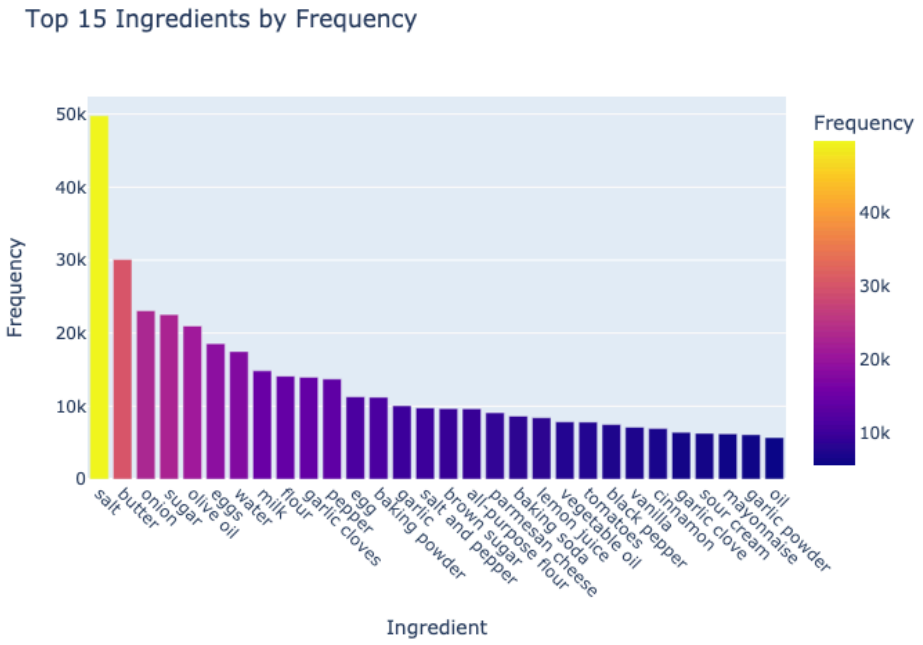

Figure 2.7

Predictive task

The predictive task that we wanted to conduct on our dataset focused on determining whether or not a specific user has interacted with (rated) a particular recipe.

Model Evaluation

The primary metric we used for evaluating the model is accuracy, which measures the proportion of correct predictions (interacted vs. not interacted) relative to the total number of predictions. We assessed the validity of our model's prediction by creating a validation set with a size of approximately 2% of our training set. Our validation set contained equal parts of both positive samples(user did rate recipe), and negative samples(user did not rate recipe). The performance of our model was determined by how accurately it could predict against this validation set.

Baseline Model

A good starting heuristic to build our baseline model off of was by judging how popular a particular recipe was, determined by the number of reviews it had. Popularity was a strong metric to determine whether or not a book was read in Assignment 1, so we also decided to include popularity as a metric in our model as well. By getting the number of reviews for each recipe in our dataset, and then sorting by that number, we were able to obtain a list of the most popular recipes. Using this list, our model will:

- Predict "yes" if the recipe is in the top half of the **popular recipes**
- Predict "no" otherwise.

Using this baseline, our model achieved an accuracy of just around 70%, which was a great starting point for our task.

Features and Processing

User Features: Information derived from user_id, such as their historical interactions or preferences.

Recipe Features: Characteristics inferred from recipe_id, including interaction counts (popularity).

Both of our main features come from our RAW_interactions csv file which contains columns user_id, recipe_id, date, rating, and review. The attributes from this dataset we care most about when we want to analyze interactions are user_id, recipe_id, and rating. We processed this data by getting rid of the columns that we didn't need, and then looking for rows with invalid or useless data. We removed rows that contained any NaN values and any row that had a rating score of 0, meaning they did not give the recipe any score.

Supplemental metrics

On top of our main popularity metric, we decided to use similarity functions as well to improve the accuracy of our model past the baseline. We used Jaccard and Cosine similarity based on the user to recipe interactions that were created using our training set. We compared all recipes that the user has previously rated and grabbed the largest Cosine and Jaccard similarity of all previous rated recipes in comparison to the recipe we were evaluating. We made many different changes to our similarity thresholds in order to find the range of similarities that yielded the most accurate results. We used a tweaked version of the Jaccard and Cosine similarity functions in lecture however we gave a stronger weight to recipes that the user gave a higher rating to in order to have a more accurate model. We also shrunk our viable list of popular recipes in order to give popularity less of a weight on whether or not the user would rate this recipe or not and give a higher weight to our similarity functions. After implementing these changes we saw an increase in the accuracy of our model, jumping from 69% with our baseline model to 79% with our optimized model.

Model Justification

We built our model based around assignment 1 because we felt it would be the most familiar model to work with and we saw how successful it was in the book dataset used in Assignment 1. Our baseline was the same as in Assignment 1 where we had a list of the most popular recipes and if the recipe was in the top 50%, then we would predict the user makes the recipe. This was our baseline for this dataset. In order to improve on this dataset, we decided to build upon our popularity heuristic while also taking the cosine and jaccard similarity between user-recipe interaction patterns. We thought that a user would have higher engagement on a recipe if the recipe was rated by a similar group of users and that the similarity heuristic + popularity metric would hold more weight than just popularity alone

Issues and Unsuccessful Attempts

Implementing both Jaccard and Cosine similarity as metrics proved to be a tough task. Using functions to calculate similarity between two sets of users, we hoped to predict that a user interacted with a recipe if their Jaccard and Cosine similarities were above a certain threshold. We iteratively tested many possible thresholds for both similarity functions, but yielded no successful results. No matter what the threshold was, our accuracy never improved. We tried evaluating this behavior by printing out a confusion matrix to see how exactly our model was predicting, and we realised that these similarity functions were leading our model to make a lot of false positive predictions. The failure of our similarity functions may be explained by the sparsity of our dataset. There are many recipes with less than 20 reviews and many users who may not be as active, with less than 10 reviews. Also, similarity between users, and the recipes they rated may not be as strong of a metric as we thought for determining whether or not a user has interacted with a recipe. In conclusion, with this particular dataset, we found out that popularity alone was a stronger heuristic in predicting interaction than Cosine and Jaccard similarity metrics.

Other Models Considered

On Kaggle, we saw other recommender systems that made use of the Term Frequency - Inverse Document Frequency (TF-IDF) values of a bag-of-words vector used on the ingredients used in the recipe. One weakness of this model is that while a recipe may have similar ingredients to another recipe, their preparation, steps, cooking style, may all be different and that maybe certain ingredients wouldn't be strong enough indicators on whether a person would like or dislike a recipe. Another weakness is that we would not know if our predictor is accurate or not. If we returned a list of the top 5 most similar recipes, how would we know if our 5 recommended recipes are reliable? We struggled a lot with having a numerical metric to measure the accuracy or validity of our recommender system. One strength using this method of recommendation is that it is easy to interpret. We show you 5 recipes with very similar ingredients to this recipe. It is also very efficient when working with a large dataset if TF-IDF values and appropriate bag-of-words vector is already established.

Relevant Literature

The Food.com dataset we selected, consisting of over 180,000 recipes and more than 700,000 interactions spanning 18 years, has been widely used for research on recipe recommendation systems, user preferences, and recipe generation. Originating from user uploads and interactions on Food.com (formerly GeniusKitchen), this dataset was a key resource in the paper "*Generating Personalized Recipes from Historical user Preferences"*(Majumder et al., 2019). This study aimed to create personalized recipes based on incomplete input specifications like partial ingredients and user preferences extracted from historical interactions. The authors utilized an encoder-decoder model with attention mechanisms to incorporate user-specific preferences, significantly improving recipe personalization compared to non-personalized baselines. They observed that personalization leads to higher coherence and plausibility in generated recipes, which aligns with findings that user preferences strongly influence recipe engagement and satisfaction. Similarly, the Kaggle project "Recipe Recommendation System" (engyyyy, 2024) applies collaborative filtering and textual analysis (e.g., TF-IDF) to recommend recipes. This

system emphasizes personalized user experiences by analyzing interactions and recipe metadata, supporting conclusions that recipes with higher ratings and simpler steps are more popular, reflecting user preferences for accessibility and quality. Both studies conclude that incorporating user preferences enhances the effectiveness of recipe personalization or recommendation. This aligns with our recipe popularity. However, while prior work focused more on advanced NLP techniques for generating or recommending recipes, our analysis focused more on using Jaccard and Cosine similarities to predict user interactions with different recipes. Together these findings validate the importance of personalization in recipe datasets.

Similar Datasets

The Recipe1M+ dataset is a large-scale dataset containing over 1 million recipes, their metadata, and images. It has been extensively studied for its applications in cross-modal tasks, such as matching recipe text with food images and generating recipes from images. Each recipe in the dataset includes structured data like titles, ingredients, instructions, and associated images, making it a valuable resource for advancing computer vision and natural language processing (NLP) in the food domain. One prominent application of Recipe1M is cross-modal retrieval, where neural embedding models connect food images and recipe text in a shared semantic space, enabling image-based recipe search and vice versa. Researchers have also used the dataset to train models for generating recipes directly from images by combining convolutional neural networks (CNNs) for image encoding with recurrent neural networks (RNNs) or transformers for text generation, producing coherent recipes based on visual input. Ingredient recognition is another focus area, with models predicting ingredients from either text descriptions or associated images. Additionally, Recipe1M+ has been utilized for nutritional analysis and recommendation systems, offering dietary suggestions, healthier alternatives, or caloric predictions based on ingredient lists. Beyond this, the dataset has inspired broader multimodal AI applications, such as joint embedding spaces for image-text retrieval, reinforcement learning to enhance recipe generation quality, and hierarchical models for aligning recipe steps with visual data. While Recipe1M emphasizes the

integration of images with textual recipe data, the Food.com dataset focuses more on user interactions, ratings, and textual metadata like ingredients and instructions. Together, these datasets provide complementary insights, with Recipe1M excelling in multimodal tasks and Food.com offering a deeper understanding of user preferences and behaviors. Recipe1M's versatility highlights its critical role in advancing AI applications in culinary research, from personalized recommendations to automated recipe creation.

Conclusion

Our model did a great job on predicting whether or not a user would rate a certain recipe, achieving almost 80% accuracy. However, we feel that we overfit the model to our predictor as we did not strongly implement the two different similarities into our model as we got much higher accuracy levels basing each user-recipe pair on the popularity of the recipe. In our EDA, we made sure to remove any outliers that, when we didn't remove, made our model very inaccurate and gave us a rough time in creating our model. After removing anomalies in our data, we found that our dense dataset was easier to look at and made our analysis a lot faster. Our model may have worked with the test and validation set we created (79% accuracy) however we feel that in other datasets where the recipes are more sparse and distributed more evenly, our model may not be the best predictor. Our model and the following heuristics we chose to measure were very simple and only sought very surface level factors. Our model failed to consider latent preferences of users like favorite ingredients, preferred preparation difficulty, preferred cuisines and tastes. Although this time around we made an accurate model for our dataset, next time around we will try other, more complicated measurements of features in order to make our model more well-equipped for other datasets.

Works Cited

engyyyy. (2024, June 27). *Recipe Recommendation System*. Kaggle.com; Kaggle. https://www.kaggle.com/code/engyyyy/recipe-recommendation-system

Majumder, B., Li, S., Ni, J., & Mcauley, J. (2019). *Generating Personalized Recipes from Historical User Preferences* (pp. 5976–5982). Association for Computational Linguistics. https://aclanthology.org/D19-1613.pdf

Recipe1M+: A Dataset for Learning Cross-Modal Embeddings for Cooking Recipes and Food Images - MIT. (2019). Mit.edu. https://im2recipe.csail.mit.edu/