Recommender Systems on Food.com Recipes

Yi Li*
yil115@ucsd.edu
UC San Diego
San Diego, California, USA

Weiyue Li
wel019@ucsd.edu
UC San Diego
San Diego, California, USA

Xiaoyue Wang
xiw027@ucsd.edu
UC San Diego
San Diego, California, USA

Ruoyu Hou
rhou@ucsd.edu
UC San Diego
San Diego, California, USA

ABSTRACT

Personalization is getting more and more crucial for user experiences on various websites. In this paper, we first performed exploratory data analysis on datasets from food.com. We then implemented various types of recommendation system models to recommend recipes to users, predict ratings based on sentiment analysis, and predict recipe categories. We have a baseline and an improved model for each predictive task for comparison purposes. After fine-tuning the hyperparameters for our improved models, we achieved 73.87% testing accuracy on the recipe recommendation model, 0.96 testing MSE on the rating predictions, and 94.43% testing accuracy on the category predictions. We further propose future research directions to explore and validate the causes of such differences between the baseline models and the improved models.

1 INTRODUCTION

The year 2020 is definitely a great hit to the global economy and people's lifestyles. The Economic Recession directly affected consumers spending 8 percent less on food in 2020 compared with 2019 [8]. Combined with the global mandatory lockdown forced more and more people to cook at home. However, cooking was new for a large portion of the population since people were in better economic conditions that allowed them to eat out more frequently and had easier access to places like school or work cafeterias. This emphasizes the significance of good recipes for newbies to cook at home. More importantly, the recipes must be clear, concise, and easy to follow; they teach people to cook delicious food! Moreover, while some people may stick on and repeat several recipes they are already familiar with, we want to bring them more different recipes that are close to their preferences. Thus, this has created a challenge for recipe websites to recommend personalized recipes to users.

Recommendation algorithms are best known for their use on e-commerce Web sites, where they use input about a customer's interests to generate a list of recommended items [4]. We can generalize similar ideas to websites like food.com, which is one of the main websites where contributors post their recipes and people(the users) review, deploy and rate them.

There are three aspects that this paper has explored. The first aspect is recipe recommendation. A study shows that the proper strategies of learning from users' information can produce acceptable recommendations [6]. Thus, we want to test how accurately we can use users' past activities to recommend a recipe they would want to try out. The second aspect is rating prediction. We are interested in exploring the rating of each recipe because that is usually the first thing that attracts people's attention to this specific recipe. Furthermore, ratings provide a quantitative evaluation for the assistance of personalization in output recipes [5]. And lastly, the third aspect is category prediction. We are interested in categorizing different recipes into specific categories to make a search on different categories much easier and more accurate when the authors forget or mislabel the correct categories of their recipes.

2 DATASET

2.1 Dataset details

The statistic for the recipes dataset is shown in Table 1 below.

Table 1: Statistics for Raw_recipes

Column	Description	Non-Null Entries	Data Type
name	Recipe name	231636	object
id	Recipe ID	231637	int64
minutes	Minutes to prepare recipe	231637	int64
tags	Food.com tags for recipe	231637	object
n_steps	Number of steps in recipe	231637	int64
nutrition	Nutrition information (calories, total fat, sugar, sodium, protein, saturated fat)	231637	object
n_ingredients	Number of ingredients	231637	int64

We broke up the nutrition list into different measurements and dropped 136910 observations that are outliers(the interquartile range method) to the distribution in each column of calories, total fat, sugar, sodium, protein, saturated fat, carbohydrates, minutes, n-steps, and n-ingredients.

Table 2: Statistics for Raw_interactions

Column	Description	Non-null Entries	Data Type
user_id	User ID	1132367	int64
recipe_id	Recipe ID	1132367	int64
rating	Rating given	1132367	int64
review	Review text submitted by user	1132198	object

The interaction dataset has 113K+ interaction entries between users and recipes also ranging from the year 2000 to 2018. The

*All authors contributed equally to this project.

statistic for this second dataset is shown in Table 2 above.

To better understand the relation between recipe details and users' reflections, we merged two dataset together based on the id in cleaned Raw_recipes and recipe_id in Raw_interactions and get 46K+ data. Besides of columns created from breaking up nutrition, we also created new columns is_dessert, 10_and_more_steps, description_len, and sentiment. The statistic for new columns in merged dataset is shown in Table 3 below.

Table 3: Statistics for New Columns in Merged Dataset

Column	Description	Non-null Entries	Data Type
calories	The calories for this recipe	462350	int64
total fat	The total amount of fat in this recipe	462350	int64
sugar	The sugar in PDV in recipe	462350	int64
saturated fat	The saturated fat in PDV in recipe	462350	int64
is_dessert	Whether the recipe contains dessert tag	462350	bool
10_and_more_steps	Whether the steps needed is over 9 or not	462350	bool
rating	Rating given	462350	int64
description_len	The length of reviews provided by user	462350	float64
sentiment	The sentiment analysis score	462350	float64

We have also found that many 0 ratings seem to have a positively strong sentiment in their reviews. After exploring the website, we found out that these ratings of 0's were because some users forgot to rate after writing a review, as the lowest rating you could give is 1. Thus, we have imputed those 0 ratings by the rating mean of each user to estimate the true distribution of ratings.

2.2 Exploratory Data Analysis

After discarding and imputing unreasonable data, we explore the statistics and properties of our dataset.

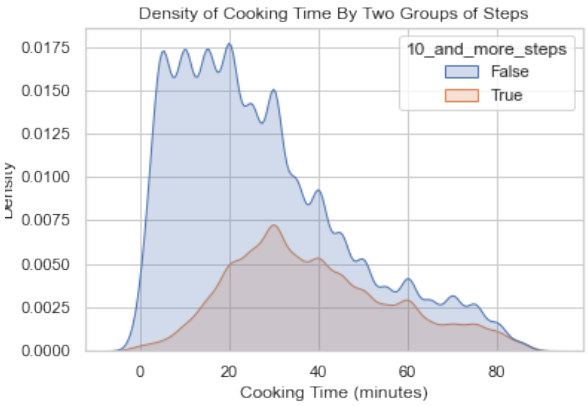

Figure 1: The Plot of Density of Cooking Time (minutes) By Whether the Number of Steps is Over Nine or Not

First, in Figure 1, we group the data into two groups according to the size of their steps, and we want to explore the relationship between the cooking time spent and the number of steps needed. We found that recipes with fewer steps need less cooking time than those with more steps.

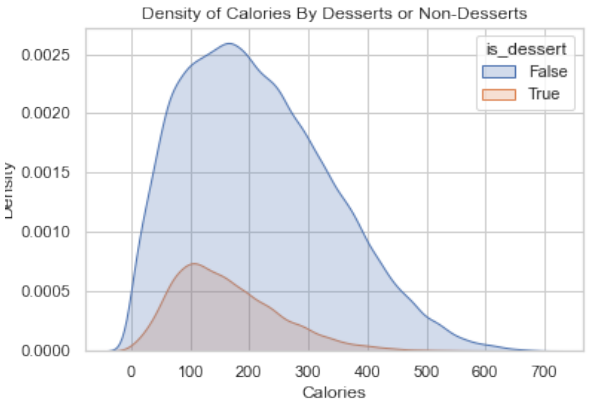

Figure 2: The Plot of Density of Calories By Whether the Recipe is Dessert or Not

We group the recipes according to one of their tags; we classify those recipes as dessert if their tags contain dessert or not dessert. The Figure 2 shows that, surprisingly, much non-dessert food has relatively more calories. This may be counter-intuitive to our knowledge, but the statistics in this dataset showed as a graph.

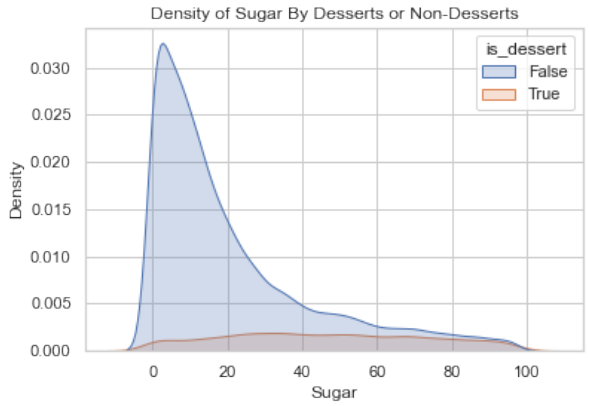

Figure 3: The Plot of Density of Sugar By Whether the Recipe is Dessert or Not

To continue exploring dessert and non-dessert groups, we plot the above graph of sugar density for each group as in Figure 3. We find out that those non-dessert recipes typically use less sugar in their food, while many dessert recipes use more sugar.

Further, we plot out the density graph for the number of steps needed for dessert and non-dessert groups in Figure 4. According to the graph, both overall distributions of the number of steps for dessert and non-dessert recipes have the most common steps of around 6-8 steps to make food. On the other hand, this shows that the dessert property, in some sense, has less effect on the number

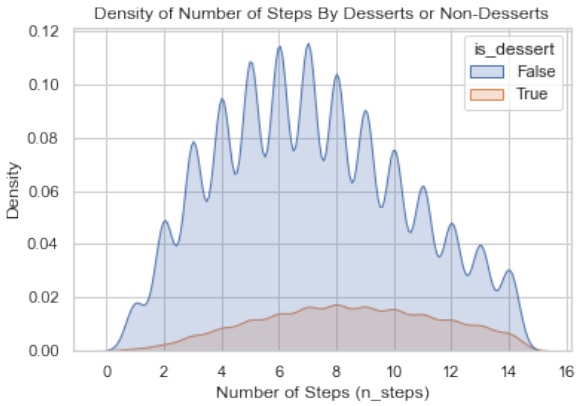

Figure 4: The Plot of Density of Number of Steps By Whether the Recipe is Dessert or Not

of steps recipes will need to prepare food.

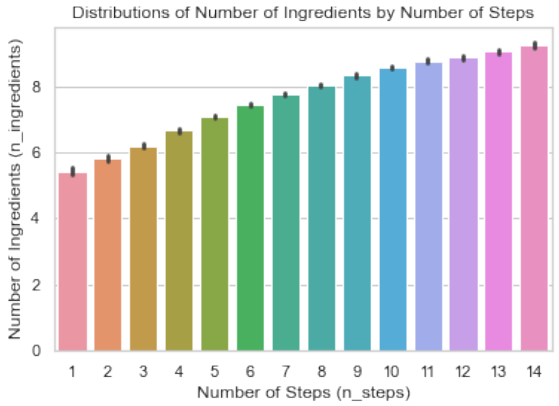

Figure 5: The Plot of Distribution of Number of Ingredients By Number of Steps

To explore the effects between the number of steps needed and the number of ingredients used, we plot the distribution of the number of ingredients by the number of steps as Figure 5. According to the graph, we find out that with more steps needed to prepare food, more ingredients are involved in the recipes. This is in line with our cognition that more ingredients need more steps to prepare and cook.

After a quick look at the densities of calories and sugar by desserts or non-desserts, we would like to have a more general sense of correlations between different nutrition measures. To best visualize it, we generated a correlation heatmap (Figure 6). From the heatmap, we can see that calories, total fat, and saturated fat are strongly positively correlated. Meanwhile, sugar has low correlations (close to 0) with all other nutrition measures. This could give us an insight into how similar recipes might be in terms of nutrition measures.

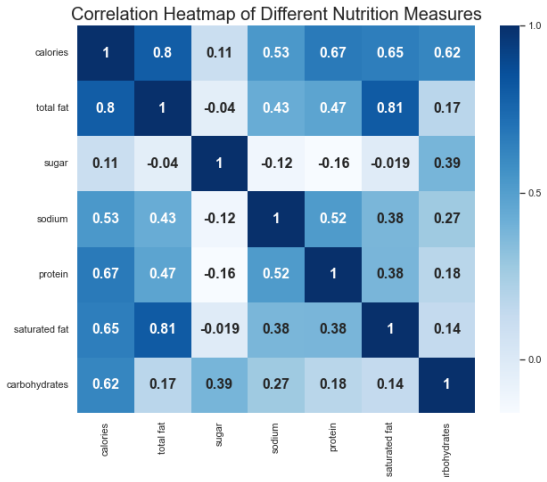

Figure 6: The Correlation Heatmap of Different Nutrition Measures

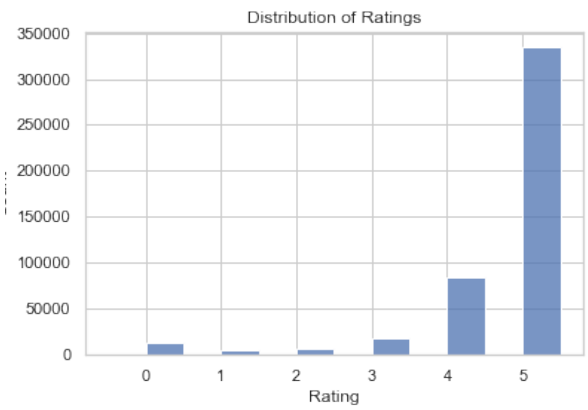

Figure 7: The Plot of Distribution of Ratings

Next, before we perform predictions on the ratings of recipes, we plot the distribution of ratings given by users to see the pattern as Figure 7. The plot shows that most ratings are either 4 or 5, while ratings from 0 to 3 only take a minor portion. Note that there is only a small portion of 0 ratings, meaning that only a small population of reviews were made without ratings. Moreover, the distribution has surprisingly peaks at the rating of 5, indicating that most interactions are positive and users are more likely to rate and review recipes they like. This pattern is reasonable if we consider the users' mentality and behaviours: users are more likely to try out those recipes they are interested in or have preferences (ingredients, style, etc.). Similarly, if a user was not satisfied with some recipe, rather than complaining and giving a negative feedback, he or she might just think this is not favored. As such, the ratings of recipes are biased as they are subjective and heavily dependent on the user's

dietary habit. Given the biased distribution, we should be careful when conducting predictions on ratings: for example, consider similarities between ingredients of some user's tried recipes.

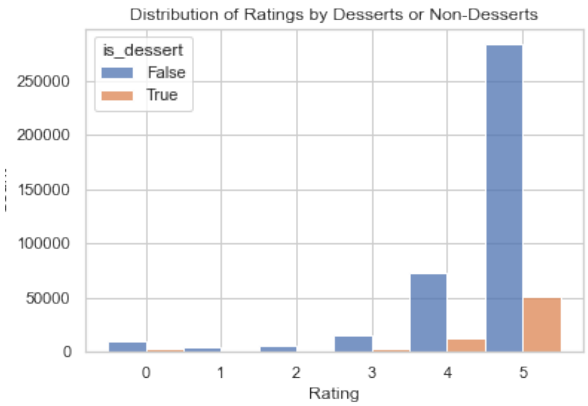

Figure 8: The Plot of Distribution of Ratings by Desserts or Non-Desserts

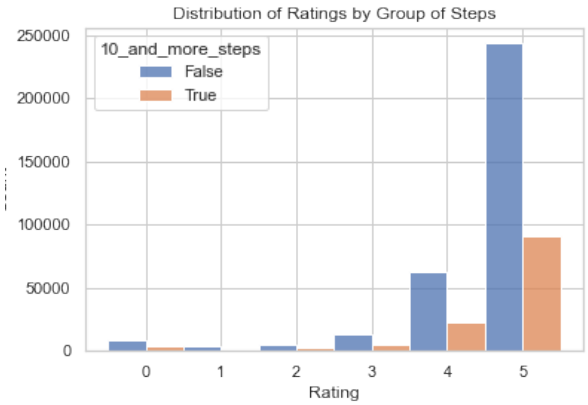

Figure 9: The Plot of Distribution of Ratings by Group of Steps

Further, we plot the distributions of ratings by group of categories (desserts or non-desserts) and by group of steps (Figure 8 and Figure 9). The plots shows that the distribution of ratings has no obvious variation across groups but displays a uniformity. The results tend to show that the bias of ratings is independent of groups.

Figure 10 and Figure 11 show that the distributions of ratings are generally uniform by number of ingredients and by number of steps, except that 1-ingredient recipes have a slightly lower mean rating and larger variation. These distributions indicate that difficulty or complexity of a recipe has little effect on the ratings it would receive.

Figure 12 is a scatter plot of saturated fat and total fat by group of desserts or non-desserts. The plot first shows that there is a positive, but loosely linear relation between saturated fat and total fat.

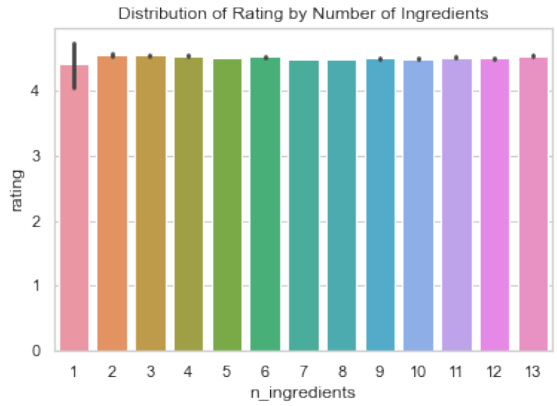

Figure 10: The Plot of Distribution of Rating by Number of Ingredients

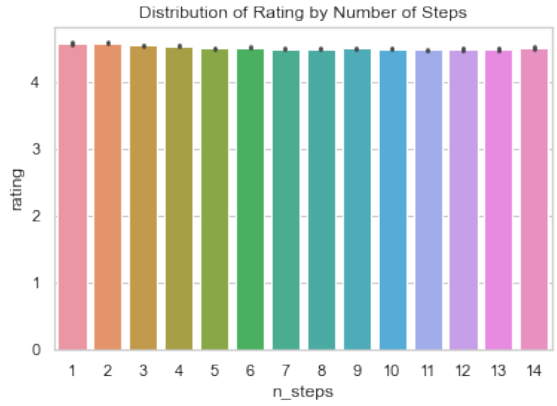

Figure 11: The Plot of Distribution of Rating by Number of Steps

Herein, the linear relationship between saturated fat and total fat is stronger for desserts compared to non-desserts, and the slope tends to be more inclined. This might indicate that for desserts, saturated fat and total fat describe more similar latent features of a recipe.

3 PREDICTIVE TASK A

3.1 Introduction for Task A

Being able to recommend recipes based on each user's personality is an essential challenge for the website, as recommending good recipes in terms of each individual user's perspective will attract them to visit the website more often, which will generate more revenue for the website. Thus, for the first predictive task, we will explore a method that can accurately predict if a user will interact with a specific recipe to make better recipe recommendations.

Figure 12: The Scatter Plot of Saturated Fat vs. Total Fat by Desserts

3.2 Feature Extraction for Task A

The purpose of Task A is to predict whether the user will use the given recipe or not based on the recipe's popularity. Our exploratory data analysis found that most "newbies" choose popular recipes as their starting point. Many experienced users also like to try out popular recipes. Therefore, we infer a positive correlation between the recipe's popularity and the likelihood of the user using the given recipe. As a result, we decide to extract the "user_id" and "recipe_id" columns from the original table for the current task. To obtain the popularity of recipes, we use the dictionary (hash table) as our data structure to count the number of users for each recipe. At the same time, we also manually create the same amount of negative samples to ensure that the model we are going to build is not biased. The negative sample, in this case, represents the user-recipe pair in which the user has not interacted with the recipe.

3.3 Descriptions of Features and Labels for Task A

After performing data pre-processing, we built a model to predict whether the user has used the given recipe through the extracted feature. The range of our predictions (labels) $y \in \mathbb{R}$ will be binary, where $y_i \in \{0, 1\}$ $\forall 0 \leq i \leq n$ (0: predicting used; 1: predicting unused), and the range r of our extracted feature (popularity of recipes) will satisfy $r \in [1, 1609]$ s.t. $r \in \mathbb{N}$.

3.4 Baseline Model for Task A

We design a naive approach for the baseline model without performing gradient descent in any sense. More specifically, we will predict that the user will interact with the given recipe if the popularity of the given recipe is in the top percentile. Finally, we will compare the predicted results with binary y-labels and calculate the overall accuracy by counting the number of correct predictions divided by the test size.

3.5 Improved Model Selection for Task A

Since Predictive Task A is a binary classification task, many suitable supervised learning models(such as Logistic Regression, Random Forest Classification, and Support Vector Machine) are options to choose from. We decide to use Logistic Regression and Random Forest Classification in our study.

- Logistic Regression: The linear relationship is represented by $z(x) = wx + b$, and the non-linearity is represent by $f(z(x)) = \frac{1}{1+e^{-z(x)}}$.
  Logistic Regression can categorize data into binary classes by learning the relationship from the training dataset. Then, it will generate the non-linearity through the Sigmoid Function. We think the Logistic Regression will work well for our predictive task A. The reasons are:
  (1) The model will not make any assumption about the distribution of the extracted features, which makes our predictions unbiased.
  (2) The model is less likely to over-fitting if the dimensions of the dataset are low.
  (3) The model will consider the coefficients as the indicators of extracted feature importance.
- Random Forest Classifier: Random Forest Classifier is designed based on ensemble learning and bagging technique. The model will combine the predictions of separate decision trees and generate the final predictions through the voting technique. Therefore, Random Forest Classifier will encounter fewer over-fitting problems and variance than other classification models. As a result, we think that the Random Forest Classifier will also work decently.

3.6 Tuning and Optimization for Task A

Table 4: Fine-Tuning Results for Task A

Model	Parameters	Accuracy
Baseline	N/A	0.60916
Logistic Regression	likelihood_threshold=0.5	0.69895
Logistic Regression	likelihood_threshold=0.4	0.73873
Random Forest Classifier	likelihood_threshold=0.5	0.73873
Random Forest Classifier	likelihood_threshold=0.4	0.73232

We use different models and parameters to tune our training/testing dataset upon our baseline model. In baseline model, we have around 0.60916 accuracy on test dataset. Then we perform logistic regression on the training dataset and test the accuracy of the model on test dataset with different likelihood threshold. We found out that with 0.5 likelihood threshold, the accuracy on test dataset is 0.69895, and with 0.4 likelihood threshold, the accuracy is 0.73873. Furthur, we also tune on Random Forest Classifier with different likelihood threshold and found that that with 0.5 likelihood threshold, the accuracy on test dataset is 0.073873, and with 0.4 likelihood threshold, the accuracy is 0.73232. Thus, we find out that Logistic Regression and Random Forest Classifier has similar accuracy performances if they both choose the optimal likelihood threshold.

4 PREDICTIVE TASK B

4.1 Introduction for Task B

People usually rate and leave a review if they have a strong opinion on a recipe. For this website specifically, people can leave a review without giving a rating score, leaving a rate of 0 in history. This will bring down the average rating of those specific recipes. Therefore, accurately predicting user ratings is crucial for data validation and preventing biased average ratings for recipes.

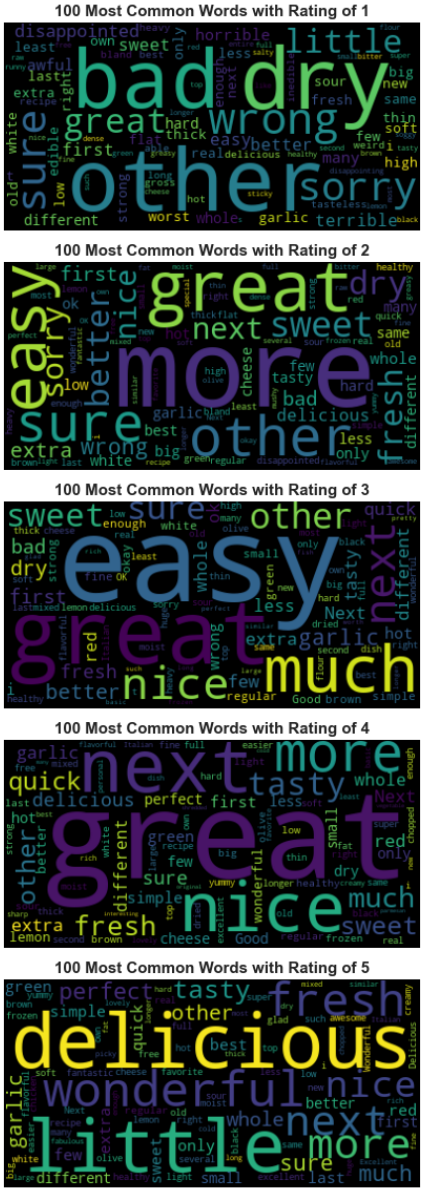

Figure 13: The WordCloud of 100 Most Common Words with Rating 1 to 5

4.2 Feature Extraction for Task B

The purpose of Task B is to predict the rating one user will give to an unseen recipe based on the previous user-recipe interactions matrix and the sentiment score of the user's review comments. Based on our exploratory data analysis, we found that the rating distribution is left-skewed. Therefore, we can utilize the patterns of user-recipe interactions (average user rating and average recipe rating) and cosine similarity to build a robust prediction model. Besides, we also observe that as in Figure 13, which demonstrate the 100 most common words in each different ratings with more frequent words be larger in graphs, the word cloud contains more negative vocabulary in the review of low rating and more positive vocabulary in the review of high rating. Based on this observation, we decided to perform sentiment analysis and use the sentiment score of the review as one variable to predict the rating as well.

4.3 Descriptions of Features and Labels for Task B

After performing data pre-processing, we built a model to predict the rating one user will give to an unseen recipe through the extracted feature. The result of our predictions (labels) will be discrete multi-class variable separated into five classes [1, 2, 3, 4, 5] (each number represent one rating). The range of the user feature will be [1,5], representing the difference between a user's average rating and the global average rating. The range of the recipe feature will also be [1,5], representing the difference between the average rating a recipe received and the global average rating. Lastly, the range of the sentiment score will be [-1,1]. The review comment will be more positive as the scoring approach 1 and more negative as the scoring approach -1.

4.4 Baseline Model for Task B

We come up with a baseline model using cosine similarity to predict ratings. More specifically, we utilize word2Vec model to help vector reviews provided by users to perform cosine similarity calculation. To better understand the performance of our baseline model, we split the dataset into train (75%) and test (25%). Then we use training data to generate word2Vec model and calculate overall average ratings and average ratings for each user. After collecting ratings for each user, we predict the ratings for each user and recipe pairs by including user's average ratings and weighted ratings according to its cosine similarity with other recipes this user have rated. Then we calculate the mean square error for train set, which is around 1.06718. For those recipes in test set that are not present in word2Vec model generated by train set, we just use 0 cosine similarity. Thus, we perform this cosine similairty prediction on test dataset, and get the MSE around 1.16827.

4.5 Improved Model Selection for Task B

Although predictive Task B is a rating prediction task, we should treat it as something other than a regression task. The reason is that the main features we used to predict the rating are the user's average and the recipe's ratings. There is no direct relationship between user/recipe features and ratings because the relationship will be different among each user/recipe pair and rating. As a result, we decide to use the SVDpp algorithm (Matrix Factorization-based

algorithms) to predict since it is a robust recommendation algorithm.

- SVDpp algorithm: $r_{ui}^{\wedge} = \mu + b_u + b_i + q_i^T (p_u + \|I_u\|^{-\frac{1}{2}} \sum_{j \in I_u} y_j)$
  The main structure of the SVDpp algorithm is the latent factor model, taking average user rating, average recipe rating, user-recipe interaction, and implicit ratings as features. The algorithm will incorporate the user/recipe bias into rating prediction during the learning process. Therefore, the algorithm will utilize stochastic gradient descent to find optimal user/recipe features (bias) and user/recipe feature vectors. Besides, the SVDpp algorithm also includes two regularization terms to avoid over-fitting, making the model more robust to the outliers. As a result, the SVDpp algorithm is the best fit to perform the rating estimation task.

- SVDpp algorithm + Linear Regression (Sentiment Score): $r_{ui} = w_1 r_{ui,SVDpp} + w_2 r_{ui,sentiment}$
  To further improve the prediction performance, we incorporate the sentiment score as an additional feature to estimate the rating. In the beginning, we used nltk's Sentiment Intensity Analyzer function to calculate the sentiment score for each review. Then, we build one linear regression model to find the optimal weights for the prediction result of the SVDpp algorithm and the prediction result of sentiment score (the idea of voting/ensemble algorithm). Ultimately, we use the optimal weights to make the rating prediction.

4.6 Tuning and Optimization for Task B

Table 5: Optimal Parameters for Improved Model

Parameters	Value
n_epochs	55
lr_all	0.004
lr_bu	0.004
lr_bi	0.002
lr_yj	0.008
reg_all	0.3
reg_bu	0.235
reg_bi	0.3
reg_yj	0.2
n_factors	3

Table 6: Fine-Tuning Results for Task B

Model	Parameters	Train MSE	Test MSE
Baseline	N/A	1.06933	1.16720
SVDpp	optimal parameters	0.61835	0.97982
SVDpp+Linear Regression	optimal parameters	0.77625	0.96496

There is a total of 462,350 ratings in our training/testing dataset. We use 346,762 ratings for training and 115,588 ratings for testing. Using the baseline model, we can obtain 1.07 MSE for training loss and 1.17 MSE for testing loss. For the SVDpp algorithm, we perform the GridSearch technique for fine-tuning. We find the most optimal parameters based on the validation results and use them to make the rating prediction. We can obtain 0.62 MSE for training loss and 0.98 MSE for testing loss. Although the SVDpp algorithm is powerful enough, we still want to incorporate an additional feature to optimize the loss function further. We choose the review's sentimental score to achieve our goal because it can directly reflect the user's attitude regarding the recipes. Since we have two models to make the prediction, we need to decide how much each prediction weighs toward the final prediction. As a result, we train a linear regression model to help us decide the weights. Finally, we can obtain 0.78 MSE for training loss and 0.96 MSE for testing loss. Although the training loss is slightly higher, we successfully reduce the testing loss by 0.02, which is a decent improvement.

5 PREDICTIVE TASK C

5.1 Introduction for Task C

For the purpose of easy search, people invented the concept of tags so that they can divide items into different categories. However, some of the contributors might forgot to label or miss-tag their recipes. For example, when a recipe contributor forgot to tag their dessert recipes as "dessert", it is likely that people will never be able to find those recipes when they will actually love those recipes if they were able to find them. Therefore, the website might need to use information within the recipes to auto-tag them as desserts if needed.

5.2 Feature Extraction for Task C

The purpose of Task C is to predict whether a recipe is about making a dessert when the given recipe does not have any associated tags. Our exploratory data analysis found that at any level of total fat, desserts usually have the highest level of saturated fat. On the other hand, we have also found that Desserts tend to have more density after 40 units of sugar. Therefore, we decide to use total fat, saturated fat, and sugar as our features.

5.3 Descriptions of Features and Labels for Task C

After performing data pre-processing, we built a model to predict whether the recipe is a dessert recipe through the extracted features. The range of our predictions (labels) $y \in \mathbb{R}$ will be binary, where $y_i \in \{0, 1\}\ \forall 0 \leq i \leq n$ (0: not dessert; 1: is dessert), and the extracted features (total fat, saturated fat, and sugar) will all be integers.

5.4 Baseline Model for Task C

We design a naive approach for the baseline model. In particular, we will predict that the recipe is dessert if the sugar level exceeds a threshold. After tuning this naive model, the best threshold is 97 and it is able to achieve around 84.5% on both the train and test set on this baseline model.

5.5 Improved Model Selection for Task C

We decide to us Random Forest Classifier as our improved model for this predictive task. Random Forest Classifier is designed based on

ensemble learning and bagging technique. The model will combine the predictions of separate decision trees and generate the final predictions through the voting technique. Therefore, Random Forest Classifier will encounter fewer over-fitting problems and variance than other classification models. As a result, we think that the Random Forest Classifier will also work decently.

5.6 Tuning and Optimization for Task C

For this task, we have mainly tuned three hyperparameters in the Random Forest Classifier model. Namely, they are max_depth (the number of trees in the forest), criterion (the function to measure the quality of a split), and n_estimators (the maximum depth of the tree). Table 7 has showcase the training and testing accuracy on the same training and testing set as the baseline model. With Gini impurity (subtracting the sum of the squared probabilities of each class from one), 17 trees and max depth of 17, we were able to achieve 95.3% accuracy on the training data and 94.4% on the testing data, which has improved the optimized baseline by around 10%.

Table 7: Fine-Tuning Results for Task C

max_depth	criterion	n_estimators	Train Accuracy	Test Accuracy
17	entropy	17	0.95237	0.94384
17	gini	17	**0.95294**	0.94438
17	entropy	10	0.95085	0.94245
10	entropy	17	0.89554	0.89514
10	entropy	10	0.89343	0.89315
10	gini	17	0.90017	0.89952

6 LITERATURE

We have downloaded the dataset from this Kaggle page. While it is a Kaggle dataset, it was scraped from food.com, which is one of the major free recipe websites available. The dataset was first collected and used in this paper [5] where it attempts to generate personalized recipes from historical user preferences. In the paper, Majumder approached their predictive task by combining Natural Language Processing's technique of data-to-text generation [2] and recommender systems' technique of personalized recommendation [6]. In particular, they explored new task of generating plausible and personalized recipes from incomplete input specifications by leveraging historical user preferences.

Similar datasets for data-to-text generation, including soccer reports and summaries of patient information in clinical contexts, were studied to automatically generate reports based on particular user-provided data [1]. Similar datasets for recommender system, such as Goodreads Datasets, were used to study the monotonic behavior chains when predicting user-book interactions and users' detailed book reviews [7]. The conclusion in the above recommender system study states "feedback signals exhibit monotonic dependency structures", which conforms our findings that when a user makes a review or rating, it implies that he or she has tried the recipe.

7 RESULTS AND CONCLUSION

In this paper, we performed exploratory data analysis on a large dataset of 23K+ recipes and 113K+ interactions (ratings and reviews) from Food.com. We first examined the relations and correlations about cooking times, number of steps, groups of categories (desserts or non-desserts), nutrition measures, and distributions of ratings under multiple situations. We also performed 3 predictive tasks:

(1) Predict if a user would interact with a specific recipe.
(2) Predict a given user's review and rating for an unseen recipe.
(3) Predict a binary label (is dessert or not) of a given recipe.

For each task, we built a baseline model and an improved model to compare the effectiveness of our predictions. All three improved models have shown significant advantages. Our first task is based on a logistic regression and a random forest classifier to reach a reasonable non-linear binary classification. The second task is based on sentiment analysis, combining with SVDpp algorithm, which is an extension of Netflix price winning SVD model that takes into account of implicit ratings [3]. Our third task is also based on a random forest classifier. The overall best result for each task is shown in Table 8.

For the first and third binary predictive task, it is clear that random forest classifier has increased the performance as it adds additional randomness to the model while growing trees, which will be robust to noise as oppose to other classification methods because it search for the best feature from a random subset instead of the entire dataset. For the second rating prediction task, we have demonstrated that sentiment from reviews can prevent usual SVDpp algorithm from overfitting of the training set as it only takes account of users and items.

Table 8: Overall Best Result for Each Task

Task	Evaluation	Best Test Result	Improvement
Recipe Recommendations	Accuracy	0.73873	+0.12957
Rating Prediction	MSE	0.96496	-0.20224
Category Prediction	Accuracy	0.94438	+0.09938

8 DISCUSSION

Our work has included optimizations and tuning to a certain extent, which gives us sufficient, convincing predictions. Yet there are still some aspects worth further investigations and improvements:

(1) The label (tag) classifier for predicting if a recipe is dessert or not is a trivial binary classifier at this point. As we have provided a feasible pipeline, we can further build a multi-category classifier upon that to better recognize and categorize unlabeled recipes.
(2) The prediction of ratings is based merely on word2Vec and cosine similarity. We can further build a model that involves and combines more different similarities like Jaccard similarity or Pearson correlation. We can also include more features in the model to have more information when predict the potential ratings of recipes.
(3) Currently, we are still determining whether the user has used the given recipe or not. If we have time, we will explore more NLP models, such as bags of words (n-grams), and use these models to create new recipes. Then, we can utilize a recommender system to assign new recipes to each user based on their previous user-recipe interactions.

ACKNOWLEDGMENTS

To Professor Julian McAuley, for the guidance and dataset to make this project possible.

REFERENCES

[1] Albert Gatt and Emiel Krahmer. 2017. Survey of the State of the Art in Natural Language Generation: Core tasks, applications and evaluation. (2017). https://doi.org/10.48550/ARXIV.1703.09902

[2] Albert Gatt and Emiel Krahmer. 2018. Survey of the State of the Art in Natural Language Generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research* 61 (Jan 2018), 65–170. https://doi.org/10.1613/jair.5477

[3] Yehuda Koren. 2008. Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Las Vegas, Nevada, USA) *(KDD '08)*. Association for Computing Machinery, New York, NY, USA, 426–434. https://doi.org/10.1145/1401890.1401944

[4] G. Linden, B. Smith, and J. York. 2003. Amazon.com recommendations: item-to-item collaborative filtering. *IEEE Internet Computing* 7, 1 (2003), 76–80. https://doi.org/10.1109/MIC.2003.1167344

[5] Bodhisattwa Prasad Majumder, Shuyang Li, Jianmo Ni, and Julian McAuley. 2019. Generating Personalized Recipes from Historical User Preferences. https://doi.org/10.48550/ARXIV.1909.00105

[6] Al Mamunur Rashid, Istvan Albert, Dan Cosley, Shyong K. Lam, Sean M. McNee, Joseph A. Konstan, and John Riedl. 2002. Getting to Know You: Learning New User Preferences in Recommender Systems. In *Proceedings of the 7th International Conference on Intelligent User Interfaces* (San Francisco, California, USA) *(IUI '02)*. Association for Computing Machinery, New York, NY, USA, 127–134. https://doi.org/10.1145/502716.502737

[7] Mengting Wan and Julian McAuley. 2018. Item Recommendation on Monotonic Behavior Chains. In *Proceedings of the 12th ACM Conference on Recommender Systems* (Vancouver, British Columbia, Canada) *(RecSys '18)*. Association for Computing Machinery, New York, NY, USA, 86–94. https://doi.org/10.1145/3240323.3240369

[8] Eliana Zeballos and Wilson Sinclair. 2021. Food Spending by U.S. Consumers Fell Almost 8 Percent in 2020. (2021). https://doi.org/amber-waves/2021/october/food-spending-by-u-s-consumers-fell-almost-8-percent-in-2020/