The Explainable Food Recommender System

Rita Bou Issa
American University of Beirut
Beirut, Lebanon
ryb07@mail.aub.edu

Omar Bazarbachi
American University of Beirut
Beirut, Lebanon
oab07@mail.aub.edu

Hassan El Khatib
American University of Beirut
Beirut, Lebanon
hme54@mail.aub.edu

Samer Farhat
American University of Beirut
Beirut, Lebanon
sbf04@mail.aub.edu

***Abstract* – Have you ever wondered what goes on inside recommender systems, and why do they recommend you the things they do? For example, were you ever shopping online and then some items pop up, with a note saying "You might like this" or "Add this item to your purchase", and you have no idea why and based on what they even suggested those items? Aside shopping, this technology has been emerging in numerous fields, namely the food industry. Based on a user's preferences, a recommender system could predict what this same user might crave and would want to eat at the give time. In this work, we introduce a recommender system that not only recommends you certain foods you might like, but also explains where they came from and the reason why they were recommended. Our final model gave great results, with an average rating of 4.1 for the recipes it recommended to users when tested on our family members. This shows that they were satisfied with the results they received.**

I. Introduction

Recommender systems have been recently widely used and have proven to be efficient in diverse situations by prompting the user to overcome the problem of information overload, assisting them with the process of decision-making and helping in changing user behavior (Ricci, Rokach & Shapira, 2011).

However, one area which has historically received comparatively little attention, especially when compared to leisure and entertainment domains, is food items recommendation. This comes as a surprise given the importance of food for human sustenance, the range of options available to select from rendering making food choices particularly challenging (Scheibehenne,Greifeneder & Todd, 2010). Recently, recommender systems are also being developed for the online food sector as more consumers buy food or search for food related content online. With 'food', we also mean food-related items, such as meal planning, recipes, ingredients, coffee shops, restaurants, restaurant menus, and grocery shopping (Min, Jiang & Jain, 2019; Trattner & Elsweiler, 2017).

The remainder of the paper is organized as follows: first, the challenges to face are reported in Section II, Section III is about the novelty our system brings to existing ones, then Section IV describes the system problem definition before Section V reports the results of the conducted experiments. Section VI concludes with final remarks.

II. Challenges

There are many reasons, however, which make food recommendation challenging, mainly in predicting what people would like to eat because this is a complex, multi-faceted, culturally determined, not to mention context-dependent. Therefore, one of the challenges our work will face would be to find and recommend accurate meals based on the user's inputs and what they like and dislike.

Another limitation faced by recommender systems is the 'cold start' problem: whenever a new user wants to make use of the platform, the system does not own any prior information about them and their preferences, so it has a hard time finding the best recommendations. A possible solution we might implement to address the problem is to have a user profile for each user where he has previously entered his preferences, which will then serve as history for each user.

III. Novelty

In a world where technology has taken the lead, and where every human-performed action is being monitored, being aware of our actions and understanding how the internet has power over us is essential. We believe that the users have the right to understand the choices they are making and the ones that are being suggested to them, instead of thinking their actions and choices are being monitored. Traditional recommender systems only offer their users the suggestions they're making, without for so much giving them reasons to why they are being given the options they are. This is not what our project is aiming for. Instead, as we want our users to understand the results that are presented to them, our target is not only to come up with an accurate model, but also one that is explainable, and here is where our project differs from existing food recommender systems.

IV. Methodology and Problem Definition

In this work, we present a food-based recommender system which, based on user inputs and ratings of different meals, dishes and desserts, recommends different foods the user might like. Depending on the number of food items the user has rated, the system could recommend up to five dishes, providing an explanation on why those specific foods were chosen, i.e. meal X was recommended to the user because he liked meal Y and meal Z.

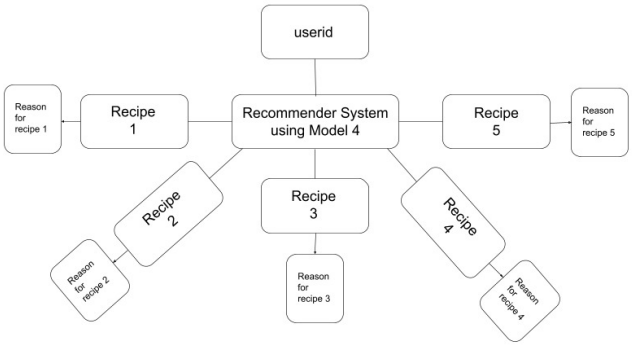
We hope to achieve our goal by using a Recommender System. First, we will need to find the datasets to be used with our algorithm, we will then need to train our algorithm to rank the proposed dishes, meals and desserts based on the users' preference and recommend different meals the user might like and would crave at that specific moment. We will describe our dataset and model in the sections below.

A. Inputs

Our system will take as input the user name, if it exists, where the user will have already rated a set of dishes according to his preferences – this terminology is further explained in the dataset section – and if it does not exist, the user will be prompted to input his new username and will be asked to rate different meals.

B. Outputs

As an output, our system will generate 5 recommendations on foods the user might like, along with an explanation of why each item got chosen. Based on how well each model performs, we will see how many foods will be generated for recommendation, and how many items it will relate each recommendation to. This is also based on the input, i.e. on how many meals the user previously rates.



C. Datasets

To further investigate our research, the first step was to collect a recipe dataset, along with a users' dataset we could use to build our recommender system. We describe our two datasets in the following section. For both datasets, extensive research was conducted by the team to find a fitting dataset. What we needed in the dataset were the names of the users, their ratings and preferences concerning different foods and cuisines. Based on that set, the recommender system will know what to recommend specifically for each user. However, the research on the web for the dataset did not produce the desired outcome. Hence, we resorted to conducting our own survey.

a. User Dataset

The user dataset simply consists of a user id, which is an id number associating each user with a number, along with the name of that user. The user ids are previously generated for all the users who completed the survey, or if the user is new, he is prompted to enter his name and an id will then be generated. If he already entered his information, he only needs to enter the name or the id number in order for him to be recommended meals.

b. Meal Rating Dataset

For our recipe ratings dataset, we conducted, a survey that took the form of a Google Docs link. We sent it out to as many users as possible, people of different occupations, age, and background There was a list of 62 dishes, and along with each dish a rating from zero to five, zero implying the user has not heard of the dish, and the number one till five gradually meaning the user hates the dish, dislikes it, is neutral towards it, likes it and loves it. The user did not need to rate all recipes. The survey generated the meals in a randomized order for each user, and asked them to rate as many meals as possible. That way, all dishes would get approximately the same number of responses.

The title accurately named the recipe and summarized its content. It must have been related to a well-known recipe and should have not include extremely specific details. This will be reiterated further in our experimental assumptions.

For our training, we were able to gather 274 participants each submitting their own responses and preferences, which will then serve to compute the average rating of a dish and the user's taste buds.

On average, each recipe was rated 255 times. Each response represents a "user" who has a unique user id that can be fetched for recommending purposes. We will then have a set with all the user ids of all users, along with ratings of different dishes that would translate their preferences for certain foods. That way, the recommender system will be able to predict which recipe can be created, based on the user's nutritional taste buds.

D. Model Description

Our plan is to train different models that have the same output: a recommendation of different meals that the user might like.

The first step will be to take the user's name – and his user id – as input. Then we will search for that user in our users' dataset. Once found we will fetch their history to be used by the recommender system. This step is common to all the models we trained. Next, we will input the user's name and history to our recommender system.

As recommender systems are traditionally based on Collaborative Filtering, which consists of making automatic predictions (filtering) about the interests of a user by collecting preferences or taste information from many users

(collaborating), we have decided to implement this algorithm on some of our models and see how well it performs. We are also planning on varying the techniques used along with collaborative filtering in each model. One of the models could be based on Matrix Factorization that works by decomposing the user-item interaction matrix into the product of two lower dimensionality rectangular matrices. We have also thought of training two of our models using the Clustering method which consists of grouping a set of objects in such a way that objects in the same group are more similar to each other than to those in other groups. We'll make use of K-Nearest Neighbor (KNN) Clustering with one model being item-based and the other user-based.

After extensive research, we also found out that we could use the computation of the correlation between all meals to come up with recommendations.

V. Experiments

A. Experimental Setup:

The experiments will be run on a system having an Intel i9-9900K Processor and an Nvidia RTX 3070 GPU. We are testing the implementation of our models using different libraries in our algorithms which are scikit-surprise, pandas, and numpy. We are running the code on Python version 3.10.

B. Experimental Assumptions:

We are assuming that meal names are standardized as in no two different names point to the same dish. (e.g., Chocolate Lava Cake and Molten Lava Cake, or rice and Risotto). We will also assume that the dataset is complete with no missing data, where a recipe not rated by the user or rated as zero (unknown by the user) was replaced by a blank, meaning its rating or rather non-rating would not affect the data. It is not accepted that a new user is added without the system knowing their history and background information, hence why they will be prompted to enter their own information and ratings. Following that logic, it is assumed that no user has no history data to them. In addition, all the ratings gathered are strictly integer values ranging from 1 to 5, as we disregarded the 0's.

C. Model Implementation

We built three of our models based on collaborative filtering to learn the users' preferences and generate a list of possible recommendations ranked according to a score that represents how likely it is for the user to enjoy it. However, each model varied slightly from the other in order to be able to generate the most accurate results.

- *Model 1*

The first model we implemented used Matrix Factorization and fine tuning. We decided to use 3 features for predictions, 500 steps for the optimization algorithm as well as a 0.001 value for alpha.

- *Models 2 and 3*

Our next two models were based on the K-Nearest Neighbor (KNN) algorithm. The second model was user-based and used cosine similarity and the third model was item-based and used Euclidian distance. We made use of the KNNWithMeans function of the scikit-surprise library. One of them was user-based and the other item-based. However, this function required that the dataset consists of three columns, those being User ID, Item ID, Rating. So, we created a function called convertDataset that takes our original dataset (in the form of a matrix where user id is the row and item id is the column) and converts it into one that this library can use.

- *Model 4*

Our fourth and final model is based on the correlation between the different meals in our dataset.

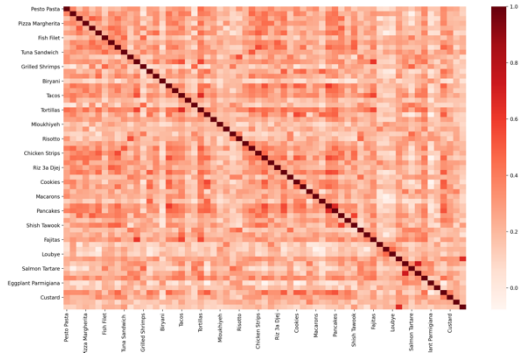


Fig. 1: Heatmap of the Correlation Matrix, with dark spots corresponding to recipes that have high correlation.

The first step of the model is to identify all recipes previously rated by the user. Next, it filters that list and only leaves the ones that the user rated as a 4 or a 5 and adds them to a list named HighlyRated (i.e., the recipes enjoyed by the user). After that, we loop over all elements of this list and compute the correlation of all other recipes with this one. By the end of this loop, we will have a matrix of the correlation of each recipe the user enjoyed with all other recipes. We then use this matrix to identify two things. First, we compute the average of the correlation of all the new recipes. This gives us a better idea of the users taste. However, we also find the maximum correlations. We found that if two recipes have a correlation of over 0.7 but don't correlate well with the other highly rated recipes, they might be ignored. So, we added an intermediate step that loops over the list of maximum correlations and if one of them is over 0.7, it automatically adds it to the top of the list while ignoring its average.

We tested out this model on ourselves and our family members as well as on several of our friends. and it seemed to show good performance. However, we were having difficulties identifying a way to properly evaluate its accuracy given that the correlations it is computing are between -1 and 1 rather than being between 1 and 5. We contemplated using a shift of scales but that didn't seem to be very accurate. So, we decided to conduct the following experiment: We had our

family members make accounts and then gave them 5 recommendations from each model. We then asked them to tell us which model they preferred and to rate each of the recipes it gave them from 1 to 5.

D. Experimental Results

The results of the survey we conducted (on 60 users) showed that Model 1 (Matrix Factorization) was the most popular with 24 votes for the best model and an average rating of 4.3. In second place was Model 4 (correlation) with 18 votes and an average of 4.1. And finally, we had Models 2 and 3 (user based, item based). In third and fourth place respectively. With 10 and 8 votes and an average rating of 3.9 and 3.8 respectively.

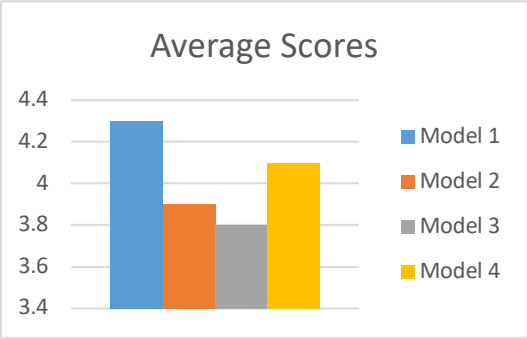


Fig. 2: Graph showing the average rating given to the recommendation of each model.

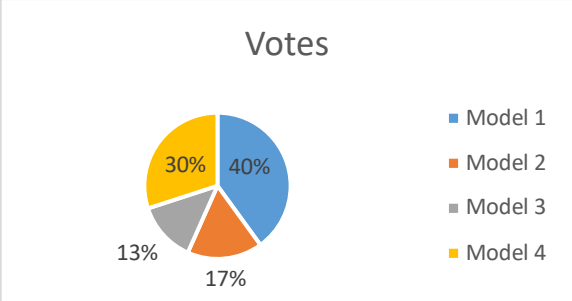


Fig. 3: Pie chart showing the distribution of votes for best overall model.

We then proceeded by measuring the accuracy of the first three models by computing the average difference between the predicted score and the real rating. We got the following results:

Model	Matrix Factorization	User Based	Item Based
Average error between prediction and real rating	0.773	1.145	1.199

As we can see our first model was the most accurate with an average error of 0.773, against values of 1.145 for the second model and 1.199 for the third. However, the technique used for Model 1 which was Matrix Factorization is famously non-explainable, and this was important for our project. That is why we proceeded with the fourth model. While it is theoretically possible to give an explanation based on our nearest neighbor's implementation, it would have been far more computationally intensive than for our correlation model, since by its very nature it finds similarities between individual recipes. In addition, Model 4 proved to be more accurate than Models 2 and 3 in our experiment. That is why we ultimately decided to go with Model 4.

Now if were to compare these results to those of [5] we would find that our scores are very similar to the ones they have. With Model 1 being very close to the best model they developed based on the Epicurious dataset, while Models 2 and 3 performed similarly to their worst preforming models.

VI. Conclusion

In this paper, we presented an efficient algorithm for an online food recommender system following a model based on the correlation between the different meals in our dataset, which was the one that showed to be the most efficient among the other three models we tested. By computing average correlation and maximum correlation, we were able to generate accurate results and reached the desired output. Our final model gave great results, with an average rating of 4.1 for the recipes it recommended to users when tested on our family members, which showed that they were satisfied with the results they received. An interesting extension to this work would be to improve the comparison to one-vs-two then three, namely one-vs-multiple instead of just comparing one-to-one.

References

[1] D. Schmidt, "Simplified-recipes-1M dataset," *simplified-recipes-1M Dataset*. [Online]. Available: https://dominikschmidt.xyz/simplified-recipes-1M/. [Accessed: 10-Nov-2021].

[2] M. Krešo, "Building a recommendation system for a culinary app," *Infinum*, 22-Jan-2020. [Online]. Available: https://infinum.com/the-capsized-eight/building-a-recommendation-system-for-a-culinary-app. [Accessed: 10-Nov-2021].

[3] P. Grover, "Various implementations of collaborative filtering," *Medium*, 31-Mar-2020. [Online]. Available: https://towardsdatascience.com/various-implementations-of-collaborative-filtering-100385c6dfe0. [Accessed: 10-Nov-2021].

[4] "Recommendation Systems Using Machine Learning - YouTube," *YouTube*. [Online]. Available: https://www.youtube.com/watch?v=EjOlN6uVBOg. [Accessed: 10-Nov-2021].

[5] J. Almeida, "Personalized food recommendations - ulisboa." [Online]. Available: https://fenix.tecnico.ulisboa.pt/downloadFile/1126295043834284/resumo.pdf. [Accessed: 11-Dec-2021].

[6] Ricci, Francesco & Rokach, Lior & Shapira, Bracha. (2010). Recommender Systems Handbook. 10.1007/978-0-387-85820-3_1.

[7] B. Scheibehenne, "Can there ever be too many options? A meta ... - scheibehenne." [Online]. Available: https://scheibehenne.com/ScheibehenneGreifenederTodd2010.pdf. [Accessed: 11-Dec-2021].

[8] M. Weiqing, "A survey on Food Computing - arxiv.org." [Online]. Available: https://arxiv.org/pdf/1808.07202.pdf. [Accessed: 11-Dec-2021].