Optimizing Recipe Recommendations for Individuals with Dietary Restrictions

Gordon Blake
Stanford University
gblake@stanford.edu

Abstract

Individuals with allergies often need to plan meals taking into account substantial dietary restrictions. Many would benefit from a system that recommends recipes that are personally appealing and diverse while avoiding allergenic ingredients. We test several models to optimize recipe recommendations by predicting user ratings from data then creating a diverse recommendation set. The collaborative filtering model estimated ratings based on the similarity of other recipes that the user had rated. Logistic regression was also used on features of the recipe such as ingredients and categories. Performance was evaluated on two datasets, one consisting of only recipes and ratings and another including detailed recipe information, using a holdout set for each. In predicting general ratings, the complex models did not substantially improve over a naive baseline of always predicting the user's average rating. However, both approaches did substantially better at detecting poorly rated recipes. Collaborative filtering performed better on the ratings only dataset while feature-reduced logistic regression outperformed it on the detailed dataset. These results suggest that machine learning techniques have promise for recipe recommendation systems but may require more ratings-dense datasets.

Introduction

Individuals suffering from allergies often need to follow complex diets to safeguard their health, including low FODMAP, dairy-free, and Gluten-free plans. These restrictions make it difficult to plan healthy, balanced, and appealing meals. Those suffering from allergies would benefit from a system that recommends recipes that are personally appealing and diverse while avoiding allergenic ingredients. This objective can be formulated as a constrained optimization problem in which the system takes a user profile and returns a diverse set of allergy safe recipes that maximizes the user's expected rating.[1]

[1]The recipe recommendation algorithm is a component of the application my team is developing for the CS 194 senior software engineering project. I have spoken with TAs from both AA 222 and CS 194 who have approved the work across classes.

The complete source code used for analysis and models is available on the project GitHub repo. Contact the author for access to data.

Past Work

Recommendation algorithms are an active field of research, including in the area of recipes. Freyne, Berkovsky, and Smith evaluated three common approaches: content-based filtering, collaborative filtering, and the logistical decision tree algorithm M5P, finding that the latter achieved the lowest error (Freyne, Berkovsky, and Smith 2011). However, these approaches rely on having a rich set of per-user ratings data. Because this algorithm will be deployed with users who recently began using an app, the personal history required for collaborative filtering may not exist.

An alternative approach is to focus on ingredients, rather than users. Teng, Lin, and Adamic adopt this approach by first constructing networks to capture the relationship between ingredients (as complements or substitutes), which can then be used to suggest functionally equivalent recipes (Teng, Lin, and Adamic 2012). This strategy may be particularly useful for the constrained problem of users with allergies who may not tolerate certain ingredients. Ueda et al. also propose an ingredient-based scoring system that takes quantity into account rather than mere presence/absence (Ueda et al. 2014). However, none of these methods were tested with the additional constraint of food allergies.

Approach

The recommendation system proceeded in two stages. First, a ratings prediction algorithm was trained such that, given a recipe and a user, that user's rating on the recipe out of five stars could be accurately predicted (a categorical variable ranging from 1 through 5). After calculating the user's predicted rating on all recipes, a set of recommended recipes that maximized predicted rating and recipe diversity was constructed using the set maximization algorithm described below. (The allergen filtering step was performed prior to making ratings predictions and is not examined in this paper.)

Data

Two datasets were used, both scraped from the website Allrecipes. The more compact Ratings Only dataset contained

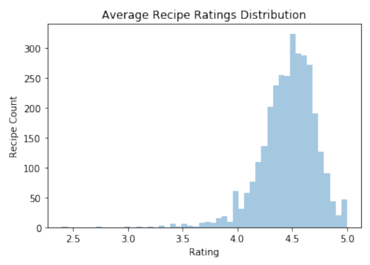

Figure 1: The distribution of recipes by average rating out of 5 stars in the Ratings Only dataset. The high mean and small standard deviation makes it difficult to derive discriminating information by comparing recipe ratings alone.

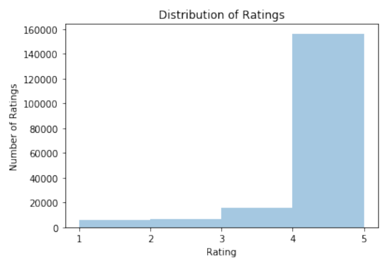

Figure 2: The frequency of ratings by category (integers 1 through 5) in the Full Recipe dataset. The Ratings Only dataset was likewise heavily skewed toward 5 stars.

only user ratings, while the Full Recipe dataset included richer details about the recipes. For each dataset, recipe ratings were held out by selecting one recipe for each user who had rated three or more recipes. 10% of these held out ratings were used for final testing. The remaining 90% of held out ratings were used to train the prediction models. Five-fold cross-validation was used on the training set to tune hyperparameter values. The final performance metrics reported here were calculated on the reserved test set.

Ratings Only The Ratings Only dataset contains recipe ids and individual user ratings scraped from the site Allrecipes. After preprocessing the data to only include recipes with at least 5 user ratings, it contains 3,201 recipes and 161,136 ratings from 87,221 users. As can be seen in Figure 1, the average recipe rating is distributed fairly normally with a high mean. However, individual user rating information is relatively sparse, as 77,208 users had only rated one or two recipes, leaving 10,013 who who rated three or more recipes.

Full Recipe The Full Recipe dataset, also scraped from Allrecipes, consists of recipe ids, individual user ratings, recipe categories, a list of ingredients and their quantities, and calories for each recipe. After equivalent preprocessing to the Ratings Only dataset, it contained of 22,515 recipes and 184,437 ratings from 120,999 users, including 9,564 users who had rated 3 or more recipes. With respect to the distributions of individual ratings and average recipe ratings, it is comparable to the Ratings Only dataset (see Figure 2). However, it had fewer ratings per recipe. The Full Recipe dataset did not have any recipes with more than 10 ratings, while Ratings Only included recipes with up to 100 ratings.

Baseline

Two simple baseline models were implemented to determine the expected performance of a naive ratings prediction system. The Recipe Mean estimator predicted a user's rating on a given recipe based on its mean rating (excluding the held-out user's rating, but including the ratings of users who had rated fewer than three recipes). The User Mean estimator used the user's mean rating on all other recipes that they had rated. Thus, for a given recipe, the Recipe Mean estimator would always return the same rating, and for a given user, the User Mean Estimator would always return the same rating.

Models

Several models were trained for this task, including collaborative filtering and logistic regression. See the Train Models Notebook for code.

One model tested was item-item collaborative filtering (Sarwar et al. 2001). For a given target recipe and user, the similarity between the target recipe and all other recipes the user had rated was computed. The similarity metric used was Pearson correlation based on the rating distribution for each recipe (thus highly dissimilar recipes could have a negative correlation). The target recipe rating was then predicted by taking a similarity-weighted average of the user's ratings on other recipes.

A second model used was logistic regression, which trained weights for recipe features using the liblinear solver (coordinate descent). For the Ratings Only dataset, these features consisted of the user's average rating, a recipe's average rating, and the user's ratings on other recipes and other user's ratings on the selected recipe (paired with the corresponding ids as feature keys). For the Full Recipe dataset, the logistic regression model incorporated a recipe's average rating, the user's average rating, the recipe's calories, categories, and ingredients. Following Udea et al., ingredient quantity was used as a feature rather than presence/absence (Ueda et al. 2014). Average ratings were always calculated by excluding relevant ratings in the holdout set so as to avoid information leakage. Hyperparameters tuned for this model include the penalty (L1), the class weight (balanced), and C, the inverse regularization strength (1.0).

Because the logistic regression estimator relied on a large feature set relative to the size of the dataset, recursive feature elimination with cross validation was used to empirically select the best features. This method left 620 features out of the original 4,075 used on the Full Recipe dataset. The performance of this RFECV Logistic estimator is reported separately from logistic regression without feature elimination.

Other models that were experimented with include stochastic gradient descent on the logistic regression feature set, a simple SGD model that tuned a weight parameter between the average rating on the target recipe and the target user's average rating, and a random forest classifier.

Model	Weighted F1 Score	Macro-Averaged F1 Score	Lower-3 Recall
Recipe Mean	0.471	0.197	0.004
User Mean	**0.595**	0.247	0.066
Collaborative Filtering	0.516	**0.270**	**0.092**
Logistic Regression	0.519	0.223	0.036
RFECV Logistic	0.523	0.227	0.083

Table 1: Performance on the Ratings Only dataset

Model	Weighted F1 Score	Macro-Averaged F1 Score	Lower-3 Recall
Recipe Mean	0.344	0.187	0.028
User Mean	**0.575**	**0.297**	0.067
Collaborative Filtering	0.447	0.229	0.046
Logistic Regression	0.500	0.252	0.110
RFECV Logistic	0.505	0.270	**0.155**

Table 2: Performance on the Full Recipe dataset

However, because these models did not perform well during initial cross-validation on the training set, they were not developed further and the results are not reported here.

Set Maximization Algorithm After all recipes were scored, a greedy algorithm was used to construct the set of recommended recipes by selecting the highest utility recipe x^*, then adjusting the scores of the remaining recipes by penalizing recipes for proximity to x^*. The subsequent recipe with the highest score was then added to the recommendation set and the scores of the remaining recipes were again adjusted. The algorithm proceeded in this manner until n recipes have been selected.

Evaluation

Three evaluation metrics were used for the ratings prediction task: Weighted F1 Score, Macro-Averaged F1 Score, and Lower-3 Recall.

The Weighted F1 score computed the precision and recall for each label, weighted by the support for that label. Thus, it accounts for label imbalance. However, as noted above, the distribution of ratings was heavily skewed toward 4 and 5 stars. Because the final recommendation set only needed to provide a small subset of highly rated recipes out of many possibilities, it was more important that the model accurately predict low-scoring recipes than high-scoring ones.

Two metrics were used to capture this goal. First, a macro-averaged F1 score was used so that, despite the small number of low ratings, performance on each rating category would receive equal weight. Second, recall was calculated for true labels of 1, 2 and 3 (the lowest scores) and macro-averaged. This Lower-3 Recall indicates how good the algorithm is at predicting rare low ratings and thus preventing them from being recommended.

Results and Discussion

In general, the more complex algorithms performed better than the Recipe Mean baseline but not consistently above the User Mean baseline, suggesting that additional features such as ingredients or collaborative filtering did not dominate simply examining the user's mean rating tendencies. Performance also varied by dataset. Collaborative filtering showed promise on the Ratings Only dataset, while on the Full Recipe dataset, the RFECV Logistic model performed nearly as well as the User Mean baseline overall while doing substantially better at detecting low ratings. Overall, results suggest that either algorithm may be a good candidate for a recommendation system that relies on filtering out poor recipes and then recommending a diverse subset from a large pool of candidates. The density of available ratings data influences which approach fares better.

The collaborative filtering algorithm performed fairly well on the Ratings Only dataset, achieving the best macro-averaged F1 score and Lower-3 recall. However, it did not do as well on the Full Recipe dataset. One reason for this could have been that individual users rated relatively few recipes. Out of 120,999 users in the Full Recipe dataset, only 9,564 users rated three or more recipes. Because each user had rated only a few recipes, predicting their rating on a new recipe through a weighted average of those ratings often led to poor results. As discussed in the Data section, the Ratings Only dataset included a denser rating structure, which may help explain the better performance of collaborative filtering on that dataset as seen in Table 1.

As can be seen in Table 2, the logistic regression models incorporating ingredient information performed better than collaborative filtering but did not improve in general over the User Mean baseline. The one exception is in the Lower-3 Recall, where logistic regression did substantially better than the baseline. The RFECV Logistic approach was the best among all tested in finding low scoring recipes while suffering only a modest overall drop in F1 scores. In comparison with logistic regression using only ratings information (see Table 1), it appears that recipe ingredient and category information was important for the Lower-3 Recall performance.

Although the performance of the set maximization algorithm was not empirically evaluated, the average pairwise overlap in ingredients for several sets of 100 recipe recommendations was found to be about 5%, indicating that recipes were fairly distinct from each other. Manual inspection of several recipe sets supported the intuition that they were fairly diverse.

Limitations

One of the most substantial limitations of the recommendation systems tested here was data sparsity. Despite the large number of users and recipes included, each user had rated only a few recipes. Predicting a user's third or fourth rating left only two or three recipes for which the user's rating was known to inform the prediction. Some experiments were conducted which increased the minimum ratings threshold for a user to be included in the holdout set to 5 or 10. However, it was found that this generally did not improve performance, as the loss in training data counterbalanced the increased number of recipes available per user. It is likely that if the ratings matrix were more dense, with 10 to 20 ratings per user, collaborative filtering would have performed better than observed here.

Future Work

Future efforts could include scraping a denser ratings dataset more focused on rating prediction, including only users who had rated a significant number of recipes. Additionally, more thorough preprocessing of ingredients could boost the signal from these features. For example, nearly identical ingredients with different names could be consolidated (such as ”chicken”, ”chicken breasts”, and ”chicken wings”), or ingredient categories such as ”Dairy” and ”Nuts” could be used instead of specific names. Following Teng et al.'s approach, a more complex ontology of ingredients could be constructed to extract the fundamental relationships captured by ingredients (Teng, Lin, and Adamic 2012). All of these approaches could boost the ratio of training data to features.

Conclusion

These results have mixed implications for machine learning techniques for recipe recommendation systems. One the one hand, they show promise in filtering out bad recipes from a user's pool of potential recommendations. On the other, additional ratings and feature information does not improve general prediction capabilities over a naive baseline. More ratings-dense datasets or refined methods of feature extraction may be required to reach this goal.

Works Cited

Freyne, Jill, Shlomo Berkovsky, and Gregory Smith. “Recipe recommendation: accuracy and reasoning”. *International conference on user modeling, adaptation, and personalization*. Springer. 2011. 99–110. Print.

Sarwar, Badrul Munir, et al. “Item-based collaborative filtering recommendation algorithms.” *Www* 1 (2001): 285–295. Print.

Teng, Chun-Yuen, Yu-Ru Lin, and Lada A Adamic. “Recipe recommendation using ingredient networks”. *Proceedings of the 4th Annual ACM Web Science Conference*. ACM. 2012. 298–307. Print.

Ueda, Mayumi, et al. “Recipe recommendation method by considering the users preference and ingredient quantity of target recipe”. *Proceedings of the International Multi-Conference of Engineers and Computer Scientists*. 2014. 12–14. Print.