Health Based Ingredient Recommender System for Recipes

Muhammad Shihab Rashid
Department of Computer Science and Engineering
University of California Riverside
mrash013@ucr.edu

Quazi Mishkatul Alam
Department of Computer Science and Engineering
University of California Riverside
qalam001@ucr.edu

Marc Giannuzzi
Department of Computer Science and Engineering
University of California Riverside
marc.giannuzzi@gmail.com

Quentin Lacroix
Department of Computer Science and Engineering
University of California Riverside
quentinlacroix0@gmail.com

Kristrian Tram
Department of Computer Science and Engineering
University of California Riverside
ktram002@ucr.edu

ABSTRACT

With obesity being a major health problem across the world, specially in the United States and an increased public interest in healthy lifestyle, the study of techniques to assist people following a healthy diet is significant. The task of predicting combinations of ingredients is also of interest. In this paper we have built an ingredient recommender system using neural networks to suggest ingredients to partial recipes. Those ingredients are then resorted according to their health value that we calculate. The neural network gives an accuracy of 77%. For categorical accuracy the model gives 47%. We further explore the future prospects in terms of replacing ingredients to recipes to achieve a better health value.

KEYWORDS

recipe, neural networks, ingredients, recommender system, health

1 INTRODUCTION

Oftentimes we have wondered whether one particular ingredient goes with existing recipes or not. This applies mostly for novel chefs. Experimenting with combinations of ingredients may even lead to newer, exciting and tastier recipes. Most of the work that has been done in the domain of food and recipes are on suggesting ingredients to recipes or improving recipes. But one thing to keep in mind is that, adding any particular ingredient to a recipe will change the nutrition measurement of that recipe. So it might happen that, suggested ingredient(s) may not be healthy for people. The works done so far does not take into consideration about the health and nutrition facts. But as stated earlier, health has now become very important.

The problem domain of the paper is significant. We want to achieve healthier audience keeping intact the taste of the recipe. Oftentimes people have experimented about adding new ingredients to recipes. In fact, most of the recipes have been generated by experimentation. Figure 1 depicts a simple blueberry cheesecake recipe. The original ingredients are blueberries, cream cheese, butter , milk cream etc. But one can easily wonder what other ingredients can go with this recipe. Here lies the solution. We propose a model where the user would give input the recipe name and our model will suggest additional ingredients that can be added to the original recipe. The suggested ingredients will also consider the health values of ingredients.

Figure 1: Blueberry cheesecake ingredients

The structure of this paper goes as follows. We will describe about the related works which describe state of the art methodologies in recommender systems. Then we will describe about the data collected and processing, our proposed methodology, evaluation and cloclude the paper with discussions.

1.1 Research 1

We propose a unique solution because we are considering the health values of the ingredients. Most of the ingredient recommender system, both in industry and research suggests ingredients that goes with the recipes. But they do not take into consideration the health values (protein, calories, fat) of those ingredients. As a result, the suggested ingredients might not be healthy, meaning the suggested ingredients may increase fat. Our model trains a neural network that will predict the new health value after the ingredient is added and re-suggests according to new health value.

2 RELATED WORK

In [3], the authors proposed a recommender system to suggest ingredients that can be added to a partial recipe based on users' ratings of a particular recipe. For that, they used item-based collaborative filtering algorithm, using a high-dimensional, sparse dataset of recipes, which inherently contains only implicit feedback. CF uses a matrix of all items and users to produce recommendations, where typically each element of the matrix represents how much a user likes an item. They implemented item-based CF by first calculating the similarities between all ingredients. Then they computed the fit of an ingredient to a recipe by summing the similarities of the target ingredients' closest neighbors that are present in the target recipe and then scaling by the sum of all similarities in the target ingredients' neighborhood. They also used PCA to reduce the number of dimensionality to speed up the computation. Their

results conclude that using the PCA reduced data consistently outperformed the non-reduced data.
The advantages of item- based CF methods can be scaled to large datasets and are well suited for sparse data. They also do not rely on item content, they are easy to implement and new data can be added incrementally. The disadvantages are that the recipes do not contain explicit feedback, the ratings are only binary. Negative rating of any recipe is not present in the dataset. The possible extension of this dataset is to include negative reviews of recipes and integrate with CF methods to generate even better ingredients.
In [6], the authors proposed a recommender system that not only offers recipe recommendations that suits the user's preference but is also able to take the user's health into account. They have designed a complete interaction process that includes preference elicitation, recommendation generation and presentation, user support for browsing and critiquing the given recommendations as well as for providing the user alternative recommendations. Based on the user profile, the long term preference and preference elicitation, the system generates a set of recipe generations and present them one by one, which later the user rates and tags. The recommendation is done by extending matrix factorization by including additional parameters to model dependencies between assigned tags and ratings. They have created a health component, which is based on a calorie balance function of the difference between the calories the user still needs and the calories of that recipe.
The advantage of this paper is that, it is one of the first papers to recommend keeping the balance between taste (preference) and health factor. Their entire platform is based on user's recommender interaction design from preference elicitation which is also novel. The disadvantage is that their health component only takes into factor how much calorie the user needs rather than computing the individual calories of the ingredients. Possible extension of this paper can be to calculate the individual health values of ingredients and take that into factor while recommending to the user.
In [4], the authors suggest numerous works on the problem of identifying user patterns and consumption preferences, and using collaborative filtering or content-based filtering to suggest ingredients in recipes. This paper tackles the unique problem of recommending missing ingredients to a recipe or creating a recipe of its own (in contrast with the user centric approach). Two deep learning methods are used for this purpose, namely, Non-negative matrix factorization and two-step regularized least squares. The first method is capable of using information in the existing recipes (ingredient combinations that often occur together) to complete a recipe. However, it cannot find recipes that are not present in the dataset. This limitation is overcome with the second method, which will not only suggest missing ingredients to a recipe, but also suggest new ingredient combinations to create a new recipe. The data used in this paper is taken from Ahn et al. (2011), which apart from having the recipes and ingredients information, also has flavor components information. The (recipe, ingredients) data is preprocessed into a m x n matrix where m is the number of recipes and n is the number of ingredients. The (ingredient, flavor component) data is preprocessed into a p x m matrix where p is the flavor components. After, that the two methods are applied using these two matrices. This paper manifests that deep learning algorithms can be used to complete recipes. The first method, namely, Non-negative matrix factorization is particularly perfect for this type of data. This is due to the fact that the data is a linear combination of non-negative (0/1) values which depicts whether or not an ingredient is present in the recipe. However, this method has two disadvantages. Firstly, it cannot take into account the flavor components information which can be quite important in food related predictions. Secondly, it cannot suggest a new recipe which is not present in the dataset. The second method, namely, two step regularized matrix factorization takes care of these limitations.
This paper [1] emphasizes on the fact that certain factors affect the ingredients that can be used to together in a recipe, such as, color, temperature, texture, sound, etc. However, palatability largely depends on the flavor components. Therefore, it can be said that the flavor components are an incisive factor when analyzing our acceptable combination of ingredients in a recipe. A hypothesis that is the center point of this paper is, "ingredients sharing flavor compounds are more likely to taste well together than ingredients that do not". To realize this the researchers adopt a network based approach to analyze the effect of flavor components on ingredient combination. Keeping a set of ingredients in one column and a set of flavor component in another column a bipartite network is formed. A projection of this bipartite network is the so-called flavor network in which the ingredients are the nodes and two nodes have an edge if the share at least one flavor component. In fact, the weight of the edges is considered to be the number of shared flavor component. In this paper these facts are brought to light that the number of ingredient and the rank-frequency behavior of ingredients in recipes among different cuisines are invariant. Also, it also discusses about how some ingredients are prevalent in a recipe, while the others are not so much.
Pros: This paper brings to light a number of significant patterns that characterizes our preference of combining ingredients to form a recipe Cons: Doesn't shed any light on any methods that we could use, rather statistical analysis is presented.
The project represented in [2] has for purpose to recommend recipes for users. The datasets that have been used are Epicurious and Food.com. Many data mining techniques are utilized such as SVM (Support Vector Machines) and TD-IDF weight. What is interesting in their approach is the adaptation of the Rocchio's algorithm for food recommendation. This algorithm enables to have some feedback on the documents used and on how to give documents that satisfy most the user.
Thus, the Rocchio's algorithm and the term frequency can be used to recommend not documents but recipes that fit with the user desired ingredients which are used as the terms or words. In fact, a recipe whose weight would not be recommended whereas a positive one would be. Also, the more the absolute value of the weight is near 1, the more the result is accurate and can be used. According to the writer of the paper, Jorge Almeida, that way of recommending brought good results. However, he wanted to know how to make results even better. What he did is to add two methods which are Min-Max Normalization in order to translate the similarity that just had been computed into ratings and another one that uses deviation. To be even more precise, a cross validation has been done with the several methods used.
In [9], Four students worked on how to recommend food based on rates of recipes, the user age, gender and other characteristics in

the purpose to answer the question: "What should I prefer to eat" and then recommend a restaurant to the client. They used neural networks to implement their recommendation system. Neural networks are based on two different steps: feed forward algorithm and back-propagation algorithm that are working on several layers that include neurons. Connections are made between each neuron of two layers that are closed with weights. The main purpose is to calculate errors with different entries to obtain a model that can fit with the one that the students have chosen. Each time an error is computed for a specific entry, the weights that are described before are updated in order to achieve the model that is wanted. Thus, the group of students developed an Android application which purpose is to recommend what food to eat to the clients. Another application has been made especially for the owner of the hotel who can feed the database with it's own recipes and ingredients. Also, an admin application has been made in order to add or retrieve some restaurants from the client application.

The paper [5] states: (1) an analysis reporting on the applicability of various personalized techniques for rating prediction, and (2) a report on the observed trends of reasoning uncovered by machine learning feature selection algorithm. Related work: Asked recipe ratings, which are transferred to ingredient ratings, is an accurate and effective method of capturing ingredient preferences -> accuracy in recommendations. Extracting important features from the recipe text -> weighted similarity measure between recipes. 3 personalized recommender algorithms: 2 recommender strategies: collaborative filtering algorithm assigns predictions for user for a target recipe, based on the weighted ratings of a set of neighbors and content-based algorithm : breaks down each rated (1 user rate) recipe into rated ingredients, making the target recipe.

There is also a machine learning strategy for rating prediction: logistical binary decision tree to predict scores based on the recipe content and metadata + splits after testing all features and computing each expected reduction in error + each leaf predicts a numeric quantity using linear regression. Algorithm accuracy and performance is assessed with recipe ratings from Mechanical Turk (amazon) (101,557 ratings of 917 users). Mean Absolute Error smaller in ML algorithm.

In [7] , apart from content-based (features of the food/item to generate user profiles and then check for similarities), collaborative filtering (other people having similar tastes), rule-based approach (rules derived from previous transactions and associations), and hybrid (multiple techniques using time, similar taste, associations rules), the authors focus on a different type of recommendation system: Sequential Pattern Mining. The purchase history of users is analyzed to find their sequential patterns (SPADE algorithm i.e Sequential PAttern Discovery using Equivalence classes) to predict the next possible food purchase. Suggest food based on previous transactions can help customer gain time. We can do better Ads and better management of food storing. Transaction history of a user is a series of transactions ordered by the transaction time. 3 Steps to do with this transaction history: collect user informations and preferences, then find sequential patterns from previous transaction stored in the database. After that one of these patterns will be shown as a special offer with discount: if the customer purchase it, the preference is stored; if not ask for feedback. Results of the experiments is well known now but: when minimum support is increased, number of patterns generated decreased. We can predict food-next purchase with sequential patterns.

Pros: It can be useful in the same location. Indeed, at the supermarket near people's home, they should buy pretty much the "same" daily things, so we can trust frequent itemsets. Cons: Useful feedback is difficult to obtain. Sequential patterns may not always be interesting because of a lack of correlation between items, just bought together. We have to extend bought ingredients patterns to recipes and it depends on the person.

The authors in [10] focus on the idea of trying to detect the ethnic cuisine type of food based on the similar ingredients. The authors found that using an SVD worked well to reduce the dimensionality of each recipe, finding the sweet spot of 400 kinds of ingredients. They used each ingredient as a separate feature, but struggled to classify recipes with many simple and common ingredients. They then looked to use an SVM method which proved to do very well in classifying the various cuisine types. Combining both methods they could increase the precision of classifying the recipes by cuisine. The authors focused on the precision and recall curves to evaluate the accuracy of their classifier. Using 5-Fold cross validation the authors found their classifier to have an 80%+ precision score for all of the cuisines they classified. The authors felt that with the increasing number rid recipe sharing online, a useful feature would to be able to automatically group recipes that are similar cuisines to better recommend recipes to others. They believe this to be very beneficial in websites that focus highly on recommended similar recipes.

The authors in [8] spent much of their efforts gathering and cleaning data to get a representative dataset that's was clearly classified. The authors used data from recipesource.com and various other American recipe websites. The authors had to reduce their dataset of over 70,000 recipes to a mere 5,900 recipes as the source they used allows anonymous submissions without strict guidelines on classify recipes. With their limited data the authors gathered only recipes from countries that had at least 50 recipes. One major issue that the authors faced was skewed data. As certain Ingredients such as Soy sauce is very prominent in Asian countries with as much as a 30% more prevalence then European countries. In contrast many Europeans countries shared many of the same ingredients.

The authors used Hierarchical clustering based on ingredient similarity. The authors chose the Euclidean distance based on the prevalence of ingredients. Using the similarities of ingredients and super catagories of the countries, the authors created ingredient networks to identify groups of ingredients to predict a country of origin. The authors train their classifier by splitting the data into 5 parts and training on 4 parts with a single test set. They found their classifier to be accurate in the 80 and 85 percentile in Asian and European ingredients.

3 PROPOSED METHODOLOGY

The proposed methodology section is structured into data collection and processing, training a neural network model and then getting the results.

3.1 Data Collection

The data was collected from Kaggle. It is named "Epicurous - Recipes With Nutrition'. The recipe and ingredientsw were fetched from epicurious website where you give one recipe as input and it lists out all the ingredients. The author of the kaggle dataset cleaned the website and crawled it to get the ingredients.
It has 27k Recipes in total with 680 columns. The columns include of two things - tags and ingredients. Tags are like "diet" , "dessert", "vegetarian" etc, which describe different traits of the recipes. Apart from the tags the rest of the columns are ingredients. If any particular recipe contains an ingredient, it is labeled as 1. Otherwise 0. Without the tags there are in total around 480 ingredients.
The dataset also contains the health values of the recipes. Calories, protein, fat are the health values. A sample dataset is depicted in figure 2.

	title	rating	calories	protein	fat	sodium	#cakewee	#wasteles	22-minute	3-ingredie	30 days of advance	p	alabama	alaska
2	Lentil, Apple, and Turkey W	2.5	426	30	7	559	0	0	0	0	0	0	0	0
3	Boudin Blanc Terrine with R	4.375	403	18	23	1439	0	0	0	0	0	0	0	0
4	Potato and Fennel Soup Hos	3.75	165	6	7	165	0	0	0	0	0	0	0	0
5	Mahi-Mahi in Tomato Olive	5					0	0	0	0	0	0	0	0
6	Spinach Noodle Casserole	3.125	547	20	32	452	0	0	0	0	0	0	0	0
7	The Best Blts	4.375	948	19	79	1042	0	0	0	0	0	0	0	0
8	Ham and Spring Vegetable S	4.375					0	0	0	0	0	0	0	0
9	Spicy-Sweet Kumquats	3.75					0	0	0	0	0	0	0	0
10	Korean Marinated Beef	4.375	170	7	10	1272	0	0	0	0	0	0	0	0
11	Ham Persillade with Mustar	3.75	602	23	41	1696	0	0	0	0	0	0	0	0
12	Yams Braised with Cream, R	3.75	256	4	5	30	0	0	0	0	0	0	0	0
13	Spicy Noodle Soup	4.375					0	0	0	0	0	0	0	0
14	Banana-Chocolate Chip Cak	4.375	766	12	48	439	0	0	0	0	0	0	0	0
15	Beef Tenderloin with Garlic	4.375	174	11	12	176	0	0	0	0	0	0	0	0

Figure 2: Original Dataset

3.2 Data Processing

From figure 2 we can see that, in the dataset there are lots of missing values and 0 values. We removed columns that are not ingredients and also the rows with the missing values. The 'rating' column is the user rating that they gave to a particular recipe. In this recipe we did not take users rating into perspective. But for future work this can be taken into consideration. After the dataset has been cleaned and processed, it looks like Figure 3.

	title	rating	calories	protein	fat	sodium	almond	anchovy	apple	apricot	artichoke	arugula	asian pear	asparagus	avocado
2	Lentil, Apple, and Tur	2.5	426	30	7	559	0	0	1	0	0	0	0	0	0
3	Boudin Blanc Terrine	4.375	403	18	23	1439	0	0	0	0	0	0	0	0	0
4	Potato and Fennel So	3.75	165	6	7	165	0	0	0	0	0	0	0	0	0
5	Spinach Noodle Casse	3.125	547	20	32	452	0	0	0	0	0	0	0	0	0
6	The Best Blts	4.375	948	19	79	1042	0	0	0	0	0	0	0	0	0
7	Korean Marinated Be	4.375	170	7	10	1272	0	0	0	0	0	0	0	0	0
8	Ham Persillade with I	3.75	602	23	41	1696	0	0	0	0	0	0	0	0	0
9	Yams Braised with Cr	3.75	256	4	5	30	0	0	0	0	0	0	0	0	0
10	Banana-Chocolate Ch	4.375	766	12	48	439	0	0	0	0	0	0	0	0	0
11	Beef Tenderloin with	4.375	174	11	12	176	0	0	0	0	0	0	0	0	0
12	Peach Mustard	3.125	134	4	3	1394	0	0	0	0	0	0	0	0	0
13	Raw Cream of Spinac	4.375	382	5	31	977	0	0	0	0	0	0	0	0	0

Figure 3: Processed Dataset

The red boxed marked ingredients depict list of ingredients.

3.3 Neural Networks

We have implemented a neural network model that lists ingredients as output. The neural network has 300 input and 300 output nodes, 4 layers of hidden nodes each having 308 nodes. The hyperparameters are listed as below:

- **Loss Function:** Categorical Cross-Entropy
- **Optimizer:** adam
- **Activation Function:** sigmoid
- 18 epochs

We have divided the original dataset into 80% training set and 20% cross-validation set. The training phase architecture is depicted in Figure 4.

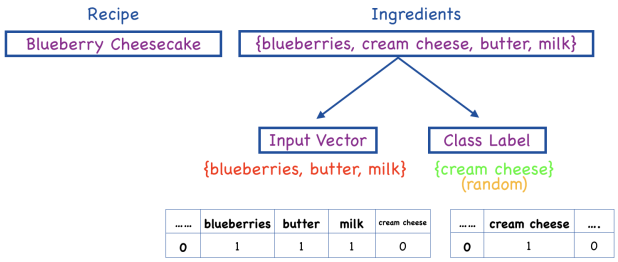

Figure 4: Training Phase

The dataset that we have does not contain any class label. So to perform neural networks, we have to create one class label for the training data. The training phase begins with splitting the list of ingredients into two vectors. As shown in Figure 4, as an example, the blueberry cheesecake recipe has 4 items as ingredients namely 'blueberries', 'cream cheese', 'butter' and 'milk'. Randomly this list of ingredients is split into two vectors. For this example, 'cream cheese' is split into class label. So now we have two vectors named 'Input Vector' and 'Class Label' for one particular row in the dataset. In the input vector cream cheese value will be 0 and the rest of the ingredients' value, that are presented in that recipe will be 1. In the class label only cream cheese will be 1 rest will be 0.
The purpose of creating this class label is to make the neural network understand the correlation between ingredients. When the model will see 'blueberries', 'butter' and 'milk' together, it will know that the corresponding class label is 'cream cheese'. That means it will suggest 'cream cheese' then. Here we create 5 random instances for each recipes, meaning each instance a different random ingredient will be taken out as the class label.
The neural network model is depicted in Figure 5.

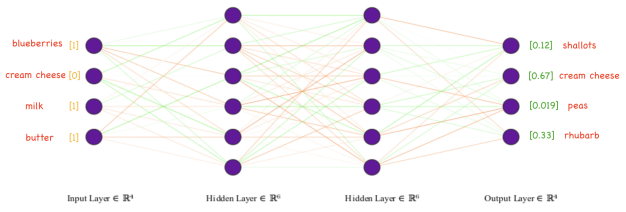

Figure 5: Neural Network Model for Ingredient Recommendation

One particular input node contains one particular ingredient. The other ingredients values will be set to zero. Then ingredient goes through the hidden layers and the model gives output as a score function. The score depicts how much correspondance is between the ingredients. For example, we can see in Figure 5 that for input of blueberries, milk and butter the model lists all possible

ingredients. But the score is not the same for all. Higher score means higher confidence that that ingredient goes with the input ingredients. 'shallots' does not go with blueberries butter or milk. That is why the score is low like 0.12. On the other hand, it gives higher score value for cream cheese and rhubard, because those ingredients can be added with the original recipe. The model shown in the figure does not show full, but a small portion of the big model. With 18 epochs, our loss function was able to reach minima.

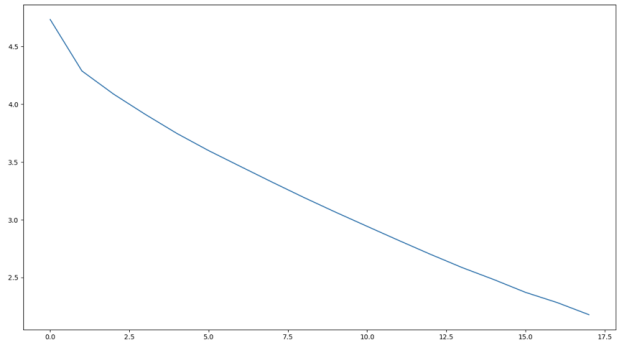

Figure 6: Loss Function Reaching Minima

We fetched out the top 5 ingredients (sorted according to score value) that can be added to partial recipe. For our blueberry cheesecake input, we see the top 5 ingredients in Figure 6.

Ingredient	Score
cream cheese	0.76
chocolate	0.64
mint	0.55
oat	0.54

Figure 7: Output of Neural Network Model

From intuition we can tell that chocolate, mint and oat all go with blueberry cheesecake. Cream cheese was originally in the recipe.

3.4 How Does Health Come In

The novel thing that our paper proposes is the integration of health values. Health values mean the amount of calories, protein and fat present in a recipe. From standard health data we can tell that a particular food should have less amount of fat and more amount of protein. So the balance between protein and calorie should be standard as well. To integrate this, we created a new column in our dataset called 'health value' which is basically a function of protein , calories and fat. The equation looks like below:

$$healthVal = f(\frac{protein}{calorieXfat}) \quad (1)$$

So to get an updated health value after adding an ingredient to a particular recipe, we need to predict it. We train another simple neural network that will predict the newer health value after adding an ingredient to it. We only predict for the top 5 ingredients that we already got from our previous neural net. After getting the new health value, we re-sort the previous results to get our final suggestions. The final suggestions are depicted in Figure 8.

Ingredient	healthVal
mint	0.87
oat	0.85
cream cheese	0.48
chocolate	0.33

Figure 8: Final Re-Sorted Order of Ingredients

According to Figure 8, we see that the order has been changed. Previously our model was suggesting cream cheese as the top recommended ingredient. But now, because the health value of mint is higher (meaning mint has less fat in it than cream cheese or chocolate), mint is now the top recommended ingredient.
Just because mint has a higher health value does not mean it does not go well with the original recipe. Rather, all the top 5 ingredient goes with the original recipe. Our model finds a balance between health value and taste. This is something novel that our model produces.

3.5 Apriori Algorithm

We also implemented Apriori which is an association rules algorithm. Apriori is often used in datasets that contain for every point of it a set of items. For each point, items could be present or not. The goal is to create different rules such as X -> Y which mean that if a point has the items of X then we should have the items of Y. Given our recipes dataset, we want to know which sets of aliments or tags are often used together in the several recipes and thus fit well together. For instance, X could be vegetarian and Y pescatarian, kosher, peanut free which is represented by this sentence: "If a recipe is vegetarian, we should have pescatarian, kosher and peanut free". We use two hyperparameters that are smin and cmin. The parameters smin and cmin represent the minimum support and the minimum confidence that a rule needs to have to be in the results. Support represents how frequently the items appear in the data, and confidence indicates the number of times the if-then statements are found true divided by the number of times we can find X and other items in the data.
Apriori is finding the correlation between our ingredients and tags. It produces some interesting results that we can see in Figure 9.

Support	Confidence	Rule
0.24	0.7273	{'kosher', 'soy free'} -> {'peanut free', 'vegetarian'}
0.25	0.7353	{'vegetarian'} -> {'pescatarian', 'fish'}
0.26	0.7567	{'lentil', 'salad'} -> {'carrots'}
0.26	0.7571	{'blueberry', 'cream cheese'} -> {'dessert', 'cake'}

Figure 9: Result of Apriori Algorithm

Apriori relies on two important properties that are:

- Anti-monotonicity property of frequent patterns: any subset of a frequent itemset must be frequent.
- Pruning principle: If there is any itemset which is infrequent, its superset should not be generated.

These two properties help us to reduce significantly the number of operations to prepare the different itemsets that we use to create the rules. However, the Epicurious dataset is very large. Thus, if s_{min} and c_{min} are very low, then the number of rules that are created is very large and the computations can take time. This is why we used the values for the parameters: s_{min} = 0.2 and c_{min} = 0.1.
From Figure 9 we can see that, 'kosher', 'soy free' is correlated with 'vegetarian' or 'pescatarian'. That means if any dish is kosher or soy free, most probably it is also vegetarian or pescatarian. The apriori algoritm is shown in Algorithm 1.

Algorithm 1: Apriori Algorithm

Result: List of correlated ingredients
i = 1;
Scan the data to get all of the possible items (size 1);
Put in an itemset called L_1;
while L_i *not empty* **do**
  Create C_i from L_{i-1} ;
  C_i an itemset that contain all of the possible itemsets than we can create from L_{i-1}. Each of the itemsets in C_i should have i as size. ;
  Create L_i which represent the itemset containing all of the itemsets from C_i that respects the conditions ($s > s_{min}$ and c > c_{min}) ;
  i = i + 1;
end

3.6 Results

The accuracy of our model is 77%. For categorical accuracy, where we compare only the top ingredient with the class label, the accuracy is 47%.

4 EXPERIMENTAL EVALUATION

For evaluation, we used leave one out cross-validation scheme. We created a validation data from the original dataset. Around 20% is used as the validation dataset. From that dataset we left one ingredient at random from the original recipe and them predict the result. The output is matched with the correspondent class label. We first considered the output to be in the top 5 of our recommendation. Following this approach, we achieved 77% accuracy.
We also did categorical accuracy, where only the top ourput is considered. We achieved 47% accuracy with this approach.
In addition to the quantitative metrics described above, we also sampled a few recipes and judge the corresponding recommendation qualitatively, simulating real-world use of our system.
Our model does better than [3] where they achieved 57.5% recall with Collaborative Filtering methods. But they did not take into consideration about the health values.

5 DISCUSSIONS & CONCLUSIONS

The accuracy that we achieved is fair because of the limitation of our dataset. Our data only consists of 27k recipe records which is not very big. If we have big amount of data, the accuracy can be higher.
Moreover, 'taste' is subjective. So it is not fair to judge the quality of recommendation based on quantitative values.
Also, in terms of health values, they can also be subjective. For some people, food with more fat is preferred and for some food with less fat is preferred. It really depends on the health type of one particular person. So we cannot always say that food with less fat is 'better'. For this paper, we considered the more in general case where food with less fat and more protein is 'healthier'.
For future work, we plan to replace the ingredients to achieve a better health value for our recipe rather than suggest a new ingredient. Sometimes, in recipes, the original ingredients may not be healthy. There may be alternative ingredients that can be used which are healthier. We plan to incorporate that in the future.

REFERENCES

[1] Yong-Yeol Ahn, Sebastian E Ahnert, James P Bagrow, and Albert-László Barabási. 2011. Flavor network and the principles of food pairing. *Scientific reports* 1 (2011), 196.
[2] Jorge Almeida. [n. d.]. Personalized Food Recommendations. ([n. d.]).
[3] Paula Fermín Cueto, Meeke Roet, and Agnieszka Słowik. 2019. Completing partial recipes using item-based collaborative filtering to recommend ingredients. *arXiv preprint arXiv:1907.12380* (2019).
[4] Marlies De Clercq, Michiel Stock, Bernard De Baets, and Willem Waegeman. 2016. Data-driven recipe completion using machine learning methods. *Trends in Food Science & Technology* 49 (2016), 1–13.
[5] Jill Freyne, Shlomo Berkovsky, and Gregory Smith. 2011. Recipe recommendation: accuracy and reasoning. In *International conference on user modeling, adaptation, and personalization*. Springer, 99–110.
[6] Mouzhi Ge, Francesco Ricci, and David Massimo. 2015. Health-aware food recommender system. In *Proceedings of the 9th ACM Conference on Recommender Systems*. ACM, 333–334.
[7] Sonali Khandagale, Sneha Mallade, Krunali Kharat, and Vishakha Bansode. 2016. Food recommendation system using sequential pattern mining. *Imperial J. Interdiscip. Res* 2, 6 (2016), 912–915.
[8] Kyung-Joong Kim and Chang-Ho Chung. 2016. Tell me what you eat, and i will tell you where you come from: A data science approach for global recipe data on the web. *IEEE Access* 4 (2016), 8199–8211.
[9] Ms Pranoti D Patil, Ms Neha P Patil, Ms Pallavi Y Geete, Ms Anuja V Nikat, and Mr Parag S Kulkarni. [n. d.]. Food Recommendation System. ([n. d.]).
[10] Han Su, Ting-Wei Lin, Cheng-Te Li, Man-Kwan Shan, and Janet Chang. 2014. Automatic recipe cuisine classification by ingredients. In *Proceedings of the 2014 ACM international joint conference on pervasive and ubiquitous computing: adjunct publication*. ACM, 565–570.