# Personalized Food Recommendations
## Exploring Content-Based Methods

Jorge Almeida, jorge.almeida@ist.utl.pt

Technical University of Lisbon - IST - Taguspark Campus,
Av. Prof. Doutor Aníbal Cavaco Silva 2744-016 Porto Salvo, Portugal

**Abstract.** Food recommendation is a relatively new area, with few systems that focus on analysing user preferences being deployed in real settings. In my M.Sc. dissertation, the applicability of content-based methods in personalized food recommendation is explored. Variations of popular approaches used in other areas, such as Rocchio's algorithm for document classification, can be adapted to provide personalized food recommendations. With the objective of exploring content-based methods in this area, a system platform was developed to evaluate a variation of the Rocchio algorithm adapted to this domain.

**Keywords:**
Recommendation Systems, Content-Based Recommendation, Food Recomendation, Machine Learning, Feature Testing

## 1 Introduction

Information filtering systems [4] seek to expose users to the information items that are relevant to them. Typically, some type of user model is employed to filter the data. Based on developments in Information Filtering (IF), the more modern recommendation systems [5] share the same purpose, but instead of presenting all the relevant information to the user, only the items that better fit the user's preferences are chosen. The process of filtering high amounts of data in a (semi)automated way, according to user preferences, can provide users with a vastly richer experience.

Recommendation systems are already very popular in e-commerce websites and on online services related to movies, music, books, social bookmarking and product sales in general. Still, food recommendation is a relatively new area, with few systems deployed in real settings that focus on user preferences. The study of current methods for supporting the development of recommendation systems, and how they can apply to food recommendation, is a topic of great interest.

In this work, the applicability of content-based methods in personalized food recommendation is explored. To do so, a recommendation system and an evaluation benchmark were developed. The study of new variations of content-based methods, adapted to food recommendation, is validated with the use of performance metrics that capture the accuracy level of the predicted ratings. In order to validate the results, the experimental component is directly compared with a set of baseline methods, amongst them the YoLP content-based and collaborative components.

The experiments performed in this work seek new variations of content-based methods using the well-known Rocchio algorithm. The idea of considering ingredients

in a recipe as similar to words in a document lead to the variation of TF-IDF developed in [11]. This work presented good results in retrieving the user's favorite ingredients, which raised the following question: could these results be further improved?

Besides the validation of the content-based algorithm explored in this work other tests were also performed. The algorithm's learning curve and the impact of the standard deviation in the recommendation error were also analysed. Furthermore, a feature test was performed to discover the feature combination that better characterizes the recipes, providing the best recommendations.

The study of this problem was supported by a scholarship at INOV, in a project related to the development of a recommendation system in the food domain. The project is entitled Your Lunch Pal[1] (YoLP), and it proposes to create a mobile application that allows the customer of a restaurant to explore the available items in the restaurant's menu, as well as to receive, based on his consumer behaviour, recommendations specifically adjusted to his personal taste.

The rest of this document is organized as follows: Section 2 provides a few fundamental concepts on recommendation systems. In Section 3, two previously proposed recommendation approaches are analysed, where interesting features in the context of personalized food recommendation are highlighted. In Section 4, the modules that compose the architecture of the developed system are described. Section 5 contains the details and results of the experiments performed in this work, and describes the evaluation metrics used to validate the algorithms implemented in the recommendation components. Lastly, in Section 6 a brief overview of the main aspects of this work is given.

## 2 Fundamental Concepts

### 2.1 Recommendation Systems

Based on how recommendations are made, recommendation systems are usually classified into the following categories [5]: Knowledge-based recommendation systems; Content-based recommendation systems; Collaborative recommendation systems; and Hybrid recommendation systems.

Collaborative filtering is currently the most popular approach for developing recommendation systems[6]. Collaborative methods focus more on *rating-based* recommendations. Content-based approaches, instead, relate more to classical Information Retrieval based methods and focus on keywords as content descriptors, to generate recommendations. Because of this, content-based methods are very popular when recommending documents, news articles or web pages, for example. Knowledge-based systems suggest products based on inferences about user's needs and preferences.

Collaborative methods, or collaborative filtering systems, try to predict the utility of items for a particular user, based on the items previously rated by other users. This approach is also known as the wisdom of the crowd and assumes that users who had similar tastes in the past, will have similar tastes in the future. In order to better understand the users' tastes, or preferences, the system has to be given item ratings either implicitly or explicitly.

[1] `http://www.yolp.eu/en-us/`

Content-based and collaborative methods have many positive characteristics but also several limitations. The idea behind hybrid systems [1] is to combine two or more different elements in order to avoid some shortcomings and even reach desirable properties not present in individual approaches.

### 2.1.1 Content-Based Methods

Content-based recommendation methods basically consist in matching up the attributes of an object with a user profile, finally recommending the objects with the highest match.

With the Vector Space Model (VSM), documents can be represented as vectors of weights associated with specific terms or keywords. Each keyword or term is considered to be an attribute and their weights relate to the relevance associated between them and the document. There are various term weighting schemes, but the Term Frequency-Inverse Document Frequency measure, TF-IDF, is perhaps the most commonly used amongst them [10]. As the name implies, TF-IDF is composed by two terms. The first, Term Frequency (TF) is defined as follows:

$$TF_{i,j} = \frac{f_{i,j}}{max_z f_{z,j}} \tag{1}$$

Where, for a document $j$ and a keyword $i$, $f_{i,j}$ corresponds to the number of times that $i$ appears in $j$. This value is divided by the maximum $f_{z,j}$ which corresponds to the maximum frequency observed from all keywords $z$ in the document $j$.

Keywords that are present in various documents do not help in distinguishing different relevance levels, so the Inverse Document Frequency measure (IDF) is also used. With this measure, rare keywords are more relevant than frequent keywords. IDF is defined as follows:

$$IDF_i = \log\left(\frac{N}{n_i}\right) \tag{2}$$

In the formula, $N$ is the total number of documents and $n_i$ represents the number of documents in which the keyword $i$ occurs. Combining the TF and IDF measures, we can define the TF-IDF weight of a keyword $i$ in a document $j$, as:

$$w_{i,j} = TF_{i,j} \times IDF_i \tag{3}$$

**Rocchio's Algorithm**

One popular extension of the vector space model for information retrieval relates to the usage of relevance feedback. Rocchio's algorithm is a widely used relevance feedback method that operates in the vector space model [9]. It allows users to rate documents returned by a retrieval system according to their information needs, later averaging this information to improve the retrieval. Rocchio's method can also be used as a classifier for content-based filtering. Documents are represented as vectors, where each component corresponds to a term, usually a word. The weight attributed to each word can be computed using the TF-IDF scheme. Using relevance feedback, document vectors of positive and negative examples are combined into a prototype vector for each class $c$. These prototype vectors represent the learning process in this

algorithm. New documents are then classified according to the similarity between the prototype vector of each class and the corresponding document vector, using for example the well-known cosine similarity metric. The document is then assigned to the class whose document vector has the highest similarity value.

More specifically, Rocchio's method computes a prototype vector $\overrightarrow{c_i} = (w_{1i}, \ldots, w_{|T|i})$ for each class $c_i$, being $T$ the vocabulary composed by the set of distinct terms in the training set. The weight for each term is given by the following formula:

$$w_{ki} = \beta \sum_{d_j \in POS_i} \frac{w_{kj}}{|POS_i|} - \gamma \sum_{d_j \in NEG_i} \frac{w_{kj}}{|NEG_i|} \tag{4}$$

In the formula, $POS_i$ and $NEG_i$ represent the positive and negative examples in the training set for class $c_j$, and $w_{kj}$ is the TF-IDF weight for term $k$ in document $d_j$. Parameters $\beta$ and $\gamma$ control the influence of the positive and negative examples. The document $d_j$ is assigned to the class $c_i$ with the highest similarity value between the prototype vector $\overrightarrow{c_i}$ and the document vector $\overrightarrow{d_j}$.

## 3 Related Work

### 3.1 Food Preference Extraction for Personalized Cooking Recipe Recommendation

Based on user's preferences extracted from recipe browsing (i.e., from recipes searched) and cooking history (i.e., recipes actually cooked), the system described in [11] recommends recipes that score highly regarding the user's favourite and disliked ingredients. To estimate the user's favourite ingredients $I_k^+$ an equation based on the idea of TF-IDF is used:

$$I_k^+ = FF_k \times IRF \tag{5}$$

$FF_k$ is the frequency of use ($F_k$) of ingredient $k$ during a period $D$:

$$FF_k = \frac{F_k}{D} \tag{6}$$

The notion of IDF (inverse document frequency) is specified in Eq.(7) through the Inverse Recipe Frequency $IRF_k$, which uses the total number of recipes $M$ and the number of recipes that contain ingredient $k$ ($M_k$).

$$IRF_k = \log \frac{M}{M_k} \tag{7}$$

The approach inspired on TF-IDF, shown in equation Eq.(5) and that was used to estimate the user's favourite ingredients is an interesting point to further explore in restaurant food recommendation.

### 3.2 Recommending Food: Reasoning on Recipes and Ingredients

A previous article has studied the applicability of recommender techniques in the food and diet domain [3]. The performance of collaborative filtering, content-based and hybrid recommender algorithms is evaluated based on a dataset of 43,000 ratings

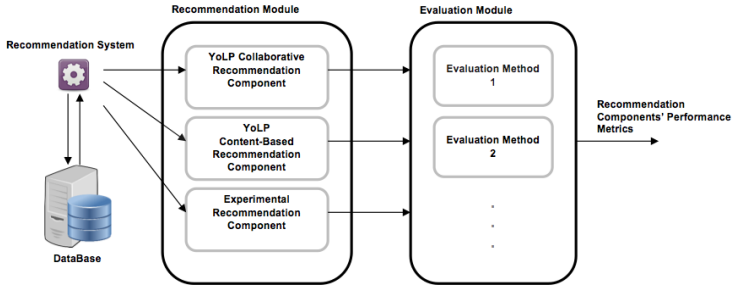


**Fig. 1.** System Architecture.

from 512 users. Although the main focus of this article is the content or ingredients of a meal, various other variables that impact a user's opinion in food recommendation are mentioned, amongst them cooking methods, ingredient costs and quantities, preparation time, and ingredient combination effects. The decomposition of recipes into ingredients implemented in this experiment is simplistic: ingredient scores were computed by averaging the ratings of recipes in which they occur.

This work shows that the content-based approach has the best overall performance, with a significant accuracy improvement over the collaborative filtering algorithm.

## 4 Architecture

The recommendation system contains three recommendation components (see Fig. 1): the YoLP collaborative recommender, the YoLP content-based recommender and an experimental recommendation component where various approaches are explored to adapt the Rocchio's algorithm for personalized food recommendations. The content-based and collaborative recommendation components implemented in YoLP are used here as baselines providing independent recommendations for the same *input*, in order to evaluate improvements in the prediction accuracy from the algorithms implemented in the experimental component. The evaluation module independently evaluates each recommendation component by measuring the performance of the algorithms using different metrics. The methods used in this module are explained in detail in the following section.

The YoLP collaborative recommendation component uses a standard item-to-item collaborative approach [2].

The YoLP content-based component generates recommendations by comparing the restaurant's recipes' features with the user profile using the cosine similarity measure. The recommended recipes are ordered from most to least similar. In this case instead of referring recipes as vectors of words, recipes are represented by vectors of different features. The features that compose a recipe are: category, region, restaurant ID and ingredients. Context features are also considered in the moment of the recommendation, these are: temperature, period of the day and season of the year. Each feature has a specific location attributed to it in the recipe and user profile sparse vectors.

### 4.1 Experimental Recommendation Component

The idea of considering ingredients in a recipe as similar to words in a document lead to the variation of TF-IDF weights developed in [11]. This work presented good results in retrieving the user's favourite ingredients, which raised the following question: could these results be further improved? As previously mentioned, the TF-IDF scheme can be used to attribute weights to words when using the popular Rocchio algorithm. Instead of simply obtaining the users' favorite ingredients using the TF-IDF variation [11], the user's overall preference in ingredients could be estimated through the prototype vector, which represents the learning in Rocchio's algorithm. These vectors would contain the users preferences, where the positive and negative examples are obtained directly from the user's rated recipes/dishes.

#### 4.1.1 Rocchio's Algorithm using FF-IRF

As mentioned in Section 3.1 of the related work, the approach inspired on TF-IDF, shown in equation Eq.(5), used to estimate the user's favourite ingredients is an interesting point to further explore in food recommendation. Since Rocchio's algorithm uses feature weights to build the prototype vectors, representing the user's preferences, and FF-IRF has shown good results for extracting the user's favourite ingredients [11], this measure could be used to attribute weights to the recipe's features and build the prototype vectors.

#### 4.1.2 Building the Users' Prototype Vector

The type of observation, positive or negative, and the weight attributed to each, determines the impact that a rated recipe has on the user prototype vector. In the experiments performed in this work positive and negative observations have an equal weight of 1. In order to determine if a rating event is considered a positive or negative observation two different approaches were studied. The first approach was simple: the lower rating values are considered a negative observation and the higher rating values are positive observations. The second approach utilizes the user's average rating value, computed from the training set.

#### 4.1.3 Generating a rating value from a similarity value

Datasets that contain user reviews on recipes or restaurant meals are presently very hard to find. *Epicurious* and *Food.com*, which will be presented next, are food related datasets with relevant information on the recipes that contain rating events from users to recipes. In order to validate the methods explored in this work, the recommendation system also needs to return a rating value. Rocchio's algorithm returns a similarity value between the recipe features vector and the user profile vector, so a method is needed to translate the similarity into a rating. Next, two methods are presented to translate the similarity value into a rating.

There are many types of normalization methods available, the technique chosen for this work was Min-Max Normalization.[2].

The second method uses the average and standard deviation values from the training set, which should in theory, bring good results and introduce only a very small error. To generate a rating value following formula was used:

[2] `http://aimotion.blogspot.pt/2009/09/data-mining-in-practice.html`

**Table 1.** Statistical characterization for the datasets used in the experiments.

| | Food.com | Epicurious | | Food.com | Epicurious |
|---|---|---|---|---|---|
| Number of users | 24741 | 8117 | Sparsity on the ratings matrix | 0.02% | 0.07% |
| Number of food items | 226025 | 14976 | Avg. rating values | 4.68 | 3.34 |
| Number of rating events | 956826 | 86574 | Avg. number of ratings per user | 38.67 | 10.67 |
| Number of ratings above avg. | 726467 | 46588 | Avg. number of ratings per item | 4.23 | 5.78 |
| Number of groups | 108 | 68 | Avg. number of ingredients per item | 8.57 | 3.71 |
| Number of ingredients | 5074 | 338 | Avg. number of categories per item | 2.33 | 0.60 |
| Number of categories | 28 | 14 | Avg. number of food groups per item | 0.87 | 0.61 |

$$Rating = \begin{cases} average\ rating + standard\ deviation, & \text{if } similarity >= U \\ average\ rating, & \text{if } L <= similarity < U \\ average\ rating - standard\ deviation, & \text{if } similarity < L \end{cases} \quad (8)$$

Three different approaches were tested: using the user's rating average and the user standard deviation; using the recipe's rating average and the recipe standard deviation; and using the combined average of the user and the recipe averages and standard deviations.

### 4.2 Datasets

The data for the experiments is provided by two datasets. The first dataset was previously made available by [8], collected from a large online[3] recipe sharing community. The second dataset is composed by crawled data obtained from a website named *Epicurious*[4]. In table 4.2 a statistical characterization for the two datasets is presented after the filter was applied.

Both datasets contain user reviews for specific recipes where each recipe is characterized by the following features: ingredients, cuisine and dietary.

## 5 Evaluation Metrics

Cross-validation was used to validate the recommendation components in this work. This technique is mainly used in systems that seek to estimate how accurately a predictive model will perform in practice [7].

The Mean Asolute Error (MAE) and Root Mean Squared Error (RMSE) measures compute the deviation between the predicted ratings and the actual ratings. These measures were used to validate the recommendation error. RMSE is very similar to MAE, but places more emphasis on higher deviations.

In order to validate the experimental context-based algorithms explored in this work, first some baselines need to be computed. Using the 5-fold cross-validation, the YoLP recommendation components presented in the previous section were evaluated. Besides these methods a few simple baselines metrics were also computed using the direct values of specific dataset averages as the predicted rating for the

[3] `http://www.food.com/`

[4] `http://www.epicurious.com/`

**Table 2.** Baselines.

| | Epicurious | | Food.com | |
|---|---|---|---|---|
| | MAE | RMSE | MAE | RMSE |
| YoLP Content-based component | 0.6389 | **0.8279** | **0.3590** | 0.6536 |
| YoLP Collaborative component | 0.6454 | 0.8678 | 0.3761 | 0.6834 |
| User Average | **0.6315** | 0.8338 | 0.4077 | **0.6207** |
| Item Average | 0.7701 | 1.0930 | 0.4385 | 0.7043 |
| Combined Average | 0.6628 | 0.8572 | 0.4180 | 0.6250 |

**Table 3.** Test Results.

| | Epicurious | | | | Food.com | | | |
|---|---|---|---|---|---|---|---|---|
| | Observation User Average | | Observation Fixed Threshold | | User Average Observation | | Observation Fixed Threshold | |
| | MAE | RMSE | MAE | RMSE | MAE | RMSE | MAE | RMSE |
| User Avg + User Standard Deviation | 0.8217 | 1.0606 | 0.7759 | 1.0283 | 0.4448 | 0.6812 | **0.4287** | 0.6624 |
| Item Avg + Item Standard Deviation | 0.8914 | 1.1550 | 0.8388 | 1.1106 | 0.4561 | 0.7251 | 0.4507 | 0.7207 |
| User/Item Avg + User and Item Standard Deviation | 0.8304 | 1.0296 | 0.7824 | **0.9927** | 0.4390 | 0.6506 | 0.4324 | **0.6449** |
| Min-Max | 0.8539 | 1.1533 | **0.7721** | 1.0705 | 0.6648 | 0.9847 | 0.6303 | 0.9384 |

recommendations. The averages computed were the following: user average rating; recipe average rating; and the combined average of the user and item averages, i.e. $(UserAvg+ItemAvg)/2$. Table 2 contains the MAE and RMSE values for the baseline methods.

As detailed in Section 4.1 two distinct ways of building the user's prototype vectors were presented: using the user average rating value as threshold for positive and negative observations; or simply using a fixed threshold in the middle of the rating range. These are referred in Table 3 as *Observation User Average* and *Observation Fixed Threshold*. Also detailed in Section 4.1 a few different methods are used to convert the similarity value, returned from Rocchio's algorithm, into a rating value. These methods are represented in the line entries of the Table 3 and are referred to as: *User Avg + User Standard Deviation*; *Item Avg + Item Standard Deviation*; *User/Item Avg + User and Item Standard Deviation*; and *Min-Max*.

Table 3 contains the first test results. The objective was to determine which method combination had the best performance so it could be further adjusted and improved. When observing the MAE and RMSE values it is clear that using the user average as threshold to build the prototype vectors results in higher error values than the fixed threshold of 3 to separate the positive and negative observations. The second conclusion that can be made from these results is that using the combination of both *user* and *item* average ratings and standard deviations has the overall lowest error values.

### 5.1 Feature Testing

In content-based methods it is important to determine if all features are helping to obtain the best recommendations, so feature testing is crucial. Using more features to describe the items in content-based methods should in theory improve the recommen-

dations, since we have more information available about them, and although this is confirmed in this test that may not always be the case. Some features, like for example the price of the meal, can increase the correlation between the user preferences and items he dislikes, so it is important to test the impact of every new feature before implementing it in the recommendation system.

### 5.2 Similarity Threshold Variation

Eq. 8, previously presented in Section 4.1.3 was used in the first experiments to transform the similarity value returned by the Rocchio's algorithm into a rating value. The initial values for the thresholds, 0.75 for $U$ and 0.25 for $L$, were good starting values to test this method but now other cases need to be tested. By fluctuating the case limits the objective of this test is to study the impact in the recommendation and discover the similarity case thresholds that return the lowest error values.

The tests showed that the lower threshold only has a negative effect in the recommendation accuracy and subtracting the standard deviation does not help. After removing the lower case from Eq. 8, the lowest results registered using the *Epicurious* dataset were 0.6544 MAE and 0.8601 RMSE. With the *Food.com* dataset the lowest MAE registered was 0.3229 and the lowest RMSE 0.6230.

Compared directly with the YoLP Content-based component, which obtained the overall lowest error rates from all the baselines, the experimental recommendation component showed better results when using the *Food.com* dataset.

### 5.3 Standard Deviation Impact in Recommendation Error

When recommending items using predicted ratings, the user standard deviation plays an important role in the recommendation error. Users with the standard deviation equal to zero, i.e. users that attributed the same rating to all their reviews, should have the lowest impact in the recommendation error. The objective of this test is to discover how significant is the impact of this variable and if the absolute error does not spike for users with higher standard deviations.

The *Food.com* dataset presented good results since it showed that the absolute error of the users starts to stagnate for users with standard deviations higher than 1. This implies that the algorithm is learning the user's preferences and returning good recommendations even to users with high standard deviations. For the *Epicurious* dataset, although the absolute error increased steadily for users with higher standard deviations, the increase rate remained consistent in these values.

## 6 Conclusions

In this M.Sc. dissertation the applicability of content-based methods in personalized food recommendation was explored. Using the well-known Rocchio algorithm several approaches were tested to further explore the breaking of recipes down into ingredients, presented in [3], and use more variables related to personalized food recommendation.

The Rocchio algorithm was never explored in food recommendation, so various approaches were tested to build the users' prototype vector and transform the sim-

ilarity value, returned by the algorithm, into a rating value needed to compute the performance of the recommendation system.

After determining the best approach to adapt the Rocchio algorithm to food recommendations, the similarity threshold test was performed to adjust the algorithm and seek improvements in the recommendation results. The final results of the experimental component showed improvements in the recommendation performance when using the *Food.com* dataset. With the *Epicurious* dataset some baselines, like the content-based method implemented in YoLP, registered lower error values. Being two datasets with very different characteristics, not improving the baseline results in both was not completely unexpected. In the *Epicurious* dataset the recipe ingredient information only contained its main ingredients, which were chosen by the user in the moment of the review, opposed to the full ingredient information that recipes have in the *Food.com* dataset. This removes a lot of detail both in the recipes and in the prototype vectors and adding the major difference in the dataset sizes, these could be some of the reasons why the difference in performance was observed.

The study of the impact that these features have on the recommendation is also another interesting point to approach in the future when datasets with more information are available. The experimental component can be configured to include more variables in the recommendation process, for example, season of the year (i.e. winter/fall or summer/spring), time of the day (i.e. lunch or dinner), total meal cost, total calories, amongst others.

## References


1. Robin Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2012.
2. Mukund Deshpande and George Karypis. Item Based Top-N Recommendation Algorithms. *ACM Transactions on Information Systems*, 22(1):143–177, 2004.
3. Jill Freyne and Shlomo Berkovsky. Recommending food: Reasoning on recipes and ingredients. In *Proceedings of the 18th International Conference on User Modeling, Adaptation, and Personalization*, volume 6075 LNCS, pages 381–386, 2010.
4. Uri Hanani, Bracha Shapira, and Peretz Shoval. Information filtering: Overview of issues, research and systems. *User Modeling and User-Adapted Interaction*, 11:203–259, 2001.
5. Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich. *Recommender Systems: An Introduction*, volume 40. Cambridge University Press, 2010.
6. Dietmar Jannach, Markus Zanker, Mouzhi Ge, and Marian Gröning. Recommender Systems in Computer Science and Information Systems - A Landscape of Research. In *E-Commerce and Web Technologies*, pages 76–87. Springer Berlin Heidelberg, 2012.
7. Ron Kohavi. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. *International Joint Conference on Artificial Intelligence*, 14(12):1137–1143, 1995.
8. Chia-jen Lin, Tsung-ting Kuo, and Shou-de Lin. *A Content-Based Matrix Factorization Model for Recipe Recommendation*, volume 8444. 2014.
9. Michael J. Pazzani and Daniel Billsus. Content-Based Recommendation Systems. *The Adaptive Web*, 4321:325–341, 2007.
10. G. Salton. *Automatic text processing*, volume 14. Addison-Wesley, 1989.
11. Mayumi Ueda, Mari Takahata, and Shinsuke Nakajima. User's food preference extraction for personalized cooking recipe recommendation. In *CEUR Workshop Proceedings*, volume 781, pages 98–105, 2011.