A Food Recommendation System using Vector Space Model

Nandakishore Puttashamachar
University of Central Florida, Orlando
nandakishore@knights.ucf.edu

Abstract

With tremendous increase in available resources about food recipes on leading websites, it is important to provide relevant food recommendations to a user/customer rather than to allow a user to browse through a whole list of websites and waste their valuable time. In order to sort out irrelevant information on food recipes and provide customized suggestions to user, a food recommendation system is built based on food properties and user profile. This project presents an approach to build a food recommendation system using vector space model.

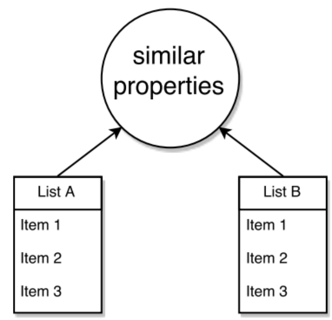
1 Introduction

Recommendation Systems have gained popularity in the recent years, especially with the Netflix prize. Human patterns are unique but also incredibly predictable, these small predictions have huge impact on the user-seller relationship. Recommendation Systems (Recommender Systems) are a set of algorithms which can predict a rating that a user can provide against an item, predict items which are similar to a user's profile or predict the most preferred item/category by learning multiple user profiles. Implementing a recommender system can please a user (customer) by suggesting items which are of preference to the user, leading to the benefit of both the seller and the user (customer).

Most of the recommendation systems follow two basic approaches - Content based filtering and Collaborative filtering.

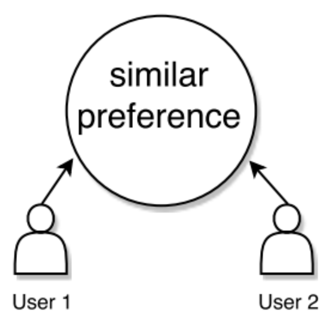
1.1 Content based filtering

In a content based recommender system, keywords or attributes are used to describe items. User profile is usually built around these attributes. Items are ranked based on their closeness measure to user profile or preferences. For instance, Pandora's Music Recommender explains how a content based filtering approach can be used to stream music to listeners application based on a user's taste.



1.2 Collaborative filtering

A recommender system of this kind is modeled around prior user behavior. Collaborative filtering uses the cumulated preferences/taste of several users and generate recommendations to a particular user. E.g. Netflix. The main advantage of using Collaborative filtering is that the algorithm need not learn about the attributes of the items, it can simply start recommending by analyzing taste of similar users.



In this project we will discuss on how a simple algorithm like vector space model can be used to build a content based recommendation system over a set of recipes.

2 Problem Definition

Aim of this project is to design a system to recommend food dishes to a user, where for a given user input query the algorithm outputs top food predictions for the user.

$$U = \{q_n\}$$

Where U is the user input query, *q* is the ingredient, and *n* is the number of ingredients input by the user.

$$Recommendations = F_m$$

Where *F* is the recommended food name and *m* is the number of recommendations.

3 Challenges

3.1 Cold Start

Cold start is a situation where there is not much available data on user profiles. This problem is of frequent occurrence in the collaborative recommendation systems. The recommendation system reaches a condition where no inference/decision can be reached without accumulating enough user information.

3.2 Scalability and Varying user preferences

User data and item profiles keep increasing with time. A good recommender system should be able to handle the computational complexity as the data set increases.

3.3 Varying user preferences

User preferences keep evolving with time. A low ranked item can move to the top rank eventually, as the intention/taste of user changes. A good recommendation should be able to adhere to these changes without much modification to the algorithm itself.

3.4 The long tail problem

Tailoring the recommendation system to each and every user is fairly unattainable. Long tail problem is a scenario where low ranked items/products never pop out from the recommendation system. Figure below shows how rank curve varies for different items. There may be a possibility where a user has interest in purchasing one of the items/products from the left out ones.

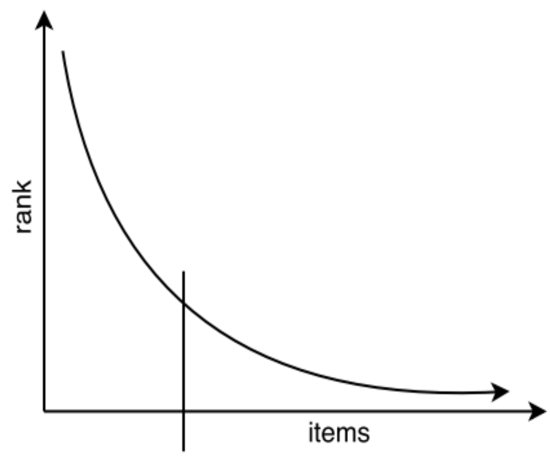


Figure 1 - Items prior to the marking will always be recommended but the items post the mark will be suppressed always

4 Dataset

Food recommendation requires good amount of data categorized into multiple cuisines, name of the food and its recipe.

4.1 Data collection

Data was extracted by crawling two popular websites - allrecipes.co.in and allrecipes.asia websites. 2581 recipes and 3837 recipes were gathered from each of the websites respectively. Recipe data is most suitable for this system as we can directly extract ingredients from the food which user likes and provide recommendations by matching those ingredients with the database. Data has been split into two categories –

Asian (Chinese and Thai recipes) and Indian recipes.

4.2 Data pre-processing

Accumulated data has been preprocessed to remove special characters, hexadecimal and html tags. All the recipes are represented in the form of documents, each document containing the recipe and document label being the food name.

5 Method

Most of the recommendation systems use similarity functions to measure the extent of likeliness between multiple items/users. Equation below shows a simple way of finding similarity between a set of recipes –

$$Similarity_j = sim\ \{r, r_j\}$$

Where *j* is the number of recipes

There are three well known similarity functions - Cosine similarity, Pearson’s correlation and Least squares. In this project cosine similarity function is used.

Given a query input by user *U* and a recipe *R* in the dataset we need to find the extent of similarity between two recipes in the range 0 to 1. Let

$$U = \{q_1, q_2, q_3, q_4 \ldots, q_n\}$$
$$\&$$
$$R = \{r_1, r_2, r_3, r_{4\ldots,} r_n\}$$

Where U and R are the *n* dimensional vectors and *q* is the query vector in *U* input by the user and *r* is the ingredient in a recipe *R*.

Similarity measure can be obtained by the magnitude of the angle between the vectors *U* and *R*. Angle 0 indicates identical vectors. The greater the angle between *U* and *R*, similarity measure is low and vice versa. Cosine of the angle between the ingredient query vector and the recipe ingredient vector is calculated using the equation–

$$\cos\theta = \frac{U.R}{\|U\|\|R\|}$$

Where *U* - ingredient query vector
R - recipe ingredient vector

||U|| and ||R|| is the Euclidean norm (L2 norm) can be calculated using the equations below –

$$\|U\| = \sqrt{\sum_{i=1}^{n} q_i^2}$$

$$\|R\| = \sqrt{\sum_{j=1}^{n} r_j^2}$$

Where *n* is the length of the ingredient and the query vector.

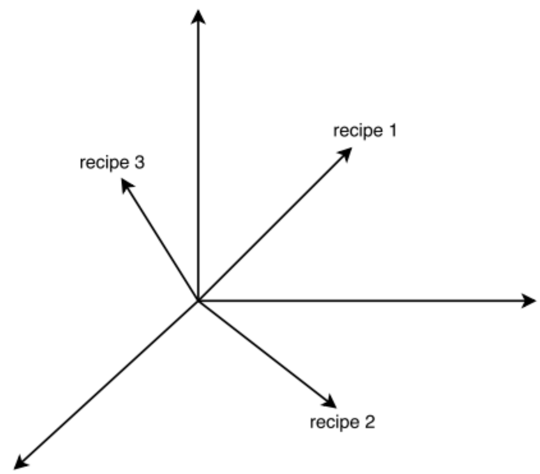


Figure 2 – Representation of 3 documents in the vector space model

6 Recommendation results

Tables below show results for three Ingredients query and the recommendation system output –

Ingredient Query	Recommendations
chicken cauliflower ghee coriander chilli powder coconut	Fried Rice

Table 1 – Results for a set of Indian ingredients

Ingredient Query	Recommendations
basmati carrot beans ginger garlic turmeric black pepper	Veg Biryani Veggie Nutty Biryani Veggie Kaju Rice Fragrant Ginger and Carrot Rice MIXED VEGETABLE PULAO

Table 2 - Recommendations similar to a dish Pulao

Ingredient Query	Recommendations
chicken hoisin sauce oyster sauce black pepper sesame oil peanut oil Chinese Shaoxing rice wine soy sauce	Spicy Chicken Drumsticks in Mushroom Hoisin Sauce Spicy Stir-Fried Chicken and Broccoli Ginger and hoisin chicken wings Asian Chicken Noodles Chinese Beef Rice Noodle

Table 3 – Recommendations for a Chinese dish

As we can see in table 1, the system produces only one recommendation output. A further more improvement in the data set can solve the problems of low/nil recommendations.

Ingredient query in table 2 is for a dish named "Vegetable Pulao" and the recommended food items are similar to it.

Table below shows how this recommender system can be evaluated –

	Recommended	Not-Recommended
Like	TP	FP
Dislike	FP	TN

Evaluation can be done by taking a user to input query ingredients and asking the user if the recommended food was of interest to them or not.

7 Future work

Recommendation system can be extended to other cuisines like American, Japanese etc. Adding more number of recipes to the existing database will solve the problems of getting zero/one recommendations. Though this system gives similar recommendation, it does not consider the quality/taste of recommended food. This can be overcome by considering an addition factor - reviews. Accumulating user reviews over several food items can help recommend top rated food to the user. Providing information on the amount of calories per serving in a dish can also be implemented.

8 Conclusion

Using vector space model, a food recommendation system is built. However, observing the results it is evident that by increasing the data set on each cuisine more recommendations can be obtained. Adding additional factors such as user reviews and calories per serving information, the quality of top food recommendation results can be improved.

References

Pandora's Music Recommender - Michael Howe

WebSci '12 Proceedings of the 4th Annual ACM Web Science Conference *Recipe recommendation using ingredient networks*

Michael D. Ekstrand, John T. Riedl and Joseph A.Konstan. *Collaborative Filtering Recommender Systems*

Seth Sorensen. *Accuracy of Similarity Measures in Recommender Systems*

Toine Bogers, Antal van den Bosch. *Collaborative and Content-based Filtering for Item Recommendation on Social Bookmarking Websites*

Yunhong Zhou, Dennis Wilkinson, Robert Schreiber and Rong Pan. *Large-scale Parallel Collaborative Filtering for the Netflix Prize*