Food Recommender System

Project Report submitted in partial fulfillment of the requirement for the degree of

Bachelor of Technology

in

Computer Science & Engineering

under the supervision of

Ms. Reema Aswani

by

Ankit Saha   111340

to

Jaypee University of Information and Technology

Waknaghat, Solan – 173234, Himachal Pradesh

Certificate

This is to certify that project report entitled “Food Recommender System”, submitted by Ankit Saha in partial fulfillment for the award of degree of Bachelor of Technology in Computer Science & Engineering to Jaypee University of Information Technology, Waknaghat, Solan has been carried out under my supervision.

This work has not been submitted partially or fully to any other University or Institute for the award of this or any other degree or diploma.

Date: 8th May, 2015 **Ms. Reema Aswani**

Assistant Professor

Acknowledgement

It is my radiant sentiment to express my sincerest regards to my project coordinator, **Ms. Reema Aswani** for her valuable inputs, able guidance, encouragement, whole-hearted cooperation and constructive criticism throughout the duration of my project.

I wish to express my profound gratitude and indebtedness to my head of department, **Prof. Dr. Satya Prakash Ghrera**, for his inspiring guidance, constructive criticism and valuable suggestion throughout the project work.

I take this opportunity to thank all my lecturers who have directly or indirectly helped my project. I pay my respects and love to my parents and all other family members for their love and encouragement throughout my career.

Last but not least, my sincere thanks to all my friends who have patiently extended all sorts of help for accomplishing this undertaking.

Date: 8th May, 2015 ANKIT SAHA

Table of Content

List of Figures

List of Tables

Abstract

With the increased interest in living a healthy lifestyle, research into applications that assist people to improve their lifestyles have also increased. This is particularly true with the development of systems and middleware which provide services utilizing various types of data. Typically, people who want to change lifestyle choices for better health and fitness, focus on their diets. As such, further research into applications that can inform consumers about appropriate food choices and that takes into account individual preferences is required.

People make decisions every day with the range varying from choice of films, places to visit, apparels, consumer goods, etc. There are too many choices and a little time to explore them all. Recommendation systems help people make decisions in these complex information spaces. Simply they compare user interest acquired from his/her profile with some reference characteristics and predict the rating that the user would give. Recommender systems have become extremely common in recent years, and are applied in a variety of applications.

CHAPTER 1

1.1 Introduction

Food is any substance consumed to provide nutritional support for the body. It is usually of plant or animal origin, and contains essential nutrients, such as fats, proteins, vitamins, or minerals. The substance is ingested by an organism and assimilated by the organism's cells to provide energy, maintain life, or stimulate growth.

Historically, people secured food through two methods: hunting and gathering, and agriculture. Today, most of the food energy required by the ever increasing population of the world is supplied by the food industry.

The right to food is a human right derived from the International Covenant on Economic, Social and Cultural Rights (ICESCR), recognizing the "right to an adequate standard of living, including adequate food," as well as the "fundamental right to be free from hunger."

Many cultures have a recognizable cuisine, a specific set of cooking traditions using various spices or a combination of flavors unique to that culture, which evolves over time. Other differences include preferences (hot or cold, spicy, etc.) and practices, the study of which is known as gastronomy. Many cultures have diversified their foods by means of preparation, cooking methods, and manufacturing. This also includes a complex food trade which helps the cultures to economically survive by way of food, not just by consumption. Some popular types of ethnic foods include Italian, French, Japanese, Chinese, American, Cajun, Thai, African, and Indian cuisine .Various cultures throughout the world study the dietary analysis of food habits. While evolutionarily speaking, as opposed to culturally, humans are omnivores, religion and social constructs such as morality, activism, or environmentalism will often affect which foods they will consume. Food is eaten and typically enjoyed through the sense of taste, the perception of flavor from eating and drinking. Certain tastes are more enjoyable than others, for evolutionary purposes.

1.2 Problem Statement

We all like to dine out once in a while. We often hear about good places to eat through various sources like friends, relatives, food shows, etc. But the issue arises when we have to choose from a hoard of items in the menu at those places. We don’t have much idea about which of the available dishes will cater to our taste buds.

The basic premise of this project is to resolve this problem. A system that can take into account one’s preferences regarding food items and recommend certain dishes after analyzing the data is the solution to the issue.

The proposed food recommendation system can enable the management of personalized well-being care in the way that it can collect data from personal contexts at run time, and interprets these in a user-centric way. Also, the proposed system can provide valuable real-time feedback to users who have specific eating habits. This information is presented to users in an intuitive and convenient real-time display.

1.3 Objective

Remembering the scenarios mentioned in the abstract, I focus on the question “What should I eat?” in the scope of this project. I try to achieve this goal with a food based recommender system. My system uses content-based recommendation technique for producing food recommendations. It is based on similarity of foods. Basically, my system constructs user profiles from the previously rated features and food profiles from the ingredients of the food, and then it recommends the most appropriate foods according to the preferences of the users.

1.4 Scope

There are four scopes in this project:

- The system is a web based system
- User has to input his preferences of ingredients
- The system works on extracted data and algorithms
- The recommendations are generated and user approves them

1.5 Organization

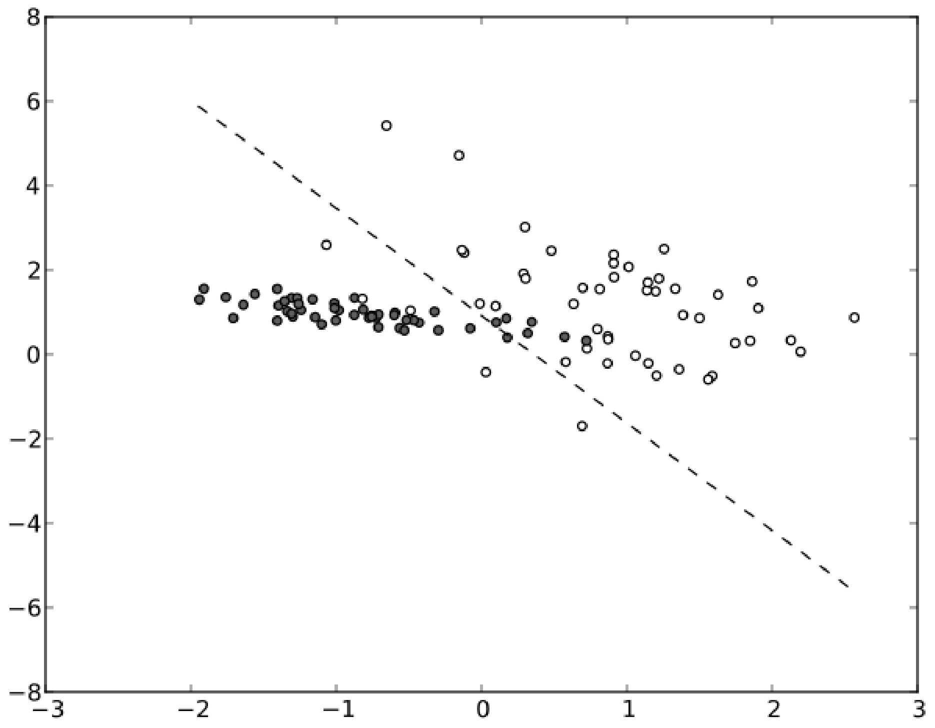
The report is organized into different chapters. Chapter 2 deals with the background of the topics. Chapter 3 contains with the literature review and analysis of existing systems and tools and technologies to be used. Chapter 4 covers the implementation aspects of the project and Chapter 5 has testing aspects. Chapter 6 deals with the evaluation of the system. The chapters are followed by references and appendix.

CHAPTER 2

2.1 Background

Machine learning is a scientific discipline that explores the construction and study of algorithms that can learn from data. Such algorithms operate by building a model based on inputs and using that to make predictions or decisions, rather than following only explicitly programmed instructions.

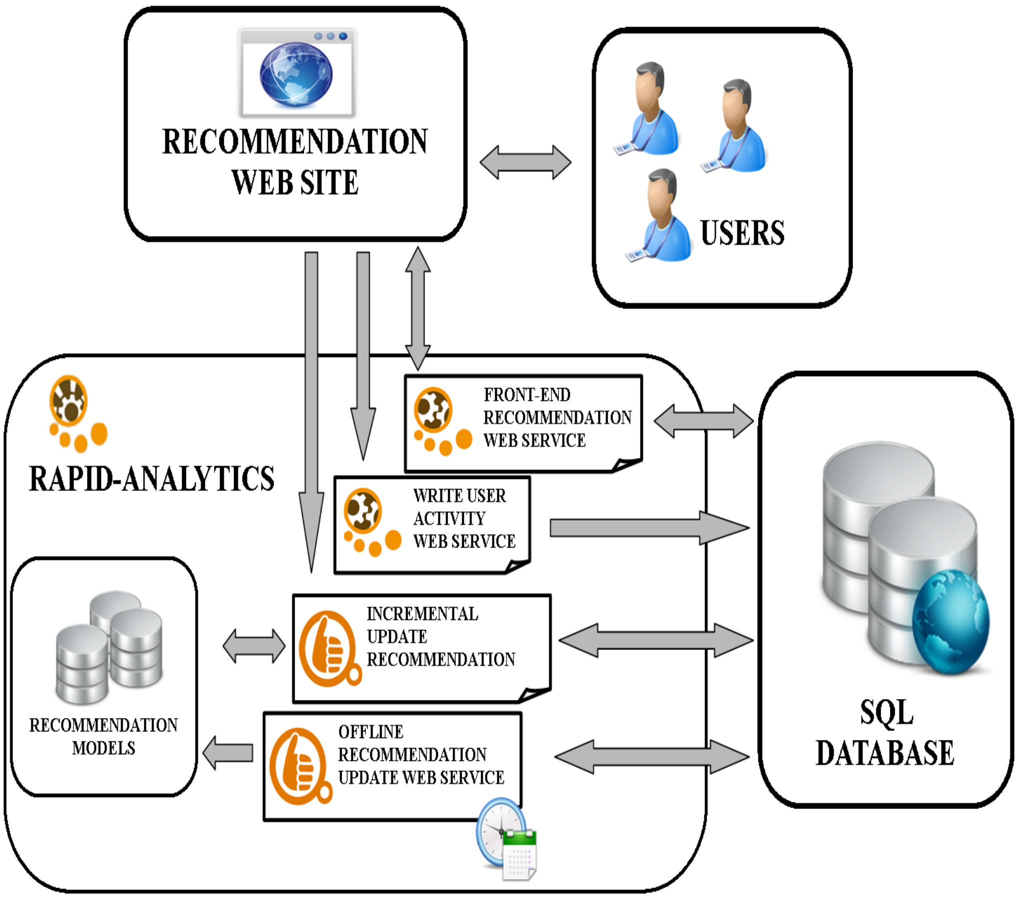
Machine learning can be considered a subfield of computer science and statistics. It has strong ties to artificial intelligence and optimization, which deliver methods, theory and application domains to the field. Machine learning is employed in a range of computing tasks where designing and programming explicit, rule-based algorithms is infeasible. Example applications include spam filtering, optical character recognition (OCR), search engines and computer vision. Machine learning is sometimes conflated with data mining, although that focuses more on exploratory data analysis. Machine learning and pattern recognition "can be viewed as two facets of the same field.



Data mining (the analysis step of the "Knowledge Discovery in Databases" process, or KDD), an interdisciplinary subfield of computer science, is the computational process of discovering patterns in large data sets involving methods at the intersection of artificial intelligence, machine learning, statistics, and database systems. The overall goal of the data mining process is to extract information from a data set and transform it into an understandable structure for further use. Aside from the raw analysis step, it involves database and data management aspects, data pre-processing, model and inference considerations, interestingness metrics, complexity considerations, post-processing of discovered structures, visualization, and online updating.

The actual data mining task is the automatic or semi-automatic analysis of large quantities of data to extract previously unknown interesting patterns such as groups of data records (cluster analysis), unusual records (anomaly detection) and dependencies (association rule mining). This usually involves using database techniques such as spatial indices. These patterns can then be seen as a kind of summary of the input data, and may be used in further analysis or, for example, in machine learning and predictive analytics.

Recommender systems or recommendation systems (sometimes replacing "system" with a synonym such as platform or engine) are a subclass of information filtering system that seek to predict the 'rating' or 'preference' that user would give to an item. Recommender systems have become extremely common in recent years, and are applied in a variety of applications. The most popular ones are probably movies, music, news, books, research articles, search queries, social tags, and products in general. However, there are also recommender systems for experts, jokes, restaurants, financial services, life insurance, persons (online dating), and Twitter followers. Amazon, Last.fm, Ulike, iLike, Netflix, Pandora are the most popular recommender systems all over the world.


RECOMMENDATION WEB SITE
USERS
RAPID-ANALYTICS
FRONT-END RECOMMENDATION WEB SERVICE
WRITE USER ACTIVITY WEB SERVICE
INCREMENTAL UPDATE RECOMMENDATION
OFFLINE RECOMMENDATION UPDATE WEB SERVICE
RECOMMENDATION MODELS
SQL DATABASE

Recommendation systems changed the way inanimate websites communicate with their users. Rather than providing a static experience in which users search for and potentially buy products, recommender systems increase interaction to provide a richer experience. Recommender systems identify recommendations autonomously for individual users based on past purchases and searches, and on other users' behavior.

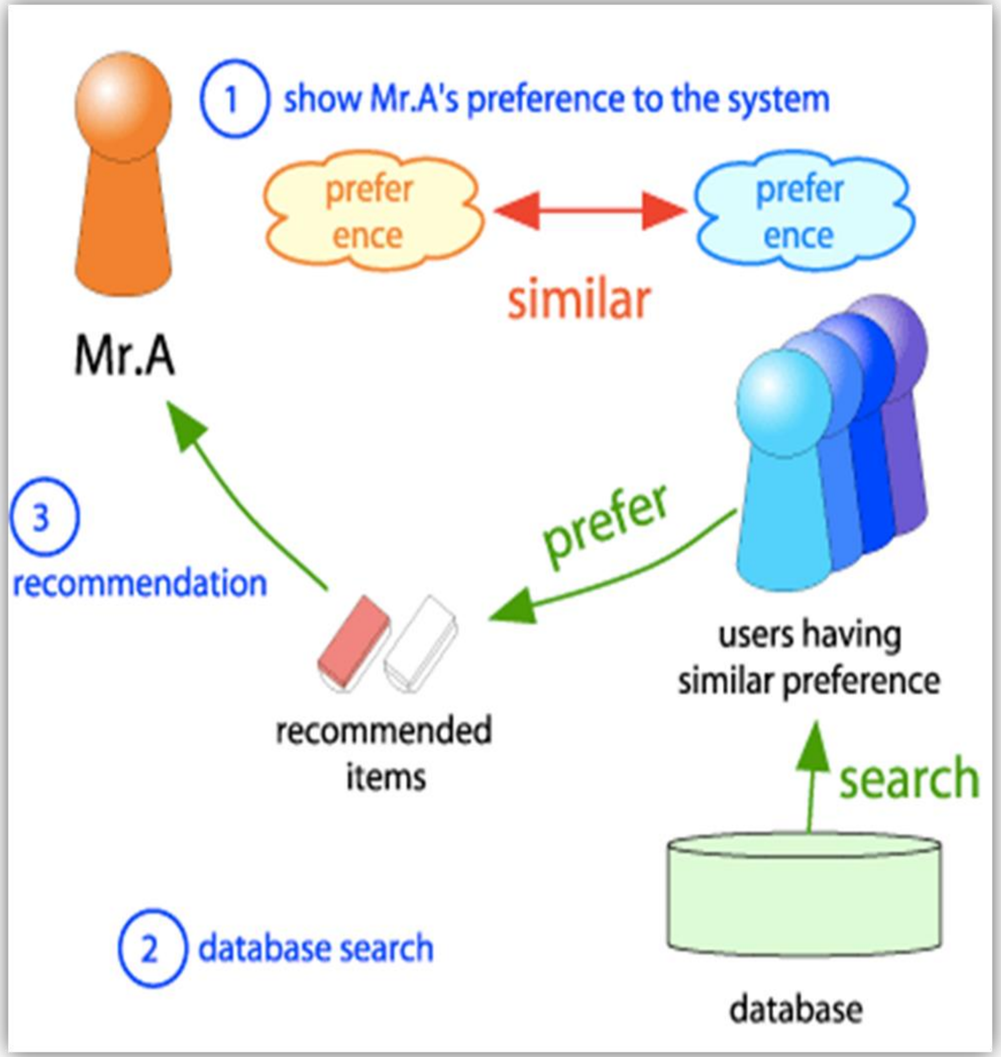
2.2 Approaches

2.2.1 Collaborative Filtering

Collaborative filtering (CF) is a technique used by some recommender systems. Collaborative filtering has two senses, a narrow one and a more general one. In general, collaborative filtering is the process of filtering for information or patterns using techniques involving collaboration among multiple agents, viewpoints, data sources, etc. Applications of collaborative filtering typically involve very large data sets. Collaborative filtering methods have been applied to many different kinds of data including: sensing and monitoring data, such as in mineral exploration, environmental sensing over large areas or multiple sensors; financial data, such as financial service institutions that integrate many financial sources; or in electronic commerce and web applications where the focus is on user data, etc.

In the newer, narrower sense, collaborative filtering is a method of making automatic predictions (filtering) about the interests of a user by collecting preferences or taste information from many users (collaborating). The underlying assumption of the collaborative filtering approach is that if a person *A* has the same opinion as a person *B* on an issue, A is more likely to have B's opinion on a different issue *x* than to have the opinion on x of a person chosen randomly.

One of the most famous examples of collaborative filtering is item-to-item collaborative filtering (people who buy x also buy y), an algorithm popularized by **Amazon**'s recommender system



Drawbacks:

- Data sparsity:

  Cold start problem. Briefly, a brand new product need to be rated by a substantial amount of users before it could be recommended. The product won't be limited if the system is using the content-based approach, which we'll see it in the next post.

- Scalability:

  The computing power needs to be very strong if the websites have a great amount of products and customers.

- Synonyms:
  Same or very similar items which have different names would not be recognize the same within the CF systems.

- Grey sheep:
  Users who are not consistent with their like and dislike. This may make the CF fail to recommend items.

- Shilling attack:
  People may give their own items a lot of good rating and bad ratings to their competitors.

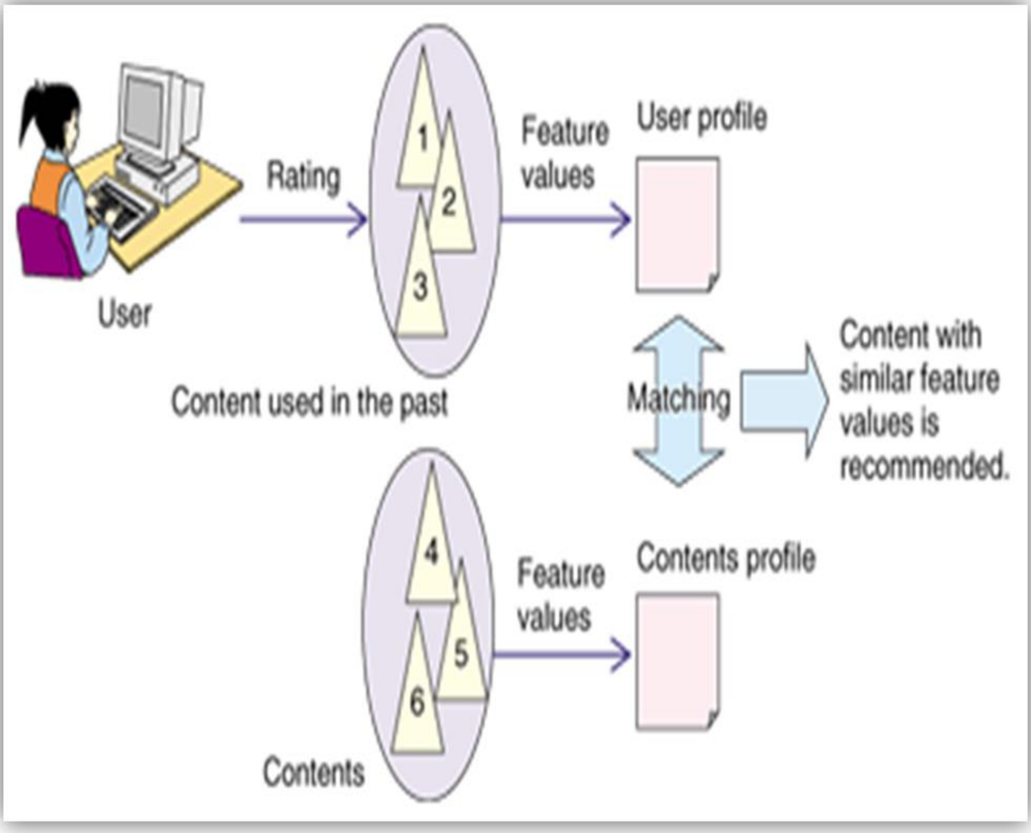
2.2.2 Content-based Filtering

Content-based filtering is using the technique to analyze a set of documents and descriptions of items previously rated by a user, and then build a profile or model of the users interests based on the features of those rated items. Using the profile, the recommender system can filter out the suggestions that would fit for the user.

Content-based filtering methods are based on a description of the item and a profile of the user’s preference. In a content-based recommender system, keywords are used to describe the items; beside, a user profile is built to indicate the type of item this user likes. They try to recommend items that are similar to those that a user liked in the past (or is examining in the present).

In particular, various candidate items are compared with items previously rated by the user and the best-matching items are recommended. To abstract the features of the items in the system, an item presentation algorithm is applied. To create user profile, the system mostly focuses on two types of information (i) a model of the user's preference (ii) a history of the user's interaction with the recommender system. Direct feedback from a user, usually in the form of a like or dislike button, can be used to assign higher or lower weights on the importance of certain attributes.

Pandora Radio is a popular example of a content-based recommender system that plays music with similar characteristics to that of a song provided by the user as an initial seed.



Drawbacks:

- Limited content analysis:
  If the content does not contain enough information to discriminate the items precisely, the recommendation will be not precisely at the end.

- Over-specialization:
  Content-based method provides a limit degree of novelty, since it has to match up the features of profile and items. A totally perfect content-based filtering may suggest nothing "surprised."

- New user:

  When there's not enough information to build a solid profile for a user, the recommendation could not be provided correctly.

Advantages:

- User independence:

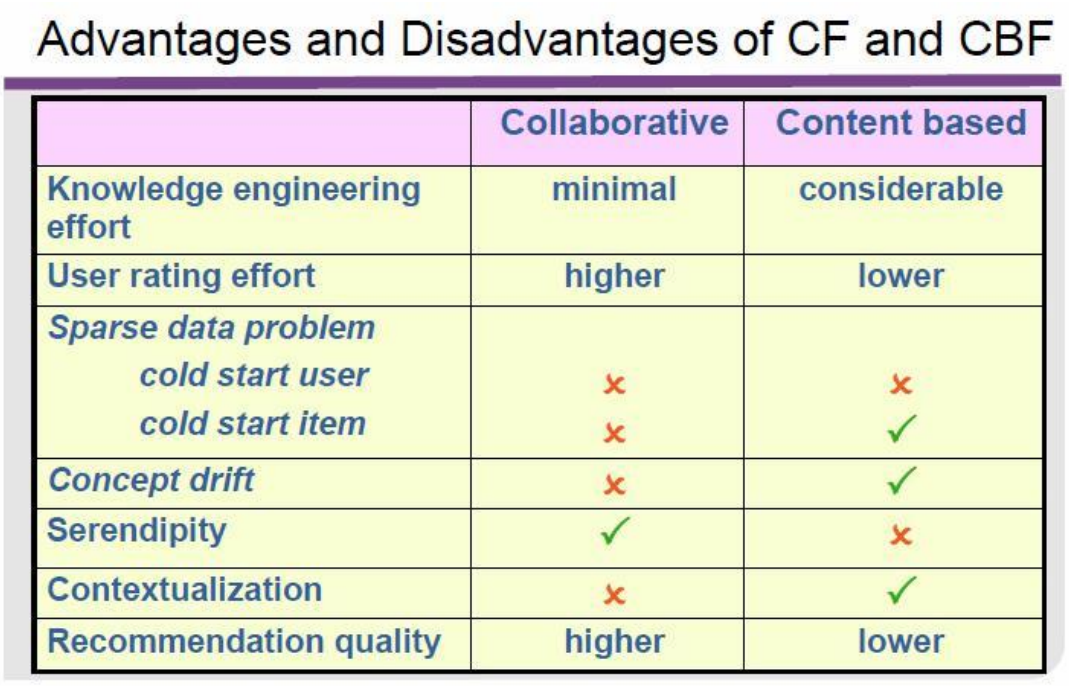
  Collaborative filtering needs other users' rating to find the similarity between the users and then give the suggestion. Instead, content-based methods only have to analyze the items and user profile for recommendation.

- Transparency:

  Collaborative method gives you the recommendation because some unknown users have the same taste like you, but content-based method can tell you they recommend you the items based on what features.

- No cold start:

  Opposite to collaborative filtering, new items can be suggested before being rated by a substantial number of users.



Advantages and Disadvantages of CF and CBF

	Collaborative	Content based
Knowledge engineering effort	minimal	considerable
User rating effort	higher	lower
Sparse data problem		
cold start user	✗	✗
cold start item	✗	✓
Concept drift	✗	✓
Serendipity	✓	✗
Contextualization	✗	✓
Recommendation quality	higher	lower

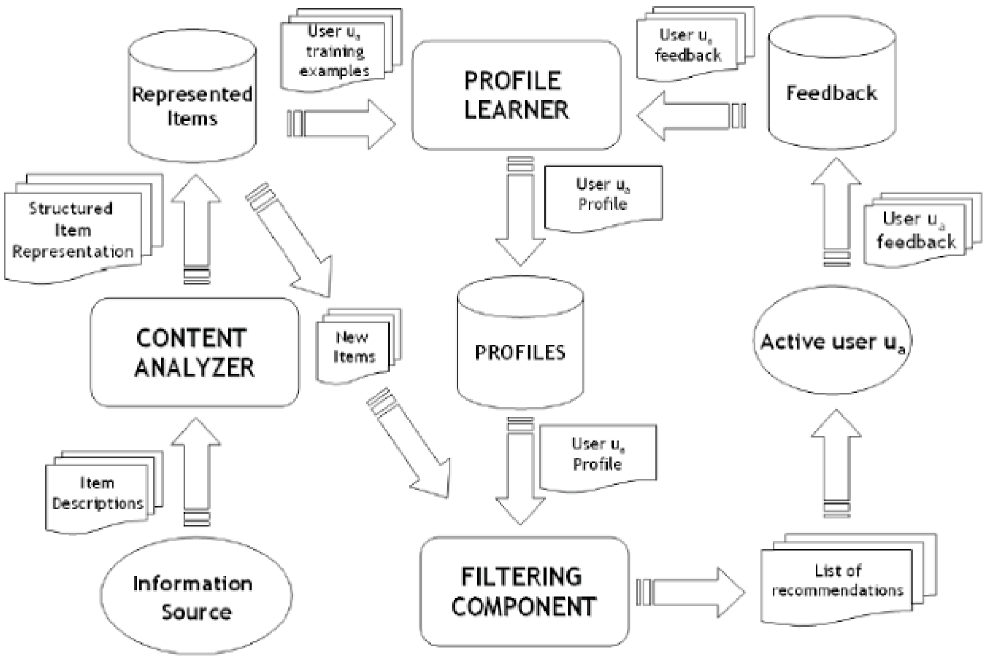
2.2.3 Hybrid Approach

Recent research has demonstrated that a hybrid approach,
combining collaborative filtering and content-based filtering could be more effective in some cases. Hybrid approaches can be implemented in several ways:

(i) by making content-based and collaborative-based predictions separately and then combining them

(ii) by adding content-based capabilities to a collaborative-based approach (and vice versa)

(iii) by unifying the approaches into one model



Several studies empirically compare the performance of the hybrid with the pure collaborative and content-based methods and demonstrate that the hybrid methods can provide more accurate recommendations than pure approaches. These methods can also be used to overcome some of the common problems in recommender systems such as cold start and the sparsity problem.

Netflix is a good example of hybrid systems. They make recommendations by comparing the watching and searching habits of similar users (i.e. collaborative filtering) as well as by offering movies that share characteristics with films that a user has rated highly (content-based filtering).

A variety of techniques have been proposed as the basis for recommender systems: collaborative, content-based, knowledge-based, and demographic techniques. Each of these techniques has known shortcomings, such as the well-known cold-start problem for collaborative and content-based systems (what to do with new users with few ratings) and the knowledge engineering bottleneck in knowledge-based approaches. A hybrid recommender system is one that combines multiple techniques together to achieve some synergy between them.

- Collaborative:
  The system generates recommendations using only information about rating profiles for different users. Collaborative systems locate peer users with a rating history similar to the current user and generate recommendations using this neighborhood.

- Content-based:
  The system generates recommendations from two sources: the features associated with products and the ratings that a user has given them. Content-based recommenders treat recommendation as a user-specific classification problem and learn a classifier for the user's likes and dislikes based on product features.

- Demographic:
  A demographic recommender provides recommendations based on a demographic profile of the user. Recommended products can be produced for different demographic niches, by combining the ratings of users in those niches.

- Knowledge-based:
  A knowledge-based recommender suggests products based on inferences about a user’s needs and preferences. This knowledge will sometimes contain explicit functional knowledge about how certain product features meet user needs.

2.3 Use of Recommender Systems

Recommender Systems (RSs) is a way to deal with the overload information/products that customers may encounter. RSs are a trend that companies tend to use because:

(i) Increase the number of items sold.
(ii) Sell more diverse items.
(iii) Increase the user satisfaction.
(iv) Increase users' fidelity
(v) Better understand what the user wants.

It's always their objective for companies to increase their revenues. For those companies which have a great deal of products, RSs are the way to increase their revenues.

First, increase the number of items sold. For example, if you want to buy a new boots on Amazon, you may first search for a specific brand that you like. But you didn't see any style that is cool enough, so you didn't buy any pairs. When you are about to give up, you'll see that Amazon would recommend you the items that are similar to the items you viewed. You may be able to find something that is fascinating from its recommendation. And the following will be that you click the button and buy the shoes, while Amazon successfully uses the RSs to increase its revenues.

Second, sell more diverse items. Here's another example from Amazon. See the cropped picture below. When I look at the MacBook air product page, you'll see there's kind of a bundle suggested by Amazon. It'll suggest you the items that other customers would buy together. Although it's not like a bundle promotion which will offer you some discount (the price is just the sum of the three products), it still increases customers' incentive to buy them together.

Third, its use ensures that the customer's satisfaction and fidelity will increase. The RSs can reduce the time that customers spending on browsing back and forth in the different categories. It can help the customers to find the stuffs they want in a relatively short time. In addition, the longer the user interacts with the websites/RSs, the more effective and accurate the recommendation will be. And, at the end, the cycling of the increasing usage and the increasing benefit will reach satisfaction and fidelity.

Finally, understand what the user wants. The data collected by the systems is not only doing good for the customers, but also helping the companies to understand what the user wants. It's a valuable data. It can help the companies to adjust many things like the inventory, the product categories, and even the business strategies. When you understand more from your customers, you are able to exploit more from them.

2.4 Evaluation

Typically, research on recommender systems is concerned about finding the most accurate recommendation algorithms. However, there is a number of factors that are also important.

Diversity - Users tend to be more satisfied with recommendations when there is higher intra-list diversity, i.e. items from e.g. different artists.

Recommender Persistence - In some situations it is more effective to re-show recommendations, or let users re-rate items, than showing new items.

Privacy - Recommender systems usually have to deal with privacy concerns because users have to reveal sensitive information. Building user profiles using collaborative filtering can be problematic from a privacy point of view. Much research has been conducted on ongoing privacy issues in this space.

User Demographics - Beel et al. found that user demographics may influence how satisfied users are with recommendations. In their paper they show that elderly users tend to be more interested in recommendations than younger users.

Robustness - When users can participate in the recommender system, the issue of fraud must be addressed.

Serendipity - Serendipity is a measure "how surprising the recommendations are". For instance, a recommender system that recommends milk to a customer in a grocery store might be perfectly accurate but still it is not a good recommendation because it is an obvious item for the customer to buy.

Trust - A recommender system is of little value for a user if the user does not trust the system. Trust can be built by a recommender system by explaining how it generates recommendations, and why it recommends an item.

Labelling - User satisfaction with recommendations may be influenced by the labeling of the recommendations.

CHAPTER 3

3.1 Literature Review

The following research papers contributed in the understanding of the topic:

S. No.	Title	Year
1.	u-BabSang: a context-aware food recommendation system	2009
2.	Intelligent Food Planning- Personalized Recipe Recommendation,	2010
3.	Smart Kitchen: Recipe Recommendation System	2010
4.	User's Food Preference Extraction for Personalized Cooking Recipe Recommendation	2011
5.	Recipe Recommendation using Ingredient Networks	2012
6.	A Mobile Food Recommendation System Based on The Traffic Light Diet	2014

u-BabSang: a context-aware food recommendation system

Yoosoo Oh Ahyoung Choi WoontackWoo

© Springer Science + Business Media, LLC 2009

Abstract

In this paper, a context-aware food recommendation system for well-being care applications is proposed. The proposed system, called u-BabSang, provides individualized food recommendation lists at the dining table, and is based dietary advice in the typical Korean medical text. Our proposed system receives a user's profile, physiological signals, and environmental information around the dining table in real time. To operate our system, we present a method for user specified analysis, and also describe time-division layered context integration which integrates the multiple contexts obtained from the sensors. Thus, our system recommends appropriate foods for each individual's health at the table in real time.

u-BabSang recommends appropriate foods in real time to each individual by utilizing each individual's profile, physiological signals, and sensed environmental information. The advantage of the proposed system is that it enables the management of personalized well-being care by integrating user contexts with other contexts during run time, and interpreting these contexts in a user centric manner. More specifically, by analyzing their personal situation (e.g., physiological status, eating custom, weather, location, etc.), the proposed system can monitor the health of individuals with specific clinical history. The proposed system has the potential to be extended to various other fields. For instance, this system can be extended to menus besides Korean menus. In particular, it can recommend food types and quantities based on the user's food preference, constitution, clinical history, and current climate conditions. Also, if we embed our system into personal devices (e.g., UMPC, mobile phone, etc.), the proposed system can be extended to recommend appropriate food items at point of purchase, e.g., supermarkets, restaurants, and other public places. In this way, users are able to protect personal information while accessing recommended products (e.g., foodstuffs, groceries, merchandise, etc.). Additionally, the proposed system could be utilized by athletics or medical services. For future work, we will evaluate our system in real environments such as homes, restaurants, and stores. After obtaining the user study, we will hope to improve convenience of using sensors and display on the table. Another possible area to explore is to track and integrate the history of users' eating habits.

Intelligent Food Planning: Personalized Recipe Recommendation

Jill Freyne and Shlomo Berkovsky
CSIRO Tasmanian ICT Centre
Hobart, Australia

Abstract

As the obesity epidemic takes hold, many medical professionals are referring users to online systems aimed at educating and persuading users to alter their lifestyle. The challenge for many of these systems is to increase initial adoption and sustain

participation for sufficient time to have real impact on the life of their users. In this work we present some preliminary investigation into the design of a recipe recommender, aimed at educating and sustaining user participation, which makes tailored recommendations of healthy recipes. We concentrate on the two initial dimensions of food recommendations: data capture and food-recipe relationships and present a study into the suitability of varying recommender algorithms for the recommendation of recipes.

As with all recommender technologies, a balance needs to be struck between accuracy, coverage and the workload of the users in providing information. This work presented an initial analysis as to the practicality of gathering food preferences and making recipe recommendations. We found that high coverage and reasonable accuracy can be achieved through content based strategies with a simple break down and construction used to relate recipes and food items. We found only marginal improvement in accuracy when collaborative filtering is employed to boost the rating matrix density.

In conclusion, we have shown that even a naive recipe break down into food items with reasoning on the latter provides more accurate recommendations than a collaborative filtering algorithm using a sparse matrix. Our future work includes an investigation into more intelligent means of reasoning on food ratings when recipe ratings are known, and vice versa, on recipe ratings when food ratings are known. Our first consideration concerns the impact of mixed ratings on recipes. For example, when breaking down recipes, a food item may receive a positive rating in one recipe and a negative in another, which are currently just averaged. However, more appropriate combinations would consider whether an ingredient in a positively and a negatively recipe should maintain only the positive rating, as it is unlikely to be the cause of the dislike in the negatively rated recipe. Furthermore, here we operate a simplistic idea of what a recipe recommender needs to do. We are, however, aware that making recipe recommendations is a far more complicated task in reality and we aim to investigate group recommendations, sequential recommendations, and diversification of recommendations.

Smart Kitchen: Recipe Recommendation System

Yap Lee Leng, Faculty of Computer Systems & Software Engineering, University Malaysia Pahang (UMP)

Abstract

Today want to eat or cook what meals? This is a common question that everyone asking, especially the housewife. Many people busy on work, some of them no time to think the meal that they want prepare to their family at night. Children are very changeable, not easy to think the meal prepare to them every day. Therefore, this project is carried out to build a prototype for recipe recommendation. It gives recommender or suggest recipe to the user based on their preference or the ingredient that available. The system is a web based system, it develop for those who are in rush. The Linear Model of Expert System Development Life Cycle is implemented in the system development. The rule is design based on the recipe that collected from web sites and book.

Nowadays, many people busy on work. Some of them did not know the meals they want to cook later need prepared what ingredient. It is normal to think that couples or family members who work at a company or a person who lives alone want to cook food for themselves as quickly as possible and no need to worry about what to cook when they are in rush. However, if every day having same food, then they will get bored. They need an easy way to get more recipes. Thinking of what to cook is also a difficult problem. To attract children liking, parent need to exchange the menu every day. Parents not only think to what recipe to changes, they also need to consider the nutrition that their children taken. Besides that, some people will forget buy ingredients to stock in their kitchen. This will become a problem when they want to prepare meal within short time. It is difficult to think what to cook with limited ingredient that in the kitchen.

User's Food Preference Extraction for Personalized Cooking Recipe Recommendation

Mayumi Ueda Mari Takahata Shinsuke Nakajima
Kyoto University Yoshida Nihonmatsu-cho, Sakyo-ku, Kyoto, Kyoto 606 {8501, Japan
Kyoto Sangyo University Motoyama, Kamigamo, Kita-Ku, Kyoto-City 603{8555, Japan

Abstract

There are many websites and researches that involve cooking recipe recommendation. However, these websites present cooking recipes on the basis of entry date, access frequency, or the recipe's user ratings. They do not reject the user's personal preferences. We have pro-posed a personalized recipe recommendation method that is based on the user's food preferences. For extracting the user's food preferences, we use his/her recipe browsing and cooking history. In this paper, we present a method for extracting the user's preferences. In the experimental results, extracting the user's favorite ingredients was detected with a 60 to 83% of precision. And extracting the not favorite ingredients was detected with 14.7% of precision, and 58% of recall. Furthermore, the F-measure value for extraction of favorite ingredients was 60.8% when we focused on the top 20 ingredients.

In this paper, a method for extracting the user’s food preferences for recipe recommendation was presented. The method estimates a user’s preferences from his/her past actions, such as through their recipe browsing and menu planning history. For extracting the preferences, our method breaks recipes down into their ingredients and scores the recipes using the frequency and specificity of ingredients. Since the method can estimate the preferences through their browsing and cooking history, the user convey his/her preferences to the system without having to carry out any articular operation. Furthermore, the user can convey the changes in his/her preferences to the system on a daily basis. In order to verify the extracting accuracy of the user’s food preferences, simple experiments were conducted. In the experimental results, extracting the user’s favorite ingredients were detected with a 60 to 83% of precision. And the F-measure was 60.8% when we focused on the top 20 ingredients. Since the average number of user's favorite ingredients was 19.2, our system should focus on the top 20 ingredients sorted by *I*+*k* to score recipes for recommendation. And extracting the disliked ingredient were detected with 14.7% of precision, and 58% of recall. In this time, we could not extract satisfactory accuracy. However, we consider that our method can improve the accuracy of extraction, by increasing the number of experiments. As future work, we plan to consider the ingredient ontology for estimating favorite/disliked ingredients. Furthermore, we plan to investigate the various weights. For example, we plan to verify *x* in Eq.(5), the ratio or frequency of avoiding the ingredients, through experiments. Moreover, we want to investigate the length of time for which the system should avoid recommending similar dishes for *d*, and the degree of similarity that the system should consider while refraining from recommending similar dishes.

Recipe Recommendation using Ingredient Networks

Chun-Yuen Teng, University of Michigan Ann Arbor, MI, USA
Yu-Ru Lin IQSS, Harvard University CCS, Northeastern University Boston, MA
Lada A. Adamic, University of Michigan Ann Arbor, MI, USA

Abstract

The recording and sharing of cooking recipes, a human activity dating back thousands of years naturally became an early and prominent social use of the web. The resulting online recipe collections are repositories of ingredient combinations and cooking methods whose large-scale and variety yield interesting insights about both the fundamentals of cooking and user preferences. At the level of an individual ingredient we measure whether it tends to be essential or can be dropped or added, and whether its quantity can be modified. We also construct two types of networks to capture the relationships between ingredients. The complement network captures which ingredients tend to co-occur frequently, and is composed of two large communities: one savory, the other sweet. The substitute network, derived from user-generated suggestions for modifications, can be decomposed into many communities of functionally equivalent ingredients, and captures users' preference for healthier

variants of a recipe. Our experiments reveal that recipe ratings can be well predicted with features derived from combinations of ingredient networks and nutrition information.

Recipes are little more than instructions for combining and processing sets of ingredients. Individual cookbooks, even the most expansive ones, contain single recipes for each dish. The web, however, permits collaborative recipe generation and modification, with tens of thousands of recipes contributed in individual websites. We have shown how this data can be used to glean insights about regional preferences and modifiability of individual ingredients, and also how it can be used to construct two kinds of networks, one of ingredient complements, the other of ingredient substitutes. These networks encode which ingredients go well together, and which can be substituted to obtain superior results, and permit one to predict, given a pair of related recipes, which one will be more highly rated by users. In future work, we plan to extend ingredient networks to incorporate the cooking methods as well. It would also be of interest to generate region-specific and diet-specific ratings, depending on the users' background and preferences. A whole host of user-interface features could be added for users who are interacting with recipes, whether the recipe is newly submitted, and hence unrated, or whether they are browsing a cookbook. In addition to automatically predicting a rating for the recipe, one could ag ingredients that can be omitted, ones whose quantity could be tweaked, as well as suggested additions and substitutions.

A Mobile Food Recommendation System Based on the Traffic Light Diet

Thienne Johnson, Jorge Vergara , Chelsea Doll, Madison Kramer, Gayathri Sundararaman, Harsha Rajendran, Alon Efrat and Melanie Hingle

University of Arizona, Tucson, AZ 85710, USA

Abstract

Innovative, real-time solutions are needed to address the mismatch between the demand for and supply of critical information to inform and motivate diet and health-related behavior change. Research suggests that interventions using mobile health technologies hold great promise for influencing knowledge, attitudes, and behaviors related to energy balance. The objective of this paper is to present insights

related to the development and testing of a mobile food recommendation system targeting fast food restaurants. The system is designed to provide consumers with information about energy density of food options combined with tips for healthier choices when dining out, accessible through a mobile phone.

The proposed system is still in its initial testing and evaluation phases, but preliminary data suggest it can be used (and is useful) for members of the community. The prototype was functional, and users evaluation suggested that the product is useful. The application and database were built to be HIPAA (Health Insurance Portability and Accountability Act [10]) compliant and designed to accommodate future upgrades; for example, expanding the tracking feature to allow users to document their choices and recording how many "red" food items they ate that day/week/month. Related to this, visualization of food choices can be done over time using formats that would be compelling to potential consumers of such information. In addition, tracking choices to location could provide valuable insight into the types of foods and energy density of foods consumed at different types of fast food restaurants. Additional information to guide users could include an option for written recommendations for the current location (restaurant) and let other users read (and give their) the recommendations about the menu items or how easy/difficult it was to make healthy choices at a particular location.

3.2 Studies on Existing System

So far, my research has found only one system that acts like a food recommender system.

u-BabSang - context-aware food recommendation system for well-being care applications.

It recommends appropriate foods in real time to each individual by utilizing each individual's profile, physiological signals, and sensed environmental information. The advantage of the system is that it enables the management of personalized well-being care by integrating user contexts with other contexts during run time, and interpreting these contexts in a user centric manner. More specifically, by analyzing their personal situation (e.g., physiological status, eating custom, weather, location, etc.), the system can monitor the health of individuals with specific clinical history.

The system has the potential to be extended to various other fields. For instance, this system can be extended to menus besides Korean menus. In particular, it can recommend food types and quantities based on the user’s food preference, constitution, clinical history, and current climate conditions. Also, if the system is embed into personal devices (e.g., UMPC, mobile phone, etc.), the system can be extended to recommend appropriate food items at point of purchase, e.g., supermarkets, restaurants, and other public places. In this way, users will be able to protect personal information while accessing recommended products (e.g., foodstuffs, groceries, merchandise, etc.). Additionally, the system could be utilized by athletics or medical services.

The system needs to be evaluated in real environments such as homes, restaurants, and stores. After obtaining the user study, the convenience of using sensors and display on the table can be improved. Another possible area to explore is to track and integrate the history of users’ eating habits.

3.3 Software Approach

3.3.1 Java

Java is a general-purpose computer programming language that is concurrent, class-based, object-oriented,] and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that code that runs on one platform does not need to be recompiled to run on another. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of computer architecture. Java is, as of 2014, one of the most popular programming languages in use, particularly for client-server web applications, with a reported 9 million developers. Java was originally developed by James Gosling at Sun Microsystems (which has since merged into Oracle Corporation) and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.

The original and reference implementation Java compilers, virtual machines, and class libraries were originally released by Sun under proprietary licenses. As of May 2007, in compliance with the specifications of the Java Community Process, Sun relicensed most of its Java technologies under the GNU General Public License. Others have also developed alternative implementations of these Sun technologies, such as the GNU Compiler for Java (bytecode compiler), GNU Classpath (standard libraries), and IcedTea-Web (browser plugin for applets).

3.3.2 MySQL

MySQL is (as of March 2014) the world's second most widely used open-source relational database management system (RDBMS). It is named after co-founder Michael Widenius's daughter, My. The SQL phrase stands for Structured Query Language.

The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. MySQL was owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, now owned by Oracle Corporation.

MySQL is a popular choice of database for use in web applications, and is a central component of the widely used LAMP open source web application software stack (and other 'AMP' stacks). LAMP is an acronym for "Linux, Apache, MySQL, Perl/PHP/Python." Free-software-open source projects that require a full-featured database management system often use MySQL.

For proprietary use, several paid editions are available, and offer additional functionality. Applications which use MySQL databases include: TYPO3, MODx, Joomla, WordPress, phpBB, MyBB, Drupal and other software. MySQL is also used in many high-profile, large-scale websites, including Google (though not for searches), Facebook, Twitter, Flickr, and YouTube.

3.3.3 Netbeans

NetBeans is an integrated development environment (IDE) for developing primarily with Java, but also with other languages, in particular PHP, C/C++, and HTML5. It is also an application platform framework for Java desktop applications and others.

The NetBeans IDE is written in Java and can run on Windows, OS X, Linux, Solaris and other platforms supporting a compatible JVM.

The NetBeans Platform allows applications to be developed from a set of modular software components called *modules*. Applications based on the NetBeans Platform (including the NetBeans IDE itself) can be extended by third party developers.

The NetBeans Team actively supports the product and seeks feature suggestions from the wider community. Every release is preceded by a time for Community testing and feedback.

3.3.4 PHP

PHP is a server-side scripting language designed for web development but also used as a general-purpose programming language. As of January 2013, PHP was installed on more than 240 million websites (39% of those sampled) and 2.1 million web servers. Originally created by Rasmus Lerdorf in 1994, the reference implementation of PHP (powered by the Zend Engine) is now produced by The PHP Group. While PHP originally stood for *Personal Home Page*, it now stands for *PHP: Hypertext Preprocessor*, which is a recursive backronym.

PHP code can be simply mixed with HTML code, or it can be used in combination with various templating engines and web frameworks. PHP code is usually processed by a PHP interpreter, which is usually implemented as a web server's native module or a Common Gateway Interface (CGI) executable. After the PHP code is interpreted and executed, the web server sends resulting output to its client, usually in form of a part of the generated web page – for example, PHP code can generate a web page's HTML code, an image, or some other data. PHP has also evolved to include a command-line interface (CLI) capability and can be used in standalone graphical applications.

The canonical PHP interpreter, powered by the Zend Engine, is free software released under the PHP License. PHP has been widely ported and can be deployed on most web servers on almost every operating system and platform, free of charge.

3.3.5 XAMPP

XAMPP is a free and open source cross-platform web server solution stack package, consisting mainly of the Apache HTTP Server, MySQL database, and interpreters for scripts written in the PHP and Perl programming languages.

XAMPP's name is an acronym for:

- X (to be read as "cross", meaning cross-platform)
- Apache HTTP Server
- MySQL
- PHP
- Perl

CHAPTER 4

4.1 Overall System Design

The system will basically consist of four components:

1. User Profiling
2. Food Profiling
3. Database Operations
4. Recommendations using appropriate algorithms

4.1.1 User Profiling

The user profiles will be created by keeping a data of each user, his preferences and the recommendations which he found suitable.

The interface will be designed in Java with the help of applet tools. The aim is to develop a user friendly and self-explanatory interface to facilitate the process of creation of user profiles.

After the registration in the system, a set of random ingredients are asked to the user in order to be rated from 0 to 5. If an ingredient has already been rated before, the new rating will be considered in the next logins after registration. I made such an estimation that if the rating is smaller than 3, the user does not like eating that ingredient. Therefore, the ratings smaller than 3 are considered as negative ratings, the ratings equal to 3 are considered as neutral ratings, and the remaining (4, 5) are considered as positive ratings. Briefly, user profile vectors are constructed with the positively rated ingredients.

The type of the user profile derived by a content-based recommender depends on the learning method employed. Decision trees, neural nets, and vector-based representations can be all used. In this project I propose to use decision vector based representations constructed with the help of user ratings. At this step, my system uses explicit data collection. Specifically, after each recommendation, user can explicitly state whether the recommendation is satisfying or not. The next recommendations will be mostly based on this user feedback.

Decision tree learning uses a decision tree as a predictive model which maps observations about an item to conclusions about the item's target value. It is one of the predictive modeling approaches used in statistics, data mining and machine learning. Tree models where the target variable can take a finite set of values are called **classification trees**. In these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels. Decision trees where the target variable can take continuous values (typically real numbers) are called **regression trees**.

In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. In data mining, a decision tree describes data but not decisions; rather the resulting classification tree can be an input for decision making.

In machine learning, **support vector machines** (**SVMs**, also **support vector networks**) are supervised learning models with associated learning algorithms that analyze data and recognize patterns, used for classification and regression analysis. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples into one category or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on.

In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

4.1.2 Food Profiling

In content based recommendation systems, every item is represented by a set of features or an attribute profile. A variety of distance measures between the feature vectors may be used to compute the similarity of two items. For example Euclidian or cosine similarity supposes that all the features have equal importance. However, human judgment of similarity between two items often gives different weights to different attributes. Moreover, document frequency is more commonplace to use for this purpose.

$$\mathrm{idf}_t = \log \frac{N}{\mathrm{df}_t}.$$

Here,
N denotes for the total number of foods in a collection, and df_t is for the total number of foods that have the ingredient “t”. Thus the idf of a rare term is high, whereas the idf of a frequent term is likely to be low. This method is called “inverse document frequency” and I propose to assign the calculated similarity as weights of the features.

Euclidean Metric

In mathematics, the **Euclidean distance** or **Euclidean metric** is the "ordinary" distance between two points in Euclidean space. With this distance, Euclidean space becomes a metric space. The associated norm is called the **Euclidean norm.** Older literature refers to the metric as **Pythagorean metric**.

The **Euclidean distance** between point’s **p** and **q** is the length of the line segment connecting them ($\overline{\mathbf{pq}}$).

In Cartesian coordinates, if **p** = (p_1, p_2,..., p_n) and **q** = (q_1, q_2,..., q_n) are two points in Euclidean *n*-space, then the distance (d) from **p** to **q**, or from **q** to **p** is given by the Pythagorean formula:

$$\mathrm{d}(\mathbf{p},\mathbf{q}) = \mathrm{d}(\mathbf{q},\mathbf{p}) = \sqrt{(q_1-p_1)^2 + (q_2-p_2)^2 + \cdots + (q_n-p_n)^2}$$

$$= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}.$$

The position of a point in a Euclidean *n*-space is a Euclidean vector. So, **p** and **q** are Euclidean vectors, starting from the origin of the space, and their tips indicate two points. The **Euclidean norm**, or **Euclidean length**, or **magnitude** of a vector measures the length of the vector:

$$\|\mathbf{p}\| = \sqrt{p_1^2 + p_2^2 + \cdots + p_n^2} = \sqrt{\mathbf{p} \cdot \mathbf{p}}$$

where the last equation involves the dot product.

A vector can be described as a directed line segment from the origin of the Euclidean space (vector tail), to a point in that space (vector tip). If we consider that its length is actually the distance from its tail to its tip, it becomes clear that the Euclidean norm of a vector is just a special case of Euclidean distance: the Euclidean distance between its tail and its tip.

The distance between points **p** and **q** may have a direction (e.g. from **p** to **q**), so it may be represented by another vector, given by

$$\mathbf{q} - \mathbf{p} = (q_1 - p_1, q_2 - p_2, \cdots, q_n - p_n)$$

In a three-dimensional space (*n*=3), this is an arrow from **p** to **q**, which can be also regarded as the position of **q** relative to **p**. It may be also called a displacement vector if **p** and **q** represent two positions of the same point at two successive instants of time.

The Euclidean distance between **p** and **q** is just the Euclidean length of this distance (or displacement) vector:

$$\|\mathbf{q} - \mathbf{p}\| = \sqrt{(\mathbf{q} - \mathbf{p}) \cdot (\mathbf{q} - \mathbf{p})}.$$

which is equivalent to equation 1, and also to:

$$\|\mathbf{q} - \mathbf{p}\| = \sqrt{\|\mathbf{p}\|^2 + \|\mathbf{q}\|^2 - 2\mathbf{p} \cdot \mathbf{q}}.$$

Term frequency Inverse Document Frequency

tf–idf, short for **term frequency–inverse document frequency**, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. It is often used as a weighting factor in information retrieval and text mining. The tf-idf value increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus, which helps to adjust for the fact that some words appear more frequently in general.

Variations of the tf–idf weighting scheme are often used by search engines as a central tool in scoring and ranking a document's relevance given a user query. tf–idf can be successfully used for stop-words filtering in various subject fields including text summarization and classification.

One of the simplest ranking functions is computed by summing the tf–idf for each query term; many more sophisticated ranking functions are variants of this simple model.

tf–idf is the product of two statistics, term frequency and inverse document frequency. Various ways for determining the exact values of both statistics exist. In the case of the **term frequency** tf (*t*, *d*), the simplest choice is to use the *raw frequency* of a term in a document, i.e. the number of times that term *t* occurs in document t*d*. If we denote the raw frequency of *t* by f(*t*, *d*), then the simple tf scheme is tf(*t* ,*d*) = f(*t* ,*d*). Other possibilities include

- Boolean "frequencies": tf(*t*, *d*) = 1 if *t* occurs in *d* and 0 otherwise;
- logarithmically scaled frequency: tf(*t*, *d*) = 1 + log f(*t*, *d*), or zero if f(*t*, *d*) is zero;
- augmented frequency, to prevent a bias towards longer documents, e.g. raw frequency divided by the maximum raw frequency of any term in the document:

$$\mathrm{tf}(t,d) = 0.5 + \frac{0.5 \times \mathrm{f}(t,d)}{\max\{\mathrm{f}(w,d) : w \in d\}}$$

The **inverse document frequency** is a measure of how much information the word provides, that is, whether the term is common or rare across all documents. It is the logarithmically scaled fraction of the documents that contain the word, obtained by dividing the total number of documents by the number of documents containing the term, and then taking the logarithm of that quotient.

$$\mathrm{idf}(t,D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

with

- N: total number of documents in the corpus
- $|\{d \in D : t \in d\}|$: number of documents where the term t appears (i.e., $\mathrm{tf}(t,d) \neq 0$). If the term is not in the corpus, this will lead to a division-by-zero. It is therefore common to adjust the denominator to $1 + |\{d \in D : t \in d\}|$.

Mathematically the base of the log function does not matter and constitutes a constant multiplicative factor towards the overall result.

Then tf–idf is calculated as

$$\mathrm{tfidf}(t,d,D) = \mathrm{tf}(t,d) \times \mathrm{idf}(t,D)$$

A high weight in tf–idf is reached by a high term frequency (in the given document) and a low document frequency of the term in the whole collection of documents; the weights hence tend to filter out common terms. Since the ratio inside the idf's log function is always greater than or equal to 1, the value of idf (and tf-idf) is greater than or equal to 0.

Cosine Similarity

Cosine similarity is a measure of similarity between two vectors of an inner product space that measures the cosine of the angle between them. The cosine of 0° is 1, and it is less than 1 for any other angle. It is thus a judgment of orientation and not magnitude: two vectors with the same orientation have a Cosine similarity of 1, two vectors at 90° have a similarity of 0, and two vectors diametrically opposed have a similarity of -1, independent of their magnitude. Cosine similarity is particularly used in positive space, where the outcome is neatly bounded in [0, 1].

These bounds apply for any number of dimensions, and Cosine similarity is most commonly used in high-dimensional positive spaces. For example, in Information Retrieval and text mining, each term is notionally assigned a different dimension and a document is characterized by a vector where the value of each dimension corresponds to the number of times that term appears in the document. Cosine similarity then gives a useful measure of how similar two documents are likely to be in terms of their subject matter.

The cosine of two vectors can be derived by using the Euclidean dot product formula:

$$\mathbf{a} \cdot \mathbf{b} = \|\mathbf{a}\| \, \|\mathbf{b}\| \cos\theta$$

Given two vectors of attributes, *A* and *B*, the cosine similarity, *cos(θ)*, is represented using a dot product and magnitude as

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum\limits_{i=1}^{n} A_i \times B_i}{\sqrt{\sum\limits_{i=1}^{n} (A_i)^2} \times \sqrt{\sum\limits_{i=1}^{n} (B_i)^2}}$$

The resulting similarity ranges from −1 meaning exactly opposite, to 1 meaning exactly the same, with 0 usually indicating independence, and in-between values indicating intermediate similarity or dissimilarity.

For text matching, the attribute vectors *A* and *B* are usually the term frequency vectors of the documents. The cosine similarity can be seen as a method of normalizing document length during comparison.

In the case of information retrieval, the cosine similarity of two documents will range from 0 to 1, since the term frequencies (tf-idf weights) cannot be negative. The angle between two term frequency vectors cannot be greater than 90°.

If the attribute vectors are normalized by subtracting the vector means (e.g., $A - \bar{A}$), the measure is called centered cosine similarity and is equivalent to the Pearson Correlation Coefficient.

Example:

Here are two very short texts to compare:

Text 1: Julie loves me more than Linda loves me

Text 2: Jane likes me more than Julie loves me

We want to know how similar these texts are, purely in terms of word counts (and ignoring word order). We begin by making a list of the words from both texts:

me Julie loves Linda than more likes Jane

Now we count the number of times each of these words appears in each text:

me 2 2

Julie 1 1

likes 0 1

loves 2 1

Jane 0 1

Linda 1 0

than 1 1

more 1 1

We are not interested in the words themselves though. We are interested only in those two vertical vectors of counts. For instance, there are two instances of 'me' in each text. We are going to decide how close these two texts are to each other by calculating one function of those two vectors, namely the cosine of the angle between them.

The two vectors are, again:

a: [2, 1, 0, 2, 0, 1, 1, 1]

b: [2, 1, 1, 1, 1, 0, 1, 1]

The cosine of the angle between them is about 0.822.

These vectors are 8-dimensional. A virtue of using cosine similarity is clearly that it converts a question that is beyond human ability to visualize to one that can be. In this case you can think of this as the angle of about 35 degrees which is some 'distance' from zero or perfect agreement.

4.1.3 Database Operations

Information retrieval (IR) is a data search technology which includes crawling, processing and indexing of content, and querying for content. Web crawling and HTML parsing has been done to create the dataset. Web crawling is the process by which we gather pages from the Web, in order to index them and support a search engine. The objective of crawling is to quickly and efficiently gather as many useful web pages as possible, together with the link structure that interconnects them. The code will connect to certain food website on the web; it will firstly extract the main food types. Keeping them in a text file, it will then extract the restaurants related to each main food type. Those restaurants will be stored in a database table. Lastly, the meals with their ingredients will be extracted. They will be stored in another database table. The meals are also parsed into their ingredients. Another database table is created for the ingredients and their calculated weights.

Crawler:

A **Web crawler** is an Internet bot that systematically browses the World Wide Web, typically for the purpose of Web indexing. A Web crawler may also be called a **Web spider**, an **ant**, an **automatic indexer**, or (in the FOAF software context) a **Web scutter**.

Web search engines and some other sites use Web crawling or spidering software to update their web content or indexes of others sites' web content. Web crawlers can copy all the pages they visit for later processing by a search engine that indexes the downloaded pages so that users can search them much more quickly.

Crawlers can validate hyperlinks and HTML code. They can also be used for web scraping (see also data-driven programming).

A Web crawler starts with a list of URLs to visit, called the *seeds*. As the crawler visits these URLs, it identifies all the hyperlinks in the page and adds them to the list of URLs to visit, called the *crawl frontier*. URLs from the frontier are recursively visited according to a set of policies. If the crawler is performing archiving of websites it copies and saves the information as it goes. Such archives are usually stored such that they can be viewed, read and navigated as they were on the live web, but are preserved as ‘snapshots'.

The large volume implies that the crawler can only download a limited number of the Web pages within a given time, so it needs to prioritize downloads. The high rate of change implies that the pages might have already been updated or even deleted.

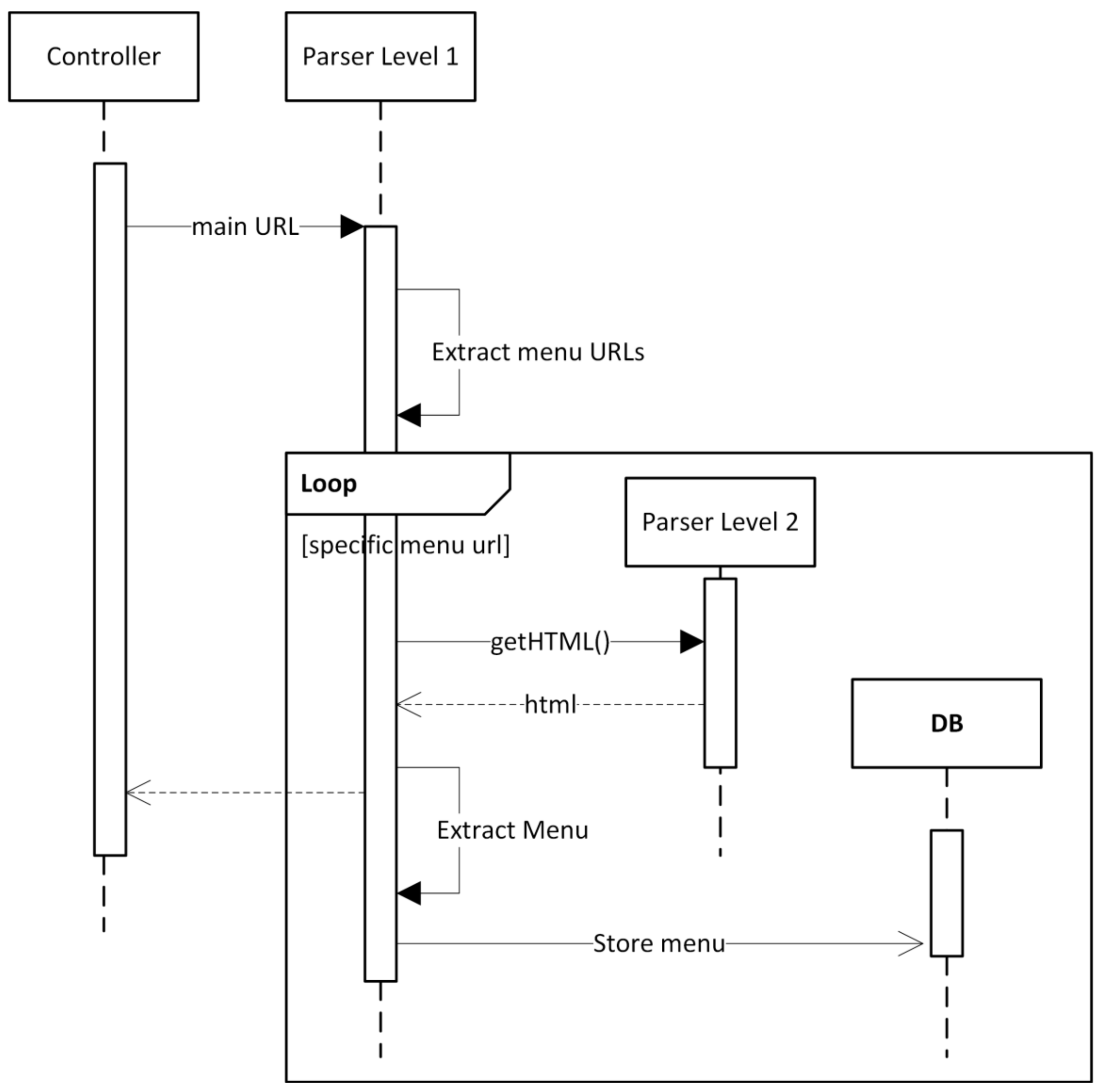
The number of possible URLs crawled being generated by server-side software has also made it difficult for web crawlers to avoid retrieving duplicate content. Endless combinations of HTTP GET (URL-based) parameters exist, of which only a small selection will actually return unique content. For example, a simple online photo gallery may offer three options to users, as specified through HTTP GET parameters in the URL. If there exist four ways to sort images, three choices of thumbnail size, two file formats, and an option to disable user-provided content, then the same set of content can be accessed with 48 different URLs, all of which may be linked on the site. This mathematical combination creates a problem for crawlers, as they must sort through endless combinations of relatively minor scripted changes in order to retrieve unique content.

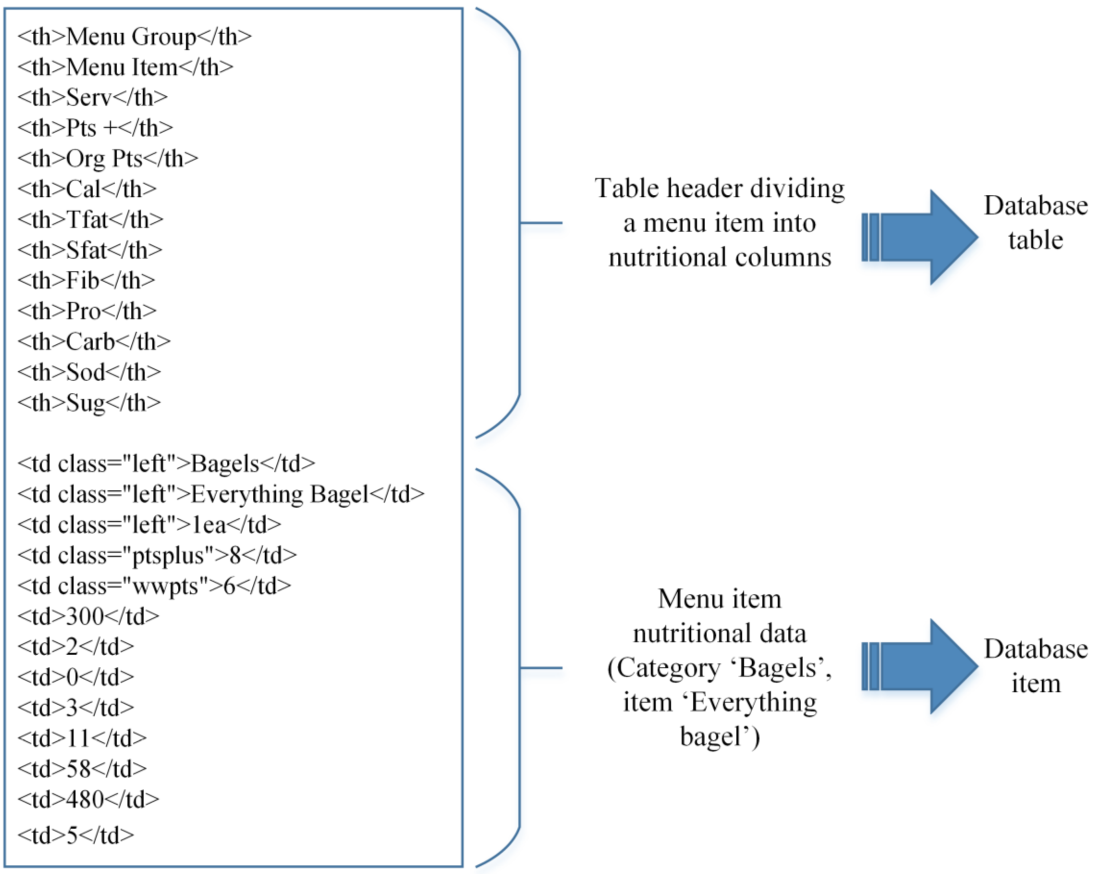
As Edwards *et al.* noted, "Given that the bandwidth for conducting crawls is neither infinite nor free, it is becoming essential to crawl the Web in not only a scalable, but efficient way, if some reasonable measure of quality or freshness is to be maintained." A crawler must carefully choose at each step which pages to visit next.

Parser:

A **parser** is a software component that takes input data (frequently text) and builds a data structure – often some kind of parse tree, abstract syntax tree or other hierarchical structure – giving a structural representation of the input, checking for correct syntax in the process. The parsing may be preceded or followed by other steps, or these may be combined into a single step. Parsers may be programmed by hand or may be automatically or semi-automatically generated by a parser generator. Parsing is complementary to templating, which produces formatted output. These may be applied to different domains, but often appear together, such as the scanf/printf pair, or the input (front end parsing) and output (back end code generation) stages of a compiler.

The input to a parser is often text in some computer language, but may also be text in a natural language or less structured textual data, in which case generally only certain parts of the text are extracted, rather than a parse tree being constructed. Parsers range from very simple functions such as scanf, to complex programs such as the frontend of a C++ compiler or the HTML parser of a web browser. The use of parsers varies by input. In the case of data languages, a parser is often found as the file reading facility of a program, such as reading in HTML or XML text; these examples are markup languages.


Controller
Parser Level 1
main URL
Extract menu URLs
Loop
[specific menu url]
Parser Level 2
getHTML()
html
DB
Extract Menu
Store menu


```
<th>Menu Group</th>
<th>Menu Item</th>
<th>Serv</th>
<th>Pts +</th>
<th>Org Pts</th>
<th>Cal</th>
<th>Tfat</th>
<th>Sfat</th>
<th>Fib</th>
<th>Pro</th>
<th>Carb</th>
<th>Sod</th>
<th>Sug</th>

<td class="left">Bagels</td>
<td class="left">Everything Bagel</td>
<td class="left">1ea</td>
<td class="ptsplus">8</td>
<td class="wwpts">6</td>
<td>300</td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>11</td>
<td>58</td>
<td>480</td>
<td>5</td>
```
Table header dividing a menu item into nutritional columns
Database table
Menu item nutritional data (Category 'Bagels', item 'Everything bagel')
Database item

4.1.4 Recommendations using appropriate algorithms

My system is a content based recommendation system. Food domain can be seen as a set of foods where each food has a set of ingredients. Content-based approaches treat the recommendation problem as a search for related items. Given a rated food, the algorithm constructs a search to find other related items with the same ingredient.

If a user likes salmon, for example the system might recommend other foods having salmon like sushi. However, this is the simplest logic behind content-based recommendation systems. In my system foods are defined by their important features and represented by vectors. Thus, feature weights are crucial in these vectors. Similarity is computed based on item attributes using appropriate distance measures.

Content-based recommendation systems share in common describing the items that may be recommended, creating a profile of the user that describes the types of items the user likes, and comparing items to the user profile to determine what to recommend.

The recommendations are done by calculating the foods which are most similar to the profile of the user. However, only the foods which have positively rated ingredients are considered in this similarity process. Moreover, users can evaluate the recommendations by stating whether they liked the recommendations or not. The assumption has been made that the user likes the recommendation according to the restaurant in which that recommended food can be eaten. This part is the user feedback step of my recommendation system. After these evaluations, the next recommendations are built on both personally liked (positively rated) ingredients and the previously liked recommendations, which are affects the similarity score positively.

4.2 Operating Environment

OS: Windows XP (SP2)/Vista/7/8, Linux, MacOS
RAM: 2GB or more
Processor: Intel Core 2 Duo and above, AMD Athlon Barton and above

- **Specific technologies:**
  + MySQL.
  + Java.
  + PHP.

- **Software Required**
  + NetBeans 8.1.
  + MySQL

- **Language requirements:**
  + Support English.

CHAPTER 5

5.1 Testing

A. Pre testing phase

During the design and implementation stage of the system, a questionnaire was designed to “pre-test” the concepts underlying the recommendation system. Pre-test questions were developed to elicit information about consumers notions about making healthy choices in fast food restaurants, and whether and what type of nutrition information or labeling might help consumers select specific foods. Ten volunteers answered questions related to healthy eating at fast food restaurants and nutrition labeling, and findings were used to refine the apps structure, features, and function and to inform the procedures and questions used during the beta testing phase.

B. Beta-Testing Phase

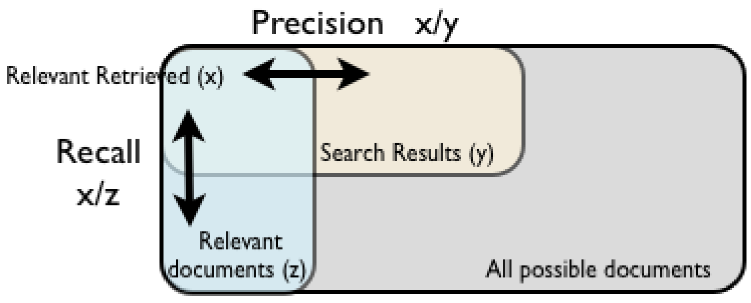
The application was beta-tested with ten users following an established protocol. Each user was asked to 1) open the application, 2) choose a restaurant based on their location, and 3) choose a menu item from the list. Immediately following the beta test, face-to-face interviews were conducted with ten individuals using a semi-structured script. The script was designed to include open-ended questions and data were analyzed per standard qualitative data methods. Following pre-testing, feedback from the ten volunteers was used to make final adjustments to the application. These adjustments were minor, and primarily related to size and location of specific features and graphics.

CHAPTER 6

6.1 Evaluation of the system

In recommender systems, for the final the user the most important result is to receive an ordered list of recommendations, from best to worst. In fact, in some cases the user doesn't care much about the exact ordering of the list - a set of few good recommendations is fine. Taking this fact into evaluation of recommender systems, we could apply classic information retrieval metrics to evaluate those engines: Precision and Recall. These metrics are widely used on information retrieving scenario and applied to domains such as search engines, which return some set of best results for a query out of many possible results.

For a search engine for example, it should not return irrelevant results in the top results, although it should be able to return as many relevant results as possible. We could say that the 'Precision' is the proportion of top results that are relevant, considering some definition of relevant for your problem domain. So if we say 'Precision at 10' would be this proportion judged from the top 10 results. The 'Recall' would measure the proportion of all relevant results included in the top results. See the Figure 1 as an example to illustrate those metrics.



Precision and Recall in the context of Search Engines

In a formal way, we could consider documents as instances and the task it to return a set of relevant documents given a search term. So the task would be assigning each document to one of two categories: relevant and not relevant. Recall is defined as the number of relevant documents (the instances that belongs to relevant category) retrieved by a search divided by the total number of existing relevant documents, while precision is defined as the number of relevant documents (the instances that belongs to relevant category) retrieved by a search divided by the total number of documents retrieved by the search.

In recommender systems those metrics could be adapted so:

"The precision is the proportion of recommendations that are good recommendations, and recall is the proportion of good recommendations that appear in top recommendations."

In recommendations domain, a perfect precision score of 1.0 means that every item recommended in the list was good (although says nothing about if all good recommendations were suggested) whereas a perfect recall score of 1.0 means that all good recommended items were suggested in the list (although says nothing how many bad recommendations were also in the list).

CHAPTER 7

7.1 Conclusion

To sum up, designing this system was a very useful exercise and I believe that the literature survey I have done was very useful for me in order to understand the logic behind the recommendation systems. I have also searched about some rule engines that I want to use in this project.

Moreover, implementing such a project demonstrated how a knowledge based system can actually encapsulate knowledge. It also showed the limitations of knowledge based systems and the assumptions that must be made when designing them. As an instance for these assumptions, I have assumed that the user likes the recommendation according to its restaurant. Moreover, while finding the similarity between user profile and candidate item profiles, I am calculating a score with the weights and ratings. This calculation logic may also be leading to some failures. I should implement a more improved version of the algorithm for finding the similarity. Moreover, my system is static that if any other restaurant of meal is added, this does not affect my system. Dynamic knowledge usage can also be a good version in the future.

The proposed system is still in its initial testing and evaluation phases, but preliminary data suggest it can be used (and is useful) for members of the community. The prototype was functional, and users' evaluation suggested that the product is useful.

In addition, tracking choices to location could provide valuable insight into the types of foods and energy density of foods consumed at different types of fast food restaurants. Additional information to guide users could include an option for written recommendations for the current location (restaurant) and let other users read (and give their) the recommendations about the menu items or how easy/difficult it was to make healthy choices at a particular location.

Research suggests that changing dietary behavior does not involve unlearning old habits, but rather, learning new ones and that the context in which learning takes place (both physical and social) significantly influences the learning process. So applications designed to help individuals engage in specific dietary behaviors should acknowledge and address the learning context, providing critical information and feedback when and where eating behaviors are occurring.

The proposed system presents choices to the users. But, for now, the system is not aware of which option the user chooses and therefore it is not possible to verify if the proposed system is effective at motivating users to choose the healthier options. If the user's choice is recorded and verified against the options presented to the user or user's historical choices, it is then possible to analyze if/how the user is changing his behavior. However, there are privacy issues if such information is recorded. Thus user needs to be aware of the tracking by explicit requests. The same issue is valid for recording user's location over time or during the app usage.

Another topic of interest is the use of ratings, such as Yelp, Urbanspoon, etc. Such services offer recommendations based on user experience (food quality, overall restaurant evaluation, etc.). It'd be easy to incorporate ratings regarding food ranking (such as proposed by our project). By acquiring user's evaluation per restaurant, it is possible to give recommendations based on overall users' evaluation. To enable this, ratings could be displayed for each restaurant in the vicinity, highlighting restaurants that offer relatively more healthy options.

Another topic of research is the adaptation of suggestions to the specific users' profiles, for example, users with intolerance to gluten, milk, etc. In this case, each food item needs to be classified per health condition before offering the item evaluation as good or bad.

References

[1] Yoosoo Oh, Ahyoung Choi, WoontackWoo, "u-BabSang: a context-aware food recommendation system", © Springer Science + Business Media, LLC 2009

[2] Ipek Tatli, "Food Recommendation System," Middle East Technical University, Ankara, Turkey, June 2009

[3] Jill Freyne and Shlomo Berkovsky, "Intelligent Food Planning-Personalized Recipe Recommendation," In IUI (2010) 321-324

[4] Mayumi Ueda, Mari Takahata and Shinsuke Nakajima, "User's Food Preference Extraction for Personalized Cooking Recipe Recommendation", Kyoto University, Kyoto, Japan, 2011

[5] Michael Hahsler, "recommenderlab: A Framework for Developing and Testing Recommendation Algorithms," Southern Methodist University, Dallas, Texas (USA), January 2013

[6] Thienne Johnson, Jorge Vergara, Chelsea Doll, Madison Kramer, Gayathri Sundararaman, Harsha Rajendran, Alon Efrat and Melanie Hingle, "A Mobile Food Recommendation System Based on The Traffic Light Diet," University of Arizona, Tucson, AZ 85710, USA, 2014

[7] Recommender System: http://en.wikipedia.org/wiki/Recommender_system

[8] Content-based, Collaborative Recommendation: www.ischool.utexas.edu/~i385d/readings/Balabanovic_Fab_97.pdf

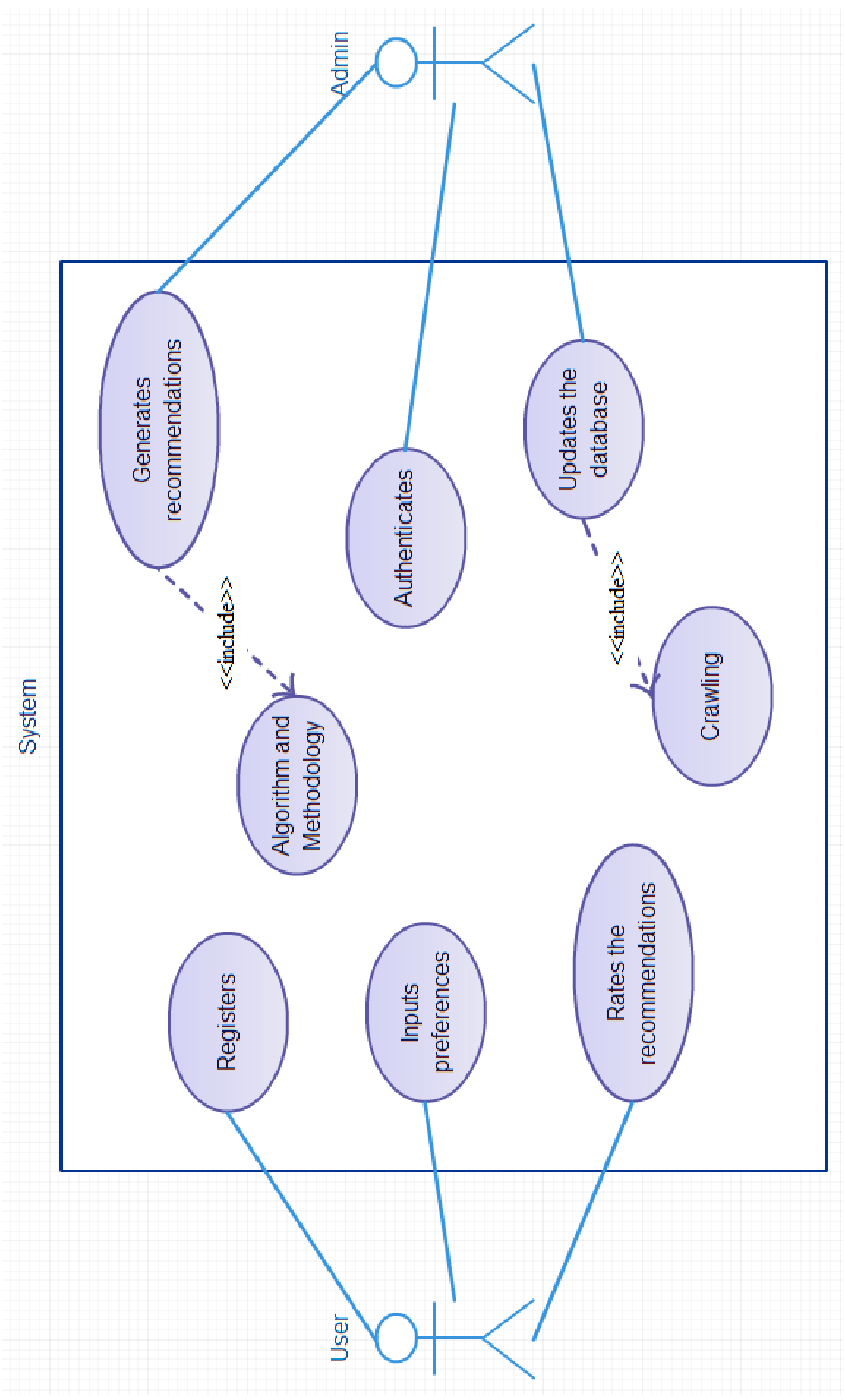
[9] Learning New User Preferences in Recommender Systems: http://www.grouplens.org/papers/pdf/voi-final.pdf

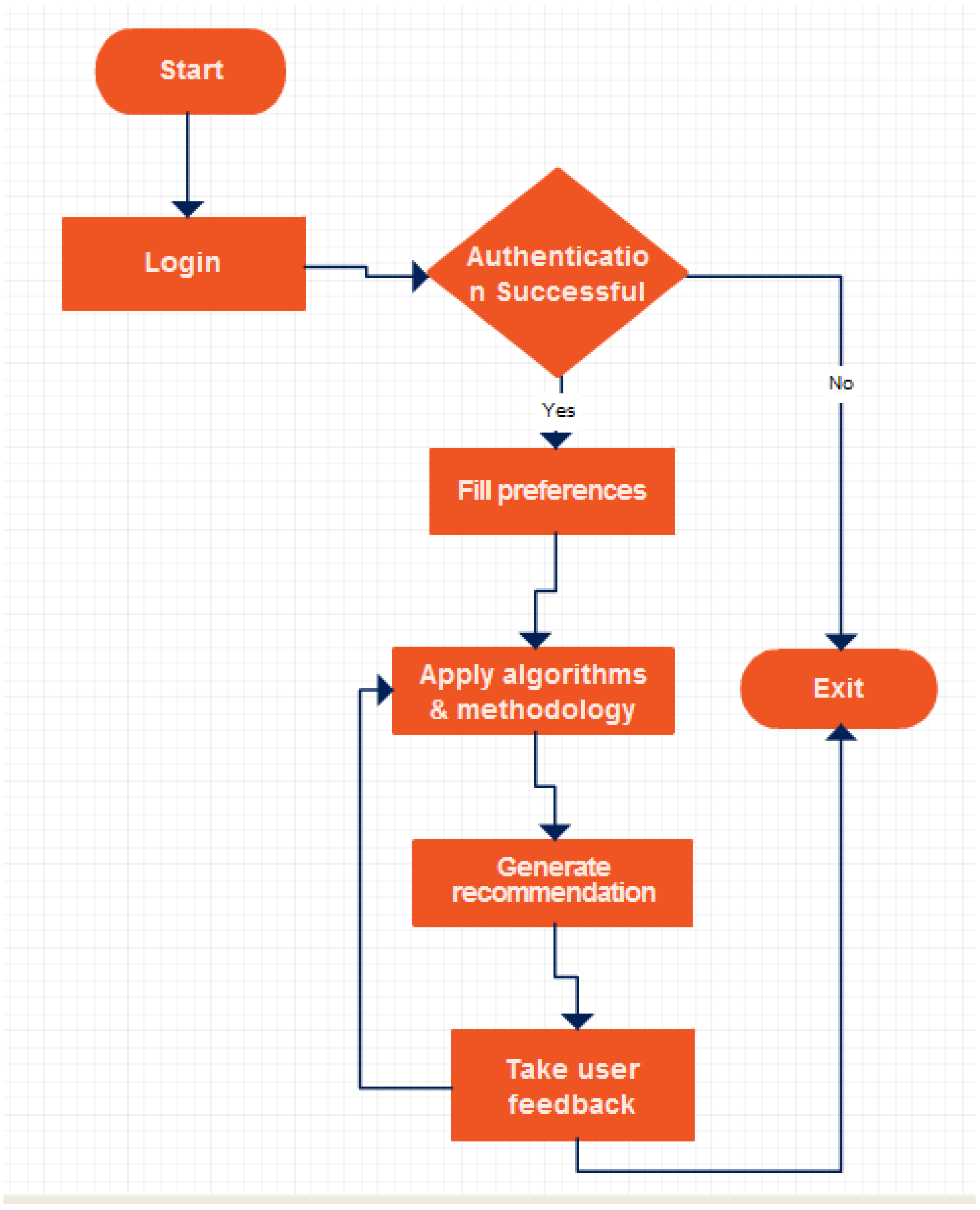
[10] Item-Based Collaborative Filtering Recommendation Algorithms: h ttp://www.groupl ens.org/papers/pdf/www10_sarwar.pdf

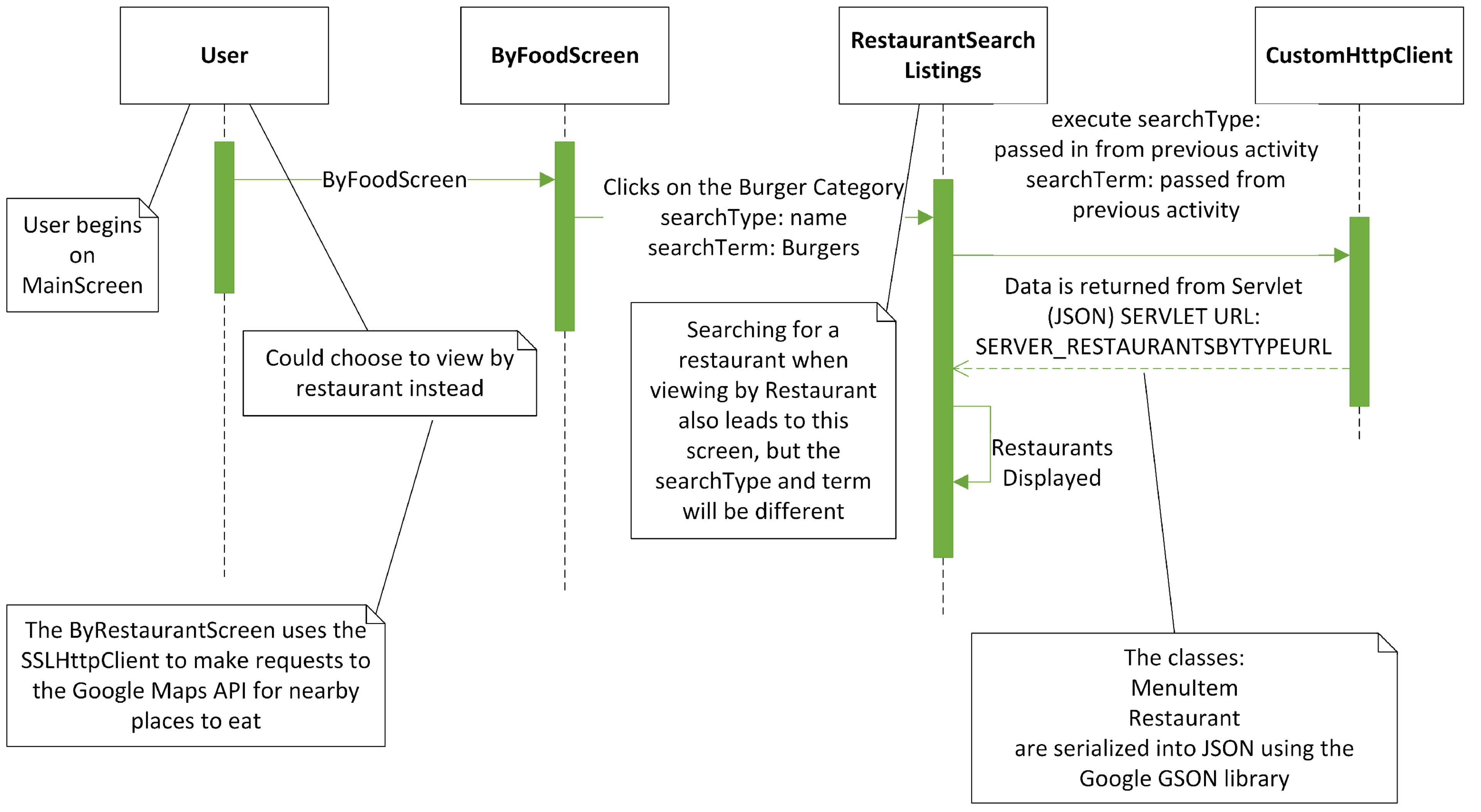
[11] Inverse Document Frequency, *http://nlp.stanford.edu/IR-book/html/htmledition/inversedocument-frequency-1.html*

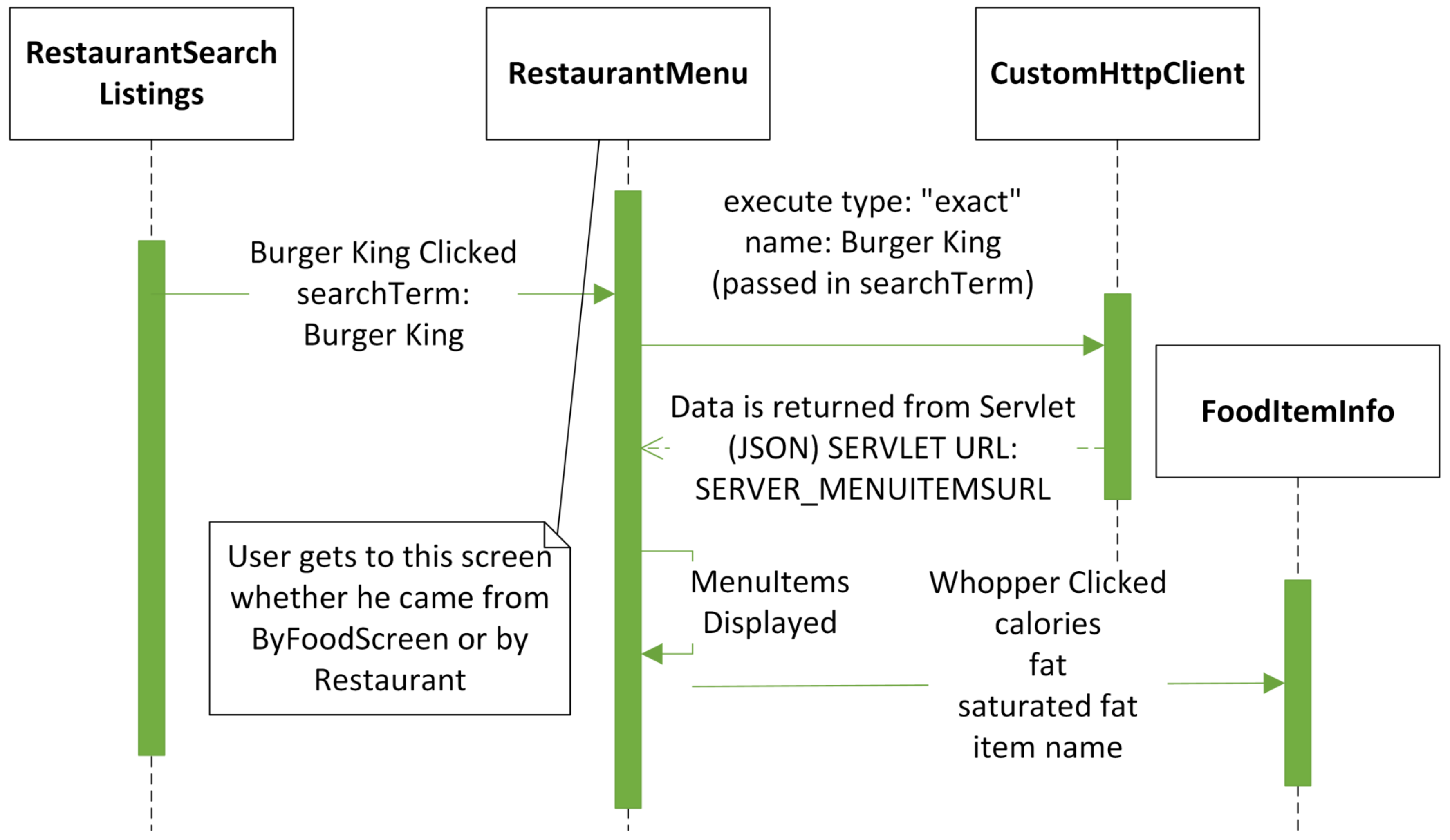
[12] Feature Weighting in Content Based Recommendation System Using Social Network Analysis http://www2008.org/papers/pdf/p1041-debnath.pdf

APPENDIX

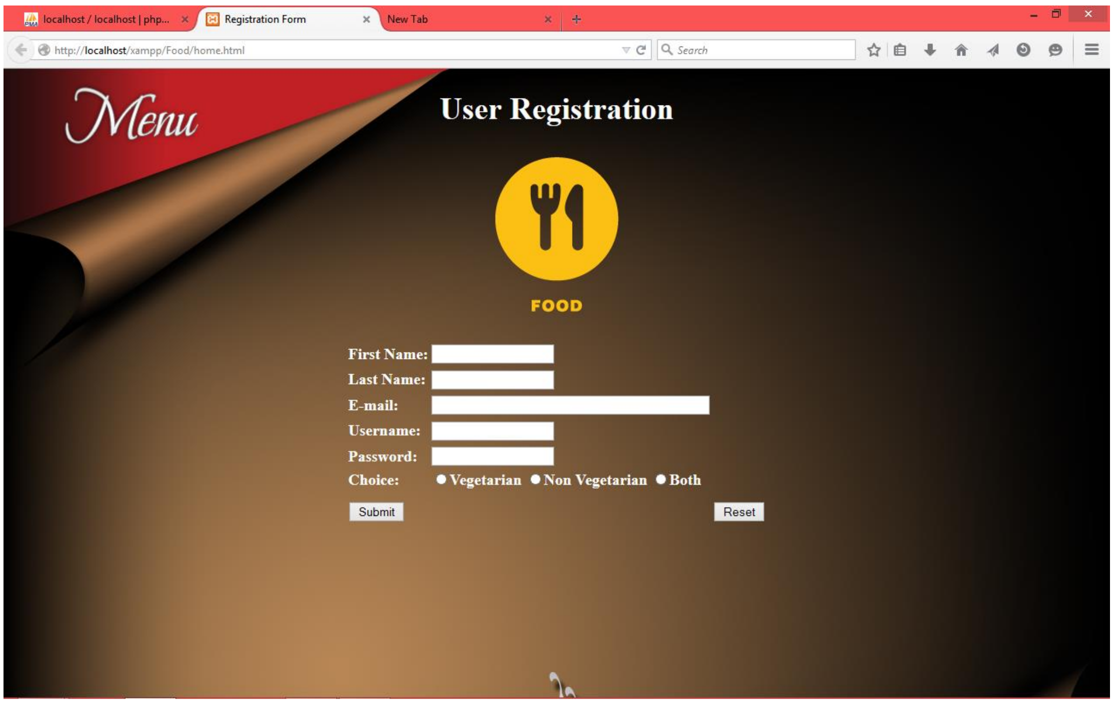

System
Admin
User
Generates recommendations
Authenticates
Updates the database
<<include>>
<<include>>
Algorithm and Methodology
Crawling
Registers
Inputs preferences
Rates the recommendations


Start
Login
Authenticatio n Successful
Yes
No
Fill preferences
Apply algorithms & methodology
Exit
Generate recommendation
Take user feedback


User
ByFoodScreen
RestaurantSearch Listings
CustomHttpClient
User begins on MainScreen
ByFoodScreen
Could choose to view by restaurant instead
The ByRestaurantScreen uses the SSLHttpClient to make requests to the Google Maps API for nearby places to eat
Clicks on the Burger Category searchType: name searchTerm: Burgers
Searching for a restaurant when viewing by Restaurant also leads to this screen, but the searchType and term will be different
execute searchType: passed in from previous activity searchTerm: passed from previous activity
Data is returned from Servlet (JSON) SERVLET URL: SERVER_RESTAURANTSBYTYPEURL
Restaurants Displayed
The classes: MenuItem Restaurant are serialized into JSON using the Google GSON library


RestaurantSearch Listings
RestaurantMenu
CustomHttpClient
FoodItemInfo
Burger King Clicked searchTerm: Burger King
execute type: "exact" name: Burger King (passed in searchTerm)
Data is returned from Servlet (JSON) SERVLET URL: SERVER_MENUITEMSURL
User gets to this screen whether he came from ByFoodScreen or by Restaurant
MenuItems Displayed
Whopper Clicked calories fat saturated fat item name

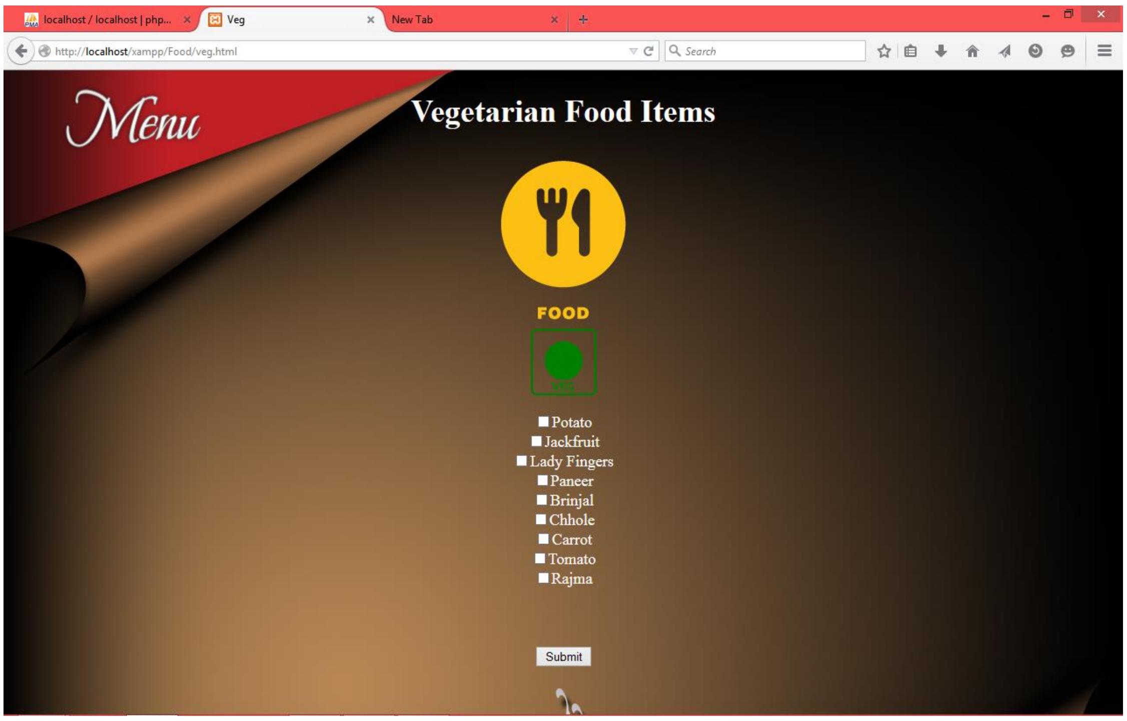
Screenshots:



localhost / localhost | php...
Registration Form
New Tab
http://localhost/xampp/Food/home.html
Search
Menu
FOOD
Submit
Reset

User Registration

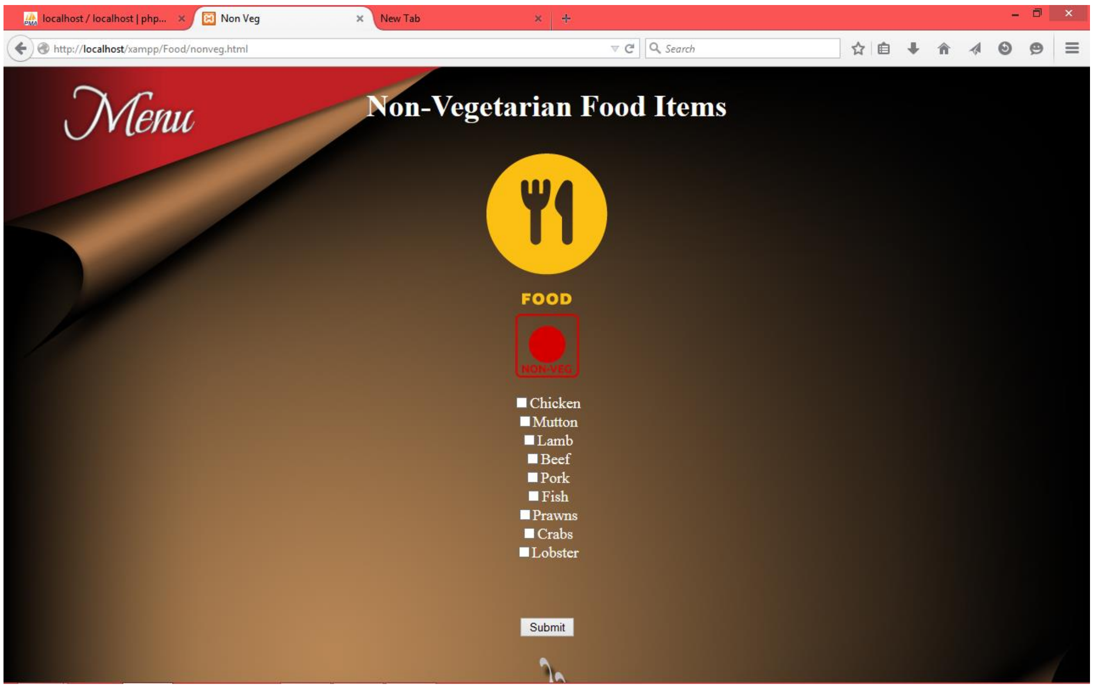
First Name:
Last Name:
E-mail:
Username:
Password:
Choice: ● Vegetarian ● Non Vegetarian ● Both



localhost / localhost | php...
Veg
New Tab
http://localhost/xampp/Food/veg.html
Search
Menu
FOOD
Submit

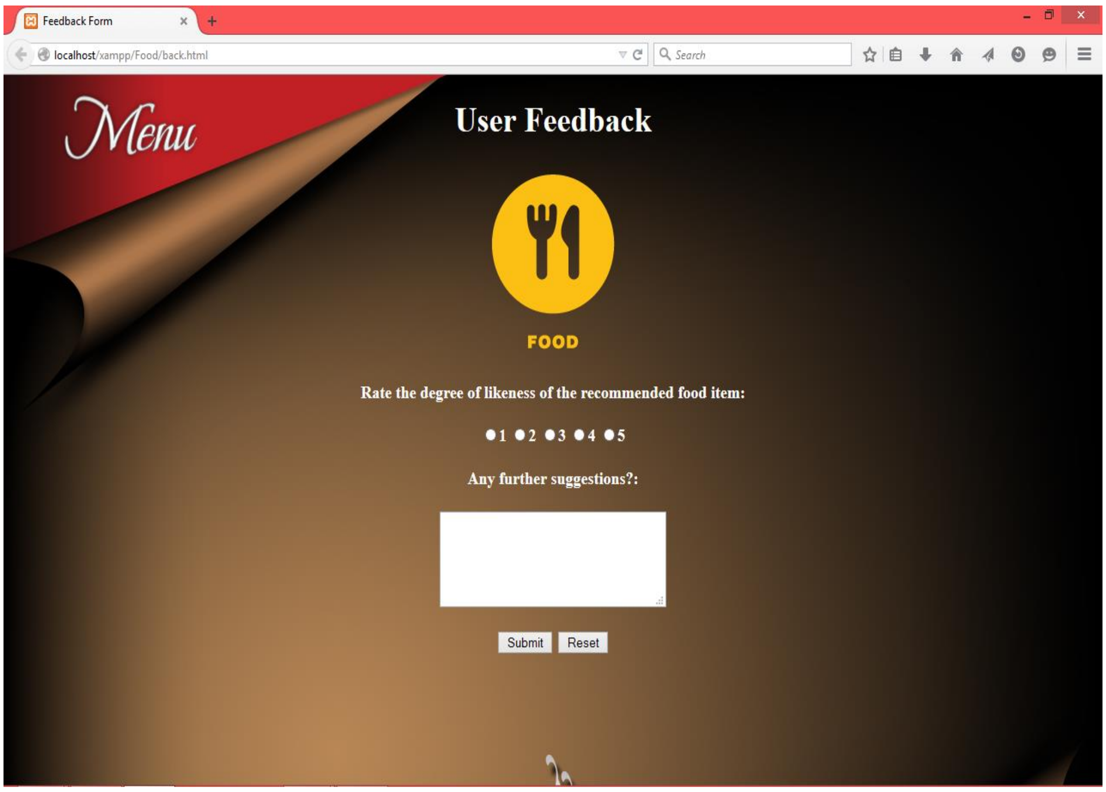
Vegetarian Food Items

- Potato
- Jackfruit
- Lady Fingers
- Paneer
- Brinjal
- Chhole
- Carrot
- Tomato
- Rajma


localhost / localhost | php...
Non Veg
New Tab
http://localhost/xampp/Food/nonveg.html
Search
Menu

Non-Vegetarian Food Items

- Chicken
- Mutton
- Lamb
- Beef
- Pork
- Fish
- Prawns
- Crabs
- Lobster



User Feedback

Rate the degree of likeness of the recommended food item:

● 1 ● 2 ● 3 ● 4 ● 5

Any further suggestions?:

Crawler Results:

```
http://simpleindianrecipes.com/asparagus.aspx
http://simpleindianrecipes.com/Basil.aspx
http://simpleindianrecipes.com/Beetroot.aspx
http://simpleindianrecipes.com/BellPeppers.aspx
http://simpleindianrecipes.com/Bittergourd.aspx
http://simpleindianrecipes.com/BokChoy.aspx
http://simpleindianrecipes.com/Bottleguard.aspx
http://simpleindianrecipes.com/Broccoli.aspx
http://simpleindianrecipes.com/BrusselsSprouts.aspx
http://simpleindianrecipes.com/Cabbage.aspx
http://simpleindianrecipes.com/Carrot.aspx
http://simpleindianrecipes.com/Cauliflower.aspx
http://simpleindianrecipes.com/Celery.aspx
http://simpleindianrecipes.com/Cilantro.aspx
http://simpleindianrecipes.com/Corn.aspx
http://simpleindianrecipes.com/Cucumber.aspx
http://simpleindianrecipes.com/CurryLeaves.aspx
http://simpleindianrecipes.com/Drumstick.aspx
http://simpleindianrecipes.com/Eggplant.aspx
http://simpleindianrecipes.com/Methi.aspx
http://simpleindianrecipes.com/Beans.aspx
http://simpleindianrecipes.com/GreenPeas.aspx
http://simpleindianrecipes.com/Greens.aspx
http://simpleindianrecipes.com/Ivygourd.aspx
http://simpleindianrecipes.com/LotusRoot.aspx
http://simpleindianrecipes.com/mint.aspx
http://simpleindianrecipes.com/Okra.aspx
http://simpleindianrecipes.com/Onion.aspx
http://simpleindianrecipes.com/Potato.aspx
http://simpleindianrecipes.com/Radish.aspx
http://simpleindianrecipes.com/Ridgegourd.aspx
http://simpleindianrecipes.com/Rutabaga.aspx
http://simpleindianrecipes.com/Scallions.aspx
http://simpleindianrecipes.com/Spinach.aspx
http://simpleindianrecipes.com/SquashVarieties.aspx
http://simpleindianrecipes.com/SweetPotato.aspx
http://simpleindianrecipes.com/Tapioca.aspx
http://simpleindianrecipes.com/Tomato.aspx
http://simpleindianrecipes.com/Turnip.aspx
http://simpleindianrecipes.com/yellowsquash.aspx
http://simpleindianrecipes.com/zuccini.aspx
http://simpleindianrecipes.com/apple.aspx
http://simpleindianrecipes.com/Avocado.aspx
http://simpleindianrecipes.com/Banana.aspx
http://simpleindianrecipes.com/berries.aspx
http://simpleindianrecipes.com/Cherry.aspx
http://simpleindianrecipes.com/Cranberry.aspx
http://simpleindianrecipes.com/Figs.aspx
http://simpleindianrecipes.com/Grapefruit.aspx
http://simpleindianrecipes.com/Grapes.aspx
http://simpleindianrecipes.com/Lemon.aspx
http://simpleindianrecipes.com/Lychee.aspx
http://simpleindianrecipes.com/Mango.aspx
http://simpleindianrecipes.com/Orange.aspx
http://simpleindianrecipes.com/RecipesByIngredients/Fruits/Papaya.aspx
http://simpleindianrecipes.com/Pineapple.aspx
http://simpleindianrecipes.com/Strawberry.aspx
http://simpleindianrecipes.com/Tamarind.aspx
```

http://simpleindianrecipes.com/Watermelon.aspx
http://simpleindianrecipes.com/RecipesByIngredients/Dairy.aspx
http://simpleindianrecipes.com/Cream.aspx
http://simpleindianrecipes.com/Curd.aspx
http://simpleindianrecipes.com/PaneerRecipes.aspx
http://simpleindianrecipes.com/Blackgram.aspx
http://simpleindianrecipes.com/Chikpea.aspx
http://simpleindianrecipes.com/GreenGram.aspx
http://simpleindianrecipes.com/millet.aspx
http://simpleindianrecipes.com/Oats.aspx
http://simpleindianrecipes.com/Semolina.aspx
http://simpleindianrecipes.com/Basil.aspx
http://simpleindianrecipes.com/Cilantro.aspx
http://simpleindianrecipes.com/CurryLeaves.aspx
http://simpleindianrecipes.com/mint.aspx
http://simpleindianrecipes.com/Scallions.aspx
http://simpleindianrecipes.com/Beef.aspx
http://simpleindianrecipes.com/Chicken.aspx
http://simpleindianrecipes.com/Crab.aspx
http://simpleindianrecipes.com/Egg.aspx
http://simpleindianrecipes.com/seafoodvarieties.aspx
http://simpleindianrecipes.com/Kheema.aspx
http://simpleindianrecipes.com/Mutton.aspx
http://simpleindianrecipes.com/Prawns.aspx
http://simpleindianrecipes.com/Quail.aspx
http://simpleindianrecipes.com/Turkey.aspx
http://simpleindianrecipes.com/almond.aspx
http://simpleindianrecipes.com/Cashew.aspx
http://simpleindianrecipes.com/Coconut.aspx
http://simpleindianrecipes.com/Peanut.aspx
http://simpleindianrecipes.com/Soya.aspx
http://simpleindianrecipes.com/Sprouts.aspx
http://simpleindianrecipes.com/Mushroom.aspx
http://simpleindianrecipes.com/Tofu.aspx
http://simpleindianrecipes.com/Specials.aspx
http://simpleindianrecipes.com/Specials/DietWithRestrictions.aspx
http://simpleindianrecipes.com/DiabeticRecipes.aspx
http://simpleindianrecipes.com/dietrecipes.aspx
http://simpleindianrecipes.com/veganrecipes.aspx
http://simpleindianrecipes.com/chaats.aspx
http://simpleindianrecipes.com/sandwich.aspx
http://simpleindianrecipes.com/breakfastvarieties.aspx
http://simpleindianrecipes.com/idlidosavarieties.aspx
http://simpleindianrecipes.com/pasta.aspx
http://simpleindianrecipes.com/PorridgeVarieties.aspx
http://simpleindianrecipes.com/rotivarieties.aspx
http://simpleindianrecipes.com/UpmaKichadiPongalVarieties.aspx
http://simpleindianrecipes.com/TraditionalTiffin.aspx
http://simpleindianrecipes.com/ExoticTiffinVarieties.aspx
http://simpleindianrecipes.com/cakesandcookies.aspx
http://simpleindianrecipes.com/cakes.aspx
http://simpleindianrecipes.com/Cookies.aspx
http://simpleindianrecipes.com/InternationalDesserts.aspx
http://simpleindianrecipes.com/PieVarieties.aspx
http://simpleindianrecipes.com/Puddings.aspx
http://simpleindianrecipes.com/DesiSweets.aspx
http://simpleindianrecipes.com/HalwaVarieties.aspx
http://simpleindianrecipes.com/IceCreamVarieties.aspx
http://simpleindianrecipes.com/KheerVarieties.aspx
http://simpleindianrecipes.com/Okra.aspx

http://simpleindianrecipes.com/Onion.aspx
http://simpleindianrecipes.com/Potato.aspx
http://simpleindianrecipes.com/Radish.aspx
http://simpleindianrecipes.com/Ridgegourd.aspx
http://simpleindianrecipes.com/Rutabaga.aspx
http://simpleindianrecipes.com/Scallions.aspx
http://simpleindianrecipes.com/Spinach.aspx
http://simpleindianrecipes.com/SquashVarieties.aspx
http://simpleindianrecipes.com/SweetPotato.aspx
http://simpleindianrecipes.com/Tapioca.aspx
http://simpleindianrecipes.com/Tomato.aspx
http://simpleindianrecipes.com/Turnip.aspx
http://simpleindianrecipes.com/yellowsquash.aspx
http://simpleindianrecipes.com/zuccini.aspx
http://simpleindianrecipes.com/apple.aspx
http://simpleindianrecipes.com/Avocado.aspx
http://simpleindianrecipes.com/Banana.aspx
http://simpleindianrecipes.com/berries.aspx
http://simpleindianrecipes.com/Cherry.aspx
http://simpleindianrecipes.com/Cranberry.aspx
http://simpleindianrecipes.com/Figs.aspx
http://simpleindianrecipes.com/RecipesByIngredients/Fruits/Gooseberry.aspx
http://simpleindianrecipes.com/Grapefruit.aspx
http://simpleindianrecipes.com/Grapes.aspx
http://simpleindianrecipes.com/RecipesByIngredients/Fruits/Guava.aspx
http://simpleindianrecipes.com/RecipesByIngredients/Fruits/Jackfruit.aspx
http://simpleindianrecipes.com/Lemon.aspx
http://simpleindianrecipes.com/Lychee.aspx
http://simpleindianrecipes.com/Mango.aspx
http://simpleindianrecipes.com/Orange.aspx
http://simpleindianrecipes.com/RecipesByIngredients/Fruits/Papaya.aspx
http://simpleindianrecipes.com/Pineapple.aspx
http://simpleindianrecipes.com/Strawberry.aspx
http://simpleindianrecipes.com/Tamarind.aspx
http://simpleindianrecipes.com/Watermelon.aspx
http://simpleindianrecipes.com/RecipesByIngredients/Dairy.aspx
http://simpleindianrecipes.com/Cream.aspx
http://simpleindianrecipes.com/Curd.aspx
http://simpleindianrecipes.com/PaneerRecipes.aspx
http://simpleindianrecipes.com/Blackgram.aspx
http://simpleindianrecipes.com/Chikpea.aspx
http://simpleindianrecipes.com/GreenGram.aspx
http://simpleindianrecipes.com/millet.aspx
http://simpleindianrecipes.com/Oats.aspx
http://simpleindianrecipes.com/Semolina.aspx
simpleindianrecipes.com/salads.aspx
simpleindianrecipes.com/soups.aspx
simpleindianrecipes.com/snacks.aspx
simpleindianrecipes.com/beverages.aspx
simpleindianrecipes.com/breakfastvarieties.aspx
simpleindianrecipes.com/snacks.aspx
simpleindianrecipes.com/Broccoli.aspx
simpleindianrecipes.com/ricevarieties.aspx
simpleindianrecipes.com/appetizers.aspx
simpleindianrecipes.com/salads.aspx
simpleindianrecipes.com/soups.aspx
simpleindianrecipes.com/snacks.aspx
simpleindianrecipes.com/beverages.aspx
simpleindianrecipes.com/chutneyvarieties.aspx
simpleindianrecipes.com/vegetariangravies.aspx

simpleindianrecipes.com/DaalVarieties.aspx
simpleindianrecipes.com/veggiedelights.aspx
simpleindianrecipes.com/nonveggravies.aspx
simpleindianrecipes.com/nonvegsidedishes.aspx
simpleindianrecipes.com/ricevarieties.aspx
simpleindianrecipes.com/chaats.aspx
simpleindianrecipes.com/breakfastvarieties.aspx
simpleindianrecipes.com/idlidosavarieties.aspx
simpleindianrecipes.com/rotivarieties.aspx
simpleindianrecipes.com/cakesandcookies.aspx
simpleindianrecipes.com/InternationalDesserts.aspx
simpleindianrecipes.com/DesiSweets.aspx
simpleindianrecipes.com/internationalrecipes.aspx
simpleindianrecipes.com/Curd.aspx
simpleindianrecipes.com/PaneerRecipes.aspx
simpleindianrecipes.com/Cream.aspx
simpleindianrecipes.com/apple.aspx
simpleindianrecipes.com/Avocado.aspx
simpleindianrecipes.com/Banana.aspx
simpleindianrecipes.com/Cranberry.aspx
simpleindianrecipes.com/Figs.aspx
simpleindianrecipes.com/Grapefruit.aspx
simpleindianrecipes.com/Cherry.aspx
simpleindianrecipes.com/Grapes.aspx
simpleindianrecipes.com/berries.aspx
simpleindianrecipes.com/Lemon.aspx
simpleindianrecipes.com/Lychee.aspx
simpleindianrecipes.com/Mango.aspx
simpleindianrecipes.com/Orange.aspx
simpleindianrecipes.com/Pineapple.aspx
simpleindianrecipes.com/Strawberry.aspx
simpleindianrecipes.com/Tamarind.aspx
simpleindianrecipes.com/Watermelon.aspx
simpleindianrecipes.com/RecipesByIngredients/Fruits/Gooseberry.aspx
simpleindianrecipes.com/Blackgram.aspx
simpleindianrecipes.com/Chikpea.aspx
simpleindianrecipes.com/GreenGram.aspx
simpleindianrecipes.com/RecipesByIngredients/GrainsCerealsPulses/Horsegram.aspx
simpleindianrecipes.com/RecipesByIngredients/GrainsCerealsPulses/SorghumJowar.aspx
simpleindianrecipes.com/millet.aspx
simpleindianrecipes.com/Oats.aspx
simpleindianrecipes.com/Semolina.aspx
simpleindianrecipes.com/Basil.aspx
simpleindianrecipes.com/RecipesByIngredients/Herbs/CaromSeeds.aspx
simpleindianrecipes.com/Cilantro.aspx
simpleindianrecipes.com/CurryLeaves.aspx
simpleindianrecipes.com/RecipesByIngredients/Herbs/Dill.aspx
simpleindianrecipes.com/mint.aspx
simpleindianrecipes.com/Scallions.aspx
simpleindianrecipes.com/Beef.aspx
simpleindianrecipes.com/Chicken.aspx
simpleindianrecipes.com/Crab.aspx
simpleindianrecipes.com/Egg.aspx
simpleindianrecipes.com/seafoodvarieties.aspx
simpleindianrecipes.com/Kheema.aspx
simpleindianrecipes.com/Mutton.aspx
simpleindianrecipes.com/Prawns.aspx
simpleindianrecipes.com/Quail.aspx
simpleindianrecipes.com/Turkey.aspx
simpleindianrecipes.com/almond.aspx

simpleindianrecipes.com/Cashew.aspx
simpleindianrecipes.com/Coconut.aspx
simpleindianrecipes.com/Peanut.aspx
simpleindianrecipes.com/asparagus.aspx
simpleindianrecipes.com/Beetroot.aspx
simpleindianrecipes.com/BellPeppers.aspx
simpleindianrecipes.com/Bittergourd.aspx
simpleindianrecipes.com/BokChoy.aspx
simpleindianrecipes.com/Basil.aspx
simpleindianrecipes.com/Bottleguard.aspx
simpleindianrecipes.com/Broccoli.aspx
simpleindianrecipes.com/BrusselsSprouts.aspx
simpleindianrecipes.com/Cabbage.aspx
simpleindianrecipes.com/Carrot.aspx
simpleindianrecipes.com/Cauliflower.aspx
simpleindianrecipes.com/Celery.aspx
simpleindianrecipes.com/Cilantro.aspx
simpleindianrecipes.com/Cucumber.aspx
simpleindianrecipes.com/CurryLeaves.aspx
simpleindianrecipes.com/Corn.aspx
simpleindianrecipes.com/Eggplant.aspx
simpleindianrecipes.com/Methi.aspx
simpleindianrecipes.com/Drumstick.aspx
simpleindianrecipes.com/Beans.aspx
simpleindianrecipes.com/Greens.aspx
simpleindianrecipes.com/Ivygourd.aspx
simpleindianrecipes.com/LotusRoot.aspx
simpleindianrecipes.com/mint.aspx
simpleindianrecipes.com/GreenPeas.aspx
simpleindianrecipes.com/Okra.aspx
simpleindianrecipes.com/Onion.aspx
simpleindianrecipes.com/Potato.aspx
simpleindianrecipes.com/Radish.aspx