A COMPARATIVE STUDY OF RECOMMENDER ALGORITHMS FOR A GASTRONOMIC SOCIAL NETWORK

MASTER OF SCIENCE THESIS

Supervisor
PhD. Marta Arias Vicente
Computer Science Department - UPC
This thesis is presented by Franz Bermeo Quezada

In fulfillment of the requirements for the degree of

Master in Innovation and Research in Informatics

with specialization in

Data Mining and Business Intelligence
ACKNOWLEDGEMENTS

I thank to God for giving me health, strength and wisdom to pursue my dreams and get this degree.

I'm really thankful the support of my supervisor PhD. Marta Arias, for her valuable advice and great patience. Thank you for your guidance, time and knowledge shared in our meetings.

My deepest gratitude to my family, to whom this dissertation is dedicated. To my lovely wife Nathalia and my little daughters Gianna and Suri to joining with me in this incredible adventure of personal and professional growth and for their patience and understand the many hours lost at home, because my responsibilities of the Master program. To my parents, Antonio and Carmita, who with their prayers, support and unconditional love has been my greatest strength.

Also, I'm indebted with kind people whom live in Catalonia but are from my hometown, Mr. Fernando "Taba" Luzuriaga, Torres-Ramon family, Gonzalez-Ayala family, Añazco-Aguirre family, Miss Adriana Salazar, Mrs. Diana Gonzalez and other paisanos with whom shared great moments. You have been like our family in this country far to our homeland, Nathalia and I will be always grateful for the love and kindness that all you gave us during our stay in Barcelona. Moltes gràcies amics, espero tornar a veure'ls en Portovelo ;)

Franz Enrique Bermeo Quezada
ABSTRACT

The always-growing popularity of social networks between the people and with a generation that takes a picture to everything they do or what they eat, generating a huge amounts of data. It is extremely difficult to deal with the available information without a tool support. The recommender systems are such tools; nowadays these are very popular because they process the information and suggest items, social elements, products or services that are likely to be of a user’s interest.

Onfan (onfan.com) is a gastronomic social network based in Barcelona that feeds on the contents that users assess and share about the specialties of any dining establishment where the users have visited.

The provision of new functionality in the site to assist and suggest users which specialty should taste, has become a priority to improve the user’s satisfaction. To add this capability, we require to understand the user activities for a correct computation of recommendations based on their preferences. Here, we report an analysis of real user ratings on a set of recipes in order to judge the applicability and practicality of custom algorithms, already tested to compute recommendations in social networks.

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 specialties.
# CONTENT

<table>
<thead>
<tr>
<th>Acknowledgements</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>3</td>
</tr>
<tr>
<td>List of Figures</td>
<td>5</td>
</tr>
<tr>
<td>1. Introduction</td>
<td>6</td>
</tr>
<tr>
<td>1.1. Problem Statement</td>
<td>6</td>
</tr>
<tr>
<td>1.2. Background and Motivation</td>
<td>6</td>
</tr>
<tr>
<td>1.3. Objectives</td>
<td>7</td>
</tr>
<tr>
<td>1.4. Outline of the Thesis</td>
<td>8</td>
</tr>
<tr>
<td>2. Related Work</td>
<td>8</td>
</tr>
<tr>
<td>2.1. State–of–the–art in Recommender Systems</td>
<td>8</td>
</tr>
<tr>
<td>2.1.1. Demographic Filtering</td>
<td>9</td>
</tr>
<tr>
<td>2.1.2. Collaborative Filtering</td>
<td>10</td>
</tr>
<tr>
<td>2.1.3. Content-Based Filtering</td>
<td>12</td>
</tr>
<tr>
<td>2.1.4. Hybrid Method</td>
<td>13</td>
</tr>
<tr>
<td>2.2. Social Recommender System for Online Social Networks</td>
<td>14</td>
</tr>
<tr>
<td>2.2.1. Published Algorithms</td>
<td>14</td>
</tr>
<tr>
<td>3. Onfan</td>
<td>16</td>
</tr>
<tr>
<td>3.1. Description</td>
<td>16</td>
</tr>
<tr>
<td>3.2. Architecture</td>
<td>19</td>
</tr>
<tr>
<td>3.2.1. Offline Component</td>
<td>20</td>
</tr>
<tr>
<td>3.2.2. Online Component</td>
<td>20</td>
</tr>
<tr>
<td>3.3. Enriching Onfan Data</td>
<td>20</td>
</tr>
<tr>
<td>3.4. Data Source</td>
<td>21</td>
</tr>
<tr>
<td>3.5. Preprocessing</td>
<td>22</td>
</tr>
<tr>
<td>4. Methods</td>
<td>23</td>
</tr>
<tr>
<td>4.1. Singular Value Decomposition</td>
<td>23</td>
</tr>
<tr>
<td>4.2. Social Recommender</td>
<td>24</td>
</tr>
<tr>
<td>5. Experiment Design and Evaluation Metrics</td>
<td>27</td>
</tr>
<tr>
<td>5.1. Mean Absolute Error (MAE)</td>
<td>28</td>
</tr>
<tr>
<td>5.2. Root Mean Square Error (RMSE)</td>
<td>28</td>
</tr>
<tr>
<td>6. Results</td>
<td>28</td>
</tr>
<tr>
<td>6.1. SVD</td>
<td>28</td>
</tr>
<tr>
<td>6.2. Item-Based CF</td>
<td>29</td>
</tr>
<tr>
<td>6.3. User-Based CF</td>
<td>30</td>
</tr>
<tr>
<td>6.4. Social Neighborhood-Based CF</td>
<td>30</td>
</tr>
<tr>
<td>6.5. Ingredient Content-Based</td>
<td>32</td>
</tr>
<tr>
<td>7. Conclusions and Future Works</td>
<td>35</td>
</tr>
<tr>
<td>8. References</td>
<td>36</td>
</tr>
</tbody>
</table>
LIST OF FIGURES.

Figure 1 Recommendation Techniques ................................................................. 9
Figure 2 Onfan User profile .................................................................................. 16
Figure 3 Onfan user profile sections ................................................................. 17
Figure 4 Onfan speciality snapshot ................................................................. 18
Figure 5 Onfan Block Diagram ....................................................................... 19
Figure 6. Online and Offline component ......................................................... 20
Figure 7 Crawling process ............................................................................... 21
Figure 8 Onfan Data Source - MySQL and REDIS ......................................... 22
Figure 9 Preprocessing and offline component .............................................. 23
Figure 10 SVD Approach ................................................................................. 24
Figure 11 CB and CF Approach ...................................................................... 26
Figure 12 Scipy solution with K different values and different sizes of test .......... 29
Figure 13 SVDLib solution with K different values ............................................. 29
Figure 14 Item-based CF solution ................................................................. 30
Figure 15 User-based CB solution .................................................................. 30
Figure 16 Social-based CF solution ................................................................. 31
Figure 17 Solution varying the ALPHA constant ............................................. 31
Figure 18 Comparison between social-based solutions .................................. 32
Figure 19 Content-based solution using original score matrix ......................... 33
Figure 20 Enhance content-based algorithm .................................................. 33
Figure 21 Comparison between content-based solutions ............................... 34
Figure 22 MAE of all algorithms ...................................................................... 35
1. INTRODUCTION

1.1. PROBLEM STATEMENT.

Onfan, a gastronomic social network with more than 8000 users is facing the exponential growth of the data generated by its users. The need of collaborative algorithms to preprocessing the data and transform it in valuable source of knowledge, has become a priority to prepare effectively the information that will be used to deal with the requirements and preferences of users more appropriately.

So far, its functionality lacks of recommendations based on the preference of the profile of a user and his/her contact list conformed by the followers/followings users with similar preferences of specialties or restaurant that they frequently visit.

The purpose of this thesis is to develop a recommender algorithm based on the existing solutions already tested for recommender system. To achieve this goal, we will use the Onfan data source and then apply different approaches to create more accurate recommendation.

Collaborative Filtering (CF) and Content-Based (CB) recommendations are the most commonly used type of algorithms, each method has its strengths and weaknesses. We will implement an algorithm to select a subset from thousands of dishes to recommend the appropriate specialty, which could fit the users’ needs and preferences. Knowing that it is almost impossible to predict precisely the users’ needs, will help people to make decisions about what dish they could try.

Furthermore, we will use decomposition of matrix, a typical technique used in recommendations as part of the collaborative strategy and thus compare its outcomes with the retrieved by the others algorithms.

1.2. BACKGROUND and MOTIVATION.

When people purchase products, they must decide which items to buy, often their choice depends on the other users’ opinion, especially in the e–commerce. Hence, they ask their friends who are trustworthy give them suggestions about which product to buy and why they should pick that one and not the other one. Based on these opinions, they decide to execute the purchase.

The growth of the Internet and the e–commerce solutions caused the development of the online recommender systems and since the mid–1990s they have become an important research domain.

Introducing into the topic of this work, before to visit any restaurant, customers are interested to know the opinion about how good is the food or the quality of service offered by. So, each user can adopt different profiles (common description/information about the user’s likes and thus to find overlaps interests with other active user) depending on what they want or based on their consumption record.
Users can have access to a huge deal of data and information related to a specific restaurant, but surely they will prefer to filter that information and get those elements that match their interests. Each of these profiles determines the different restaurant to visit.

When users decide to taste a recommended dish in base to his/her preferences, the site can offer additional items in order to increase the cross-sell. This leads to maintain the loyalty of the customers and encourages them to come back a next time to the restaurant. Actually, in the Internet and e-commerce where the competition exist in significant amount, this feature is a crucial advantage of the recommender systems.

For the purpose of improving the USER’S experience, recommender systems offer many solutions to process the data of the users and their neighborhood of contacts and thus give them customized information according to their needs.

The amount of information available to manage and the lack of experience of users to deal with it, make it clear the need of automatic systems capable of helping users in the decision-making process through directions and suggestions.

The idea of both online social networks and recommender systems working together, was initially proposed by researchers who foresaw the exponential growth of information it was coming in the future. The motivation is, with help of the knowledge of computer science, build a solution framework with the appropriate algorithm adapting specific needs or preferences of users. For online social network not only the preferences of a person who expects some suggestions should be considered, but also the expectations of a person who will be suggested.

1.3. OBJECTIVES.

The purpose of this thesis is to make a comparative study of state-of-the-art where we propose popular algorithms already tested in the recommendations’ field, design and implement a recommender layer in the current Onfan platform and then measured its accuracy of its retrieved recommendations. With this improvement, we hope to increase the loyalty of the current users and in the case of users who only surf through the site, encourage them to become in active foodies.

To enhance the outcomes of traditional collaborative filtering, we use a collaborative and content-based implementation which builds a users-items matrix based on credibility of users drawn from their trustworthiness taken from social network of an active user.

Test the performance and verify the accuracy in collaborative algorithms, taking into account both the similarities matrix and the social information of all users to compute the predictions and improve the recommendations based on the neighborhood of the active user.

Finally, we study and compare the characteristics of each algorithm, highlighting its strengths and weaknesses. Several experiments have been performed, using the most popular metrics and algorithms.
1.4. OUTLINE OF THE THESIS.

INTRODUCTION. This chapter is focused on the statement of the current problem in the recommendations of Onfan.com, the motivation and the objectives that we want to achieve at the final of this work.

RELATED WORK. This chapter reviews the state-of-the-art about recommender systems, the classification of the recommender algorithms, its application into the social networks and some published algorithms that we will consider in this work.

ONFAN. This chapter describes the structure of the data, where it hosted, how is managed currently and how is the preprocessing phase to leave the data ready to compute the recommendations.

METHODS. This chapter explains the collaborative and content-based methods, which conform the filtering strategy and a decomposition matrix technique as second solution applied on the Onfan data.

EVALUATION METRICS. This chapter shows the measure strategies to evaluate the performance of the algorithms which computes the recommendations.

RESULTS. This chapter shows all the outcomes, errors, accuracy values and the comparison between the methods explained in the section above.

CONCLUSIONS. The final chapter summarize the contributions of this work.

2. RELATED WORK.

Recommender systems emerged as an independent research area when researchers started focusing on recommendation problems that explicitly rely on the ratings structure [1]. The goal is to estimate ratings for the items that have not been seen by a user, based on the ratings given by this user to other items previously rated and on some other related information. Once defined the strategy to estimate ratings for the unrated items, we can apply it to compute the recommendations and thus to offer to the user the items with the highest estimated rating.

2.1. STATE–OF–THE–ART IN RECOMMENDER SYSTEMS.

We will take the formal definition given by G. Adomavicius and A. Tuzhilin [1]:

Let \( C \) be the set of all users and let \( S \) be the set of all possible items that can be recommended. The space \( S \) of possible items can be very large, ranging in hundreds of thousands or even millions of items in some applications. Let \( u \) be a utility function that measures the usefulness of item \( s \) to user \( c \).

\[
u: C \times S \rightarrow R
\]

Where \( R \) is a totally ordered set. Then, for each user \( c \in C \), we want to choose such item \( s' \in S \) that maximizes the user’s utility.

\[
\forall c \in C, s'_c = \arg \max_{s \in S} u(c, s)
\]
In recommender systems, the utility of an item is usually represented by a rating, which indicates how a user liked a target item. The recommendation engine should be able to predict the ratings of the non-rated user/item combinations and issue appropriate recommendations based on these predictions. [1] In the recommendation process, the existing ratings is usually very small compared to the ratings that need to be predicted.

The central element of recommender systems is the user model, containing knowledge about the user preferences. So, the user data is built, containing different information recorded during the user’s interactions with web-based systems, capturing interests, preferences or behavior to define how relevant this data is.

Once the unknown ratings are computed, item recommendations are made by selecting the highest rating among all the estimated ratings for that user. Furthermore, we can recommend the N best items to a user or a set of users to an item. The not-yet-rated items can be estimated using methods from machine learning, approximation theory and heuristic algorithms [1].

Recommendation techniques can be distinguished on the basis of their knowledge sources. In some systems, this knowledge is the knowledge of other users’ preferences. In others, it is ontological or inferential knowledge about the domain, added by a human knowledge engineer [11].

Recommender systems have background data, which is the information that user stores into the database through the web application. This is the source used by the algorithms to generate a recommendation and thus deliver its suggestions. On this basis, we can distinguish four recommendation techniques (as you see in the Figure 2)

2.1.1. DEMOGRAPHIC FILTERING.

Provides a categorization of users taking their attributes (age, income level, education, race, sex, economic status, employment, etc) and make recommendations based on specific classes. The advantage of this approach is that the user-item ratings matrix is not used. We can get recommendations before they have rated any item, because the technique is domain independent.
In this work we will focus on the following strategies:

### 2.1.2. COLLABORATIVE FILTERING.

Try to predict the utility of items for a particular user based on the items rated by other users. That is, the utility $u(c, s)$ of item $s$ for user $c$ is estimated based on the utilities $u(c_j, s)$ assigned to item $s$ by those $c_j \in C$ similar to user $c$.

According to Breese and Heckerman [2], algorithms for collaborative recommendations can be grouped into two general classes:

- **MEMORY-BASED ALGORITHMS.**

  The task in collaborative algorithms is to predict the rating of a particular user from a database of user ratings or population of other users.

  In memory-based algorithms, we predict the rating of the active user based on partial information regarding the active user and a set of weights calculated from the user preferences. The predicted rating $p_{a,j}$ is a weighted sum of the ratings of the other users [2]:

  $$ p_{a,j} = k \sum_{i=1}^{n} s(a, i) \cdot v_{i,j} $$

  Where $v_{i,j}$ is the rating of user $i$ on the item $j$, $n$ is the number of users involved in the collaborative algorithm with non-zero rating. The weights $s(a, i)$ can be distance, correlation or similarity between each user $i$ and the active user. Finally, $K$ is a normalizing factor such that the absolute values of the weights sum to unity.

In similarity we have available two measures:

1. **User-User similarity.** Find users whose rating behavior is similar to that of the current user and use their ratings on other items to predict what the current user will like.

2. **Item-Item similarity.** Based on their ratings of items that two users have rated. Is a heuristic algorithm that is introduced in order to be able to simplify the rating’s estimation. For this case, the definition for the predicted rating is the following:

   $$ p_{a,j} = k \sum_{i=1}^{m} s(j, i) \cdot v_{a,i} $$

   Where $v_{a,i}$ is the rating of active user $a$ on the item $i$, $m$ is the number of items similar to $j$ with non-zero rating and the weights $s(j, i)$ is similarity between the target item $j$ and each item $i$.

To present the most popular approaches which compute the similarity, let $S_{xy}$ be the set of all items co-rated by both users $x$ and $y$, that is used mainly as an intermediate result for calculating the trusted neighborhood of user $x$. 


Correlation-based approach: The Pearson correlation coefficient is used to measure the similarity.

\[
sim(x, y) = \frac{\sum_{s \in S_{xy}} (r_{xs} - \bar{r}_x)(r_{ys} - \bar{r}_y)}{\sqrt{\sum_{s \in S_{xy}} (r_{xs} - \bar{r}_x)^2} \sqrt{\sum_{s \in S_{xy}} (r_{ys} - \bar{r}_y)^2}}
\]

Cosine-based approach: The users x and y are treated as two vectors in m-dimensional space.

\[
sim(x, y) = \frac{\sum_{s \in S_{xy}} r_{xs} \cdot r_{ys}}{\sqrt{\sum_{s \in S_{xy}} r_{xs}^2} \cdot \sqrt{\sum_{s \in S_{xy}} r_{ys}^2}}
\]

- MODEL-BASED ALGORITHMS.

Use the collection of ratings to learn a model, which is then used to make rating predictions and it is assumed that rating values are integers between 0 and N, predicting the user’s ratings of the previously rated items. [1]

To estimate this probability, Breese, Heckerman and Kadie [2] proposes two alternative probabilistic models:

Cluster model: Groups or types of users are captured in a common set of preferences and tastes. Given the class, the preferences regarding the various items are independent. The probability model relating joint probability of class and votes to a tractable set of conditional and marginal distributions is the standard Naïve Bayes.

\[
Pr(C = c, v_1, ..., v_n) = Pr(C = c) \prod_{i=1}^{n} Pr(v_i | C = c)
\]

The probability of observing an individual of a particular class c (we must employ methods to construct models with c variables) and a complete set of rating values \(v_i\) in \(n\) number of users.

Bayesian Network model: Represents each item in the domain as a node in a Bayesian network, where the states of each node correspond to the possible rating values for each item. Both the structure of the network and the conditional probabilities are learned from the data. One limitation of this approach is that each user can be clustered into a single cluster, whereas some recommendation applications may benefit from the ability to cluster users into several categories at once.

- CHALLENGES OF CF ALGORITHMS.

Sparsity. Since the correlation coefficient is defined between users who have rated at least two items, we found that even active users have rated under 1% of the total of items. So, the algorithms may be unable to make accuracy recommendations for a particular user because the coverage in rating is reduced [8].
**Scalability.** Since nearest neighbor algorithms require computation that grows with both the number of users and items, all of this encapsulated in a web-based environment, will exist algorithms which present scalability problems [8].

**Loss of Neighbor Transitivity.** User $u_i$ and user $u_k$ are highly correlated and user $u_k$ also highly correlated with user $u_i$, so there exist a possibility that users $u_i$ and $u_l$ correlate highly too. But, the transitivity is not captured, unless users $u_i$ and $u_l$ have purchased or rated common items [14].

**Synonymy.** Different item names can refer to the similar objects. Correlation-based recommender systems are usually unable to discover the latent association between products, which have different names, but still refer to similar objects. A method should be used to deal those latent associations in order to generate better recommendations [14].

### 2.1.3. CONTENT-BASED FILTERING.

The content-based filtering system recommends items based on the correlation between the item’s content and user’s preferences, so different as collaborative algorithms that suggest items based on users/items’ similarities. The content of each item is represented as a set of terms, commonly the words contained in a document.

When we make the decision to implement the content-based strategy, initially we must to extract the terms from the container document either in automatically or manually way and thus the information retrieved from users and items can be compared in a meaningful way. Then, construct an algorithm responsible to learn the user profile based on rated items to make more accuracy recommendations.

Next, we describe the representation types used in content-based approach [13]:

- **ITEM REPRESENTATION.**

  This approach takes items that the target user has rated and computes the similarity between its attributes with the target item and then selects the most similar. Then, the prediction is then computed by taking a weighted average of the target user’s ratings on these similar items. These Items are often stored in a database table, where the columns can be defined as attributes and each item is described by the same set of attributes.

  In many content-based filtering solutions, item descriptions are textual features extracted from web pages, news articles or product descriptions. Unlike structured data, there are no attributes with well-defined values. Item features is a challenge because this features are created through user feedback from the application interface, therefore, will be involved all problems concerning to natural language.

- **USER PROFILES.**

  A profile that consist in a number of different types of information of the user’s interests is used by most recommendation systems. There are the following types of profiles:
**User’s preferences.** Description of the types of items that interest the user. There are many alternative representations of this description, but one common representation is a function that for any item predicts the likelihood that the user is interested in that item. The function can be used to retrieve the items most likely to be of interest to the user.

**History of the user’s interactions.** Includes storing the items that a user has viewed together with other information about the user’s interaction. Other types of history include queries typed by the user.

A use for history interaction can be systems which displays the last visited items, to facilitate the user returning to these items. Other case, the system can filter out from a recommendation an item that the user has already purchased or read.

Other techniques for content-based recommendation have also been used, such as Bayesian classifiers and various machine learning techniques, including clustering, decision trees and artificial neural networks which construct a learning model from the underlying data using statistical learning and machine learning techniques. [1]

**2.1.4. HYBRID METHOD.**

Born of the combination of collaborative and content-based methods, which helps to avoid certain limitations of content-based and collaborative systems. Hybrid recommendation systems can also be augmented by knowledge-based techniques [3], such as case-based reasoning, in order to improve the accuracy and to address some of the limitations of traditional recommender systems.

The survey by Robin Burke [12], identified the many types of hybrid recommendation, but we will focus only in the following:

- **WEIGHTED.**

  A weighted recommender is an approach where the score of a recommended item is computed from the results of all of the available recommendation techniques present in the system. A simplest hybrid would be a linear combination of recommendation scores.

  The advantage to use a weighted solution is that all of the system’s capabilities are made on the recommendation process in a straightforward way and it is easy to perform post-hoc credit assignment. The implicit assumption in this technique is that the relative value of the different techniques is more or less uniform across the space of possible items.

Based on the weighted type, there are different ways to combine collaborative and content-based methods into a hybrid recommender system. They can be classified as follows:

- **COMBINING SEPARATE RECOMMENDERS.**

  Implement a collaborative and content-based by separated. We take the ratings obtained from individual systems into one final recommendation using either a linear combination of ratings or a voting scheme.
**ADDING CONTENT-BASED CHARACTERISTICS TO COLLABORATIVE MODELS.**

The profiles are used to calculate the similarity between two users. This allows to overcome the sparsity problems of collaborative filtering since not many pairs of users will have a significant number of commonly rated items [1].

Another benefit is that users can recommend an item not only when is rated with a high value by users, but also using a collaborative approach where the user’s ratings is augmented with additional ratings, which are calculated using a content-based predictor [1].

**ADDING COLLABORATIVE CHARACTERISTICS TO CONTENT-BASED MODELS.**

Use some dimensionality reduction technique on a group of content-based profiles where user profiles are represented by term vectors, resulting in a performance improvement compared to the pure content-based approach [1].

### 2.2. SOCIAL RECOMMENDER SYSTEM FOR ONLINE SOCIAL NETWORKS.

The growth of social networks in recent years has developed new business such as the social user data. These social data are becoming important for many companies and are often used to determine user interests for items in order to propose or advertise items to them.

Social recommender as a new type among recommender systems, will recommend the products/services that user’s contacts, users in the social network whom the user trust, liked in the past. In other words, its role is to suggest items that a user might be interested in the content of people who he/she follows.

The authors Freyne, Berkovsky, Daly and Geyer [17] in their work said that the social networks keep users up to date with their contacts activities on the network by including news or activity feeds. The content of the feed relates to the actions of a target user’s contacts and informs the user of new content contributed (photos, groups, comments), new friendships made, groups joined, status message updates and other actions. Social recommender systems aims to determine the relevance of a network item to a user by examining the observed user interactions of the individual over a period of time.

#### 2.2.1. PUBLISHED ALGORITHMS.

There is a diversity of algorithms and approaches that help creating recommendations, but for this works we focused mainly in the popular collaborative and content-based filtering solutions. Based on the available source and the problem to solve, we present the recommender approaches that we will take to test and get the better results as possible.

**DIMENSIONALITY REDUCTION IN RECOMMENDER SYSTEM [8].**

This work presents experiments where they explore one technology called Singular Value Decomposition (SVD) to reduce the dimensionality of recommender system databases. Each experiment compares the quality (effectiveness predicting consumer
preferences based on a database of ratings of products and to produce Top-N lists based on a real-life customer purchase database) of a recommender system using SVD with the quality of a recommender system using collaborative filtering.

Sarwar, Karypis, Konstan and Riedl [8] concluded that the SVD-based approach produced results that were better than a traditional collaborative filtering algorithm. This technique leads to very fast online performance, requiring just a few simple arithmetic operations for each recommendation.

- **PROBABILISTIC MATRIX FACTORIZATION [6].**

A trust-aware collaborative filtering method for recommender systems is proposed. In this work, the collaborative filtering process is informed by the reputation of users which is computed by propagating trust. Trust values are computed in addition to similarity measures between users.

In this paper, using latent factor analysis with probabilistic matrix factorization, the authors catch to the user latent feature space and item latent feature space by employing a user social network and a user-item matrix simultaneously and seamlessly.

- **RECIPE RECOMMENDATION [7].**

This work aims to investigate how individuals reason in relation to food and in particular recipes. We examine real user rating data to see if patterns of reasoning exist for individuals. This analysis presented here aims to understand reasoning on recipes only and disregards the context of meal planning and scheduling.

Each recipe in the corpus has a basic structure including a title, ingredient and instructions. From this basic information, automatically extract additional information. Finally, Freyne, Berkovsky and Smith [7] decipher two indicators of recipe complexity: The number of ingredients and the number of steps required to complete the recipe.

- **TRUST-BASED RECOMMENDATION [16].**

The model deals with agents which have to decide for a particular item that they do not yet know based on recommendations of other agents. When facing the purchase of an item, agents query their neighborhood for recommendations on the item to purchase.

Neighbors in turn pass on a query to their neighbors in case that they cannot provide a reply themselves. In this way, the network replies to a query of an individual by offering a set of recommendations. It is important to stress that trust relationships only exist between neighbors in the social network.

Traditional recommender systems generate meaningful items recommendations that might interest to users and continue to analyze the data to develop notions of affinity between users and items. In the other hand, social-based recommender system is defined as
enhanced recommender which incorporates social contextual information that can help to improve the quality prediction especially when the available data is sparse.

In other words, the social recommender in addition to have as bases the interest on an item that comes from a user, incorporates preferences of a network of trusted contacts of a target user, thus improving the recommendation.

3. ONFAN.

3.1. DESCRIPTION.

Onfan.com is a gastronomic social network site in which users express what dishes they have eaten and whether they have liked them when visiting restaurants. Onfan revolves around specialties (or dishes) and not restaurants. Furthermore, users can follow other users and they can see what dishes these followed users liked.

The users must register for free. Then, upload the image of the speciality and the geo-position of the restaurant and starting from this point, you can start writing their personal opinions of those specialties that they have tried in a specific venue.

Based on the contact list, Onfan presents a network of trust relationships between users and thus use it to re-order the specialties reviews such that a user first sees reviews by users that he/she trust.
The above screen shows a user profile page. We can see the detail section where show the summarize information about the activity of the user. In the upside the fans and prescriptors section representing the contact list of the user.

1. **Fans.** Registered users who follows to a specific user because they like or looks appetizing the dishes that this user commonly upload in his/her onfan’s wall.

2. **Prescriptors.** Registered users who a specific user follows because he/she like or looks appetizing the dishes that those users commonly upload in their onfan's wall.

Furthermore, under contacts section we found a historical of the specialities which the user have created and the venues where the user have visited to try its dishes.

In addition, in the bottom we found other sections as specialities which have tasted and specialities which user wish, where we can see a preliminary information about the dishes and also the comments section where we can see all comment posted by the user in a speciality which him/her is the owner or any speciality uploaded by his/her contacts.

![Onfan user profile sections](image-url)
As a social network, Onfan offers the possibility to pick and choose people whose preferences could be similar to yours within the network, thus gives to users the capability to grow your network of contacts and the opportunity to comment not only his/her own dishes but the dishes uploaded by other users.

Therefore, we can access the speciality profile of any user where we can found information as the restaurant and its detail information where the speciality is served. Furthermore, similar specialities section where shows other specialities corresponding to the same classification as the current dish. In addition, there exists the possibility to add comments to the dish.

But the most important functionality in this page corresponding to the three buttons available under the speciality image (WISH, TASTE and LIKE) which will be useful to Onfan to have an idea about the user's preference.

Figure 4 Onfan speciality snapshot
1. **WISH.** The most important according to the needs for recommendations. When a user prefers to want a speciality, will press this button.

2. **TASTE.** When a user have tried the savor of a speciality, should press this button.

3. **LIKE.** This button will be pressed when user to see the speciality image, it seems desirable to him/her.

### 3.2. ARCHITECTURE.

The current architecture of Onfan is a web-based platform accessed either from mobile or computer browser. All its layers (UI, Business and Data components) are implemented in the widely used Python language, in the version 2.7.

The data sources used by Onfan are hosted in two components:

1. **Redis**, which is an open-source key-store server with optional durability, where is stored the complete graph of all links between users registered in the platform and also all users' activities over the specialities.

2. **MySQL**, the most popular open-source RDBM System, where is stored the data corresponding to all Onfan business logic.

![Onfan Block Diagram](image)

In this work we will implement all algorithms for the recommendation layer, responsible for computing similarities, predicting the rating to fill the blank cells to finally compute the items recommendations.

Generally, collaborative algorithms require a huge amount of computation time to extract the data, preprocess and finally generate both the users-items rating matrix as the similarities matrices. That’s why as part of our solution we have divided data process into two parts [8]:

3.2.1. **OFFLINE COMPONENT.**

Is the portion of the algorithm that requires an enormous amount of computation especially during the construction of users-items matrix. During the offline computation, issues with memory and secondary storage requirement should be taken with care, as the growth of data sources.

![Diagram showing offline component](image)

In our solution, the offline component is responsible to preprocess the data taking the raw data from MySQL and then to generate all necessary matrices for the computations of recommendations. Once finished the computations, all resulting matrices are stored as object in a key-store database.

3.2.2. **ONLINE COMPONENT.**

Is the portion of the algorithm that is dynamically computed to provide predictions to users using data from the offline component.

The online computation scales independently of the number of customers and number of items involved. It is possible to improve the relevance of the recommendations by using segments, but it is very likely that user–segment classification becomes almost as expensive as compute the user similarities [9].

For our particular case, the online component is in charge to get all matrices from the key-store database and then to compute the corresponding item recommendations based on a particular user.

3.3. **ENRICHING ONFAN DATA.**

Onfan web have not a specific section to add the corresponding ingredients after to add a new speciality but whether the user can typing all information that he/she wants in the description field. So, the data stored in Onfan database have no established reference nor structure links between the specialities submitted by users and the ingredients contained in that dish. That's why before start to apply the recommender algorithms, we tried to complete this information in the database taking the existing data from the field’s recipe.
NAME and DESCRIPTION and validate it using data matching process implemented in python with information recipes existing in the web.

![Figure 7 Crawling process](image)

As a result, we obtain only few matched registers with the downloaded recipes because the data typed by the users corresponding to the speciality in Onfan.com in many cases aren’t words related with recipes title but feeling expressions or in some cases the ingredients names are typed. So, after several tests getting very poor results we ruled-out this part of our solution.

Furthermore, we also scraped data related with cuisine ingredients to populate a table in MySQL database, which then we used to validate during the execution of content-based algorithm. The following are the source urls of the cuisine glossary words.

1. www.gourmed.com/glossary/all
2. www.sabormediterraneo.com/diccionario1/

With this strategy we ensure that the words that serve as inputs in the content-based algorithm, are related with recipes or cuisine, thus we’ll improving the accuracy in the results of the recommendations.

3.4. DATA SOURCE.

As we have mentioned in the above section 3.2, Onfan have two instance for storage of the whole data, both with different capabilities when is required to manage data, one where is hosted the all tables containing the core data business and the other in charge to keep the relations between users and their activities on the specialities.

Currently, Onfan have more than 18000 registered users and more than 8000 specialities with its own image and user feedback stored in MySQL database. As we can see in the below Figure 8, the left-side is a partial schema of the Onfan business model that works in MySQL database. For the recommendations, the objects involved as data source will be the tables SPECIALITY, SPECIALITY_USER and ONFANUSER where is hosted the required data to use in the filtering solutions.
From speciality table we take the NAME field, which is the name of the speciality typed by the user through Onfan interface. Next, we go to speciality_user table because it is charge to host the relationships of each users with their specialities which he/she have submitted, from this table we will take DESCRIPTION and COMMENT fields. In the case of the ONFANUSER, this table is responsible to store ID, username, name, surname, contact phones and address of the Onfan users. Unlike ID, the remaining information is not relevant in the computations made by the recommendations algorithms.

In the right-side, the oval represent the graph stored in REDIS, the key-store database in charge of to manage all graphs generated for the network contacts as user followers and user followings and also host the user’s activity as which speciality the user wish or he/she have tasted or maybe want to try.

3.5. PREPROCESSING.

Before generating all required matrix for the computations, is a healthy practice to preprocess the data, in this way, we reduce the computing time required for the recommendation algorithms. After removing stop-words from title field of SPECIALITY table and description and comment fields corresponding to SPECIALITY_USER table of MySQL, we continue filtering the data this time validating with wGlossary table where are stored words related with cuisine (mentioned in section 3.2).
In the users-specialities matrix, we'll only consider the cells which have been rated or contain predicted value greater than zero. That is, for the computations we select only set of users with at least one speciality marked in their profile before the system can generate accurate predictions and useful recommendations for that user.

4. METHODS.

In this work we explore the techniques mentioned in the above section 2.3.1. First, the Singular Value Decomposition (SVD) to reduce the dimensionality of the users-specialities rating matrix and then as a second method, a strategy that combines Collaborative + Content-based filtering where decompose in cuisine words to generate a new users-ingredients rating matrix, thus testing the strength and weakness for each of applied methods.

In both strategies, after obtaining the updated users-items rating matrix, we will remove a bit the sparsity by filling each cells with the average ratings for the corresponding user. Once obtained this reconstructed matrix, we can use it to recommend items for a target user.

4.1. SINGULAR VALUE DECOMPOSITION.

SVD is a matrix factorization technique that takes a $R_{n \times m}$ matrix and decompose it into three matrix applying $k$ dimension of reduction:

$$U * S * V^T = \text{SVD}(R, k)$$

Where $U_{n \times k}$ and $V_{n \times k}$ are two orthogonal matrices and $S_{k \times k}$ is a diagonal matrix having all singular values of matrix $R$, all values positive and in decreasing order. The SVD provides the best lower rank approximations of the original matrix $R_{n \times m}$. After reducing to U and V matrices, if we multiply all resulting above matrices, the reconstructed matrix $R'$ of $k$ eigenvectors is the following:

$$R'_{n \times m} = U_k * S_k * V_k^T$$

SVD is used to capture latent relationships between users and items. The dimensionality reduction process helps each user to map items into the space corresponding to its eigenvectors, allowing us to compute the predicted rating of a user-item cell, producing a
new representation of the original matrix space $R$ and thus to have a reduced neighborhood.

$$R_{n \times m} \equiv User \times Item \ rating \ matrix$$

$$k \equiv dimensions$$

$$sTest \equiv Test \ set$$

$$U_{n \times k}, S_{k \times k}, V_{k \times m}^T = SVD(R, k)$$

We can take the less sparse decomposed matrices $U$, $S$, and $V$ to capture the relationships among pairs of users based on ratings of specialties. SVD provides the best low-rank approximation of the original matrix, which according to pointed by M. Berry, S. Dumais and G. O'Brien [19] in their tests, is better than the original space itself due to the filtering out of the small values that introduce noise in the user-item matrix.

As we can see in the above Figure, the offline component is the responsible to generate the users-specialities rating matrix and also in charge to select from non-zero rated cells, a percentage size as test set.

4.2. SOCIAL RECOMMENDER.

In many research about recommender system, the incorporation of social contextual information can help to improve the quality of prediction, especially when the data is sparse. That said, as part of our filtering strategy we decided to incorporate the fans/prescriptors relations existing between Onfan users into the user's similarity approach as one of proposed collaborative solutions.

In the other hand, we implement an effective method of capturing ingredient preferences to improve the accuracy of recommendations, assigning scores for a target item for a user, based on the average of all the ratings provided by its ingredients. Regardless of whether ratings are gathered on ingredients or specialties, the output of the recommender algorithms is an item recommendation.
Following, we’ll present all filtering strategies applied on the available Onfan data:

\[
R_{n \times m} \equiv \text{User x Item rating matrix}
\]

\[
SPath_{n \times n} \equiv \text{Shortest path matrix}
\]

\[
ncR_{n \times m} = \text{normalize}(R,\ \text{columns}')
\]

\[
l_{ixl_{m \times m}} = ncR^T * ncR \equiv \text{Items Similarity Matrix}
\]

\[
nrR_{n \times m} = \text{normalize}(R,\ \text{rows}')
\]

\[
UxU_{n \times n} = nrR * nrR^T \equiv \text{Users Similarity Matrix}
\]

\[
N_{\in SPath} \equiv \text{Trusted neighbors}
\]

\[
M_{\in lxI} \equiv \text{Similar items}
\]

\[
L_{\in UxU} \equiv \text{Similar users}
\]

- **ITEM-BASED CF TECHNIQUE.**

Our first strategy is a collaborative filtering algorithm. We assign predictions of user \(u_a\) for a target speciality \(r_t\), based on a set \(M\) of similar items than \(r_t\). That’s why when we reach this point, we should have already generated the matrix of similarities between items, necessary to perform the computations according to the definition put forward below:

\[
\text{pred}(u_a, r_t) = \frac{\sum_{m \in M} \text{sim}(r_t, r_m) * R[u_a, r_m]}{\sum_{m \in M} \text{sim}(r_t, r_m)}
\]

- **USER-BASED CF TECHNIQUE.**

The following approach is based on the similarity between users. We’ll take the user-speciality rating matrix to predict the rate which user \(u_a\) gave to speciality \(r_t\), considering the \(L\) most similar users

\[
\text{pred}(u_a, r_t) = \frac{\sum_{n \in L} \text{sim}(u_a, u_n) * R[u_n, r_t]}{\sum_{n \in L} \text{sim}(u_a, u_n)}
\]

- **SOCIAL NEIGHBORHOOD-BASED CF TECHNIQUE.**

An alternative strategy of the above mentioned is other collaborative algorithm. The rating predictions of user \(u_a\) for a target speciality \(r_t\), is based on set \(N\) of trusted neighbors taken from the shortest path matrix. In this case, the users-users similarity matrix is required to compute the predictions. In addition, the shortest-path matrix
also should to be generated and thus help us to know the trust neighborhood of a specific user.

\[
\text{newR} = \text{fillRatingCells}(R, UxU, SPath)
\]

\[
\text{pred}(u_a, r_t) = \frac{\sum_{n \in N} \text{sim}(u_a, u_n) \times R[u_n, r_t]}{\sum_{n \in N} \text{sim}(u_a, u_n)}
\]

Based on this solution, we can vary slightly the computation of rating predictions. We will no longer trusted users of a particular active user. What we do is create a new user-user similarity matrix, combining the original similarity matrix and the shortest-paths matrix using a constant \( \alpha \) which have an action rate between [0,1].

\[
UxU' = \alpha \times UxU + (1 - \alpha) \times SPath
\]

- **INGREDIENT CONTENT-BASED TECHNIQUE.**

Finally and to complete all solutions, the content-based algorithm which breaks down each rated specialty into ingredients and creating a new user-items matrix with a new rating computed to each ingredient. Then, taking this generated matrix we predict a score for the target specialty \( r_t \) based on the average of all the scores provided by the user \( u_a \) on ingredients making up the specialty.

\[
\text{Sc}(u_a, ingr_b) = \frac{\sum_{r \in \text{recipes}(ingr_b)} R[u_a, r]}{\text{count(\text{recipes}(ingr_b))}}
\]

\[
\text{newR} = \text{fillRatingCells}(R, Sc)
\]

\[
\text{pred}(u_a, r_t) = \frac{\sum_{b \in \text{ingredients}(r_t)} \text{Sc}[u_a, b]}{\text{count(\text{ingredients}(r_t))}}
\]

The update process will be applied to fill most of the cells in the users-specialities matrix where having a zero value. The Once obtained the updated matrix \( \text{newR}_{n \times m} \), we can make on it all computations to recommend items for a target user.

---

**Figure 11 CB and CF Approach**
In comparison with the SVD strategy, for this solution the offline component have more computation tasks to generate besides the users-specialities matrix, the similarity matrices and the shortest-path matrix, all this matrices essentials to compute the rating predictions.

5. EXPERIMENT DESIGN AND EVALUATION METRICS.

For all tested solutions in this work are required two objects as the main input parameter and the values contained in these matrices, perform the necessary computations for recommendations. The following matrices are based on our studied approach:

1. **USER-SPECIALITY MATRIX**: Matrix whose rows are formed by all users’ ID who have rated a speciality and the columns are all specialities ID rated by those users. Their cells contain the rating gave by a user to a particular specialty. With the current data the dimensions of the matrix is 2150x4430.

2. **USER-INGREDIENT MATRIX**: Matrix whose rows are formed by the same users' ID as the user-speciality matrix and the columns are the names of all ingredients that have been involved in the specialities that users have tasted. To calculate the rating of its cells, we will use a content-based algorithm using user-speciality matrix as input parameter to determining the value that a user gives a certain ingredient. From this process we obtain a matrix of 2150x300 of dimensions.

Because Onfan have not a rating option to quantify the user opinion on to a specific dish as part of functionality of its web, but grants a like, wish and tasted through a button click, we had to create a n intern ratings, these values will be assigned to each cells during the generation of user-speciality matrix.

Importantly, since Onfan allows users give till 3 ratings to the same speciality through the WISH, TASTE and LIKE buttons. Just as the cells in the user-specialty matrix will have the total sum of the values which are involved.

The table below shows the interpretation of the created ratings:

<table>
<thead>
<tr>
<th>RATING</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>WISH</td>
<td>3</td>
</tr>
<tr>
<td>TASTE</td>
<td>2</td>
</tr>
<tr>
<td>LIKE</td>
<td>1</td>
</tr>
</tbody>
</table>

The procedure to be followed to execute the experiments depends on the strategy used. For experiments using SVD, at the beginning we have executed iterations varying the constant dimension variable K, the idea is to find the best value as possible to use for both matrix user-specialties as user-ingredients matrices, all experiments using a test set of 10% total size of non-zero cells.

Furthermore, for the filtering algorithms the first experiments have varied the size of the test set, from 5% to 75% size of non-zero cells. Once we identified that the best results we obtain
with a 10% of total size. As next step, we have executed 10 iterations and later compute the average of these retrieved outcomes.

A number of different measures have been used in order to evaluate the performance of the filtering algorithms employed by Recommender Systems. Next, we will explain the two used to evaluate the accuracy in this work:

5.1. MEAN ABSOLUTE ERROR (MAE).

Measures the deviation of predictions generated by the Recommender System from the true rating values. The MAE is measured only for those items, for which user $u_i$ has rated on it. The predictions generated for those items are $p_{ij}$, for $j = 1, 2, ..., n_i$, while the real ratings provided from the user are denoted by $r_{ij}$, for $j = 1, 2, ..., n_i$.

$$MAE = \frac{\sum_{j=1}^{n_i} |r_{ij} - p_{ij}|}{n_i}$$

5.2. ROOT MEAN SQUARE ERROR (RMSE).

Puts more emphasis on larger absolute error. Is used to measure the residuals between values predicted by a model and the values actually observed from the environment that is being modelled.

$$RMSE = \sqrt{\frac{\sum_{j=1}^{n_i} (r_{ij} - p_{ij})^2}{n_i}}$$

6. RESULTS.

For the recommendation experiment, we vary the percentage of the test set size, taking as a reference set just the total of rated cells, obtaining a set of 10654 rated cells and the training data will be the remaining rated cells of the user-specialties matrix. Each chosen cell selected by the test set will be set to zero in its position on the original user-specialties matrix.

Furthermore, for SVD implementation we use different values of K in a range of [50, 275] to test the algorithms using scipy and svdlib, two python modules created to work with matrix decompositions.

6.1. SVD.

We run the low dimensional experiments for different fixed dimension K in specialties as ingredients. Furthermore, we have varied the size of test set in a range of [5, 20] percentage of rated cells, to validate how it influences the results set size.
In the left-side we can see the outcomes processing the user-speciality matrix and in the right-side we present the outcomes computed for user-ingredient as input parameter. We observe that behavior for both cases are complete different, while in specialities the values are not diverge a lot and tend to be a straight line, in ingredients the curve falls if the value of $K$ increases. Another factor to consider is the amount of sparsity in the data, it is 0.112% of rated cells (10654 non-zero cells taken from a 2150x4430 matrix).

As we can see in the below graphs, the behavior of the obtained curve using the SVDLib module is very similar to those obtained in Scipy. Here we note also that for best results, the value of $K$ should differ depending on the rating matrix used. Therefore, we can say that to deal the user-speciality matrix we can use both python modules with $K = 150$ because we got the lowest MAE, but for the users-ingredients matrix we should use $K = 275$.

The strategy to get better accuracy values, is always adjusted to the parameters which retrieve best results and since both matrices are processed by the SVD technique independently, it is possible to use two $K$ different values according to our interest.

6.2. ITEM-BASED CF.

In this experiment we have gradually increased the size of test set. We note that while larger our test set is (or smaller the training set), the error in prediction of ratings increases. So, only for the smaller size of test set, the error is lower.
6.3. USER-BASED CF.

Again, we have gradually increased the size of test set and as in item-based approach the best results were obtained with the smaller test set. If we compare these results with the before strategy, we see that the item-based gets lower error rates.

6.4. SOCIAL NEIGHBORHOOD-BASED CF.

The main advantage of social recommendation approach is that it incorporates the social network information, which helps predict users’ preferences.

The main advantage of social recommendation approach is that it incorporates the social network information, which helps predict users’ preferences. In the first social approach we add the shortest-path matrix catching only the users who are neighbors till 2 levels in-depth (that's it, the neighbor of my neighbor) as part of the trust neighborhood of a specific active user.
Comparing the results of this solution with the previous approach which involved only similarity matrix between users, we observe that the accuracy is very similar with a growing tendency if we use a bigger test set each time. These results are only a small improvement in predictions of ratings.

Finally as our last social-based solution, we have used a parameter $\alpha$, which balances the information from the user-user similarity matrix and the shortest path user matrix. If $\alpha = 0$, we only mine the user social matrix for matrix factorization but if $\alpha = 1$, we will use the similarity between users. In other words, we fuse information from the user-speciality rating matrix and the user social network for probabilistic matrix factorization.

$$UxU' = \alpha \cdot UxU + (1-\alpha) \cdot SPath$$

We observe that shortest path matrix impact the results, improving the accuracy. As $\alpha$ increases, the error is influenced just a bit and finally when $\alpha = 1$ the error grows faster. We
can say that using the social network graph we can get a better accuracy than just using user-user similarity.

As we can observe in all above plots, the best accuracy is obtained when we take a size of 10% of the total non-zero ratings as size of test set and as we have already mentioned that shortest-path matrix is most influential than the user similarity matrix, we have executed an additional test considering \( \alpha = 0 \) (or just taking shortest-path matrix) and the also execute the other two solutions where the user similarity is involved to verify which strategy retrieve better results.

We have applied a type of k-fold cross-validation process, repeating 10 times the computation. Each subsamples will be in a range between \([5, 20]\)% of size assigned to test set and the remaining as the training data. At final, the obtained results will be averaged to produce a single estimation per size.

![Figure 18 Comparison between social-based solutions](image)

As we see in the above specialties plot, we will obtain better results if we take into account only the shortest-path matrix as input parameter to compute the predictions, so we compute more accuracy recommendations based only in the trust neighborhood of each user.

6.5. INGREDIENT CONTENT-BASED.

Now, we will show the results obtained from the content-based solution. As we mentioned in its corresponding section, the ingredients are the words referenced to cuisine contained in the title and description of each speciality. In this case, the social strategy using the shortest-path matrix is not involved in the computations to predict the ratings.
Now, working with the same content approach we go to complete most of zero cells of user-ingredient matrix before to make the predictions of the test set. Thus, the new user-ingredient matrix will be less sparse than the original.

As we can see in the above plot, we have improved the accuracy of the rating prediction leaving this solution at the same level of results obtained by the SVD approach.

The following is the comparison graph between content-bases strategies. We use a k-fold process to obtain an average result for each test size. We can observe that the enhanced content-based solution retrieves better results than using the original user-ingredient ratings matrix and again only taking the 10% percentage of the total non-zero rated cells.
The specialties strategies return the lowest MAE when we use item-item similarity matrix. We do not see a significant error decrement when the user-user similarity or the shortest path matrix is involved in the computations. However we do see a difference in accuracy between specialties and ingredient strategies. It seems that more accurate when the prediction is based on ingredients rather than on specialties showing us that depending of the needs the strategy must be a CF process if we want to deal with user-speciality matrix or CB process with SVD if we must to work with user-ingredient rating matrix.

The following table summarizes the best computed results for each tested algorithm:

<table>
<thead>
<tr>
<th></th>
<th>RMSE</th>
<th>MAE</th>
<th>RMSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SPECIALTIES</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SCIPY</td>
<td>2,25</td>
<td>1,866</td>
<td>0,637</td>
<td>0,343</td>
</tr>
<tr>
<td>SVDLIB</td>
<td>2,245</td>
<td>1,865</td>
<td>0,663</td>
<td>0,367</td>
</tr>
<tr>
<td>IxI</td>
<td>1,058</td>
<td>0,666</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>UxU</td>
<td>1,447</td>
<td>1,130</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>UxU + SPATH</td>
<td>1,419</td>
<td>1,122</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>INGREDIENTS</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SCIPY</td>
<td>0,637</td>
<td>0,343</td>
<td>0,663</td>
<td>0,367</td>
</tr>
<tr>
<td>SVDLIB</td>
<td>0,637</td>
<td>0,343</td>
<td>0,663</td>
<td>0,367</td>
</tr>
<tr>
<td>IxI</td>
<td>0,666</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>UxU</td>
<td>2</td>
<td>1,17</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>UxI</td>
<td>0,966</td>
<td>0,558</td>
<td>0,674</td>
<td>0,389</td>
</tr>
<tr>
<td>UxI2</td>
<td>0,966</td>
<td>0,558</td>
<td>0,674</td>
<td>0,389</td>
</tr>
</tbody>
</table>

As we can see in the below scatter plot, if we use the collaborative filtering solution we obtain best results applying the items-items similarity matrix in the computation to predict values. But as we have mentioned, the best overall result is obtained by using the score users-ingredients matrix applying decomposition matrix technique SVD with the scipy module.
The gastronomic recommender will process the huge volume of existing information using the above mentioned algorithms and thus help to the Onfan users, to find specialties they wish or want to taste based on the preference of their trusted neighborhood, improving their experience through the social network.

In the computation of the recommendations, a balance needs to be struck between accuracy and the workload of the data. A reasonable accuracy can be achieved through strategies considering specialties using the similarities between items or content-based ingredients approach as part of the rating matrix. We have shown that if we break the specialties into food items, provides more accurate recommendations.

We tried different python modules that use SVD for generating recommendations and predictions and we see that reduces the dimension of the ratings matrix. The preprocess time was lower than the filtering algorithms and in accuracy the SVD-based approach was better than the other solutions processing users-ingredients matrix.

The content-based ingredient strategy improve in accuracy and always is better than the specialties strategy. So, this shows that the decomposition into ingredient items is beneficial for the purposes of recommendation in our data.

Within the bag of words contained in the description and comment fields, there are many words which denote feelings of happiness, satisfaction and in other cases sadness, discontent. For future works could use sentiment analysis to process this data and adapt these results to the current solution to improve the accuracy of the recommendations.

When we applies the filtering strategy on large data set the showed performance is impractical, unless it uses dimensionality reduction, sampling or partitioning. Compute all-pairs similarities can take an enormous amount of computation, it is not faster, it is more parallelizable, but entails doing more work.

The sparsity in the users-specialties matrix is a big problem during the computation of the predictions. The combination of two approaches makes more accurate predictions for a particular user, although the cost of execution time is high.

Since REDIS keys objects can contain strings, hashes and lists and currently it have contained the user and specialties relationships, the use of Map-Reduce would be a more effectively solution to process huge amounts of data, thus we can use ML algorithms to parallelize the
workload, accelerating the recommendation. The social-based recsys have been proved that they have better recommendation precision than traditional recommender systems, but they also face the problem of scalability. In the near future to continue improving the performance of the solution, we can use the work of Ch. He, Y. Tang, Zh. Yang, K. Zheng and Gu. Chen [21] which proposes the design of social recommender system based on Hadoop, composed by the following cores: Friends recommendation module, similar user, user community and content-based recommendation module to provide a perfect scalable solution and leaves ready the solution to deal with Big data cases.

In addition to the cuisine recipes already downloaded in English and Catalanz language, continue repopulating the recipes storage with at least 3 languages more: Spanish, French and Italian for instances. Furthermore, having a big repository and improving the current data matching process, we can match the Onfan data with the new downloaded recipes. Thus, we can use ingredients network techniques to measure whether it tends to be essential or can be dropped or added from recipe, capture the relationships between ingredients, ingredient's frequency on recipes and capture the users’ preference for healthier variants of a recipe. For instance we can implement a solution similar like the worked by C. Teng, Y. Lin and L. Adamic, in their research [20].

8. REFERENCES.


