Roommates and Apartments Recommendation System

Zhicheng Xu
Electrical Engineering
Email: zx2170@columbia.edu

Yuzhong Ji
Electrical Engineering
Email: yj2345@columbia.edu

***Abstract*—The goal of a Recommender System is to generate meaningful recommendations to a collections of users for items or products that might interest them. Suggestions for books on Amazon, or movies on Netflix, are real world examples of the operation of industry-strength recommender systems. Recommender systems differ in the way they analyze these data sources to develop notions of affinity between users and items which can be used to identify well-matched pairs. The traditional systems use algorithm like k-nearest neighbor collaborative filtering based ones, are achieving widespread success on the Web. The tremendous growth in the amount of available information and the number of visitors to Web site in recent years poses some key challenges for recommender systems. They need to produce high quality recommendations, perform many recommendation per second for millions of users and items, and achieve high coverage in the face of data sparsity. So new recommender system technologies are needed that can quickly produce high quality recommendations, even for large-scale problems.**

In this paper we build our system based on collaborative filtering techniques. We analyze item similarity computation algorithms including Cosine-based Similarity and Correlation-based Similarity. And we look into different recommendation generation algorithm including neighborhood-based collaborative filtering algorithm and item-based collaborative filtering algorithm. The goal of our system is to recommend roommates and apartments for users in our website. To test our system, we use the data from meetup.com. And we use neo4j to convert the relationship netwrok to visualized graph.

Keywords*—*Recommend System, Collaborative Filtering, Similarity, Housing Website, Meetup, Neo4j.

I. Introduction

The amount of information in the world is increasing far more quickly than our ability to process it. Technology has been developed to reduce the barriers to publishing and distributing information. In our former project, we built a website for house rental around universities like Columbia University or NYU. One of the features of our website is that everyone can collect the houses or apartments information and discuss with someone who is also interested in this apartment. This is really beneficial for students who want to share apartments with others. We all know that finding a good roommate is important and living with someone you don't like is very annoying. So based on the original idea, we want to do it further and recommend roommates and apartments for people who is looking for apartments or roommates.

So now the problem is to find technologies that can help us go through all the available information to find that which is most valuable to us. One of the most promising technologies is collaborative filtering. Collaborative filtering works by building a database of preference for items by users. Using this preference database, we can find the preference similarity between two people. And based on these similarities, we can recommend items for users. Collaborative filtering has been very successful in both research and practice especially information filtering applications and E-commerce applications.

This paper is organized as following. In the second part, we introduce some related work with recommend system and collaborative algorithm. In the third, we give an overview about our recommend system and data collected from Meetup network. In part IV and V, we talk about our work in pulling data and implementation of collaborative filtering algorithm. In part VI, we give result of our simulation and use Neo4j to present our result visually. In the last part, we present the real application in our website.

II. Related Work

The term "collaborative filtering" was introduced in the context of first commercial recommend system, called Tapestry, which was designed to recommend documents drawn from newsgroups to a collection of users. The motivation was to leverage social collaboration in order to prevent users from getting inundated by a large volume of streaming documents.

Initial formulation for recommender systems were based on straightforward correlation statistics and predictive modeling, not engaging the wider range of practices in statistics and machine learning literature. Further research was spurred by the public availability of dataset on the web, and the interest generated due to direct relevance to e-commerce. Netflix, an online streaming video and DVD rental service, released a large-scale dataset containing 100 million ratings given by about half-a-million users to thousands of movies tittles, and announced an open competition for the best collaborative filtering algorithm in this domain. Currently, Recommender Systems remain an active area of research, with a dedicated ACM conference, interested several sub-discipline of statistics, machine learning, data mining and information retrievals. Application have been pursued in diverse domains ranging from recommending webpages to music, books, movies and other consumer products.

III. System Overview

In this section, we first present the theoretical framework of constructing and analyzing the components of our recommend

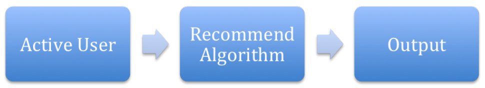


Fig. 1. Basic System Architecture

$$\begin{array}{cc|c|c|c|c|c|c|}
 & & \multicolumn{6}{c}{\textit{Items}} \\
 & & \textit{1} & \textit{2} & \ldots & i & \ldots & m \\
\cline{3-8}
 & \textit{1} & 5 & 3 & & 1 & 2 & \\
\cline{3-8}
 & \textit{2} & & 2 & & & & 4 \\
\cline{3-8}
\textit{Users} & : & & & 5 & & & \\
\cline{3-8}
 & u & 3 & 4 & & 2 & 1 & \\
\cline{3-8}
 & : & & & & & 4 & \\
\cline{3-8}
 & n & & & 3 & 2 & & \\
\cline{3-8}
\multicolumn{8}{c}{} \\
\cline{3-8}
 & a & 3 & 5 & & ? & 1 & \\
\cline{3-8}
\end{array}$$

Fig. 2. User Ratings Matrix

system. And then, we would like to introduce the structure of Meetup.com social network and how to apply the data from Meetup.com to our housing recommendation system.

A. Basic System Architecture

The system structure is composed of three parts (shown in figure 1): Active User Information Input, Recommendation Algorithm and Recommended Output. This is a very clear structure to implement our system.

The input information of active user contains ratings of different items. In recommendation algorithm part, we have a user preferences database. The most general setting which presents our recommend system is shown in figure 2. The database is represented as a matrix of n users and m items. The user ratings matrix is typically sparse, because user don't rate all items. The output of our system is people who have the same interest with you and items which you may like.

We use collaborative filtering algorithm in Recommendation Algorithm part. The goal of a collaborative filtering algorithm is to suggest new items or to predict the utility of a certain item for a particular user based on the user's previous preference and the options of other like-minded users. Here we define n users $U = \{u_1, u_2, u_3, ..., u_n\}$ and a list of m items $L = \{i_1, i_2, i_3, ..., i_n\}$. Each user u_i has a list of items I_{u_i}, which the user has expressed his/her preference. Preference can be explicitly given by the user as a rating score, generally within a certain numerical scale. There exists a distinguish user $u_a \in U$ called the active user for whom the task of a collaborative filtering algorithm is to find an item likeliness that can be of two forms.

- **Prediction** is a numerical value, $P_{a,j}$ expressing the predicted likeliness of item $j \notin I_{u,a}$ for the active user u_a. This predicted value is within the same scale as the opinion values provided by u_a.
- **Recommendation** is a list of N items, $I_r \subset I$, that the active user will like the most. Note that the recommended list must be on items not already purchased by the active user i.e., $I_r \cap I_{u_a} = \Phi$.

B. Meetup Social Network

Since our website is still developing, we have little data about user preference. To solve this problem and test our recommend system, we come up with an idea to use data from social website like Twitter, Facebook. Finally, we decide to use Meetup.com because the structure of this social website corresponds to our demand perfectly.

Different from the classical social network structure as presented in the case of Facebook and Twitter, the group/event-based characteristic of Meetup has made it an interesting and special type of network. The Meetup online network is organized around group: users can choose to join multiple groups of their inters. While the offline network is organized around events that are hosted by groups. Sander and Seminar attended 40 social events in Meetup and concluded that participants in Meetup social events have social structures instead of just strangers meeting strangers. Liu, and his colleges also conducted a comprehensive research on the Meetup community structure using web-crawled data from Oct 2011 to Jan 2012: clear definitions are given and properties are analyzed.

Given the characteristics, we hope to use data from Meetup.com to simulate the preference for different apartments. We can view the groups that one user joined as the apartments that user collected so now we have ”users” and their favorite ”apartments”. Besides, data from Meetup is also one way to recommend roommates since people prefer to live with someone who shares interests with him or her.

C. Recommendation System Platform

Given the meetup data source, we can divide our system into the three components which we will discuss in later chapters.

1) Data Collections and Data Clean.
2) Algorithm Implementation and Result Analysis.
3) Web Application Demonstration

IV. Meetup Data Collection and Clean

We write a java program to pull data from meetup.com. Meetup.com server provide APIs (Shown in figure 3) for data collection including group ID, tag ID, event ID, user ID as well as the detailed information about event, groups and users. For our system, we hope to use the relationship between users and groups to simulate people's fond for apartments and houses. So we remain the user ids and group ids for computing convenience. To get the detailed information about the users or groups, we can send url request to meet up API and get the response with detailed information.

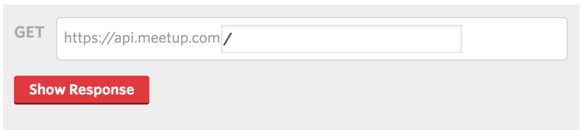


Fig. 3. Meetup API

Total data included 18212 groups and 1294827 users. However, about 50% users join only one group. These samples are less useful in our algorithm so we try to remove them before processing the data. We compare the result between data before removing these users and data after removing these users and find that there's not much difference. The reason may be that users with only one group is less like to have great similarity between other users except the other user has exactly the same group.

V. Collaborative Filtering Algorithm

In this section we introduce Collaborative Filtering Algorithm and steps to implement the algorithm. Different methods are introduced and compared in this part.

Collaborative Filtering (CF) systems work by collecting user feedback in the form of rating for items in a given domain and exploiting similarities in rating behavior among several users in determining how to recommend an item. CF methods can be further divided into two different methods: **neighborhood-based** and **model-based** approaches. In our system, we use neighborhood-based collaborative filtering.

A. *Neighborhood-based Collaborative Filtering*

In neighborhood-based techniques, a subset of users are chosen based on their similarity to the active users. The prediction is based on the similarity weights and ratings of known users. The steps of neighborhood-based collaborative filtering can be summarized as following:

1) Assign a weight to all users with respect to similarity with the active user.
2) Select k users that have the highest similarity with the active user, we can them neighborhood.
3) Compare a prediction from a weighted combination of the selected neighbors' ratings.

In step 1, the weight $w_{a,u}$ is a measure of similarity between the user u and the active user a. The most commonly used measure of similarity is the Pearson correlation coefficient between the ratings of the two users, defined as:

$$w_{a,u} = \frac{\sum_{i \in I}(r_{a,i} - \bar{r}_a)(r_{u,i} - \bar{r}_u)}{\sqrt{\sum_{i \in I}(r_{a,i} - \bar{r}_a)^2 \sum_{i \in I}(r_{u,i} - \bar{r}_u)^2}} \tag{1}$$

Where I is the set of items rated by both users, $r_{u,i}$ is the rating given to item i by user u, and $\bar{r}_u$ is the mean rating given by user u.

Another common used similarity computation is cosine-based similarity. We can treat the ratings of two users as a vector in an m-dimensional space, and compute similarity based on the cosine of the angle between them, given by:

$$w_{a,u} = cos(\vec{r}_a, \vec{r}_u) = \frac{\vec{r}_a \cdot \vec{r}_u}{||\vec{r}_a||_2 \times ||\vec{r}_u||_2} \tag{2}$$

When computing cosine similarity, we cannot have negative ratings, and unrated items are treated as having a rating of zero. Empirical studies have found that Pearson correlation generally performs better so in our system we use Pearson correlation for similarity computation. There are also several other similarity measures used in the literature, including Spearman rank correlation, Kendall's τ correlation, mean squared differences, entropy, and adjusted cosine similarity.

In step 3, predictions are generally computed as the weighted average of deviations from the neighbor's mean, as in:

$$p_{a,i} = \bar{r}_a + \frac{\sum_{u \in K}(r_{u,i} - \bar{r}_u) \times w_{a,u}}{\sum_{u \in K} w_{a,u}} \tag{3}$$

Where $p_{a,i}$ is the prediction for the active user a for item i, $w_{a,u}$ is the similarity between users a and u, and K is the neighborhood or set of most similar users.

B. *Item-based Collaborative Filtering*

When applied to millions of users and items, conventional neighborhood-based CF algorithms do not scale well, because of the computational complexity of the search for similar users. As an alternative, Linden proposed $item-to-item$ Collaborative Filtering where rather than matching similar users, they match a user's rated items to similar items. In practice, this approach leads to faster online systems, and often results in improved recommendations.

In this approach similarities between pairs of items i and j are computed offline using Pearson correlation, given by

$$w_{i,j} = \frac{\sum_{u \in U}(r_{u,i} - \bar{r}_i)(r_{u,j} - \bar{r}_j)}{\sqrt{\sum_{u \in U}(r_{u,i} - \bar{r}_i)^2 \sum_{u \in U}(r_{u,j} - \bar{r}_j)^2}} \tag{4}$$

Where U is the set of all users who have rated both items i and j, $r_{u,i}$ is the rating of user u on item i, and $\bar{r}_i$ is the average rating of the ith item across users.

Now, the rating for item i for user a can be predicted using a simple weighted average, as in:

$$p_{a,i} = \frac{\sum_{j \in K} r_{a,j} w_{i,j}}{\sum_{j \in K} |w_{i,j}|} \tag{5}$$

Where K is the neighborhood set of the k items rated by a that are most similar to i.

C. *Implementation of Algorithm*

In our system we write C++ program to implement the algorithm because of the fast data processing characteristics of C++. We use two similarity computing methods introduced above: Cosine-based Similarity and Pearson Correlation. For each algorithm, we will find the five most similar users and recommend them to active user.

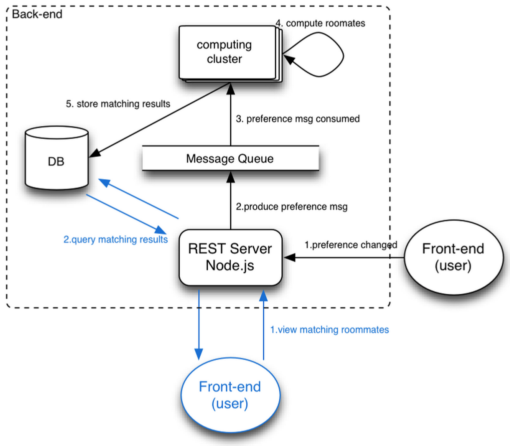


Fig. 4. Web Application Architecture

As for group recommendation, we implement both neighborhood-based collaborative filtering and item-based collaborative filtering. The comparisons between the two algorithms are shown in the last part. We will recommend five groups with the highest weight to active users.

The biggest problem for data is that we can't get the exact rates about the groups. All we get is whether a user is in a group. So we simplify the algorithm and set the rate to be 1 if user is in the group and set the rate to be 0 if user is not in the group.

VI. Web Application Architecture

A. Introduction of web

This website is designed to help you reserve your ideal apartments and find suitable roommates. Once you register in the website, the system can recommend suitable apartments and roommates according to your preference and browser history. Our algorithm is applied in the most important part-recommendation system.

B. Work flow of web

In the web application, users can add their suitable apartments in a wishlist. In this way, we can get the data from the front end when users view our website. By default, the recommendation system will retrieve the data from database and compute the recommendation data every two hour. After finishing, it will store the recommended result in the database. The system will return the recommended result once users send 'recommend apartments' request. In addition, we can help users to match their roommates. Every time, uses can edit their ideal roommates preference. Then the server will produce preference message which be sent to a message queue. The queue will notify the computing cluster to compute users' potential roommates. After waiting several seconds, the users can get their matching roommates. Feature 4 shows the web application architecture of our recommend system.

TABLE I. Active User Group

Group Id	Group Name
113455	Resilience NYC Meetup Group
176399	NY Tech Meetup
225219	The New York City Meetup Organizer Meetup Group
306773	The New York Permaculture Meetup
366067	Goodnik
431480	New York Society for Ethical Culture
771777	Brooklyn Drum Circles
976696	NYC BLUEGRASS SLOW JAM (for Adv beginners and Int players)
1183598	Health 2.0 NYC - The New York Healthcare Innovation Group

TABLE II. Recommended Users and weights of Cosine-Based Similarity and Pearson Correlation

Cosine-Based Users	Weight	Pearson Correlation Users	Weight
8684032	0.866025	8684032	0.866025
13577150	0.866025	13577150	0.866025
2361358	0.816497	2361358	0.816497
30149182	0.816497	30149182	0.816497
22579941	0.816497	22579941	0.816497
22399581	0.816497	22399581	0.816497
32630102	0.816497	32630102	0.816497
153513272	0.816497	153513272	0.816497
149402652	0.816497	149402652	0.816497

VII. Result and Discussion

A. Introduction of Neo4j Graph Database

After data collection and algorithm implementation, we have complete recommendation data result. The next thing to do is to find a good way to present the results to users. We find that Graph Database is suitable considering our data structure.

The Neo4j graph system is composed of three essential elements: node, property, and edge. The Meetup.com network has 4 types of node: group, user, event and tag. As for our system, the network is composed of two nodes: groups and users. The edges between users and groups indicate that users are these groups.

B. Roommate Recommendation

The result of user recommendation are shown in table 1. We use the active user (USER ID: 6). Column 1 shows the groups active user is in. The second column is the recommend roommates and the third column is the weight value.

The recommend users of Cosine-Based Similarity and Pearson Correlation are shown in Table II. We can see from the table that the results of two methods are exactly the same. That's because in large data, the correlation method is similar to the Cosine-Based similarity. However, as for the execution time. The cosine-based similarity is much faster than Pearson Correlation.

Using the Neo4j Graph database, we can have more clear observation of the result. The relationship network is shown in Feature 4. The blue nodes are the users and the grey nodes are the groups. The edges shows that users are in the groups. In the center of the network is the active user (ID 6). Grey nodes around the active user are groups that active user is in. We can see that most recommend shares one or more groups with active user, that proves the correctness of our algorithm.

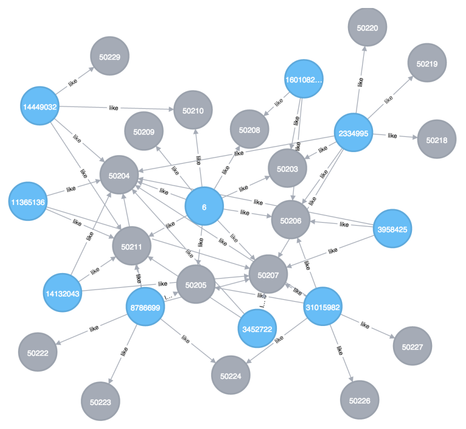


Fig. 5. Users and Groups Network

TABLE III. RECOMMENDED GROUP ID AND NAME OF NEIGHBORHOOD-BASED CF

Group Id	Group Name
338496	The Piscataway Zen Meditation Meetup
509584	The Lehigh Valley Art of Living Meditation
808691	The Partners of People with M4K Mustaches Support Group
316744	The Literary RPG Society of Westchester
1184411	Malvern Ultimate Frisbee
1164380	Science Fiction & Fantasy Writers Round Table
1054385	East Coaster Crusaders
557066	Suffolk County Writers Workshop
176399	NY Tech Meetup
976696	Philadelphia Area Society of Intuitives,Mediums

TABLE IV. RECOMMENDED GROUP ID AND NAME OF ITEM-BASED CF

Group Id	Group Name
344002	The Brooklyn Parents Meetup Group
648888	Greater New Haven Unknown Zone Meetup Group
696005	Branford Ballroom Meetup
372546	Women's Small Business Association (WSBA) - Lehigh County
785216	Be Wild Woman
288727	Meetup Groups v2
455847	The NYC Uterine Fibroids Meetup Group
882193	The New York OpenAFS User Group
836316	Indians in Westchester and Fairfield Co
380315	Philadelphia Area Society of Intuitives,Mediums & Empaths

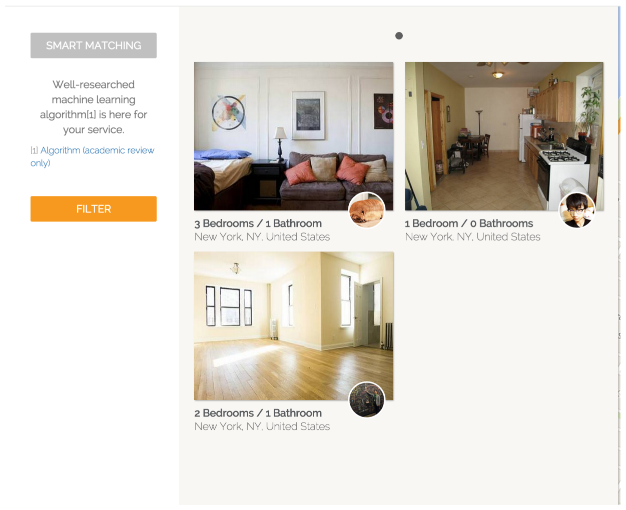
C. Collaborative Filtering Analytics

For the groups(apartments) recommendation, we compare the results of Neighborhood-Based CF and Item-Based CF. The result of the two algorithm are shown in Table III and Table IV.

We can see that the two recommendation are totally different. The result of Neighborhood-Based CF is relevant to the original group. However, the Item-Based CF is not accurate because the ratings of groups are either 1 or 0. So the calculated weight of each group is either 1 or 0. In this way, we can't tell the difference between different groups. However, this algorithm will be useful if the rating is not fixed to two values. We compare the exciting time of the two algorithms and find that Neighborhood-Based CF needs 93.882164 seconds and Item-Based CF only needs 4.41963 seconds. That will be particularly useful in website application.

D. Real Website Demonstration

The real website is shown in Feature 6. When clicking the ”Smart Matching” button, we will recommend four apartments for you. And someone who also collect this apartment is shown in the lower right corner.



Well-researched machine learning algorithm[1] is here for your service.

[1] Algorithm (academic review only)

3 Bedrooms / 1 Bathroom
New York, NY, United States

1 Bedroom / 0 Bathrooms
New York, NY, United States

2 Bedrooms / 1 Bathroom
New York, NY, United States

Fig. 6. Users and Groups Network