Collaborative Filtering Based Simple Restaurant Recommender

Umar Farooque
Dept. of Computer Science
Jamia Hamdard University
New Delhi, India
ger.ballack@gmail.com

Bilal Khan
Dept. of Computer Science
Jamia Hamdard University
New Delhi, India
bkthegerman@gmail.com

Abidullah Bin Junaid
Dept. of Computer Science
Jamia Hamdard University
New Delhi, India
abid.abidullah@gmail.com

Akash Gupta
Dept. of Computer Science
Jamia Hamdard University
New Delhi, India
ak.akashgupta@hotmail.com

***Abstract*—The use of Collaborative Filtering is becoming very popular in designing a simple yet efficient recommender system. A recommender system based on Collaborative Filtering basically predicts a user's interest in some item on the basis of the scores generated and the correlation calculated between the users. In this paper we propose a basic structure and steps of designing a recommender system that uses Collaborative Filtering (user based) along with applications of partitioning and clustering of data, thus designing a Restaurant Recommender System. The proposed system reduces the complexity and gives a clear view of the basic approach to build a recommender system from scratch.**

***Keywords*—Collaborative Filtering, Pearson Correlation, Recommender System, Vector cosine similarity, Z- score.**

I. INTRODUCTION

In the past decade the world has observed a steep rise in the use of e-commerce [12] in various domains of life. This has been possible due to the advancements in the field of online data security. People feel safe to make online transactions and hence, the market of online services has evolved to a great extent. In today's era, the consumer relies a lot on peer reviews and ratings to make a choice. This makes online ratings and reviews a significant asset in the process of online marketing and form the backbone of the recommender system [14]. Though there are a lot of online service providers that make use of a number of rating parameters to target their marketing campaigns, but still there is room for improvements [6]. In this paper, we propose a basic recommender system [15] that gives personalized ratings for a user and recommends products or services based on the ratings, generated by the system [13]. A rating is basically a measure of user's taste and preferences. The system predicts how much a user would like or dislike a particular item and generates a numerical measure of that [3].

II. RELATED WORK

Researchers in the field of data mining are shifting their focus from normal mining operations to more complicated computations that are done on the mined data [5]. There are many mining techniques available which are efficient and simple enough to extract meaningful data, but there is large amount of scope for improvements and innovations in use of that meaningful data obtained [4]. One of the uses comes in the form of recommender system. Collaborative Filtering, a term coined by Goldberg [10], is a concept of information filtering [9]. Goldberg, published a paper using collaborative filtering techniques in an email filtering system named, Tapestry. Nicholas Ampazis [11] applied the methods of decomposition in order to achieve the same on Netflix Dataset.

.

III. OVERVIEW OF THE SYSTEM

Before The whole process of recommending the restaurants to the user and summary of the steps involved can be described in the following stages and the fig 1**.**

A. Stage 1

The user enters the budget for his meal. Furthermore, the user can specify the cuisine along with the budget for the meal. This would partition our dataset and help in reducing the number of operations performed on the dataset considerably.

978-93-80544-12-0/14/$31.00

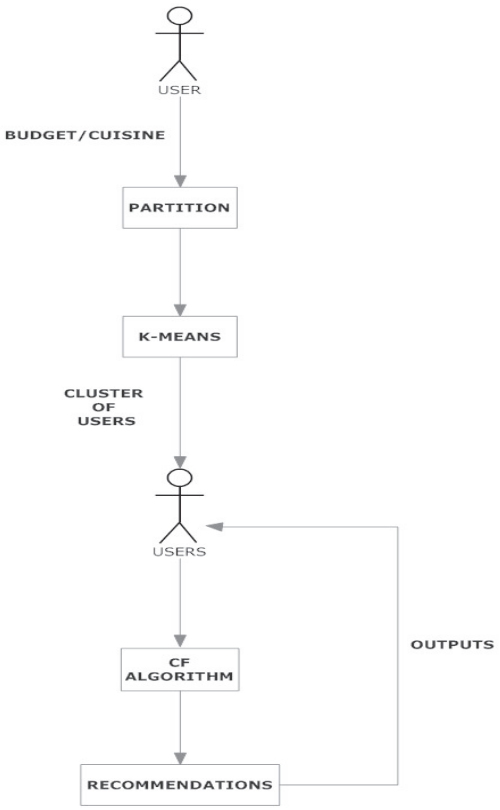

Fig. 1 Overview of the system

B. Stage 2

After partitioning, K mean clustering algorithm would be applied on the user dataset in order to identify the cluster user belongs to [18]. If the current user is a new user, then the system would ask the user to select and rate the restaurants that have been visited by the user. This would help in obtaining information about the user's preference and hence put the user in suitable cluster.

C. Stage 3

After applying clustering, the cluster for the current user will be identified. Now, the members of the cluster identified will be referred as neighbors [5]. All the operations performed in finding the various weights, score, and further generating the recommendations would include the current user's neighbor.

D. Stage 4

Various similarity functions would be used to generate the weight for each neighbor (member of the current user's cluster) in relation to the current user. After the similarities have been discovered, scores [2] will be generated using the similarities, for the various restaurants and the user would be displayed the top 5 restaurants based on the score of the restaurants.

IV. RESEARCH METHODOLOGY

This section describes the order of the computation steps involved, the formulae used and their description.
How to proceed:

1. Finding and selecting the neighborhood for current user.
2. Computing similarities using various similarity functions.
3. Normalizing data.
4. Scoring items based on neighborhoods.

A. Select neighborhood (Clustering Step)

This step defines the neighbors of the current user. Basically it involves the simple K-mean clustering of the user dataset [5]. The number of cluster would depend on the total size of the datasets. Ideally, the size of cluster should be around 30-50 members each. Some other methods to find the neighbors if clustering is to be avoided are as follows:

a. Selecting all neighbors.
b. Selecting users on basis of threshold of similarity or differences.
c. Selecting random neighbors.

It has been found that clustering doesn't have many advantages in some situations. It can increase the number of computations. In such cases, the methods stated above can be used as they don't tend to create much variations on the results produced.

B. Compute similarity [sim(a,u)]

This is an essential step involved in generating the predictions. Similarity is also known as weight in some prediction methods. It is calculated in order to find the relation of the current user with its neighbors. The function determines that how much value or weight should be given to a neighbor's rating for a particular item, depending on the similarity or dissimilarity of the neighbor with the current user. The value or weight of the neighbor multiplied by the ratings given by the neighbor for a particular item gives us the true state of the rating in reference to our user. There are many similarity functions used. Some of them discussed and used in the paper are:

1) Pearson's Correlation as similarity [6].

$$sim(a,u) = \frac{\sum_i (r_{ai} - \mu_a) * (r_{ui} - \mu_u)}{\sqrt{\sum_i (r_{ai} - \mu_a)^2} * \sqrt{\sum_i (r_{ui} - \mu_u)^2}} \quad (1)$$

where,
r = rating of an item by a user.

a = the user whose similarity is to be compared.
u = users in the neighborhood of user a.
i = items rated by both user a and u (mutual items).
μ = mean or the average rating by user.

Advantage of using Pearson's correlation as similarity is that there is no need for user normalization or user mean normalization as it is already subtracting the user's mean.
There are certain drawbacks in the Pearson's correlation as similarity**.** Consider a situation where there is only 1 rating in common for the users, then the result would be 1. This is not a desirable situation, as having 1 common rating would make the similarity 1, whereas the user may have different ratings for the non-mutual items. Having just 1 mutual isn't a true measure of similarity and this would not be convincing enough to have confidence on the ratings generated.
This issue can be tackled by using some significance weighting. In significance weighting, the number of ratings below a certain minimum, would cut the ratings or reduce the significance of the ratings on calculation of the similarity. In other words, this would down scale the similarity, if the number or ratings are below a certain minimum cut off.

2) Vector Cosine Similarity

$$sim(a,u) = \frac{\sum_i \hat{r}_{ai} \cdot \hat{r}_{ui}}{\sqrt{\sum_i (\hat{r}_{ai})^2}\sqrt{\sum_i (\hat{r}_{ui})^2}} \quad (2)$$

where,

$$\hat{r}_{ai} = r_{ai} - \mu_a$$
$$\hat{r}_{ui} = r_{ui} - \mu_u$$

r = rating of an item by a user.
a = the user whose similarity is to be compared.
u = users in the neighborhood of user a.
i = items rated by both user a and u (mutual items).
μ = mean or the average rating by user.

Vector similarity [6] is computed for common ratings, that is 0 is considered for a rating given by one user but not the other. Unlike Pearson's correlation similarity, normalization can be applied in vector cosine similarity. In Pearson's correlation normalization wasn't much useful for similarity as it had an in built normalization factor (user's mean) in the formula. This may appear exactly same as the Pearson's correlation but it has a different suggestion and removes the need of significance weighting. The normalized vector cosine similarity assumes an average rating for the unrated items by the users for the purpose of similarity. But more importantly, it scales the ratings by a factor:

$$\frac{|R_a \cap R_u|}{|R_a||R_u|} \quad (8)$$

where,
R_a = rating by user a.
R_u = rating by user u in neighborhood of user a.

So, normalized vector cosine similarity has a built in dampening effect that comes in to action when the number of mutual ratings is low. So it removes the need of any significance weighting and is preferred over Pearson's correlation as similarity.

C. Normalization

Normalization is done in order to compensate for the difference in rating style of the various users. A user may have different rating style like, some user would rate the items very highly, some would give moderate ratings, some would give extreme ratings that is, minimum or maximum, etc. In order to bring these rating style of various users on a compensated scale where the ratings can be compared with each other without ignoring these considerations, normalization is done. Averaging the ratings can ignore this consideration rather than creating a mutual scale, so it is not advised to use it. Some other methods are available which deal with the problems present in simple averaging, they are as follows:

1) Mean Centering

$$P(a,i) = \frac{\sum_u (r_{ui} - \mu_u) * sim(a,u)}{\sum_u sim(a,u)} + \mu_a \quad (3)$$

where,
r = rating of an item by a user.
a = the user whose similarity is to be compared.
u = users in the neighborhood of user a.
i = items rated by both user a and u (mutual items).
μ = mean or the average rating by user.

In mean centering of the data the average of deviance of each user's mean rating is taken into consideration, which overcomes the problems of averaging.
Another method to normalize the ratings is by using, what is called the Z score of the users.

2) Z-score

$$Z_{ui} = \frac{r_{ui} - \mu_u}{\sigma_u} \quad (4)$$

where,
r = rating of an item by a user.
u = users in the neighborhood of user a.
i = items rated by both user a and u (mutual items).
μ = mean or the average rating by user
σ = standard deviation

D. Scoring items based on neighborhoods.

This section deals with the scoring items or generating the prediction for an item with reference to a particular user. Here the concept of collaborative filtering (CF) is discussed in brief. Collaborative Filtering [7] is basically a technique used for generating scores for various items in a recommender system by comparing the ratings of a current user with that of another

similar user and predicting rating for the current user for a unrated item. The similarity between the users is taken into account in collaborative filtering [10].The concept of collaborative filtering is based on the fact that often people with similar taste as the person in consideration give the best recommendations. There is basically two mechanisms of collaborative filtering, user-user collaborative filtering and item-item collaborative filtering. In our approach, we have used user-user collaborative filtering.
The main steps involved in the collaborative filtering are:

1) In this method basically users with similar rating patterns like the current user are identified.
2) Using the ratings of the similar users found in the above step, the prediction is calculated for the current user's rating.

Personalized Prediction method used in the computation of the scores for the user [1].

$$P(a,i) = \bar{r}_a + \frac{\sum_{u=1}^{n}(r_{ui} - \bar{r}_u) * \sum_{u=1}^{n} w_{au}}{\sum_{u=1}^{n} w_{au}} \quad (5)$$

where,

$$w_{au} = \frac{\sum_{i=1}^{m}(r_{ai} - \bar{r}_a) * (r_{ui} - \bar{r}_u)}{\sigma_a \sigma_u} \quad (6)$$

where,
r = rating of an item by a user.
u = users in the neighborhood of user a.
i = items rated by both user *a* and *u*(mutual items).
μ = mean or the average rating by user
σ = standard deviation
n = total users in the neighborhood.

The formula (5), (6) takes into account all the factors required for predicting a rating for a user for a particular item. It deals with the following:

a. For every user u in the cluster (or for every neighbor of the current user a) the rating r for an item i, compared with their normal rating.
b. For every user u in the cluster (or for every neighbor of the current user a) the weight (similarity) for the user u in comparisons to the current user a.
c. Adding the number generated back to the average rating of user a.

Note that, this formula of predicting score (5) almost remains the same with just a minor variations, it's the similarity factor that changes for different approaches and hence the different results. In other words, similarity or the weighing factor plays the major role in determining the ratings/predictions.

E. Floor and ceiling Functions

This has a certain issue. The predictions generated for item i's rating can get out of range. In other words, the ratings predicted for a particular item have the tendency to get out of range of the acceptable rating. For a range of ratings 1 to 5, the prediction generated maybe more than 5 or less than 6.This is quite usual in such systems. It can be easily brought back into range by applying ceiling and floor functions to the predictions. This doesn't have any effect on the predictions as a high prediction of rating 6 for an item on a rating range of 1 to 5 would still mean 5 and similarly for a low prediction rating.

V. SYSTEM EVALUATION

This system of recommendation was evaluated on a subset of the Restaurant ratings data collected from ChefMoz data set. ChefMoz has more than 100,000 reviews. Random numbers of users were taken and various methods of scoring and similarity were implemented. First the similarity functions, Cosine vector (2) and Pearson's correlations (1) were implemented on the dataset, then normalization was applied on the ratings and finally prediction/scoring methods were used. The prediction/scoring methods used were Pearson' weighted mean (5), (6), Z-Score prediction (7), and Normalized cosine vector score (2), (5).
Scoring / Prediction formula for Z-score.

$$P(a,i) = \frac{\sum_u Z_{ui} * sim(a,u)}{\sum_u |sim(a,u)|} * \sigma_a + \mu_a \quad (7)$$

Using the formulae the respective predictions/scoring were produced. The following table displays the results produced by the various prediction (scoring) methods.

TABLE I. RESULT TABLE

PREDICTIONS FOR USER U1068 USING VARIOUS METHODS.				
SCALE : 0-BAD, 1-AVERAGE, 2-GOOD				
ID#	**ACTUAL RATING**	**PEARSON' WEIGHTED MEAN**	**Z-SCORE**	**COSINE VECTOR (NORM)**
132584	-	0.0	0.0	0.0
132733	1.0	2.0	2.0	2.0
132732	0.0	0.0	0.0	0.0
132630	-	0.0	0.0	0.0
135104	1.0	1.0	1.0	1.0

VI. CONCLUSION

The partitioning we used before the clustering plays a vital role in reducing the computations and increasing the accuracy of the score/predictions. The partitioning step further reduces the load on clustering as it already divides the data set and keeps the relevant data in relation to the user's preference that is , in our case user's budget or cuisine of preference. Finally, out of various scoring functions available, we found Normalized Cosine Vector scoring to be optimum for our dataset. The advantage of Normalized Cosine Vector scoring being that it has a built in dampening effect that eliminates the need of any significance weighting and is preferred over Pearson's correlation as similarity. The ratings generated by using the vector cosine similarity were found to be more accurate and easy to obtain.

REFERENCES

[1]. Joseph Konstan and John Riedl. "AI Techniques for Personalized Recommendation", IJCAI 2003 University of Minnesotta, Minneapolis, pp 126-131.

[2]. Mark O'Connor and Jon Herlocker. Clustering Items for Collaborative Filtering. ACM-SIGIR Workshop on Recommender Systems,1999,58-62

[3]. Daniel Lemire and Anna Maclachlan. Slope One Predictors for Online Rating-Based Collaborative Filtering. Proceedings of SIAM DataMining (SDM'05), 2005 pp 471-476

[4]. Mittal, N., Nayak, R. ,Govil, M.C.; "Recommender System Framework Using Clustering and Collaborative Filtering" Emerging Trends in Engineering and Technology (ICETET), 2010 3rd International Conference ,pp 978-0-7695-4246-1 .

[5]. Han, J., and Kamber, M. 2000. Data mining: Concepts and Techniques. New York: Morgan- Kaufman. page 135-140.

[6]. Daniel Lemire Anna Maclachlan' Slope One Predictors for Online Rating-Based Collaborative Filtering In SIAM Data Mining (SDM'05), California, April 21-23, 2005.

[7]. Glenn Fung. A Comprehensive Overview of Basic Clustering Algorithms. June 22, 2001:153-158

[8]. Sarwar, B. M., Karypis, G., Konstan, J. A. and Riedl, J..2001. Item-based Collaborative Filtering Recommendation Algorithms. WWW Conf. 2001, pp. 285- 295.

[9]. ACM. Special issue on information filtering. Communications of the ACM, 35(12), Dec 1992.

[10]. David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. Communication of the ACM, 35(12):61-70, December 1992.

[11]. Nicholas Ampazis.Collaborative Filtering via Concept Decomposition on the Netflix Dataset. The netflix prize challenge, SIGKDD Explorer. News, 2007: 43-47.

[12]. N. Leavitt, "Recommendation technology: will it boost: ecommerce?", IEEE Computing Society, vol. 39, no. 5, pp.13-16,May 2006.

[13]. K S. Lam, J. Riedl Shilling recommender systems for fun and profit. Proceedings of the 13th international conference on World Wide Web ,ACM Press, pp 393-402, 2004.

[14]. Hung-Wen Tung; "A personalized restaurant recommender agent for mobile e-service," 2004 IEEE International Conference, pp 0-7695-2073-1.

[15]. L. Schmidt-Thieme, A. Felfernig and G. Friedrich. "Guest Editor Introduction: Recommender Systems", IEEE Intelligent Systems, pp. 18-21, 2007.

[16]. L. Schmidt-Thieme, "Compound Classification Models for Recommender Systems," Proc. IEEE Int',l Conf. Data Mining (ICDM 05), IEEE Press, 2005, pp. 387&ndash,385.