ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 4, April 2016

Scalable Filtering Approaches for Recommendation Systems in E-Commerce

Sneha Khatwani[1], Dr. M. B. Chandak[2]

M.Tech Student, Dept. of CSE, Shri Ramdeobaba College of Engineering and Management, Nagpur, India[1]

HOD, Dept. of CSE, Shri Ramdeobaba College of Engineering and Management, Nagpur, India[2]

ABSTRACT: Various contents are provided through the internet. Especially the contents of e-Commerce such as music, movies, and books are indispensable for modern life style. However, it is not easy to find favorite contents among huge amounts of contents in terms of user's preference. An effective approach to solve such the problem is to develop "Recommender System." The Recommender System of Amazon site selects and recommends the contents to meet user's preference automatically using various data stored in database. There is need to filter, prioritize and efficiently deliver relevant information in order to alleviate the problem of information overload, which has created a potential problem to many Internet users. Recommender systems solve this problem by searching through large volume of dynamically generated information to provide users with personalized content and services. The essential technique in the Recommender System is information filtering. Among the various types of information filtering that have been proposed, the techniques fall into two categories: content-based filtering and collaborative filtering. Recommender system is a kind of web intelligence technique to make daily information filtering for people. Clustering techniques can be applied to the user-based collaborative filtering framework to solve the cold start problem. This paper covers different techniques which are used in recommendation system and also proposes a system for hybrid recommendation system. Recommendation based on hybrid collaborative filtering i.e. using few techniques of collaborative filtering approach. This paper explores the different characteristics and potentials of different prediction techniques in recommendation systems in order to serve as a compass for research and practice in the field of recommendation systems.

KEYWORDS: Collaborative Filtering, Content Based Filtering, Clustering.

I. Introduction

Recommender systems are personalized or non-personalized information sources that provide recommendations: prediction or suggestions for items which could be of use to a user. The data used by a recommender system can be broadly divided into three parts (i) data of items contained by the database, the information that the system has about the items, (ii) users data, the information about ratings of all users for items which they have purchased, and (iii) the information about interaction of user with the system so that a prediction or recommendation can be generated which would take the user and item data. Recommender systems work in three phases: (i) Information collection phase: This collects important information of users which generates a profile for user including user's attribute, behaviors or content of the resources the user accesses. Recommender systems rely on different types of input such as the most convenient high quality explicit feedback i.e. the system normally prompts the user through the system interface to provide ratings for items in order to construct and improve his model or implicit feedback i.e. the system automatically infers the user's preferences by monitoring the different actions of users such as the history of purchases, navigation history, and time spent on some web pages, links followed by the user, content of e-mail and button clicks among others. The strengths of both implicit and explicit feedback can be combined in a hybrid system in order to minimize their weaknesses. (ii) Learning phase: It applies a learning algorithm to filter and exploit the user's features from the feedback gathered in information collection phase.

(iii) Prediction/recommendation phase: It recommends or predicts what kind of items the user may prefer. This can be made either directly based on the dataset collected in information collection phase which could be memory based or model based or through the system's observed activities of the user.

Recommendation techniques can be classified into few filtering techniques. One of the most prominent personalization techniques is Collaborative Filtering (CF). It is the process of finding information using the opinion of other users. Predictions about user interests are made by collecting information from users who have made similar choices. It is assumed that those individuals agreed in the past tend to agree again in the future. Another filtering technique is Content-Based filtering which makes recommendations based on the users previous choices or interests. Personalized profiles are created automatically through user feedback, and they describe the type of items a person likes. In order to achieve better recommendation results the collaborative filtering and content based filtering techniques can be combined to build hybrid recommender systems. Knowledge-based systems recommend items based on specific domain knowledge about how certain item features meet users needs and preferences and, ultimately, how the item is useful for the user. In these systems a similarity function estimates how much the user needs (problem description) match the recommendations (solutions of the problem). The similarity score or the calculated prediction can be directly interpreted as the utility of the recommendation for the user.

II. RELATED WORK

In the field of collaborative filtering, both Herlocker et al. [8] and Breese et al. [14] have provided overviews and frameworks for evaluating CF algorithms. Many algorithms beyond the original k-nearest neighbor algorithm [16] have been proposed and used for collaborative filtering. These include item-based algorithms [17] and model-based algorithms such as Bayesian networks [8] and clustering [8]. Researchers have experimented with CF systems in a wide variety of domains, including Usenet news [16], jokes [13], movies [14,15] . Collaborative filtering has succeeded in helping users in all of these domains. We use the following CF algorithms in our experiments User-Item CF is the original k-Nearest Neighbor CF algorithm [16]. Given the ratings matrix, the User-Item algorithm compares rows in the matrix to create a neighborhood of the most similar papers to the target paper. The algorithm uses a cosine similarity metric. The algorithm recommends movies with the highest weighted counts. Instead of building neighborhoods among users, Item-Item CF compares similar items [17]. The Item-Item algorithm compares columns in the ratings matrix to create a neighborhood, an 'item' neighborhood, of the closest movies to each citation in the basket. Again, we use cosine similarity metric. The Naïve Bayesian classifier [8, 11] calculates probabilities that any given citation in the dataset is related to the input basket. The algorithm sorts the citations by probability and recommends citations from highest to lowest probability. The classifier is trained on citation lists from the dataset. Even in domains where the naïve Bayes principle does not hold, naïve Bayesian classifiers still work remarkably well

III. EVALUATION TECHNIQUES

A function is used to measure the accuracy of Recommender techniques and it computes the following aspects:

- Root mean square error (RMSE): This is the standard deviation of the difference between the real and predicted ratings.
- Mean squared error (MSE): This is the mean of the squared difference between the real and predicted ratings. It's the square of RMSE, so it contains the same information.
- Mean absolute error (MAE): This is the mean of the absolute difference between the real and predicted ratings.

IV. RECOMMENDER TECHNIQUES

(a) Cosine Similarity

There are many similarity measures which are used to compare similarity between different items or similarity between different users. Some common examples are Pearson (correlation) based similarity, Jaccard Coefficient and Cosine angle Similarity. Cosine similarity is a measure of similarity between two vectors of an inner product space that measures the cosine of the angle between them. Cosine similarity is given by this equation:

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|}$$

In Collaborative filtering, recommendations are based on a few customers who are most similar to the active users. It comprehends the similarity of two users by using cosine of the angle between the two vectors: The collaborative filtering can be adapted with neighborhood methods, whose focus is on relationship between the items or, alternatively between the users. They are:

I. User-based CF: For each user, it computes correlation with other users. For each item, aggregate the rating of the users highly correlated with each user.

Problem: Sparsity, easy to attack
II. Item-based CF: For each item, compute correlation with other items. For each user, aggregate his rating of the items highly correlated with each item
Advantages: Collaboration filtering approach doesn't need a representation of items in terms of features but it is based only on the judgement of participating user community.
Disadvantages: The item can't be recommended to any user until and unless the item is either rated by another user(s) or correlated with other similar items.

(b) Recommender lab

R has a package recommenderlab that provides infrastructure to develop and test recommendation algorithm. This package focusses on recommendation algorithm which is based on collaborative filtering. We implement item based collaborative filtering by using this package. It helps computing similarities between items for similar users. Recommenderlab can be used to get insight into collaborative filtering algorithms and evaluate the performance of different algorithm available in the framework on Movie Lens 100k data set.
Here two sets are constructed. First is the training set which includes users from which the model runs and second is the test set which includes users to whom items are recommended. Few steps are performed prior to using the package are:
Data Exploration:
MovieLens is dataset about movie ratings. Each row corresponds to a user, each column to a movie, and each value to a rating. Here we explore the values of the rating by converting it to a vector. We can then explore the movies have been viewed and then extracting quick results using column counts which will give the number of non-missing values for each column and column means which gives the average value for each column. Using the average ratings for each movie we can identify top rated movies.
Data preparation
In this step we prepare the data which is to be used in recommender models. This is done by selecting the relevant data and then normalizing it. After exploring the movies it is observed that some movies that have been viewed only a few times. Their ratings might be biased because of lack of data and some users who rated only a few movies may have given biased ratings. Normalizing the data takes into consideration those users who give high (or low) ratings to all their movies and hence it might bias the results. This can be avoided by normalizing the data in such a way that the average rating of each user is 0.
Then using the package we implement the algorithm which is based on the following steps:
1. For each two items, measure how similar they are in terms of having received similar ratings by similar users
2. For each item, identify the k-most similar items
3. For each user, identify the items that are most similar to the user's purchases
Then, the algorithm ranks each similar item in the following way:

It extracts the user rating of each purchase associated with this item. The rating is used as a weight. The algorithm then extracts the similarity of the item with each purchase associated with this item. Then the weight calculated in the previous step is multiplied with the related similarity.

(c) K means clustering

K-means is an unsupervised, iterative algorithm where k is the number of clusters to be formed from the data. Clustering is achieved in two steps as shown in Fig-1:

1. Cluster assignment step: In this step, we randomly choose two cluster points and assign each data point to the cluster point that is closer to it
2. Move centroid step: In this step, we take the average of the points of all the examples in each group and move the centroid to the new position, that is, mean position calculated. The preceding steps are repeated until all the data points are grouped into two groups and the mean of the data points after moving the centroid doesn't change.

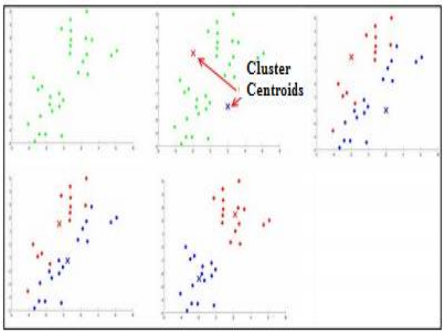


Fig-1 Steps in Cluster Analysis

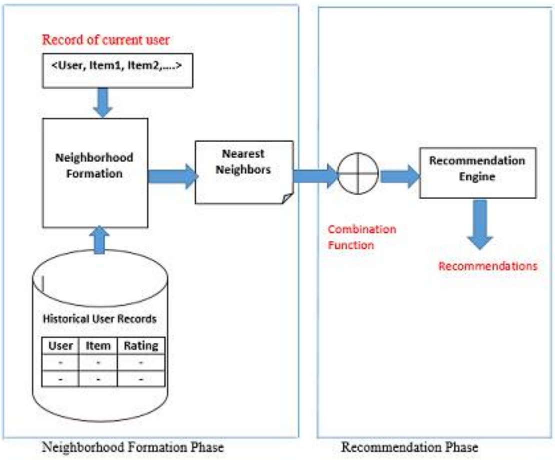


Fig-2: User Based Collaborative Filtering Approach

Recommendation model is built using k nearest algorithm by implementing user based collaborative filtering. This is done in two phases. First is the neighborhood formation phase which is depicted in the figure below (Fig-2). In this phase by using centroid method as discussed above we can formulate the neighborhood of items for a current user by taking into consideration previous user records. After the formation of nearest items for a current user the items can be recommended to the user. This is done in the second phase that is the recommendation phase by applying a combination function which would filter results and predict top k items for the user.

In this approach, given a new user, we will identify its similar users. Then, we will recommend the top-rated items purchased by similar users. For each new user, these are the steps:

1. Measure how similar each user is to the new one. Like IBCF, popular similarity measures are correlation and cosine.

2. Identify the most similar users using top k users (k-nearest neighbors)

3. Rate the items purchased by the most similar users. The rating is the average rating among similar users and the approaches are: ° Average rating ° Weighted average rating, using the similarities as weights

4. Pick the top-rated items.

(d) Content Based Filtering

For a dataset of movies we can find content of the videos using several techniques. Metadata is attached to video for making it easy to access. Different types of information that can be associated with videos or images are: Content independent metadata that is related to the image or video, but does not describe it directly. For example, name of user, date, location, etc. It cannot be extracted from the image or video. Content dependent metadata refers to low-level and intermediate-level features. Various low-level features can be found from the video and from individual video frames. These features can be used for annotation. Lowlevel features that can be used are shape, color, texture, edge, motion, etc. MPEG-7 visual descriptors can also be used. MPEG-7 color descriptor and edge descriptor are commonly used.

(e) Graph Based Filtering

Graph-based learning is a semi-supervised method. Graph with labelled and unlabeled vertices are used. These vertices are samples; the edges reflect the similarities between sample pairs. A function is estimated on the graph based on a label smoothness assumption. These methods have already been successfully applied in image and video content analysis.

V. EXPERIMENTAL STUDY

The dataset used in this experiment is obtained from
Movie Lens project. This data set consists of:
* 100,000 ratings (1-5) from 943 users on 1682 movies.
* Each user has rated at least 20 movies.
* Simple demographic info for the users (age, gender, occupation, zip)

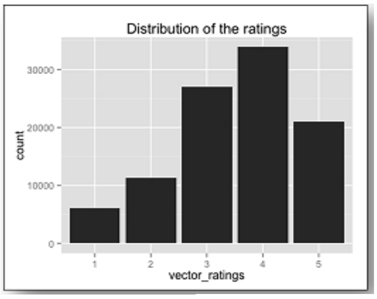
All ratings are between 1(bad) and 5 (Excellent). In our experiment, we selected 50% of the data as training set and compute the recommendation for the remaining 50% of the movies. The experiments were conducted on three different recommendation algorithms: [1] Item based collaborative filtering using similarity metric, [2] item based collaborative filtering, using package of R language, i.e. RecommenderLab [3] User based collaborative filtering using K-Means Clustering Algorithm.

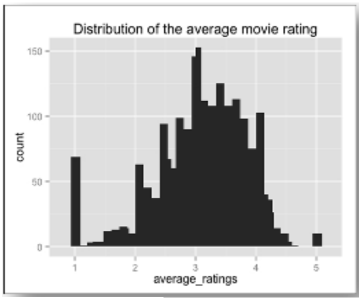
Algorithm for Item-Item Collaborative Filtering:

- We split user dataset into train/test sets
- For each active user a in the test set:
- Split a's votes into observed (I) and to-predict (P) and measure average absolute **deviation** between predicted and actual votes in P
- We then predict votes in P, and form a **ranked list**
- assume (a) utility of k-th item in list is max(va,j-d,0), where d is a "default vote" (b) probability of reaching rank k drops exponentially in k. Score a list by its expected utility Ra
- Average Ra over all test users

Before implementing the package RecommenderLab we explore and prepare the data. A graph of vector ratings vs count is plotted as shown in Fig-3. It is observed that most of the ratings are above 2, and the most common is 4.

The occurrence of rating is observed and a graph of count of movies vs average ratings is plotted. It is observed that the highest value is around 3, and there are a few movies whose rating is either 1 or 5. The probable reason is that these movies received a rating from a few people only, so these movies are not taken into account. This can be seen in Fig-4. After exploring the most relevant data and preparing the data by normalizing and binarizing it we apply item based collaborative filtering method by using similarity metrics like Pearson similarity coefficient and cosine similarity.





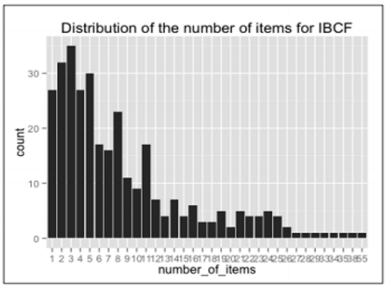


Fig-3: Distribution of Ratings vs Vector_Ratings

Fig-4 Average_Ratings vs Count

Fig-5: No. of items vs Count

For each user, the algorithm extracts its rated movies. For each movie, it identifies all its similar items, starting from the similarity matrix.

The graph shown in Fig-5 shows the distribution of number of movies for Item Based Collaborative Filtering. IBCF recommends items on the basis of the similarity matrix. IBCF model once built, it doesn't need to access the initial data. For each item, the model stores the k-most similar, so the amount of information is small once the model is built. This is an advantage in the presence of lots of data. In addition, this algorithm is efficient and scalable, so it works well with big rating matrices. Its accuracy is good, compared with other recommendation models.

User-based collaborative filtering In the previous section, the algorithm was based on items and the steps to identify recommendations were as follows: • Identify which items are similar in terms of having been purchased by the same people • Recommend to a new user the items that are similar to its purchases In this section, we will use the opposite approach.

First, given a new user, we will identify its similar users. Then, we will recommend the top-rated items purchased by similar users. This approach is called user-based collaborative filtering. For each new user, these are the steps: 1. Measure how similar each user is to the new one. Like IBCF, popular similarity measures are correlation and cosine. 2. Identify the most similar users. The options are: ° Take account of the top k users (k-nearest_neighbors) ° Take account of the users whose similarity is above a defined threshold 3. Rate the items purchased by the most similar users. The rating is the average rating among similar users and the approaches are: ° Average rating ° Weighted average rating, using the similarities as weights 4. Pick the top-rated items.

After running k-means, the plot of the number of assigned users to each cluster showed a power-law curve for all values of $k \geq 5$, where the majority of users were assigned to first cluster and then a bump on the curve with 2-3 equally sized clusters, and then a long tail with small clusters. This can be seen in Fig-6. The plot here below shows the number of users assigned to each cluster for k = 10.
Prediction accuracy of the clustering method is calculated and compared with the Root-Mean-Square Error of the assigned ratings compared to ratings in a test set.

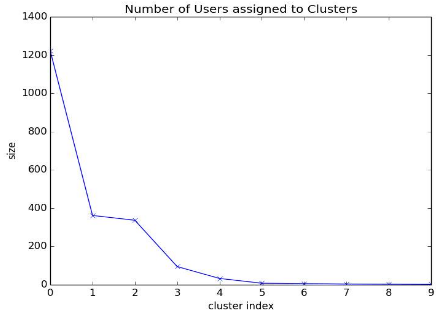


Fig-6: No. of users assigned to clusters

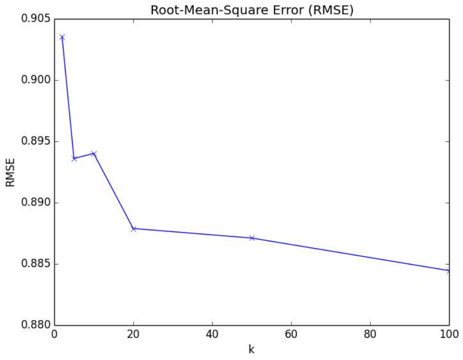


Fig-7: Root Mean Square Error

The baseline predictor showed an RMSE = 0.900 and the best achieved result with the approach described in this paper was RMSE = 0.884, which is an improvement of 1.81% compared with baseline.

VI. CONCLUSION

Over the last years recommender systems emerged as a significant information filtering system. It uses several techniques for recommendation which includes content-based, collaborative and hybrid methods. All existing recommender systems employ one or more of a handful of basic techniques: content-based, collaborative, demographic, utility-based and knowledge-based. A survey of these techniques shows that they have complementary advantages and disadvantages. This fact has provided incentive for research in hybrid recommender systems that combine techniques for improved performance. In this paper we have discussed how to make recommender system models. User based and item based collaborative filtering models using different techniques have been discussed. We should evaluate recommendation algorithms so as to select the best algorithm from a set of candidates. These metrics put emphasis on the quality of the recommender system. Through the Experiments, we have compared three algorithms which vary from user to user.

In the future, we are interested in studying the hybrid recommendation model which takes into consideration building collaborative filters using big data.

REFERENCES

1. Gediminas Adomavicius, Alexander Tuzhilin , "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible ExtensionsIEEE transactions on knowledge and data engineering, vol. 17, no. 6, june 2005
2. Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl, Item-based Collaborative Filtering Recommendation Algorithms", WWW10, May 1-5, 2001, Hong Kong.
3. Bharat Bhaskar, K. Srikumar Recommender systems in E-commerce, Methodologies and applications of data mining.
4. Greg Linden, Brent Smith, and Jeremy York "Amazon.com Recommendations Item-to-Item Collaborative Filtering" Published by the IEEE Computer Society
5. Dr. Sarika Jain, Anjali Grover, Praveen Singh Thakur, Sourabh Kumar Choudhary, "Trends, Problems And Solutions of Recommender System" International Conference on Computing, Communication and Automation (ICCCA2015) and expert system application.
6. D. Jannach, M. Zanker, A.Felfernig and G. Friedrich, Recommender Systems – an introduction Cambridge University Press, 2010
7. Dr. M.B. Chandak, Kushboo Khurana, "Study of Various Video Annotation Techniques", International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 1, January 2013
8. Breese, J., Heckerman, D., and Kadie, C. Empirical Analysis of Predictive Algorithms for Collaborative Filtering. In Proc. UAI 98, Madison, 1998, 43–52.
9. Bollacker, K., Lawrence, S., and Giles, C. L. Discovering relevant scientific literature on the web. IEEE Intelligent Systems, 15(2), 42–47, 2000.
10. Egghe, L., and Rousseau, R. Introduction to Informetrics. Elsevier, Amsterdam, 1990.
11. Friedman, N., Gieger, M., and Goldszmidt, M. Bayesian Network Classifiers. Machine Learning, 29, 131–163, 1997.
12. Garfield, E. Citation Indexing: Its Theory and Application in Science, Technology, and Humanities. Wiley, New York, 1979.
13. Goldberg, K., Roeder, T., Gupta, D., and Perkins, K. Eigentaste: A Constant Time Collaborative Filtering Algorithm. Information Retrieval Journal, 4(2), 133– 151. 2001.
14. Herlocker, J., Konstan, J. A., Borchers, A., and Riedl, J. An Algorithmic Framework for Performing Collaborative Filtering. In Proc. SIGIR 99, Berkeley, 1999, 230–237.
15. Hill, W., Stead, L., Rosenstein, M. and Furnas, G. Recommending and evaluating choices in a virtual community of use. In Proc. CHI 1995, Denver, 1995, 194–201.
16. Resnick, P., Iacovou, N., Sushak, M., Bergstrom, P., and Riedl, J. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In Proc. CSCW 94, Chapel Hill, 1994, 175–186.
17. Sarwar, B., Karypis, G, Konstan, J. A., and Riedl, J. Item-based Collaborative Filtering Recommendation Algorithms. In Proc. WWW 10, Hong Kong, 2001, 285–295.
18. Scott, J. Social Network Analysis: A Handbook, 2nd Edition. Sage Publications, London, 2000.