RECOMMENDER SYSTEMS: AN ALGORITHM TO PREDICT "WHO RATE WHAT"

by

RAHUL SINGHAL

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

THE UNIVERSITY OF TEXAS AT ARLINGTON

December 2010

To my Parents, Brother and Friends.

ACKNOWLEDGEMENTS

I would like to thank my supervising professor Dr.Chris Ding for constantly motivating and encouraging me, and also for his invaluable advice during the course of my Masters studies. I wish to thank my committee members Dr.Heng Huang and Dr.Chengkai Li for their interest in my research and for taking time to serve in my dissertation committee.

I would like to thank my parents and friends for constantly encouraging and inspiring me to continue my masters studies. Finally, I would like to express my deep gratitude to my Father who have encouraged and inspired me and sponsored my undergraduate and graduate studies. I am extremely fortunate to be so blessed.

November 19, 2010

ABSTRACT

RECOMMENDER SYSTEMS: AN ALGORITHM TO PREDICT
"WHO RATE WHAT"

RAHUL SINGHAL, M.S.

The University of Texas at Arlington, 2010

Supervising Professor: Chris H.Q. Ding

Recommender systems are systems that recommend content for us by looking at certain factors including what other people are doing as well as what we are doing. Examples of such systems present today are Amazon.com recommending books, CDs, and other products; Netflix recommending movies etc. These systems basically recommend items or movies to customers based on the interests of the present customer and other similar customers who purchased or liked the same item or movie. Our paper goes beyond the concept of overall generic ranking and provides personalized recommendation to users. Despite all the advancements, recommender systems still face problems regarding sparseness of the known ratings within the input matrix. The ratings are given in the range of (1-5) and present systems predict "What are the ratings" but here we propose a new algorithm to predict "Who rate what" by finding contrast points in user-item input matrix. Contrast points are the points which are farthest from the known rated items and most unlikely to be rated in future. We experimentally validate that our algorithm is better than traditional Sin-

gular Value Decomposition (SVD) method in terms of effectiveness measured through precision/recall.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Recommender Systems

Recommender systems form or work from a specific type of information filtering system technique that attempts to recommend information items (movies, video on demand, music, books, news, images, web pages, etc.) that are likely to be of interest to the user. Typically, a recommender system compares a user profile to some reference characteristics, and seeks to predict the 'rating' that a user would give to an item they had not yet considered. These characteristics may be from the information item (the content-based approach) or the user's social environment (the collaborative filtering approach).

When building the user's profile a distinction is made between explicit and implicit forms of data collection.

Examples of explicit data collection include the following:

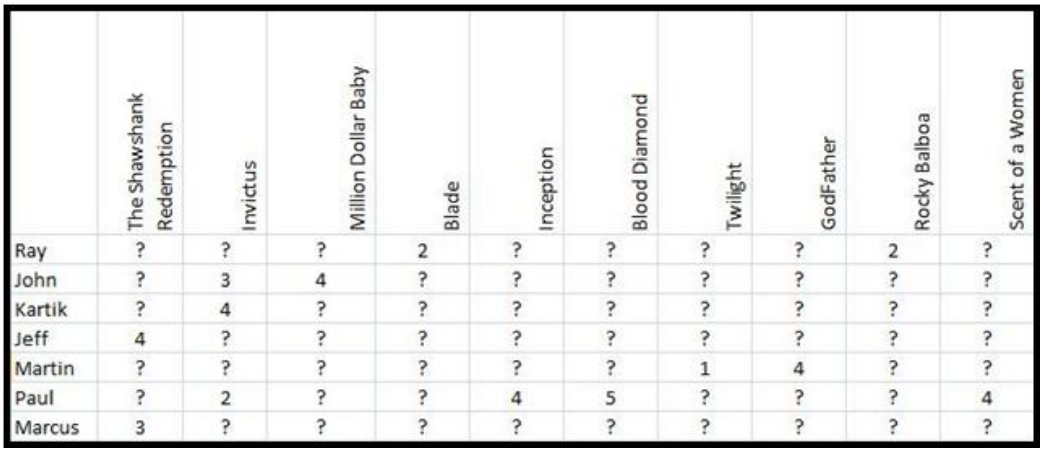
- Asking a user to rate an item on a rating scale.
- Asking a user to rank a collection of items from favorite to least favorite.
- Presenting two items to a user and asking him/her to choose the best one.
- Asking a user to create a list of items that he/she likes.

Examples of implicit data collection include the following:

- Observing the items that a user views in an on-line store.
- Keeping a record of the items that a user purchases on-line.
- Obtaining a list of items that a user has listened to or watched on his/her computer.

- Analyzing item/user viewing times.
- Analyzing the user's social network and discovering similar likes and dislikes.

The recommender system compares the collected data to similar and not similar data collected from others and calculates a list of recommended items for the user. Several commercial and non-commercial recommendation search engines are available in market. Examples include Pandora Radio [1] which is an automated music recommendation service and custodian of the Music Genome Project. Users enter a song or artist that they enjoy, and the service responds by playing selections that are musically similar. Users provide feedback on approval or disapproval of individual songs, which Pandora takes into account for future selections. Another example is MovieLens [2] which is a recommender system and virtual community website that recommends films for its users to watch, based on their film preferences and using collaborative filtering. The website is a project of GroupLens Research, a research lab in the Department of Computer Science and Engineering at the University of Minnesota which created MovieLens in 1997. The website's ultimate goal is to gather research data on personalized recommendations systems. Recommender systems are a useful alternative to search algorithms since they help users discover items they might not have found by themselves. Figure 1.1 shows an example how users rate items (movies) and how ratings are stored in 2-D recommender systems. Here ?'s show the unrated user-item pairs.



	The Shawshank Redemption	Invictus	Million Dollar Baby	Blade	Inception	Blood Diamond	Twilight	GodFather	Rocky Balboa	Scent of a Women
Ray	?	?	?	2	?	?	?	?	2	?
John	?	3	4	?	?	?	?	?	?	?
Kartik	?	4	?	?	?	?	?	?	?	?
Jeff	4	?	?	?	?	?	?	?	?	?
Martin	?	?	?	?	?	?	1	4	?	?
Paul	?	2	?	?	4	5	?	?	?	4
Marcus	3	?	?	?	?	?	?	?	?	?

Figure 1.1. 2-D matrix with user ratings on movies.

1.2 Who Rate What

The problem "Who Rate What" means to predict which users will rate what items in future. The actual rating is irrelevant in this case, we need to predict which item is going to be rated in near future. The other problem "What are the ratings" means to predict what are the actual ratings of the items. "Who rate what" goes beyond the scope of giving generic overall ranking to users which may or may not be useful to particular useful. We basically here are trying to do the personalized recommendation by predicting which users are going to rate what items in future. Doing personalized recommendation is difficult because we do not deal with the ratings of items but we deal with rated user-item pairs in dataset. Even providing personalized recommendation is commercially beneficial to companies as users get the better and more detailed picture of recommendation as compared to generic overall recommendation.

In this paper, we propose an algorithm to predict "Who rate What". Here user - item matrix contains only ones for rated items and all others are unrated items. We compare our method with traditional Singular Value Decomposition method. One of

the major problem that recommender systems face is data sparsity. Very less number of items are rated by users and mostly remain unrated. Due to this the quality of recommendation is not good to users. Here we address this problem by finding some contrast points which are originally the unrated items. Contrast points means the items which are most unlikely to be rated in future and thereby improving the accuracy of results calculated in terms of effectiveness measured through precision/recall.

One of the main intents of this paper is to understand our approach with the standard Singular Value Decomposition method. Our main contributions are:

- We propose two approaches for finding contrast points. The first one is "Distance approach" based on Cosine similarity measure and second one is "Efficient Distance Approach". But only second one is experimentally proved on movie lens dataset, not first one because it is very slow due to its high time complexity.
- The initialization of masked out values with the average of both dimension i.e., user and item instead of zeros. The masked out values are the predictions. All these masked out values are stored in test matrix which we use for assessing the predictions in future.
- The Comparison of our algorithm with traditional SVD method in terms of recommendation accuracy measured in precision/recall, with emphasis on parameters that improve results.

1.3 Our Results and Organization

The rest of the paper is organized as follows. In chapter 2, we describe the related work i.e Traditional Singular Value Decomposition approach. We describe the contrast method of finding contrast points and the motivation behind finding these points in input user-item matrix. We also describe that effectiveness of recommendations.

We describe in chapter 4, Distance Approach of finding contrast points which is based on cosine-based similarity. We show mathematically how this approach can find contrast points and also this approach is not applicable on real and large datasets as it has very high complexity. In chapter 1 we describe our new and faster approach of finding contrast points in original user - item matrix which is Efficient distance approach (EDA). The original input matrix contains ones for rated records and zeros for unrated records. We try to find some of the user-item pairs(contrast points) in matrix which are most unlikely to be rated in future

The chapter 6 describes about experimental evaluation and contains the description of dataset used for experimentation and our experimental setup. In chapter 7 We compare the results of our algorithm with the traditional Singular Value Decomposition approach. Finally chapter 8 summarizes our most important contribution. Appendix A.1 shows all the symbols used in this paper.

CHAPTER 2

RELATED WORK

2.1 Related Work

Recommenders based on the rank k approximation of the rating matrix based on the first k singular vectors are described in [3, 4] and many others near year 2000.

The Singular Value Decomposition(SVD) of a rank k matrix A is given by $A = U\Sigma\ V^T$ with U an $m \times k$, Σ a $k \times k$ and V an $n \times k$ matrix such that U and V are orthogonal. By the Eckart- Young theorem [5] the best rank-k approximation of A with respect to the Frobenius norm is

$$||A - U_k\Sigma_k V_k^T||_F^2 = \sum_{ij}(w_{ij} - \sum_k \sigma_k u_{ki} v_{kj})^2 \tag{2.1}$$

where U_k is an $m \times k$ and V_k is an $n \times k$ matrix containing the first k columns of U and V and the diagonal Σ_k containing first k entries of Σ.

$$RMSE^2 = \sum_{ij \in R} {err_{ij}}^2 \ where\ err_ij = w_{ij} - \sum_k \sigma_k u_{ki} v_{kj} \tag{2.2}$$

where R denoted either the training or the test set. To simplify notation we extend err_{ij} with value 0 for $ij \notin R$.

The crux in using SVD for recommender lies in handling missing values in the rating matrix A. Goldeberg et al [6] for example require a gauge set where all ratings are known, an assumption clearly infeasible on netflix or movielens data scale. Azar et al. [6] prove asymptotic results on replacing missing values by zeros and scaling known ratings inversely proportional to the probability of being observed.

If we see carefully, all these algorithms talk about the "What are the ratings" problem in recommender systems. In our paper we emphasize on "Who rate what" problem.

SVD has an important property that makes it particularly interesting for our algorithm. SVD provides the best *low-rank* approximation of the original matrix A. It is possible to retain only k ¡¡ r singular values by eliminating other irrelevant records. We term this reduced matrix as Σ_k. The records in Σ_k are in sorted order i.e., $\Sigma_1 \geq \Sigma_2 \geq \Sigma_3 \geq ... \geq \Sigma_r$, the reduction process is performed by retaining first k singular values. The matrices U and V are also reduced to produce matrices U_k and V_k, respectively. The reconstructed matrix

$$A_k = U_k \Sigma_k V_k^T \tag{2.3}$$

is a $rank - k$ matrix that is closest approximation to the original matrix A. Authors [7, 8] pointed out that *low-order* approximation of original matrix is better than original matrix as it helps in filtering out that small singular values that create "noise" in user-item relationship.

CHAPTER 3

METHOD OVERVIEW

3.1 Traditional SVD method

In this paper, we predict "Who Rate What" i.e., which items are most likely to be rated in future based on previous rated user-item pairs. The dataset is represented in a 2-d matrix of users and items. Users rate the items and rated items are given the certain ratings on a scale of 1 to 5. Rated user-item pairs are called as records. These records show that particular user x rated the item y with certain rating. We change all the original ratings in the input matrix into +1. The Singular Value Decomposition(SVD) of a rank k matrix A is given by $A = U\Sigma\, V^T$ with U an $m \times k$, Σ a $k \times k$ and V an $n \times k$ matrix such that U and V are orthogonal. By the Eckart-Young theorem [5] the best rank-k approximation of A with respect to the Frobenius norm is

$$||A - U_k \Sigma_k V_k^T||_F^2 = \sum_{ij} (w_{ij} - \sum_k \sigma_k u_{ki} v_{kj})^2 \tag{3.1}$$

where U_k is an $m \times k$ and V_k is an $n \times k$ matrix containing the first k columns of U and V and the diagonal Σ_k containing first k entries of Σ where u_{ki} and v_{kj} denotes the strength of user i and movie j in factor k, respectively, while σ_k is the factor importance weight.

After doing this we do the predictions in Top-N fashion as described in chapter 3.2.3. This whole process of predictions using SVD is called as Traditional Singular Value Decomposition method. But we can see here that we use full $[0, 1]$ matrix of all known ratings for training. Figure 3.1 gives an example of user-item matrix with

ratings given by certain users. We convert all the ratings into 1's which is shown in figure 3.1. Then we do SVD on $[0, 1]$ matrix and get matrix which is shown in figure 3.1. After this we predict the items which have value greater than 0.5(threshold value) in matrix shown in figure 3.1 are predicted. The predicted items are shown in red color, original rated records are shown in black color and non rated records in blue color. Basically, we are giving an example how predictions take place in real recommender systems with certain threshold value. But we have not set any threshold value, here we check our prediction performance using traditional precision and recall methods in a Top-N fashion discussed in chapter 3.2.3.

?	?	?	2	?	?	?	?	2	?
?	3	4	?	?	?	?	?	?	?
?	4	?	?	?	?	?	?	?	?
4	?	?	?	?	?	?	?	?	?
?	?	?	?	?	?	1	4	?	?
?	2	?	?	4	5	?	?	?	4
3	?	?	?	?	?	?	?	?	?

Figure 3.1. Matrix with [0,5] ratings.

0	0	0	1	0	0	0	0	1	0
0	1	1	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	0	0
0	1	0	0	1	1	0	0	0	1
1	0	0	0	0	0	0	0	0	0

Figure 3.2. Matrix with [0,1] ratings.

0.88	0.34	0.22	1	0.05	0.23	0	0	1	0.32
-0.06	1	1	0.06	-0.01	-0.04	0	0	-0.01	-0.02
-0.04	1	0.99	0.09	-0	-0.01	0	0	0.008	0.009
1	-0.02	-0.06	0.74	-0.05	-0.22	0	0	1	-0.14
0	0	0	0	0	0	1	1	0	0
-0.21	0.41	0.24	0.52	0.25	1	0	0	0.007	1
0.03	0.2	0.19	0.08	0.008	0.03	0	0	0.06	0.04

Figure 3.3. Matrix formed after doing SVD.

1	0	0	1	0	0	0	0	1	0
0	1	1	0	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	1	0
0	0	0	0	0	0	1	1	0	0
0	1	0	1	1	1	0	0	0	1
1	0	0	0	0	0	0	0	0	0

Figure 3.4. Matrix with predicted and unpredicted user-items pairs.

3.2 Contrast Method

3.2.1 Contrast Points

Our task here is to predict "Who rate What" i.e., which users will rate what items in future. This means that we are predicting on the basis of personalized recommendations of users because we are taking into account the user-item pairs or records. Recommender systems are a useful alternative to search algorithms since they help users discover items they might not have found by themselves. Most of the recommender systems or search engines provides generic overall ranking to users which may or may not be useful to customers. We change the original ratings given to items to +1 because, suppose if we keep the ratings and then try to predict which

items are going to be rated in future; then it could be possible that low rated items will never be predicted and in this way the system will be biased.

The traditional SVD method discussed in chapter 3.1 gives good results but there is scope of improvement. This gives us the motivation of think about a method which can do magic. We come up with a Contrast method. In Contrast method, we try to find "Contrast Points" which are most unlikely to be rated in future. These points have maximum distance from rated user-item pairs in matrix i.e., most dissimilar from rated records. We assign the values to contrast points as -1, because they have contrast behavior to original rated user-item pairs. Now we show an example how Contrast method works on 2-D matrices. Figure 3.1 gives an example of user-item matrix with ratings given by certain users. We covert all the ratings into 1's which is shown in figure 3.1. After applying contrast method we get matrix with $[0, 1, -1]$ values shown in figure 3.2.1. Then we do SVD on previous matrix and gets matrix shown in figure 3.2.1. Then we do the predictions; suppose we take threshold value as 0.5, so values higher than 0.5 are predicted and below are unpredicted. The predicted matrix is shown in figure 3.2.1 with red color values are predicted, black color values are original 1's and contrast points, and blue color values are unpredicted. If we compare our novel "Contrast Method" with "Traditional SVD method", then the difference is the input matrix that goes for predictions. In our method also we do the predictions using SVD.

0	0	-1	1	-1	0	0	0	1	0
0	1	1	0	0	0	0	0	0	-1
0	1	0	0	0	0	-1	0	0	0
1	0	0	0	-1	0	0	0	1	0
0	0	-1	0	0	0	1	1	-1	0
0	1	0	0	1	1	0	0	0	1
1	0	0	0	0	0	0	-1	0	0

Figure 3.5. Matrix with [0,1,-1] values.

0.068	0.113	-1	1	-1	0.028	-0	0.753	1	0.229
-0.15	1	1	0.002	0.32	0.037	0.047	0.128	0.246	-1
0.52	1	0.028	0.011	-0.33	0.099	-1	-0.19	-0.21	0.091
1	-0.1	0.097	0.205	-1	-0.02	-0.02	-0.71	1	0.106
-0.54	0.584	-1	0.001	0.425	0.091	0.006	0.252	-1	0.875
0.168	1	0.03	0.275	-0.05	0.73	0.015	-0.31	-0.33	1
1	0.254	0.236	-0.19	0.054	0.025	-0.01	-1	0.364	0.342

Figure 3.6. Matrix formed after applying SVD on matrix with contrast points.

0	0	-1	1	-1	0	0	1	1	0
0	1	1	0	0	0	0	0	0	-1
1	1	0	0	0	0	-1	0	0	0
1	0	0	0	-1	0	0	0	1	0
0	1	-1	0	0	0	1	1	-1	1
0	1	0	0	1	1	0	0	0	1
1	0	0	0	0	0	0	-1	0	0

Figure 3.7. Matrix with predicted user-item pairs with Contrast points.

3.2.2 Methods to Compute Score

Since recommender systems face the major problem of data sparsity i.e., most of the items remain unrated by many users. Rated User-item pairs or records in matrix

are very less as compared to all the records. So we can think of those pairs which have the least probability of getting rated. We call these points as Contrast points. Why we call these points as Contrast points, because they have opposite behavior to rated records in matrix. Now the question arises how to find these points. We give two approaches "Distance Approach" and "Efficient Distance Approach" discussed in chapter 4 and chapter 1 respectively. Logically these contrast points are farthest from the rated records i.e., in similarity wise they are least similar to rated records.

There are two similarity approaches that are commonly used which are correlation-based approach and cosine-based approach. The similarity measure between users c and $c^{'}$, $sim\left(c, c^{'}\right)$, is essentially the distance measure and is used as a weight, i.e., the more similar users are c and $c^{'}$ are, the more weight rating $r_{c^{'},s}$ will carry in the prediction of $r_{c,s}$, where c,$c^{'}$ are users and s is an item. Note that $sim\left(x, y\right)$ is a heuristic artifact that is introduced in order to be able to differentiate between levels of user similarity (i.e., to be able to find a set of "closest peers" or "nearest neighbors" for each user) and, at the same time simplify the rating estimation procedure. To present correlation and cosine based popular similarity approaches , suppose S_{xy} be the set of all items co-rated by both users x and y, i.e.,

$$S_{xy} = \{s \in S | r_{x,s} \neq \oslash \ \& \ r_{y,s} \neq \oslash\} \tag{3.2}$$

In collaborative recommendation systems, S_{xy} is used mainly as an intermediate result for calculating the "nearest neighbors" of user x and is often computed in a straightforward manner, i.e., by computing the intersections of sets S_x and S_y. In the correlation-based approach, the Pearson correlation coefficient is used to measure the similarity [9]:

$$sim\,(x,y) = \frac{\sum_{s \in S_{xy}} (r_{x,s} - \bar{r}_x)\,(r_{y,s} - \bar{r}_y)}{\sqrt{\sum_{s \in S_{xy}} (r_{x,s} - \bar{r}_x)^2}\sqrt{\sum_{s \in S_{xy}} (r_{y,s} - \bar{r}_y)^2}} \tag{3.3}$$

In the cosine-based approach [10], the two users x and y are treated as two vectors in m-dimensional space, where $m = |S_{xy}|$. Then, the similarity between two vectors can be measured by computing the cosine of the angle between them:

$$sim\,(x,y) = cos\,(\vec{x},\vec{y}) = \frac{\vec{x}\cdot\vec{y}}{||\vec{x}||_2 \times ||\vec{y}||_2} = \frac{\sum_{s \in S_{xy}} r_{x,s} r_{y,s}}{\sqrt{\sum_{s \in S_{xy}} r_{x,s}^2}\sqrt{\sum_{s \in S_{xy}} r_{y,s}^2}} \tag{3.4}$$

where $\vec{x}\cdot\vec{y}$ denotes the dot-product between the vectors $\vec{x}$ and $\vec{y}$.

3.2.3 Top N Approach

We examine the top-N ranked list, which is recommended to a test user, starting from the top item. In this situation, the recall and precision vary as we proceed with the examination of the top-N list. We do the personalized recommendation to a test user but in a Top-N fashion i.e., recommending Top-N items. We predict that these top-N list of items will be rated by a test user in future. We can predict exactly which item will be rated by user but in that case we need to have the exact threshold. Google search engine does the same thing by predicting Top-N list web-pages to users but non-personalized. Google finds the importance of pages by calculating the weights of pages in terms of forward and backward links.

The similarity approaches are used to compute the similarity scores between records and using this we find the records which are least similar i.e., least similarity scores. These records are considered as Contrast Points which increase our chances of predicting more relevant and more similar items to users thereby increasing accuracy.

3.3 Effectiveness of Recommendations

The problem of developing good metrics to measure effectiveness of recommendations has been extensively addressed in the recommender systems literature. In most of the recommender systems literature, the performance evaluation of recommendation algorithms is usually done in terms of the coverage and accuracy metrics. Coverage measures the percentage of items for which a recommender system is capable of making predictions [11]. Accuracy measures can be either statistical or decision-support [11]. Statistical accuracy metrics mainly compare the estimated ratings against the actual ratings R in the $User \times Item$ matrix, and include Mean Absolute Error (MAE), root mean squared error, and correlation between predictions and ratings. Decision-support measures determine how well a recommender system can make predictions of high-relevance items (i.e., items that would be rated highly by the user). They include classical IR measures of precision (the percentage of truly high ratings among those that were predicted to be high by the recommender system), recall (the percentage of correctly predicted high ratings among all the ratings known to be high), F-measure (a harmonic mean of precision and recall), and Receiver Operating Characteristic (ROC) measure demonstrating the tradeoff between true positive and false positive rates in recommender systems [11].

Although popular, these empirical evaluation measures have certain limitations. One limitation is that these measures are typically performed on test data that the users chose to rate. However, items that users choose to rate are likely to constitute a skewed sample, e.g., users may rate mostly the items that they like. In other words, the empirical evaluation results typically only show how accurate the system is on items the user decided to rate, whereas the ability of the system to properly evaluate a random item (which it should be able to do during its normal reallife use) is not tested. Understandably, it is expensive and time-consuming to conduct controlled ex-

periments with users in the recommender systems settings, therefore, the experiments that test recommendation quality on an unbiased random sample are rare. However, the high-quality experiments are necessary in order to truly understand the benefits and limitations of the proposed recommendation techniques.

CHAPTER 4

DISTANCE APPROACH

4.1 Distance Approach

Various approaches have been used to compute the similarity $sim(c, c')$ between users in collaborative recommender systems. In most of these approaches, the similarity between users is based on their ratings of items that both users have rated. The two most popular approaches are correlation[12] based approach and cosine-based[12] approach. We use the cosine based similarity to measure distance between different users and items. To present it, suppose S_{xy} be the set of all items co-rated by both users x and y, i.e.,

$$S_{xy} = \{s \in S | r_{x,s} \neq \oslash \ \& \ r_{y,s} \neq \oslash\} \tag{4.1}$$

In the cosine-based approach [12], the two users x and y are treated as two vectors in m-dimensional space, where m = $|S_{xy}|$. Then, the similarity between two vectors can be measured by computing the cosine of the angle between them:

$$sim\,(x, y) = cos\,(\vec{x}, \vec{y}) = \frac{\vec{x} \cdot \vec{y}}{||\vec{x}||_2 \times ||\vec{y}||_2} = \frac{\sum_{s \in S_{xy}} r_{x,s} r_{y,s}}{\sqrt{\sum_{s \in S_{xy}} r_{x,s}^2} \sqrt{\sum_{s \in S_{xy}} r_{y,s}^2}} \tag{4.2}$$

where $\vec{x} \cdot \vec{y}$ denotes the dot-product between the vectors $\vec{x}$ and $\vec{y}$.

We use the cosine-based similarity measure to calculate the distance between two points in a user-item matrix. The two points have a row-row distance and column-column distance. We first precompute the row-row and column-column similarity matrices of an input user-item matrix using equation 4.2. Using precomputed row-

row and column-column similarities we find the distance of each unrated user-item pair from original rated user-item pairs.

The equation for finding the distance between each unrated and all rated records is as follows:

$$d\left[(r,c),(r',c')\right] = \left(\frac{1}{|R|}\sum_{r'\in R\setminus r} d\left(r,r'\right) + \frac{1}{|C|}\sum_{c'\in C\setminus c} d\left(c,c'\right)\right) \tag{4.3}$$

where (r,c) is unrated user-item pair and (r',c') is rated user-item pair. $d\left(r,r'\right) = cos\left(r,r'\right)$; similarly, $d\left(c,c'\right) = cos\left(c,c'\right)$. $|R|$ and $|C|$ is initial number of rows and columns in user-item matrix. We use them in equation as balancing weights. We maintain the sets $S^+ = \{R^+, C^+\}$ *and* $S^- = \{R^-, C^-\}$. At any time $S^+ \bigcup S^- = S$, Initially when we have no contrast points in our matrix $S^+ = S$ *and* $S^- = \emptyset$.

So using equation 4.3, we get the distances of all the zeros or unknown records and replace them with their respective distances. We pick the maximum value and convert it into +1. This maximum value record is our first contrast point. We keep track of these maximum value records for future use. After we get the first contrast point, set $S^+ = S^+ - 1$ *i.e.*, $R^+ = R^+ - 1$ *and* $C^+ = C^+ - 1$; *and* $S^- = S^- + 1$ *i.e.*, $R^- = R^- + 1$ *and* $C^- = C^- - 1$. We use equation 4.3 for finding first contrast point and then it is never used for exploring subsequent contrast points.

Then we precompute the user-user and item-item similarity matrices again for newly formed matrix which contains original 0's, 1's and contrast point with value +1. During finding of contrast points we keep their values as +1, not -1. But after finding all the contrast points we change their values to -1. The equation for finding second and subsequent contrast points is as follows:

$$d\left[(r,c),\{(r',c'),(r'',c'')\}\right] = \left[\left(\frac{1}{|R^+|}\sum_{r'\in R^+\setminus r} d(r,r') + \frac{1}{|C^+|}\sum_{c'\in C^+\setminus c} d(c,c')\right) + \left(\frac{1}{|R^-|}\sum_{r''\in R^-\setminus r} d(r,r'') + \frac{1}{|C^-|}\sum_{c''\in C^-\setminus c} d(c,c'')\right)\right] \tag{4.4}$$

where (r,c) is unrated user-item pair; $(r',c') \in S^+ \; and \; (r'',c'') \in S^-$. $|R^+|$, $|C^+|$, $|R^-|$ and $|C^-|$ act as balancing weights for distances between points. If suppose our first contrast point index is (434 , 282) and second contrast point index is (434 , 831) i.e., same row as of first one. In this case, value of $|R^-|$ will not be increased by 1 but value of $|C^-|$ will be increased by 1. Similarly value of $|R^+|$ will not be decreased but value of $|C^+|$ will be decreased by 1. So we need to keep track of indexes of every contrast point. The equation 4.4 is used for finding second and subsequent contrast points. The concept is, when we go for exploring more contrast points, we try to find those contrast points which are not only farthest from original rated user-item pairs but also from previously found contrast points. The equation 4.4 find the distance between each unrated user-item pair from all original rated user-item pairs as well as from previously computed contrast points. Whichever unrated user-item pair gives maximum value is our contrast point.

To prove that our distance approach is correct, we find the overall distance from each contrast point to all other contrast points and original rated records which is summed over all contrast points. The equation is given below:

$$OD = \sum_{l\in S^-}\left(\frac{1}{|S^+|}\sum_{l_1\in S^+\setminus l} d(l,l_1) + \frac{1}{|S^-|}\sum_{l_2\in S^-\setminus l} d(l,l_2)\right) \tag{4.5}$$

We expand the equation 4.5 as:

$$OD = \sum_{(r,c)\in S^-} \left[\left(\frac{1}{|R^+|} \sum_{r_1 \in R^+ \setminus r} d(r, r_1) + \frac{1}{|C^+|} \sum_{c_1 \in C^+ \setminus c} d(c, c_1) \right) + \left(\frac{1}{|R^-|} \sum_{r_2 \in R^- \setminus r} d(r, r_2) + \frac{1}{|C^-|} \sum_{c_2 \in C^- \setminus c} d(c, c_2) \right) \right] \tag{4.6}$$

where $(r_1, c_1) \in S^+ \; and \; (r_2, c_2) \in S^-$. Values of $|R^+|, |C^+|, |R^-| \; and \; |C^-|$ are taken from equation 4.4 result i.e., when we have all the contrast points.

We compare our approach with traditional SVD method in which we find the contrast points by doing SVD of input user-item matrix and reconstruct it again. After reconstruction, we find all those points which have smallest values and their number is equal to number of original rated items in input matrix. Then we assign those points as -1 value in original matrix. We use equation 4.6 to find the overall distance as in our approach. we find the overall distance from our approach is more than SVD method, which shows that contrast points in our approach are farthest. So in this way, we prove our approach is better than SVD method. We examined it on small synthesized matrix, not on real dataset as time complexity is very high. Refer to table 5.1 for time and space complexity comparison of algorithms. As we could not able to implement our novel Distance Approach on real dataset, we come up new approach which is basically an approximation of Distance Approach. We name it "Efficient Distance Approach" and discussed in chapter 1.

4.1.1 Time Complexity

Suppose m is number of rows and n is number of columns in input matrix. p is the number of original rated user-item pairs. Total number of entries in matrix is $m * n$. Total number of zeros in matrix is $(mn - p)$. To find first contrast point, we need to find the distance of each zero from all p's. Time complexity to find

first contrast point is $O(pmn - p^2)$. Then time complexity to find all remaining contrast points is $O(p^2mn - p^3)$. If we see mn is much greater than p because of data sparsity problem in recommender systems. Therefore upper bound time complexity of Distance Approach is $O(p^2mn)$.

CHAPTER 5

EFFIECIENT DISTANCE APPROACH

5.1 Efficient Distance Approach

This approach is an approximation of our original Distance Approach. In this approach, our main aim is to find the points in user-item matrix which are most unlikely to be rated in future. We call these points as "Contrast Points", which are basically farthest from the rated entries. The philosophy of this approach and original Distance Approach is same but the way of achieving is different. We use Singular Value Decomposition as an underlying matrix factorization technique.

The algorithm takes four arguments as input. These are User-Item input matrix with rated and unrated records, number of contrast points, rank of matrix for SVD and empty index matrix of size $c \times 2$. The output of the matrix is X with all desired contrast points. In line 3 we assign the input matrix A to temporary matrix T. Line 4 of algorithm is entry point of for-loop where number of iterations are equal to number of contrast points(c). In line 5 we do the Singular Value Decomposition of input user-item matrix(A) which contains original zeros(unrated recored) and original ones(rated records). After doing SVD we get three matrices U, S and V, where k is the rank of the matrix(A). In line 6, we take first k columns of matrix U and assign to intermediate matrix U1. In line 7 we take first k rows and first k columns of matrix S1 and assign to intermediate matrix S1. In line 8, we take first k columns of matrix V and assign to intermediate matrix V1. In line 9 we reconstruct the matrix T, which was decomposed in line 5 as $T = U1 * S1 * V1^T$. Then in line 10 we find the index of minimum value of reconstructed matrix T and assign to index matrix Y. This

minimum value record is basically our contrast point. The purpose of Index Matrix Y is to store the indexes of contrast points which will be used when we come out of the for-loop. Last step of for-loop i.e., in line 11 we replace the minimum value records with the value $\frac{\sqrt{c}}{i}$, where c is number of contrast points and i is iteration number of for-loop. We assign the contrast point index with $\frac{\sqrt{c}}{i}$ because it acts as weight and when we get the second and subsequent contrast points they should be distant from themselves also. When the for loop is complete, we replace the values of indexes stored in index matrix Y, in original input matrix A with -1. Last step of algorithm is to assign matrix A to matrix X. This way we achieve our purpose of finding points which are farthest from the original rated records. We use this matrix to get the test and training portions for our experiments. The experimental procedure is defined in chapter 6. The table 5.1 shows time and space complexities of algorithms used in this paper.

Table 5.1. Algorithms Complexity Comparison

Methods	Time Complexity	Space Complexity
EDA	$O(min(cnm^2, cmn^2))$	O(mn)
DA	$O(p^2mn)$	O(mn)
SVD	$O(min(nm^2, mn^2))$	O(mn)

Algorithm 1 Efficient Distance Approach

1: Input:

A: User-Item Matrix with rated and unrated records

c: Number of Contrast Points

k: Rank of matrix for SVD

Y: Empty Index matrix of size $c \times 2$

2: Output:

X: Resultant Matrix with all desired contrast points

3: $T \leftarrow A$

4: **for** $i = 1$ to c **do**

5: $[U, S, V] \leftarrow SVD\,(T)$

6: $U1 \leftarrow U\,(:, 1:k)$

7: $S1 \leftarrow S\,(1:k, 1:k)$

8: $V1 \leftarrow V\,(:, 1:k)$

9: $T \leftarrow U1 * S1 * V1^T$

10: $Y\,[i, :] \leftarrow IndexOf\,(min\,(T))$

11: $T\,[Y] \leftarrow \sqrt{c}/i$

12: **end for**

13: $A\,[Y] \leftarrow -1$

14: $X \leftarrow A$

CHAPTER 6

EXPERIMENTAL EVALUATION

This section describes the experimental verification of our algorithm to find contrast points. We first present our experimental platform - the data set, the evaluation metric , and the computational environment. Then we present our experimental procedure.

6.1 Experimental platform

Data set. We used the MovieLens data set [13]. MovieLens is a web-based research recommender system that debuted in Fall 1997. The data set was converted into user-item matrix that had 943 rows (users) and 1682 columns (movies). The matrix had 100,000 ratings (1-5) and all unrated items had value zero, for our experiments We converted the value of all rated user-item pairs into +1 as we are concerned to predict "Who rate What" rather than "What are the ratings". For getting test data, we used user and item percentages. We set 30% as user percentage and 40% as item percentage for our experiments.

6.2 Evaluation Methodology

For our experiments, we measure the prediction quality using traditional Precision-Recall methods in a Top-N fashion. They are widely used statistical accuracy metrics. Precision is defined as the ratio of relevant items selected to number of items selected. Precision represents the probability that a selected item is relevant. Precision is defined mathematically as,

$$Precision = \frac{N_{rs}}{N_s} \tag{6.1}$$

where, N_{rs} is number of relevant items selected and N_s is number of items selected.

Recall is defined as the ratio of relevant items selected to total number of relevant items available. Recall represents the probability that a relevant item will be selected. Recall is defined mathematically as,

$$Recall = \frac{N_{rs}}{N_r} \tag{6.2}$$

where, N_{rs} is number of relevant items selected and N_r is total number of relevant items available.

Precision and recall depend on the separation of relevant and non-relevant items. For our dataset, originally items are rated from 1-5 but we do not use as such because we are concerned with the rated records only, not with the ratings. We predict "who rate what", not "What are the ratings". We change the ratings and give all the rated records as value +1. MovieLens dataset has a rating scale of 1 to 5 and is commonly transformed into a binary scale by converting every rating of 4 or 5 to relevant and all ratings of 1 to 3 to non-relevant". But in our case we consider all rated records i.e., having value as +1 as "relevant" and all unrated records as "non-relevant".

6.3 Environment

All our experiments are done using MATLAB, running on Windows machine with 2.1 GHz Intel Pentium core 2 duo processor with 3 GB RAM, and 2 MB of cache memory.

6.4 Experimental Procedure

We introduce two approaches to achieve our goal of finding contrast points in input user-item matrix. First one is Distance Approach where we use cosine based similarity to find distance between unrated and rated records. But as we mentioned earlier that we could not able to implement this approach on real movielens data set due to its high time complexity. Due to this hurdle, we examined it on small test matrix.

The second approach is basically an approximation of distance approach. We call it an Efficient Distance Approach. We first take input user-item matrix formed after changing all the ratings (1-5) into 1, as we are concerned with rated records only not with their ratings. Then in next step we implement our Efficient Distance Approach as described in chapter 1. We find all the desired contrast points in matrix. Next step is to do initialization and dividing the matrix into *training* and *test* portion by using user and item percentages. We first randomly choose all the required number of users as set by user percentage and then randomly choose user-item pairs. If we find 1 then we change it with 2, for 0 we change it with 3 in matrix. These are called masked out values and we do the initialization of them by filling them with the average of both dimensions (user and item). This is better than using zero for initialization. The masked out values are the predictions. All these masked out values are stored in test matrix which we use for assessing the predictions in future. We do not touch the contrast points for initialization, as we do not want to loose them. But for test matrix we change contrast points value to 4 just to make them differentiable.

After doing initialization and forming test portion of matrix, We use the initialized matrix to do rank k approximation based on the first k singular vectors i.e. we use Singular Value Decomposition(SVD) in order to reduce noise i.e., error in the

matrix. Then We measure the prediction quality using traditional Precision-Recall methods in a top-N fashion. We sort the predicted values. We pick $N = 1, 2, 3, \cdots, 10$ top values. We assess the predicted values with the known masked out information. For comparing with traditional SVD method where we follow all the experimental procedure steps except finding contrast points which is basically the key of improving accuracy.

CHAPTER 7

RESULTS AND DISCUSSION

7.1 Results and Discussion

We examined our original distance approach on synthesized dataset which contains 67 users and 70 items with 281 rated records i.e., around 6% of total records. The overall distance in equation 4.6 in our case came higher than traditional SVD method, which proves that our motive of finding farthest contrast points is achieved. As we mentioned earlier we can not implement this approach on real dataset as its time complexity is very high which is its main limitation. Due to this we come up with the Efficient Distance Approach to find contrast points. We tested it on real data set i.e., Movielens dataset of size 943 users and 1682 items with 100,000 rated records. The known records is around 6.3% of total records. We have shown results for different ranks of matrix i.e., k = 10, 20, 30 and 40 for EDA and SVD respectively. For each rank, we have shown results when number of contrast points is c(contrast points) = 200, 500, 1000 and 2000 for EDA. For all the results, the user and item percentage was kept fixed at 30% and 40% respectively. The figure 7.1 shows Precision v/s Recall graph of MovieLens Dataset when k = 10. The table 7.1 shows the performance comparisons of recommendation methods when k = 10. Similarly there are figures 7.2, 7.3, 7.4 and tables 7.2, 7.3, 7.4 shown for k = 20, 30 and 40 respectively. When we examine each Precision v/s Recall graph of MovieLens Dataset for particular k, we find the maximum precision and recall for c = 2000. Refer figure 7.5 to see maximum Precision and Recall for different ranks. If we compare the graphs for all

k's, then we find the maximum precision and recall when k = 30. The experimental results illustrate the effectiveness of Efficient Distance Approach.

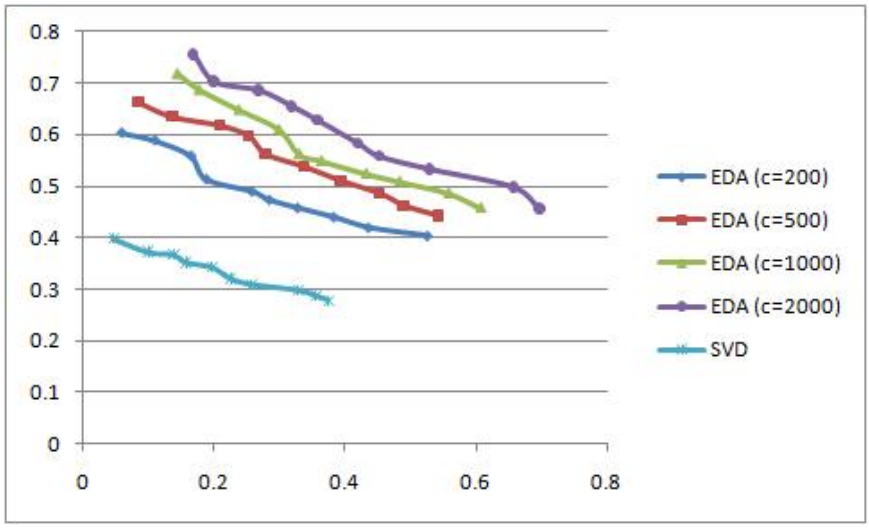


Figure 7.1. Precision v/s Recall graph of MovieLens Dataset (k=10).

Table 7.1. Performance Comparisons of Recommendation Methods (k=10)

Methods	Contrast Points (c)	Max. Precision	Max. Recall
EDA	200	0.6028	0.5263
	500	0.6628	0.5425
	1000	0.7183	0.6072
	2000	0.7562	0.6972
SVD	N/A	0.3988	0.3751

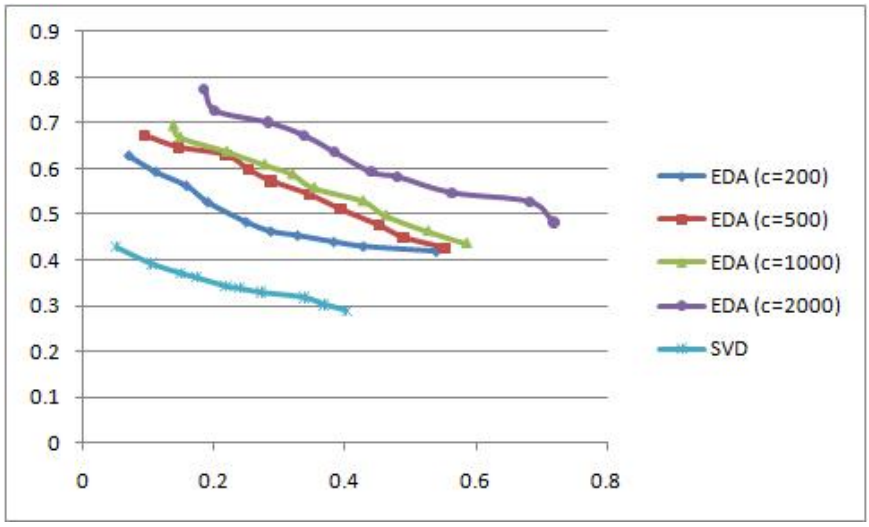


Figure 7.2. Precision v/s Recall graph of MovieLens Dataset (k=20).

Table 7.2. Performance Comparisons of Recommendation Methods (k=20)

Methods	Contrast Points (c)	Max. Precision	Max. Recall
EDA	200	0.6281	0.5392
	500	0.6724	0.5524
	1000	0.6952	0.5862
	2000	0.7729	0.7193
SVD	N/A	0.4294	0.4025

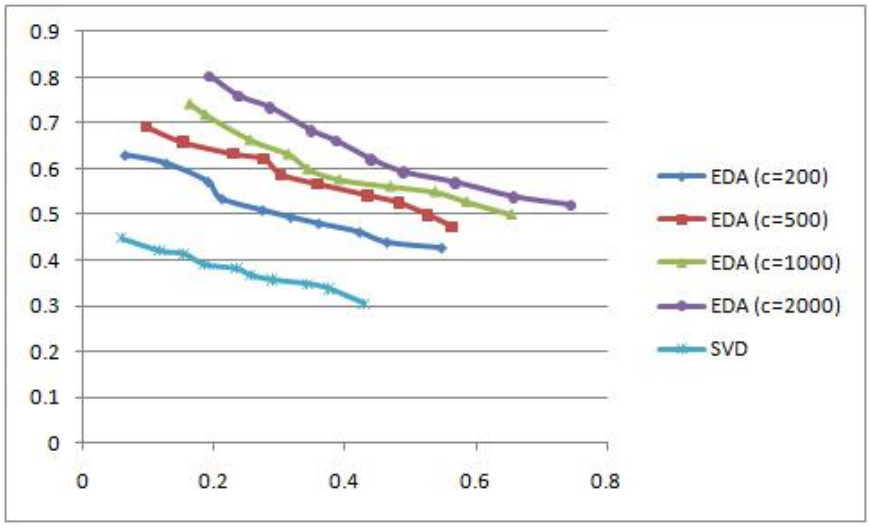


Figure 7.3. Precision v/s Recall graph of MovieLens Dataset (k=30).

Table 7.3. Performance Comparisons of Recommendation Methods (k=30)

Methods	Contrast Points (c)	Max. Precision	Max. Recall
EDA	200	0.6312	0.5473
	500	0.6917	0.5627
	1000	0.7417	0.6551
	2000	0.8017	0.7451
SVD	N/A	0.4486	0.4292

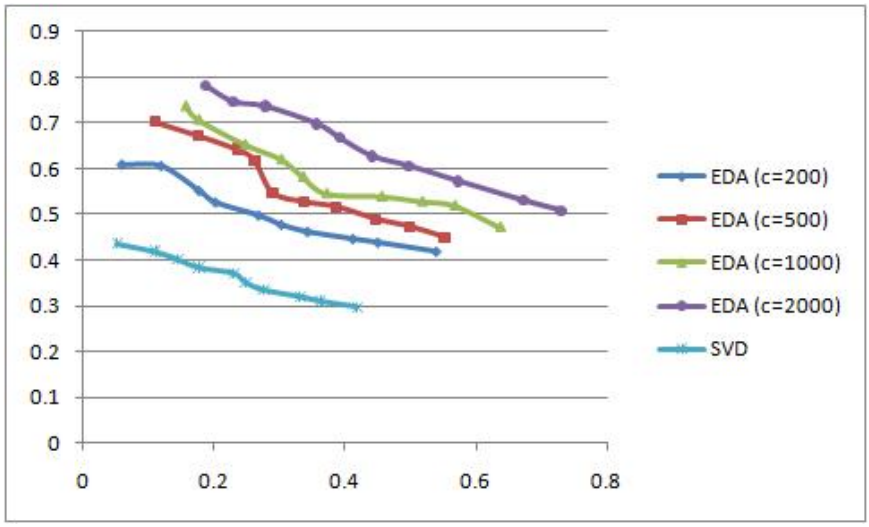


Figure 7.4. Precision v/s Recall graph of MovieLens Dataset (k=40).

Table 7.4. Performance Comparisons of Recommendation Methods (k=40)

Methods	Contrast Points (c)	Max. Precision	Max. Recall
EDA	200	0.6103	0.5382
	500	0.7028	0.5517
	1000	0.7362	0.6381
	2000	0.7828	0.7303
SVD	N/A	0.4367	0.4189

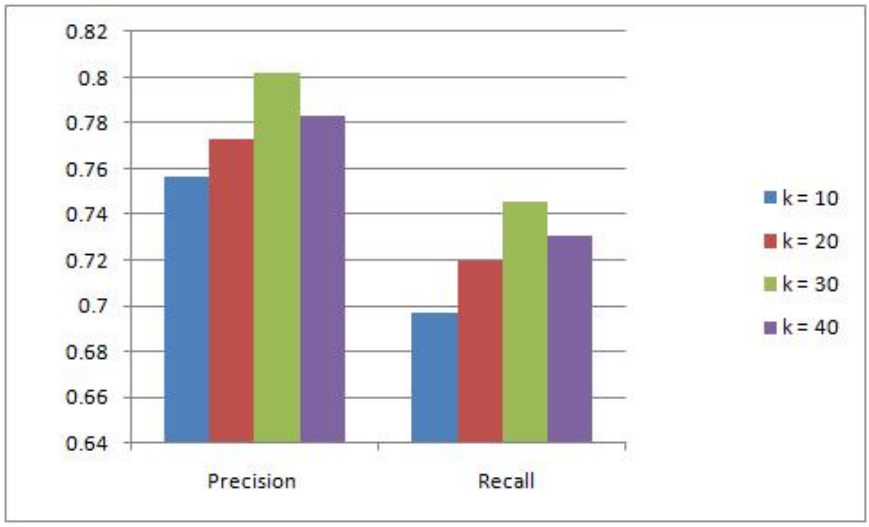


Figure 7.5. Maximum Precision and Recall for different ranks.

CHAPTER 8

SUMMARY

8.1 Summary

In this paper, we propose an algorithm to predict "Who rate what". We show that we are not concerned with the ratings of items instead the rated items. The algorithm finds the contrast points in matrix which are farthest from rated records and themselves, due to which we eliminate the chances of those points being predicted in future, thereby improving results. We give two approaches of finding contrast points, (1) Distance Approach (DA) using cosine similarity and (2) Efficient Distance Approach (EDA) using SVD as underlying matrix factorization technique. Philosophy of both approaches is same i.e. using distance to find contrast points but the underlying methodology is different. We clarify that we can not use first approach on real dataset because of high time complexity due to which we propose EDA. Our novel idea of finding contrast points is unique. Experimental results indicate our approach outperform traditional Singular Value Decomposition method.

APPENDIX A

SYMBOLS TABLE

A.1 Table of Symbols Used

Table A.1. Table of Symbols used.

Symbols	Definition
S_{xy}	Set of all items co-rated by both users x and y
$sim\,(x, y)$	Similarity between two vectors x and y
$cos\,(\vec{x}, \vec{y})$	Cosine of the angle between two vectors x and y
m	Number of rows in matrix
n	Number of columns in matrix
p	Total number of original rated records in matrix
R	Number of rows/users in matrix
C	Number of columns/items in matrix
R^+	Number of rows for rated records
C^+	Number of columns for rated records
R^-	Number of rows for contrast points
C^-	Number of columns for contrast point
c	Number of Contrast points
N_{rs}	Number of relevant items selected
N_s	Number of items selected
N_r	Total number of relevant items
U_k	$m \times k$ matrix containing first k columns of U
V_k	$n \times k$ matrix containing first k columns of V
Σ_k	$k \times k$ matrix containing first k entries of Σ

REFERENCES

[1] Pandora, http://www.pandora.com.

[2] Grouplens, http://http://movielens.umn.edu/.

[3] M. H. Pryor, "The Effects of Singular value decomposition on Collaborative filtering," 2001.

[4] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Application of dimensionality reduction in recommender systems-a case study," 2000.

[5] G. H. Golub and C. F. Loan, "Matrix Computations." 1983.

[6] K. Goldberg, T. Roeder, D. Gupta, , and C. Perkins, "Eigenstate: A constant time collaborative filtering algorithm," 2001, pp. 133 – 151.

[7] S. T. D. M. W. Berry and G. W. O'Brian, "Using Linear Algebra for Intelligent Information Retrieval," 1995, p. 37(4).

[8] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, , and R. Harshman, "Indexing by Latent Semantic Analysis," 1990, p. 42(6).

[9] U. Shardanand and P. Maes, "Social information filtering: Algorithms for automating "word of mouth"," 1995.

[10] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-Based Collaborative Filtering Recommendation Algorithms," 2001, pp. 285–295.

[11] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "Social Information Filtering: Algorithms for Automating "word of mouth"," 1995.

[12] G. Adomavicius and A. Tuzhilin, "Toward the Next Generation of Recommender Systems: A Survey of the state-of-the-Art and Possible Extensions," 2004.

[13] MovieLens, http://www.grouplens.org/node/73.

[14] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Incremental Singular Value Decomposition Algorithms for Highly Scalable Recommender Systems."

[15] A. B. M. Kurucz and K. Csalogany, "Methods for large scale SVD with missing values."

BIOGRAPHICAL STATEMENT

Rahul Singhal was born in Agra, India, in 1986. He received his Bachelor of Technology degree in Computer Engineering from Govind Ballabh Pant University of Agriculture and Technology, Pantnagar, India in 2008, his Masters of Science degree in Computer Science from The University of Texas at Arlington in 2010. He did his Summer Internship from Indian Institute of Technology, Kanpur, India during June-July 2007. He worked as Graduate Research Assistant, System Administrator under Computer Science and Engineering department. His current research interests is in the area of Algorithms and Data Structures.