CHAPTER 1: INTRODUCTION

1.1 RECOMMENDER SYSTEMS

With the increase in the amount of information over web, there are a lot of options available to the users to choose from. This phenomenon is known as Information Overload. Recommender Systems or Recommendation Systems (RS) that are a type of Information Filtering Systems helps users to identify items that are aligned to their tastes and preferences from the huge amount of item set that is actually present.

Recommender Systems have many applications and are used in diverse areas like social networking, news articles, music, movies, web pages, shopping etc.

Recommendations are made generally by using the past purchases or preferences of the users and making a rating prediction for other items apart from the ones already liked by the user. Finally the items are ranked according to the predicted values and the ones having highest ratings are recommended to the users.

Figure 1.1 shows the various approaches by which the recommendations can be done [25].

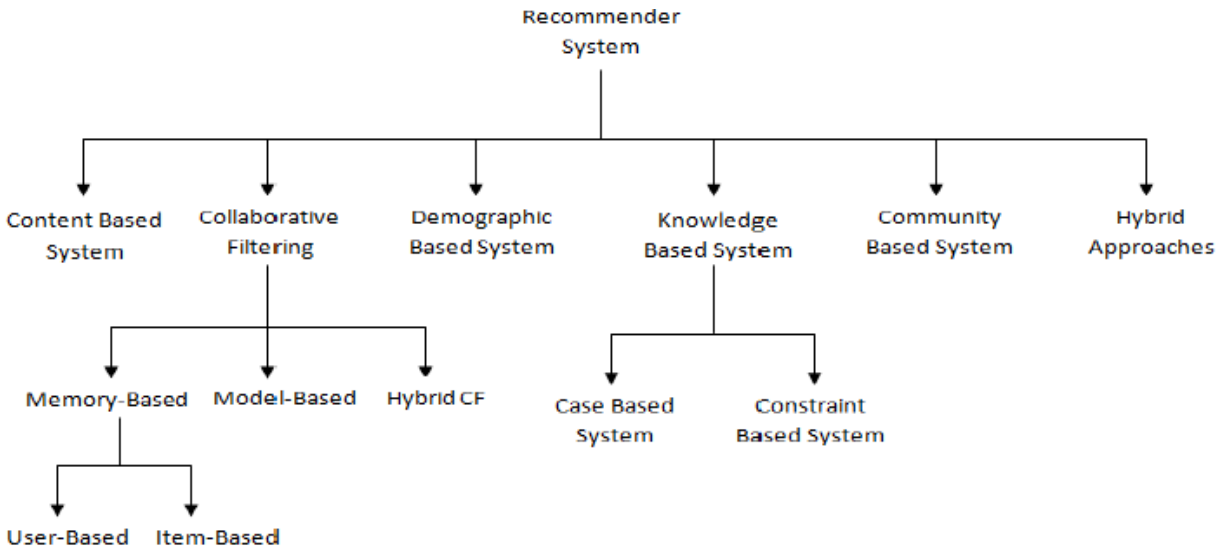


Figure 1.1 Various Recommender System Approaches

A brief description of all these approaches is provided as follows:-

- Collaborative filtering which is the most popular approach make recommendations based on preferences of other users in the system that are most similar to the target user.
- Content based approaches construct user profiles and item profiles that consist of their most important characteristics. For example, information filtering techniques like TF-IDF are used to construct the profiles of documents as 'bag of words' or the set of words that describe the document.
- Demographic based approach to RS considers the demographic information of its users like age, occupation, gender etc. to make groups of its users. It is based on the idea that users having similar demographic information tend to have similar tastes and hence can be used for making recommendations to other members of the group.
- Knowledge based approach uses explicit knowledge about the users to make quality recommendations. Through such information, the RS can decide whether a particular item is useful for a user or not. Such systems do not face cold start or ramp up problem. They are further classified as case based or constraint based systems depending on how they gather information or knowledge.
- Community based approach also called the social recommender system, is based on recommending items to a user that are liked by his/her friend on social media platforms. Rather than taking preferences of random similar users into account, the friends of a user are considered for making predictions of his preferences.
- Hybrid approached combines two or more of the above approaches to utilize the advantages of each individual approach.

1.2 SCOPE OF WORK

In this era of information overload, it has become extremely important to study information filtering techniques to present a meaningful set of items to the users from a large set that is relevant to the needs of the users. Hence it is very important to study recommender systems. Experiences have shown that improvement in the quality and accuracy of recommendations leads to increasing profits for the businesses. Collaborative filtering which is the most popular algorithm among the different RS algorithm is widely used for making recommendations e.g. Amazon, Netflix etc. But the conventional models are prone to several attacks. Some of these attacks include sparsity, scalability, shilling attacks, cold start problem etc. Because of its

immense impact and scope for improvement, various modifications and optimizations have been proposed to the conventional Algorithm in research. This thesis gives a survey of the new techniques that have been proposed and compares them over different parameters like dataset, evaluation criteria used etc. Also the various techniques that are a part of collaborative filtering are discussed with all the details and steps involved in the process. Hence it gives an insight and a complete knowledge about the details to all the researchers who are working in this area. Also, a collaborative filtering RS that simultaneously alleviates the shilling attack, sparsity problem and cold start problem has been proposed. The results are presented in an effective manner using the commonly used evaluation metrics whose details are provided.

1.3 RESEARCH GOALS

The following are the research goals for this work:-

- To study the various techniques of collaborative filtering.
- To study the various attacks/ challenges to collaborative filtering.
- To present a survey of the various improvements to the conventional algorithm.
- To alleviate the shilling attacks.
- To solve the sparsity problem.
- To provide various techniques for providing quality recommendations to cold start users.

1.4 ORGANISATION OF THE THESIS

The thesis comprises of 8 chapters and 3 appendices. The rest of the thesis is organized as follows:

Chapter 2:

Provides a Literature Survey of the Collaborative Filtering Research; the different papers considered are compared across different parameters which are author/year, technique(s) used, domain/dataset and evaluation metric.

Chapter 3:

This chapter gives framework and details of various approaches to collaborative filtering. Also the various attacks that are possible are also studied with their causes

Chapter 4:

This chapter provides details about the various types of shilling attacks. It also gives details of the algorithm used for its detection and removal i.e. principal components analysis (PCA) along with its optimization.

Chapter 5:

This chapter provides insights on how to alleviate the sparsity and cold start problems.

Chapter 6:

The proposed system architecture along with the pseudocode of the same is presented in this chapter.

Chapter 7:

The implementation platforms, results and analysis of the proposed system are described here.

Chapter 8:

We conclude the project here and also mention the future work.

Description of the appendices is as below:-

Appendix A: Code Snippets

Appendix B: Snapshots of the System

Appendix C: Publication

CHAPTER 2: LITERATURE SURVEY

Studies and Experiences have shown that Recommender Systems substantially increase sales at on-line stores and other online businesses that have many products to offer to their users. Hence, a lot of research is going on in this direction from the past decade. The two major categories of RS i.e. content based and collaborative filtering work well on small data set or item set but for real world scenario where millions of users and items are involved, some modification or improvement is required in such algorithms.

As already stated, in this work we focus the most popular class of recommender systems i.e. collaborative filtering, its challenges and solutions for increasing its performance and accuracy.

Collaborative Filtering works by computing similarity between users and/or items. But it faces many issues like sparsity, scalability issue, shilling attacks and so on [5] [6]. All these issues and challenges affect the accuracy and performance of RS and hence are needed to be minimized. Many hybrid algorithms are proposed to benefit from the advantages of individual techniques.

For improving the accuracy of CF based RS, many techniques are proposed. One of the techniques is to construct a trust network first and then perform the recommendation [10]. Other technique is to so user / item clustering using any of the well-established clustering algorithms like k-means or k-medoids [18] [26]. Some Algorithms propose to use the domain knowledge of the users as that can help in recommending the items [14]. Hybrid techniques are also used to increase the recommendation accuracy [18].

Context- aware RS have also been proposed in the recent years to take context of the users into account. For instance, users like to listen to particular kind of music on a particular time of the day. [6].

Hence Collaborative Filtering based RS are an important class of RS and an area of active research. If a significant improvement in such algorithms is done, it will impact the businesses all around the globe and would increase profitability.

We present a comparative study of Collaborative Filtering Improvements as follows.

S.No	Author, Year	Technique(s) used	Domain/Dataset	Evaluation Metric
1	(Huizhi Liang, Yue Xu et al., 2008) [19]	Tag Information Incorporation in CF technique	Amazon.com	Precision, Recall
2	(SongJie Gong, GuangHua Cheng, 2008) [13]	User Interest Mining	MovieLens	MAE
3	(P. Braak, N. Abdullah et al., 2009) [26]	User Profile Clustering using genre interest	Netflix	Average Time for Prediction
4	(Gilda Moradi Dakhel, Mehregan Mahdavi, 2011) [8]	k-means clustering and neighbor's voting	MovieLens	NMAE, Time Comparison
5	(Pooyan Adibi, Behrouz Tork Ladani, 2013) [1]	Rating Timestamp, group membership, interest and similarity usage in CF technique	MovieLens	MAE, Coverage
6	(Jun Zou, Faramarz Fekri, 2013) [31]	Shilling Attack Detection by a belief propagation approach	MovieLens 100k	Precision
7	(Abhishek Kaleroun, Dr. Shalini Batra, 2014) [15]	Ant Colony Optimization, Trust based Approach	MovieLens	Precision, Recall, F Measure
8	(Abinash Pujahari, Vineet Padmanabhan, 2015) [24]	Combining user-user and item-item CF techniques for Group Recommender Systems (GRS)	MovieLens	Precision

9	(Feng Xie, Zhen Chen et al., 2015) [29]	Item Similarity Learning Methods using Stochastic Gradient Descent (SGD) Algorithm	MovieLens100k and MovieLens1M	MAE, RMSE
10	(Faris Alqadah, Chandan K. Reddy · Junling Hu, Hatim F. Alqadah, 2015) [3]	Bi-cluster Neighborhood Framework	Paypal_big, Paypal_small, delic_bookmarks, lastfm_friends, lastfm	Five-time Leave-One-Out cross validation (LOOCV), Hit Rate (HR), Average Reciprocal Hit-Rank (ARHR)
11	(D. Yongping, D.Xiaoyan et al., 2016) [10]	Trust Network Construction	Epinions	MAE, RMSE
12	(Anand Shanker Tewari, Asim Gopal Barman, 2016) [27]	Trust based Social Network, Association Rule Mining	Live data, trust Network of NIT Patna, India	Precision
13	(Bushra Alhijawi, Yousef Kilani, 2016)[2]	Genetic Algorithm for Similarity Computation	MovieLens, Synthetic Data	MAE, Precision, Recall
14	(Rahul Kataria and O.P. Verma, 2016) [16]	Particle Swarm Optimization, K-means Clustering, Fuzzy c-means	MovieLens	MAE
15	(Mahdi Nasiri, Behrouz Minaei, 2016) [23]	Matrix Factorization	Movielense 100k	RMSE
16	(Farnaz Ghaznavi, Sasan H.	Usage of negative	Extended	RMSE, MAE

	Alizadeh, 2017) [12]	Similarity and Distrust Information	Epinions and Epinions	
17	(Sundus Ayyaz, Usman Qamar, 2017) [4]	Effective User neighborhood using KNN and threshold based neighbors.	MovieLens 1M	RMSE
18	(Vahid Faridani, Mehrdad Jalali, Majid Vafaei Jahan, 2017) [11]	Effective Trust Computation	Flixster	MAE, Ratings Coverage (RC), F-Measure

Table 2.1 Comparative Study of Collaborative Filtering research

CHAPTER 3: COLLABORATIVE FILTERING

3.1 OVERVIEW

Collaborative filtering is one of the most important and popular algorithms for recommender systems that generally predict a particular user's rating based on the ratings that are given to the target item by similar users. The term was first coined by D. Goldberg et al. where they proposed a manual collaborative filtering technique for tapestry mailing system. The idea behind the algorithm is that the users who have rated the common items in a similar rating fashion will most probably have similar rating preferences for other items as well.

The similarity among users and/or items is obtained using common similarity measures like jaccard similarity, cosine or adjusted cosine, pearson correlation coefficient etc. to be discussed late in this chapter.

One of the advantages of considering like-minded users for making recommendations is that it overcomes the problem of 'over-specialization'. Over-specialization means that the items that are recommended are always of the same type i.e. there is no diversity in recommendations which is generally the case with content based approach. Focusing on user ratings rather than content helps to avoid such a problem.

3.2 FRAMEWORK OF COLLABORATIVE FILTERING

The general collaborative filtering framework consists of the following three steps [30]:

1. Data Collection
2. Preprocessing
3. Collaborative Filtering

This is shown in the Fig. 3.1

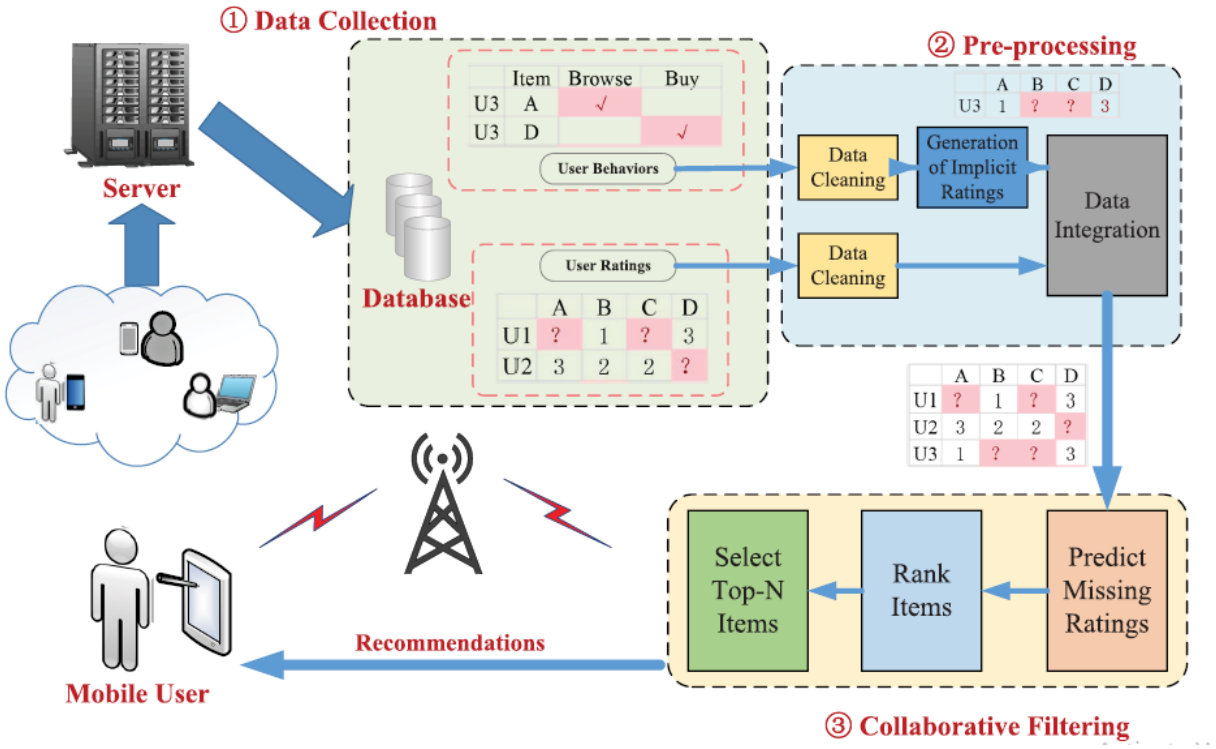


Figure 3.1 Framework of Collaborative Filtering

The data is collected from various sources and preprocessing step is performed for its homogenization. After this step, we obtain a matrix known as the rating matrix or utility matrix with blank entries which are predicted by the CF Algorithm. A sample rating matrix is shown in Fig. 3.2

User | Item	i1	i2	i3	i4	i5
u1		r12		r14	r15
u2	r21	r22			r25
u3		r32		r34	
u4			r43		r45

Figure 3.2 Rating or Utility Matrix

The steps of the framework are explained in detail below.

DATA COLLECTION

It is one of the most important activities of the entire process and the data mainly fall into the following 4 categories:-

1. Demographic data: It consists of the personal information of the users like name, telephone number, age etc. which helps businesses to construct users' profiles.
2. Production data: Here, classification of commodities is done based on their brands, functions etc. e.g., video tagging
3. User Behavior: e.g. playing duration of songs, book purchasing date etc.
4. User Rating: The actual ratings provided by the users.

PREPROCESSING STEP

The data as collected above is in various formats due to the heterogeneity of the devices and networks; hence preprocessing is done to ensure a consistent format. There are 3 sub-steps in this step:-

1. Data Cleaning: Due to transmission errors or equipment failures, noisy data may be present in the system. Also, users may arbitrarily rate the items to save time. Hence here, we apply certain outlier detection algorithms to perform cleaning of the data.
2. Generation of Implicit Ratings: The rating matrix obtained is severely sparse leading to the data sparsity issue. We can use the users' behaviors and apply machine learning techniques like neural networks to make a prediction model which can covert user behaviors into implicit ratings.
3. Data Integration: Explicit and implicit ratings are combined to form the rating matrix as shown in Fig. 3.2.

COLLABORATIVE FILTERING

Finally, Collaborative filtering is applied to make predictions regarding the user preferences. The detailed algorithm is explained in the next section.

3.3 APPROACHES TO COLLABORATIVE FILTERING

Collaborative Filtering Approaches are further classified as memory based, model based and hybrid techniques as shown in the Fig. 3.3

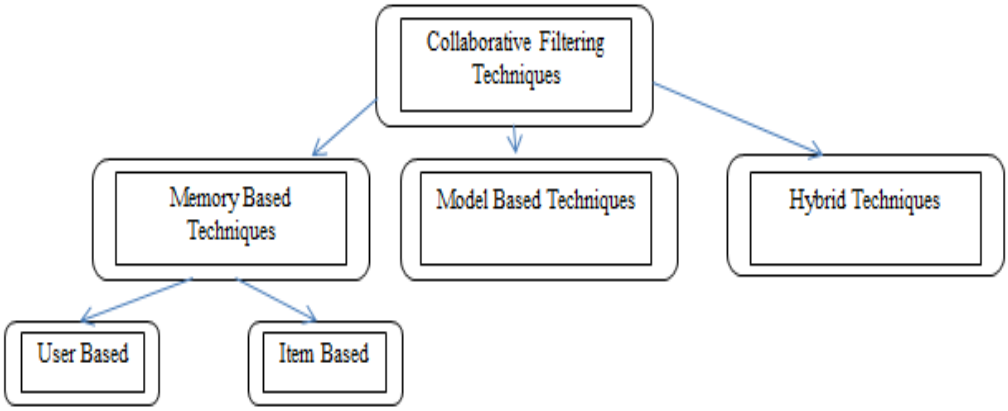


Figure 3.3 Classification of CF Techniques

These techniques are explained in detail as below:-

3.3.1 MEMORY BASED CF ALGORITHMS

These methods use the rating matrix directly for making predictions of the ratings. They are easy to implement but have large memory requirements for storing the complete rating matrix.

The general Memory based Model consists of the following steps [24]:-

1. Similarity Computation between users/items
2. Neighbor Selection
3. Prediction
4. Items Ranking
5. Selection of top k items

SIMILARITY METRICS

In order to find similarity values between users and/or items, the following similarity metrics are generally used.

1. Jaccard Similarity

Consider two users u and v. Jaccard similarity between these users is defined as:-

$$sim(u,v)^{Jaccard} = \frac{|I_u| \cap |I_v|}{|I_u| \cup |I_v|} \quad (3.1)$$

Where I_u and I_v are the sets of items rated by user u and user v respectively.

2. Cosine Similarity

$$sim(u,v)^{COS} = \frac{\vec{r}_u \cdot \vec{r}_v}{\|\vec{r}_u\| \cdot \|\vec{r}_v\|} \quad (3.2)$$

Here r_u and r_v respectively represent the rating vectors of users u and v. If a user has not rated a particular item, that rating is considered as zero.

3. Adjusted cosine similarity (ACOS)

$$sim(u,v)^{ACOS} = \frac{\sum_{p \in P} (r_{u,p} - \bar{r}_u)(r_{v,p} - \bar{r}_v)}{\sqrt{\sum_{p \in P} (r_{u,p} - \bar{r}_u)^2} \cdot \sqrt{\sum_{p \in P} (r_{v,p} - \bar{r}_v)^2}} \quad (3.3)$$

Here, In order to remove the user bias i.e. the fact that different users give different ratings to an item even if they like it to same extent (different rating scales).

P represents the set of all items. $r_{u,p}$ is the rating given by user u to item p. r_u(bar) is the average rating of user u. Similar notations are followed for user v.

4. Pearson Correlation Coefficient (PCC)

$$sim(u,v)^{PCC} = \frac{\sum_{p \in I} (r_{u,p} - \bar{r}_u)(r_{v,p} - \bar{r}_v)}{\sqrt{\sum_{p \in I} (r_{u,p} - \bar{r}_u)^2} \cdot \sqrt{\sum_{p \in I} (r_{v,p} - \bar{r}_v)^2}} \quad (3.4)$$

Here, I is the set of items that are rated by both the users u and v. The difference between ACOS and PCC is that PCC uses only the co-rated items. PCC generally performs better than the other metrics.

The value of similarity metric lies between -1 and 1.

Example:-

Consider the following rating matrix:-

	The Matrix	Titanic	Die Hard	Forrest Gump	Wall-E
John	5	1		2	2
Lucy	1	5	2	5	5
Eric	2	?	3	5	4
Diane	4	3	5	3	

Figure 3.4 Sample Rating Matrix

Where rows of the matrix represent users and columns are representing movies. Now, using Pearson correlation coefficient, the similarity values between each pair of users is given by the following matrix:-

	John	Lucy	Eric	Diane
John	1.000	-0.938	-0.839	0.659
Lucy	-0.938	1.000	0.922	-0.787
Eric	-0.839	0.922	1.000	-0.659
Diane	0.659	-0.787	-0.659	1.000

Figure 3.5 User Based Pearson Correlation

Similarly, we can use these methods for item similarity computation. For example, the Pearson correlation values for the pair of items in the above rating matrix are computes as follows:-

	The Matrix	Titanic	Die Hard	Forrest Gump	Wall-E
Matrix	1.000	-0.943	0.882	-0.974	-0.977
Titanic	-0.943	1.000	-0.625	0.931	0.994
Die Hard	0.882	-0.625	1.000	-0.804	-1.000
Forrest Gump	-0.974	0.931	-0.804	1.000	0.930
Wall-E	-0.977	0.994	-1.000	0.930	1.000

Figure 3.6 Item Based Pearson Correlation

Memory based CF Algorithms are further classified into 2 categories based on how whether we calculate similarity between users or items:-

1. **USER BASED APPROACH**

This algorithm works in two phases:-

1. User Neighborhood Formation phase: In this phase, we calculate the similarity between the target user u and other users using common similarity metrics as described above and select the top k neighbors who have rated the target item i. We denote these neighbors by $N_i(u)$
2. Recommendation phase: The predicted value of concerned rating is computed as follows:-

$$\hat{r}_{ui} = \frac{\sum_{v \in \mathcal{N}_i(u)} w_{uv} r_{vi}}{\sum_{v \in \mathcal{N}_i(u)} |w_{uv}|}. \tag{3.5}$$

where w_{uv} represent the similarity value between the target user and the nearest neighbor under consideration.

In the denominator, modulus is used so that the predicted ratings are within the legitimate range of rating scale.

Also, rating normalization with mean-clustering is performed to remove the user-bias. Therefore, the predicted rating is now computed as:

$$\hat{r}_{ui} = \bar{r}_u + \frac{\sum_{v \in \mathcal{N}_i(u)} w_{uv} \left(r_{vi} - \bar{r}_v \right)}{\sum_{v \in \mathcal{N}_i(u)} |w_{uv}|}. \tag{3.6}$$

The user based approach is pictorially depicted in the Fig. 3.7.

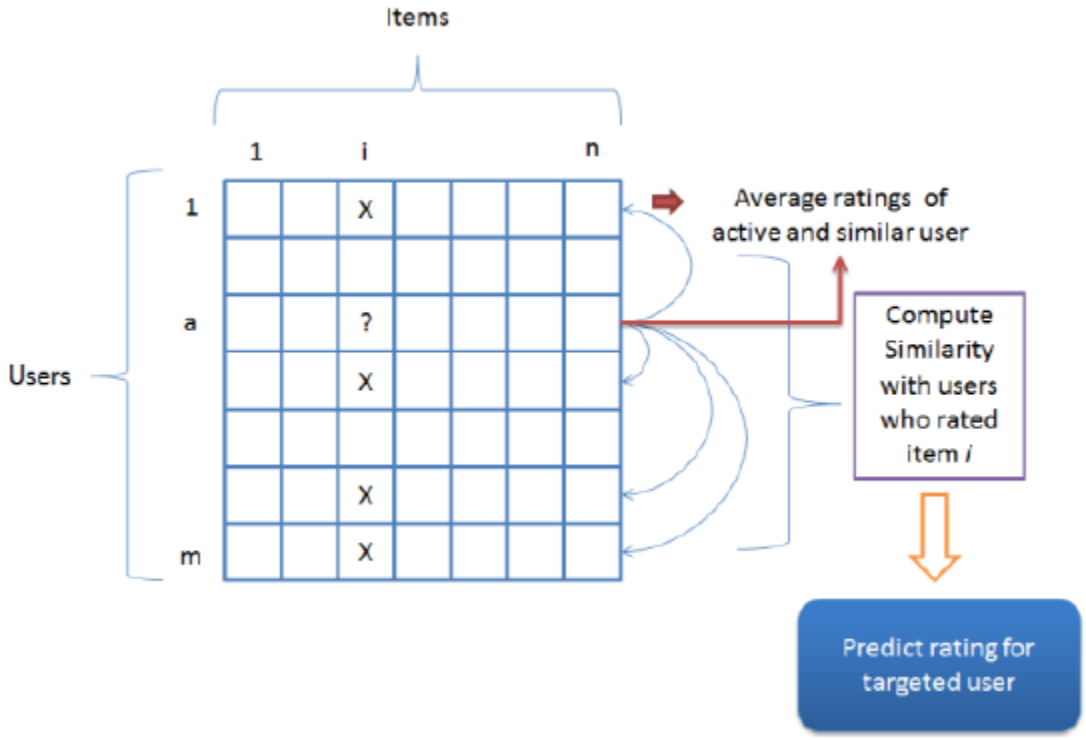


Figure 3.7 User based CF Process

Using this approach, Entry marked as '?' in Fig 3.4 is estimated as 4.75.

2. ITEM BASED APPROACH

Similar to the user-based method, item based method also works in two phases:-

1. Item Neighborhood Formation phase: In this phase, we calculate the similarity between the target item i and other items using common similarity metrics as described above and select the top k neighbors that are rated by the target user u. We denote these neighbors by $N_u(i)$

2. Recommendation phase: The predicted value of concerned rating is computed as follows:-

$$\hat{r}_{ui} = \frac{\sum_{j \in \mathcal{N}_u(i)} w_{ij} r_{uj}}{\sum_{j \in \mathcal{N}_u(i)} |w_{ij}|}. \tag{3.7}$$

where w_{uv} represent the similarity value between the target item and the nearest neighbor under consideration.

Also, rating normalization with mean-clustering is performed to remove the user-bias. Therefore, the predicted rating is now computed as:

$$\hat{r}_{ui} = \bar{r}_i + \frac{\sum_{j \in \mathcal{N}_u(i)} w_{ij} (r_{uj} - \bar{r}_j)}{\sum_{j \in \mathcal{N}_u(i)} |w_{ij}|}. \tag{3.8}$$

Table 3.1 shows a comparison between user based and item based approaches of the memory based CF Algorithms.

Categories of CF	Similarity	Application scenarios
User-based CF	User-user similarity	• The number of items is larger than that of users. • Users do not change frequently.
Item-based CF	Item-item similarity	• The number of users is larger than that of items. • Items do not change frequently.

Table 3.1 Comparison of user-based and item-based approaches

3.3.2 MODEL BASED CF ALGORITHMS

Due to the large memory requirements of the memory based techniques, model based techniques are used where a prediction model is constructed using data mining or machine learning

techniques. Examples of such techniques include matrix factorization, singular value decomposition, clustering based approaches etc.

3.3.3 HYBRID CF ALGORITHMS

These algorithms are a combination of memory-based and model based algorithms.

3.4 CHALLENGES AND ATTACKS TO COLLABORATIVE FILTERING

SCALABILTY ISSUE:

With the increasing users and items in the system, it becomes difficult for the CF algorithms to make predictions in real time. To compute similarity between users and/or items that re millions in number, increases the time complexity of the CF Algorithm.

DATA SPARSITY PROBLEM:

The utility or the rating database is quite large due to a large number of users and items. Users tend to rate only a few items leading to a large number of blank entries in the rating matrix and hence low prediction accuracy of CF recommender systems.

COLD START PROBLEM:

Cold start problems is one of the problems caused by data sparsity. It is difficult to recommend items to a new user as we have no information about his past preferences. This is called user cold start problem. Similarly, we have item cold start problem for new items that enter into the system.

CHANGING DATA SET:

Users and items are constantly added in the rating database with time, leading to the problem of changing dataset.

SHILLING ATTACKS:

Collaborative filtering based recommender systems are prone to shilling attacks wherein shilling users are inserted into the system. These users are injected into in order to manipulate the

recommender system [7] [9]. Biasing of recommendations of the target item is intended by the attacker so that he can promote/demote these items.

GRAY SHEEP PROBLEM:

'Gray sheep' users are the ones who have unusual item preferences and their interests do not match with other people in the rating database but since CF based techniques are based on similarity computation between users, it becomes difficult to make recommendations to gray sheep users.

LONG TAIL ISSUE:

Since collaborative filtering is based on past purchase history of users, it does not provide diversity in the recommendations they provide. Only a few popular items that are rated by most of the customers are used in recommendation leading to a 'rich-get-richer' effect.

The above mentioned challenges make CF recommender systems an active area of research. The challenges along with their root cause(s) are formulated in table 3.2

SNO	CHALLENGE/ATTACK	CAUSE(S)
1.	Scalability Issue	Presence of millions of users and/or items
2.	Data Sparsity Problem	Large number of missing or blank entries in the rating matrix
3.	Cold Start Problem	New users and/or items
4.	Changing Dataset	Dynamic nature of the rating matrix with constant inclusion of new users and/or items
5.	Shilling Attacks	Wrong or fake ratings provided by some users
6.	Gray Sheep Problem	Unusual interests of some users for items that is different from other users
7.	Long Tail Issue	Lack of diversity in recommendation

Table 3.2 Challenges and Attacks to Collaborative Filtering along with their cause(s)

CHAPTER 4: SHILLING ATTACKS

4.1 OVERVIEW

Collaborative filtering based recommender systems are prone to shilling attacks wherein shilling users are inserted into the system. These users are injected into in order to manipulate the recommender system. Biasing of recommendations of the target item is intended by the attacker so that he can promote/demote these items. Hence there are two types of attacks:

Push Attack: Here, attacker gives maximum rating to the target item in order to popularize or promote his/her own products or items.

Nuke Attack: Here, attacker gives minimum rating to the target item in order to demote the competitors' products or items.

Riedl and Lam first coined the term "shilling" and gave two attack models i.e. Average Bot and Random Bot.

A general shilling attack profile is shown in table 4.1:-

item	i_1^S	...	i_k^S	i_1^F	...	i_l^F	$i_1^\emptyset$	...	$i_1^\emptyset$	i^T
rating	$\delta(i_1^S)$	...	$\delta(i_k^S)$	$\sigma(i_1^F)$	...	$\sigma(i_l^F)$	$null$	$null$	$null$	$\gamma(i^T)$

Table 4.1 Attacker's Profile Structure

There are 4 main components in this structure:-

1. Selected items (i^S): These are a set of randomly selected items.

2. Filler items (i^F): These are given ratings in accordance with the attack strategy.

3. Unrated items (i^ϕ): These items are not given any ratings.

4. Target items (i^T): These are the set of items which are given maximum (push) or minimum (nuke) ratings in order to bias the recommendations given by the system.

The items along with their corresponding distribution functions are shown in the table. E.g. items labeled as i^S have distribution function $\delta(i^S)$.

Also, the following two terms are used in literature to indicate the strength of the shilling attack:-

1. **Filler size** is the ratio between the number of items that are rated in a profile and the total items present in the system.
2. **Attack size** refers to the ratio between the number of shilling users and the number of all the users.

4.2 TYPES OF SHILLING ATTACK MODELS

The most commonly used attack models are the following:-

1. **Random Attack**: This type of attack is a low knowledge attack where the attacker selects the filler items randomly and ratings are given by distribution of all the items e.g. normal distribution with standard deviation and mean of the system. Selected items are given null or no ratings while target items are given maximum or minimum ratings depending on the type of attack (push or nuke).
2. **Average Attack**: This type of attack is a high knowledge attack where like random attack, the attacker selects the filler items randomly but ratings are given according to distribution of the individual items.
3. **Bandwagon Attack**: This type of attack is a low knowledge attack where the filler items are filled just like random attack. But the selected items and target items both are given maximum rating. The selected items used are actually popular items that are rated by most of the users. This is done to increase the similarity with the authentic users and to increase the strength of the attack.
4. **Segment Attack**: This type of attack is also a low knowledge attack. Segment attack is done in order to popularize the target items among a specific group of users or a segment. Here, maximum ratings are given to the selected items and minimum ratings are given to the filler items to increase the similarity measure between the shillers and the segment for which the attack is being performed.

Hence, these attacks are injected into the system in order to alter the recommendations that are produced by the system for its users. The attackers become neighbors to the normal users, by virtue of the similarity metric values and thereby promote or demote the target item.

Table 4.2 summarizes the different types of attack models. rmin refers to the minimum rating in the system and rmax refers to the maximum rating in the system. For example, if the rating scale is from 1 to 5 then the value of rmin=1 and the value of rmax=5.

Attack Models	$\mathbf{I^F}$	$\mathbf{I^S}$	$\mathbf{I^T}$ **(nuke/push)**	$\mathbf{I^N}$
Random Attack	overall mean	ϕ	rmin/ rmax	ϕ
Average Attack	item mean	ϕ	rmin/ rmax	ϕ
Bandwagon Attack	overall mean	rmin/ rmax	rmin/ rmax	ϕ
Segment Attack	rmax/rmin	rmin/ rmax	rmin/ rmax	ϕ

Table 4.2 Features of different types of Attack Models

4.3 PRINCIPAL COMPONENT ANALYSIS

Principal Component Analysis (PCA) is a dimensionality reduction technique in which a high dimensionality space is projected onto a low dimensionality space with minimum loss of information. In simpler terms, it reduces a bigger set of variables into a smaller set with as much information as the original set. It is also known as the Karhunen-Loeve transform, or the Hotelling transform. By dimensionality reduction, the same analytical results are produced with an advantage of handling fewer amounts of data.

A Principal Components is a linear combination of the variables that define the system and number of principal components is equal to the number of original variables.

Some Applications of PCA as used in data mining are as follows:-

1. Data Compression
2. Visualization of the data
3. Feature Selection

Figure 4.1 shows the application of PCA on a dataset and the first two principle components as obtained by it.

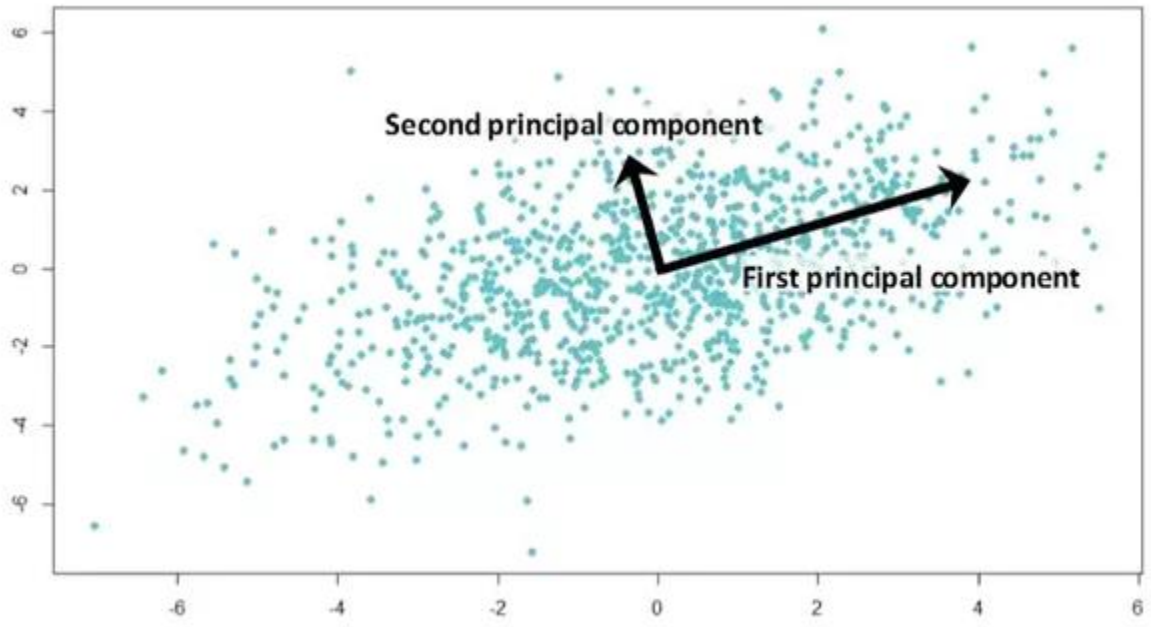


Figure 4.1 Principal Components obtained by PCA on a sample data

Principal components are equivalent to the Eigen vectors of the covariance matrix of the original data.

4.3.1 EIGEN VECTORS AND EIGEN VALUES

An eigenvector is a vector whose direction does not change even after some linear transformation is applied to it. Three vectors are shown in the Fig. 4.2. Red colored vectors do not change direction while yellow colored vector changes direction when a transformation is applied to it.

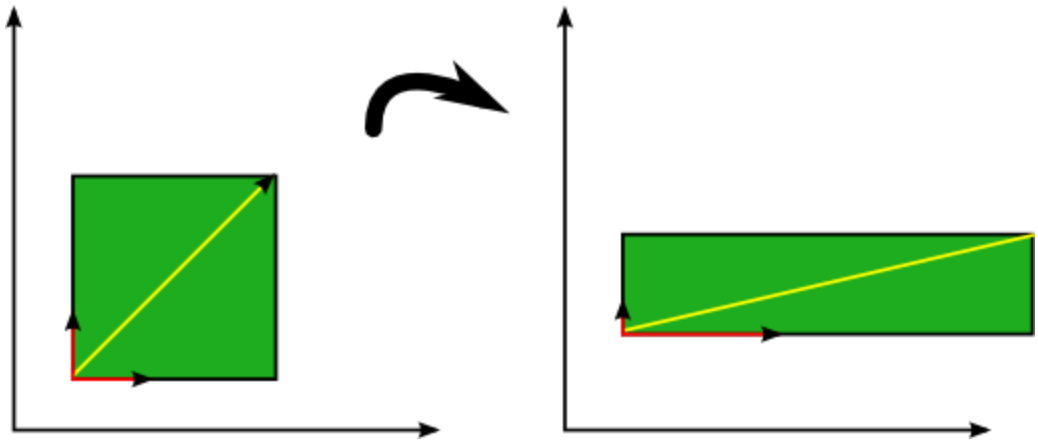


Figure 4.2 Eigen vectors (shown in red)

The above transformation (scaling) is defined by the matrix A as:-

$$A = \begin{bmatrix} 2 & 0 \\ 0 & 0.5 \end{bmatrix}. \tag{4.1}$$

Definition:-

In general, the **eigenvector** $\vec{v}$ of an n X n matrix A is the vector such that:

$$A\vec{v} = \lambda\vec{v} \tag{4.2}$$

where λ is a scalar value called the '**eigenvalue**'. This implies that the linear transformation A on vector $\vec{v}$ is defined completely by λ.

Rewriting the above equation (4.2) as:

$$\begin{aligned} & A\vec{v} - \lambda\vec{v} = 0 \\ \Rightarrow\ & \vec{v}(A - \lambda I) = 0, \end{aligned} \tag{4.3}$$

where I is the identity matrix of order n.

The above equation can be defined only if (A-λI) is non-invertible i.e. its determinant is equal to zero. Hence in order to find the eigen vectors of matrix A, we need to solve the below equation:

$$Det(A - \lambda I) = 0. \tag{4.4}$$

Example:

$$A = \begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix}.$$

Solving equation Det(A-λI) = 0, we get;

$$Det\begin{pmatrix} 2-\lambda & 3 \\ 2 & 1-\lambda \end{pmatrix} = 0.$$

Expanding the determinant :-

$$(2-\lambda)(1-\lambda) - 6 = 0$$
$$\Rightarrow 2 - 2\lambda - \lambda - \lambda^2 - 6 = 0$$
$$\Rightarrow \lambda^2 - 3\lambda - 4 = 0.$$

The discriminant of the above equation is:-

$$D = b^2 - 4ac = (-3)^2 - 4 * 1 * (-4) = 9 + 16 = 25.$$

Since D>0, The two unique values of λ are:-

$$\lambda_1 = \frac{-b - \sqrt{D}}{2a} = \frac{3-5}{2} = -1,$$
$$\lambda_2 = \frac{-b + \sqrt{D}}{2a} = \frac{3+5}{2} = 4.$$

We have now calculated the two eigenvalues λ_1 and λ_2. Corresponding to each eigenvalue, we have an eigenvector.

Substituting the eigenvalue λ_1, we can find the corresponding eigenvector (first):

$$\begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} x_{11} \\ x_{12} \end{bmatrix} = -1 \begin{bmatrix} x_{11} \\ x_{12} \end{bmatrix}.$$

And solving we get,

$$\vec{v_1} = \begin{bmatrix} -1 \\ 1 \end{bmatrix}.$$

Similarly, the second eigenvector is computed as:

$$\vec{v_2} = \begin{bmatrix} 3 \\ 2 \end{bmatrix}.$$

Hence, in this way, we find the eigenvalues and eigenvectors of a square matrix.

4.3.2 STEPS FOR PERFORMING PCA

We represent the data by a m*n matrix X where each column represents the observations denoted by xi = {xi,1, xi,2, ..., xi,m}.

1. The covariance matrix is defined by the following equation.

$$\mathbf{C} = \frac{1}{n-1}\mathbf{X}.\mathbf{X}^T \tag{4.5}$$

2. Now, Applying the Spectral decomposition theorem, C can be written as

$$\mathbf{C} = \mathbf{U}\lambda\mathbf{U}^T \tag{4.6}$$

   Where λ is the diagonal matrix with diagonal entries as eigen values of C.

3. Finally, The Principal Components (PCs) are given by the rows of the matrix S and S is defined as:-

$$\mathbf{S} = \mathbf{U}^T\mathbf{X} \tag{4.7}$$

To get PC's in ascending order, we can rearrange the rows of matrix U in the order of eigen values of the covariance matrix C. Depending upon the application, we require to arrange the PC's in ascending or descending order.

4.4 PCA FOR SHILLING DETECTION

Some properties of shilling attack construction are presented below that helps to detect such users in the system [21].

1. They have low deviation from the mean rating of the system but high deviation from the mean rating the item that is attacked by them. The statistical measures such as Rating Deviation from Mean Agreement (RDMA) use this property for attack detection.

2. They have high similarity with large number of users to bias their recommendations.

3. Shilling users work together in the sense that they intensify each other's effect resulting in low recommendation accuracy and promoting the attacked item.

4. Shilling users are highly correlated due to same underlying model that is used for their construction. In other words, their covariance is low.

4.4.1 VARIABLE SELECTION USING PCA

Shilling users are highly correlated and therefore we tend to employ clustering techniques like k-means clustering to form their separate cluster and filter them out but the fact that they are constructed in a way such that they are quite similar to other users as well, makes clustering approach difficult for their detection.

Principal Component Analysis can be used for shilling users' detection since the shilling users have low covariance amongst themselves as well as with the other users in the system. Also, other users (normal) have high covariance between themselves. Therefore shilling users are identified as the users having least covariance with other users. This is called Variable Selection using PCA since we are selecting some variables (users) from the complete set of variables.

Below, we present the Algorithm for shilling users' detection using PCA:-

Algorithm: PCASelectUsers

Input:
Dataset, D
Cut off Parameter, r

Output: Return r users with smallest *Distance* values.

1: $\mathbf{D} \leftarrow$ z-scores ($\mathbf{D}$)
2: $COV \leftarrow \mathbf{D}\mathbf{D}^T$ {Covariance of $\mathbf{D}^T$}
3: $\mathbf{U}\lambda\mathbf{U}^T$ = Eigenvalue-Decomposition (COV)
4: $PCA1 \leftarrow \mathbf{U}(:, 1)$ {1*st* Eigenvector of COV
5: $PCA2 \leftarrow \mathbf{U}(:, 2)$ {2*nd* Eigenvector of COV
6: for all columnid *user* in $\mathbf{D}$ do
7: $Distance(user) \leftarrow PCA1(user)^2 + PCA2(user)^2$
8: end for
9: Sort *Distance*

Here we first compute the z scores of the original dataset D. Z-score value of a rating given by user u for an item y is defined as:-

$$z_{u,y} = \frac{v_{u,y} - \hat{v}_u}{\sigma_u} \quad (4.8)$$

Where, $\hat{v}_u$ is the average rating and σ_u is the standard deviation of the user under consideration. After performing the Eigen decomposition of the covariance matrix, we can obtain the top two

principal components which are essentially the eigenvectors that are obtained with respect to the largest eigenvalues. Finally we select the shilling users as the users having smallest coefficients in these principal components.

Principal components are orthogonal to each other since the eigenvectors of a symmetric matrix are always orthogonal.

4.4.2 PCA WITH PERTURBATION

In this section, a modified algorithm based on PCA for shilling attack detection is presented [9].

Algorithm: PCAPerturbation

Input:
R, users' rating matrix (n users, m items);
k, cutoff parameter of PCA;
α, strength of perturbation

Output: A list of shilling users;

```
1. Att1 ← k profiles selected by PCA on matrix R
2. for i: 1 → n (each user in R) do
3.       for j: 1 → m (each item in R) do
4.             Randomly select a number t from normal distribution
5.             Rnew(i, j) = R(i, j) + α × t;
6.             if Rnew(i, j) > ratingupperBound then
7.                   Rnew(i, j) = ratingupperBound;
8.             else if Rnew(i, j) < ratinglowerBound then
9.                   Rnew(i, j) = ratinglowerBound;
10.            end if
11.      end for
12. end for
13. Att2 ← k profiles selected by PCA on matrix Rnew
14. AttackList = Att1 U Att2;
15. Return AttackList;
```

First, the shilling attackers are identified with the conventional PCA Algorithm. After this, perturbation in the form of Gaussian noise is introduced in the system. Using this noise, some of the shilling profiles that are considered to be authentic will be identified correctly. The parameter α is used to control the strength of the noise and its value is obtained as 0.8 experimentally. (Optimal value).

Finally, the shilling attackers are obtained by taking the union of the lists obtained by PCA and PCA with perturbation.

CHAPTER 5: ALLEVIATING SPARSITY AND COLD START PROBLEMS

5.1 ALLEVIATING SPARSITY

The utility or the rating database is quite large due to a large number of users and items. Users tend to rate only a few items leading to a large number of blank entries in the rating matrix and hence low prediction accuracy of CF recommender systems.

In order to remove sparsity from the dataset to increase recommendation accuracy, we employ weighted slope one technique. In the following section, the technique is explained in detail.

5.1.1 WEIGHTED SLOPE ONE TECHNIQUE

'Slope One' technique was given by Daniel Lemire and Anna Maclachlan in 2005[17]. They are simple to implement and at the same time, have a good accuracy comparable to many compute-intensive techniques. This is a type of item based collaborative filtering.

This class of algorithms is based on the idea of "popularity differential" between items for users. In this algorithm, we compute how much more a particular item is liked as compared to the other item for each pair of items. It uses a linear model for predicting the rating of an item i.e. $f(x) = x + b$ where b is a constant and x represents a variable equal to the average difference between the ratings of the items.. This is why the name of the technique is slope one.

Example:-

Consider the following rating matrix consisting of 4 users and 5 items:-

	item1	item2	item3	item4	item5
user1	2	3		4	4
user2	2	3	3		4
user3	1		4	5	3
user4	2	4			?

Figure 5.1 Sample rating matrix for weighted slope one scheme

Suppose we need to predict the rating of user4 for item5.we will find the deviation of item5 with respect to all the items that user4 rated.

1. Average Deviation between item5 and item1 is ((4-2) + (4-2) + (3-1))/3 = 2
2. Average Deviation between item5 and item2 is ((4-3) + (4-3))/2 = 1

Therefore the predicted rating for the cell marked as '?' = ((2+2) + (4+1))/2 = 4.5

Formally, the algorithm can be stated as follows:-

1. For a particular training set, the average deviation between the target item j and any item i can be computed by the following equation:-

$$\mathrm{dev}_{j,i} = \sum_{u \in S_{j,i(\chi)}} \frac{u_j - u_i}{\mathrm{card}\left(S_{j,i}(\chi)\right)} \qquad (5.1)$$

Here, uj and ui are the ratings given by the user u to item j and item i respectively.

Sj,i(χ) is the set of users who have rated both the item i and j

card(A) represents the number of elements in a set A.

2. Now, $dev_{j,i} + u_i$ is the predicted value of u_j if u_i is given; so taking average for all values of i, we get the below formula for the predicted value of item j for the target user u.

$$P(u)_j = \frac{1}{card(R_j)} \sum_{i \in R_j} dev_{j,i} + u_i \qquad (5.2)$$

   Rj is the set of all the items that are rated by the target user except the target item, whose deviation from the target item can be determined.

   This is the result as suggested by the slope one scheme.

3. Consider a user X whose rating for the item A is to be predicted. Now, we consider two other items B and C that are rated by user X to make an estimate for the item A. If 3000 users rated both the items A and B, whereas 30 users rated both the items A and C; then rating of the item B is more appropriate as a predictor for the item A's rating. Hence the weighted slope one algorithm introduced the following formula for making prediction of the item j:-

$$P^{wSl}(u)_j = \frac{\sum_{i \in S(u) - \{j\}} (dev_{j,i} + u_i) c_{j,i}}{\sum_{i \in S(u) - \{j\}} c_{j,i}} \qquad (5.3)$$

   Where $cj,i = card\ (\ Sj,i\ (\chi))$.

By using the weighted slope concept, the rating the cell marked '?' now becomes ((2+2)***3** + (4+1)***2**) / 2 = 4.4.

Hence we use this algorithm for sparsity removal from the considered dataset.

5.2 ALLEVIATING COLD START PROBLEM

Cold start problems is one of the problems caused by data sparsity. It is difficult to recommend items to a new user as we have no information about his past preferences. Till the time, the user

has not rated sufficient number of items so that we can find accurate nearest neighbors to him, no quality recommendations can be made. This is called user cold start problem.

Similarly, we have item cold start problem for new items that enter into the system.

To solve the user cold start problem, methods that are based on 'ask-to-rate' technique have been proposed. In this technique, whenever a new user registers into the system, he is asked about his rating preferences for certain items. In this way, the user is subjected to an interview so that his profile can be constructed (which is row in the rating matrix with a judicious number of ratings) and quality recommendations can be made to him [22]. Fig. 5.2 describes this process.

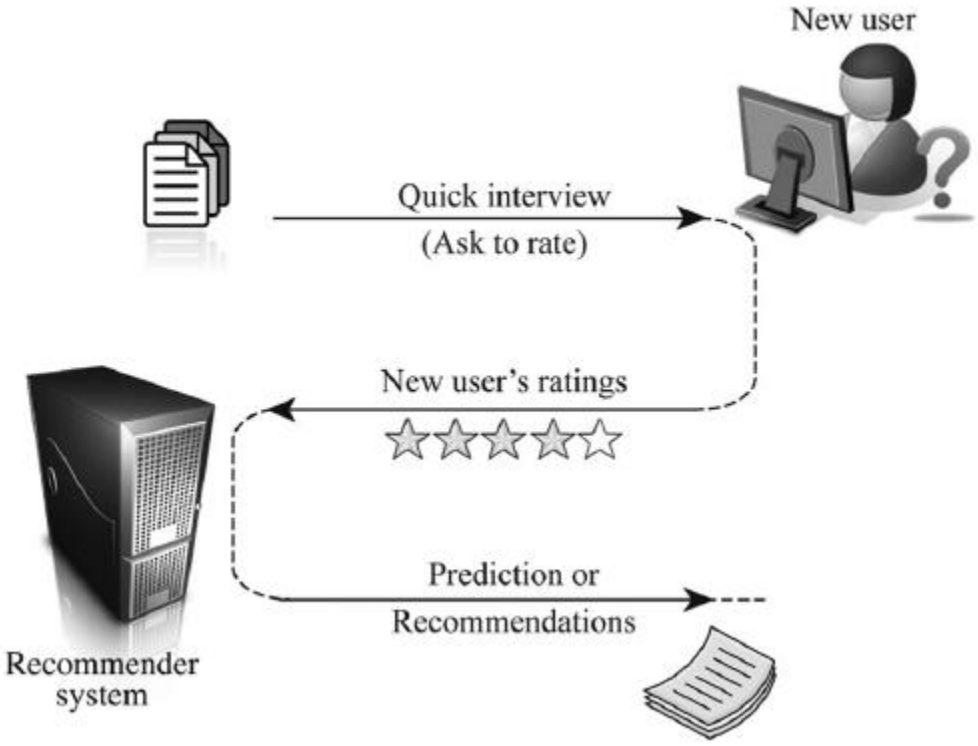


Figure 5.2 Alleviating user cold start problem using ask-to-rate process

- The items that are presented by these methods for the users to rate should be well chosen so that the chances of users to rate those items are maximized.
- In particular, we want to optimize to 2 parameters i.e. Maximize the **recommendation accuracy** and Minimize the **user effort**.

The ask-to-rate techniques can be further divided into Non-adaptive or Adaptive methods. In non-adaptive methods, the same set of items is presented to the user for interview always irrespective of the change in the knowledge of users with subsequent ratings given by him.

Whereas, in adaptive strategies, the items presented to the new user are changed in accordance with the subsequent ratings provided by him. In other words, the system adapts to the preferences of the user. Here, we only consider the non-adaptive strategies for user cold start problem resolution.

Some of the most important strategies or methods to select the items to be used for interviews of the new users are as follows:-

1. **Random Strategy:**
   This strategy obtains the items to be presented in a random fashion. It does not utilize the patterns of the rating matrix for item selection and hence is not effective practically where a case may arrive that user might not have any idea about the item that he is required to rate. It is a baseline strategy and is more often used for making comparisons with other strategies.

2. **Popularity Strategy:**
   This strategy uses the frequency of ratings of an item or its popularity. The number of users who have rated an item are counted for each item and based on this the most popular items are derived. These items are then presented to the users for providing their ratings. The popularity of an item a_t is calculated as:-

$$\text{Popularity } (a_t) = |a_t| \qquad (5.4)$$

   In this equation, $|a_t|$ refers to the number of users who have rated the item a_t.
   The strategy is simple to implement but provides less information about the new users as most of the users' rate the popular items; hence we are not able to comprehend the specific tastes of the cold start users.

3. **Pure Entropy:**

Entropy is the measure of randomness in the data. We want the users to rate the items that gives maximum information about their interests. By presenting the items having mixed ratings in the data i.e. the case when some users like a particular item while the others do not; we increase the chances to understand the preferences of the users better. The items are arranged in descending order of their entropy value and presented to the user.

Formally the entropy is defined as follows:-

$$\text{Entropy}(a_t) = \sum_{i=1}^{5} -p_i * log p_i \qquad (5.5)$$

Where p_i is the proportion of ratings for a particular rating value on the rating scale. For example, in MovieLens dataset rating values vary from 1 to 5.

Now, we present a pseudocode for entropy calculation of an item :-

```
Function Entropy (a_t)
entropy (a_t) = 0
for each item a_t in dataset
        for i as each of the possible rating values // in movielens i = 1 . . . 5
                if a_t's rating= i
                        value[i] += 1 // rating frequencies
        end for
        proportioni =value[i]   //total number of users who rate at
        entropy (a_t) += proportion_i * Math.log (proportion_i, 2)
end for each
entropy (a_t) = −entropy (a_t)
End
```

This method is also not much effective because it does not incorporate frequency of the items into account.

4. **Balanced Strategy:**

This strategy combines the advantages of both the above techniques of popularity and entropy. Through popularity, chances of getting ratings from the users are very high and through entropy, value from each rating is high. Also, logarithm of popularity or frequency is taken to reduce the dominance of popularity over entropy. Hence the metric used for selecting the item becomes *(log popularity) *entropy.*

5. **HELF (Harmonic mean of Entropy and Logarithm of Frequency)**
   This strategy is a modification to the balanced strategy where rather than taking the multiplication, harmonic mean of the two important factors i.e. Entropy and Frequency is taken.

   Also appropriate normalization of the two factors is done so that no factor can dominate the other one.

   The HELF of an item a_i is defined as:-

$$HELF_{a_i} = \frac{2 * LF'_{a_i} * H'(a_i)}{LF'_{a_i} + H'(a_i)} \qquad (5.6)$$

   where, LF'_{ai} is logarithm of the frequency or popularity of a_i and is normalized by a factor as well(lg(|U|)): **$lg(|a_i|)/lg(|U|)$,**
   Similarly, $H'(a_i)$ is the entropy of a_i and normalized by a factor of lg(5): **$H(a_i)/lg(5)$.**

Hence, these are the various strategies used for presenting items to cold start users. Apart from these there are other heuristics as well that are proposed in literature.

CHAPTER 6: THE PROPOSED FRAMEWORK

6.1 ARCHITECTURE OF THE PROPOSED SYSTEM

Fig. 6.1 shows the architecture of the proposed recommender system that simultaneously alleviates the three most common attacks/challenges to collaborative filtering i.e. shilling attack, sparsity and the cold start user problem.

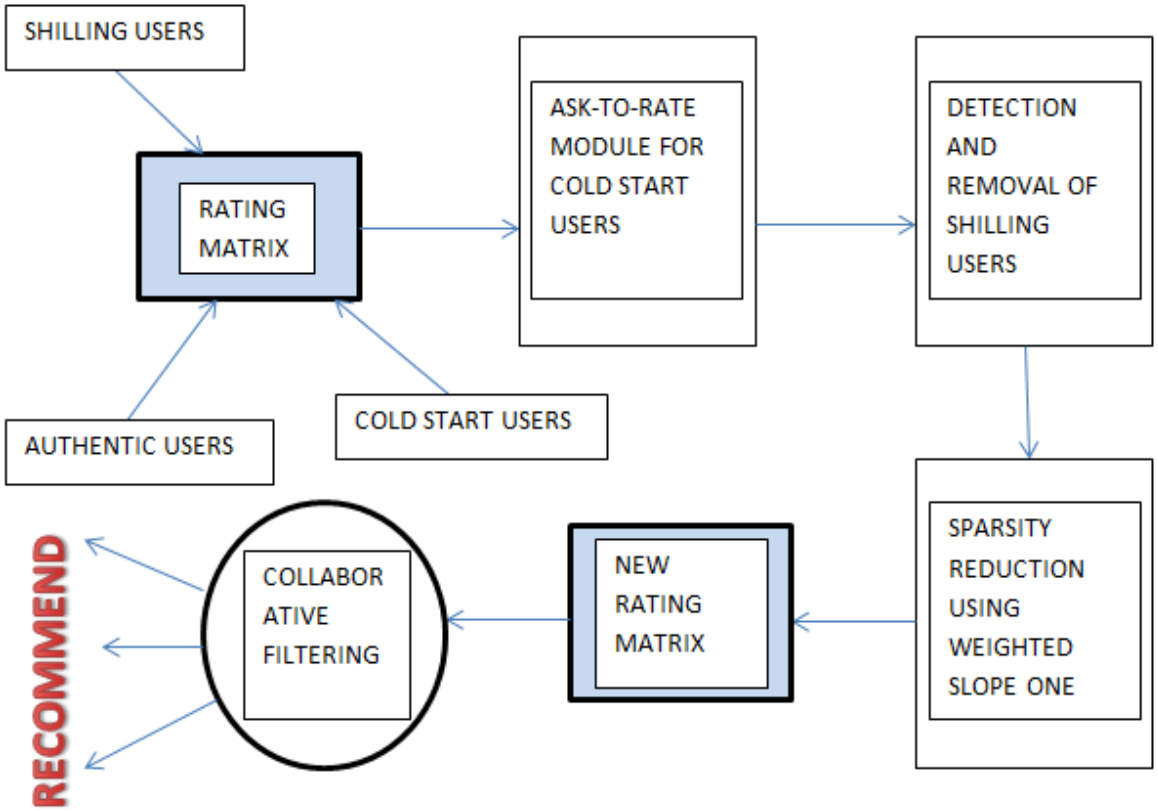


Figure 6.1 Architecture of the Proposed System

As the figure shows, the rating matrix consists of three types of users' i.e. shilling users, authentic users and cold start users. We pass the matrix through different phases. In the first phase an efficient ask-to-rate system is used to handle the cold start users as it gathers sufficient information from them so that recommendations can be made to them.

After this, the shilling users are detected and removed using PCA algorithm. The reduced dataset now passes through the sparsity reduction module where weighted slope technique is used to

make a dense rating matrix and finally the collaborative filtering algorithm is applied to make quality recommendations to users.

6.1.1 GENRE BASED HELF

We propose a new technique for presenting items to the cold start users i.e. Genre based HELF (Harmonic mean of Entropy and Logarithm of Frequency).

In this approach, we utilize the fact that users can have specific interests for a particular genre (a particular category in case of items). We ask a cold start user whether he is interested in a particular genre or not and if yes, the user is asked to select that genre.

After this all the items of the concerned genre are extracted from the dataset and HELF value is obtained for only those items using equation (5.6). These items are sorted based on the HELF values and are presented to the user. The detailed algorithm is given in the next section.

6.2 PROPOSED ALGORITHM

The pseudo code of the proposed system is described as follows:-

INPUT: Rating or Utility Matrix R with cold start users, shilling users or attackers and authentic users.

OUTPUT: Top-K items for each user as recommendations

STEPS:-

1. Identify the cold start users from the rating matrix.
2. for each cold start user,
3. Obtain the specific genre interest of the user from the existing set of genres.
4. Extract all the movies of genre specified by the user.
5. Calculate the normalized logarithm of frequency for each extracted item.
6. Calculate normalized entropy of each extracted item.
7. Calculate the HELF value by equation (5.6) and sort items according to this value in descending order.
8. Present the items to the user for his/her rating and update the rating matrix. (Ask-to-Rate Mechanism)
9. end for

10. Apply the algorithm PCASelectUsers or PCAPerturbation to detect shilling attackers in the dataset and remove x users where y is the predetermined threshold, to obtain the updated rating matrix.
11. Calculate the entries in the deviation matrix using equation (5.1) for every pair of items and also store the count of common ratings for each such pair.
12. Fill the blank ratings in the rating matrix using equation (5.2) to make a dense rating matrix. This step reduces the sparsity problem of collaborative filtering.
13. For each target user, compute its similarity value with other users in the system using pearson correlation coefficient.
14. Sort the neighbors according to the similarity value in descending order.
15. Take top y users who have rated the target item where y is predetermined number.
16. Finally, apply the weighted similarity formula to get the estimated or predicted value for target item.
17. Repeat above procedure for different items and sort the items according to predicted ratings.
18. Present top k items to each target user as a set of recommended items.

Steps 1 to 9 resolve the cold start user problem by utilizing an effective ask-to-rate strategy. Step 10 detects and removes shilling attackers. Step 11 and 12 removes the sparsity problem. Steps 13 to 18 apply collaborative filtering to produce recommendations to each target user.

CHAPTER 7: IMPLEMENTATION AND RESULTS

7.1 EVALUATION METRICS

In order to evaluate the recommender models, we have the following Evaluation Metrics.

- Mean Absolute Error (MAE)

$$MAE = \frac{\sum_{(u,i)\in T} |r_{u,i} - \hat{r}_{u,i}|}{|T|} \tag{7.1}$$

Here we are summing over the difference between actual and predicted values and T consists of all the user-item ordered pairs that are present in the test set. |T | is the number of all such pairs.

- Root Mean Square Error (RMSE)

$$RMSE = \sqrt{\frac{\sum_{(u,i)\in T} (r_{u,i} - \hat{r}_{u,i})^2}{|T|}} \tag{7.2}$$

- Precision (P)

$$P = \frac{TP}{TP+FP} \tag{7.3}$$

It is defined as the Number of True positives out of the total number of positives or recommendations.

- Recall (R)

$$R = \frac{TP}{TP+FN} \tag{7.4}$$

It is defined as the Number of True positives out of the total number of relevant recommendations.

- F- Measure or F1 Score (F)

$$F = \frac{2*P*R}{P+R} \quad (7.5)$$

It is defined as the Harmonic Mean of Precision and Recall to accommodate both the factors into one metric or measure.

Along with these metrics, sometimes we also use Prediction Time as CF based techniques require Similarity Computation among a large number of users and/or items.

7.2 IMPLEMENTATION DETAILS

The Implementation platforms for the project are two-fold:-

1. RStudio 3.4.1 has been used for the purpose of dataset analysis and MAE calculation. R has an advantage of extensive support of in-built tools and packages for implementation purpose and helps in manipulating the data efficiently. Hence it is an ideal platform for recommendation analysis.
2. Dev C++ 5.11 compiler is used for building a menu-driven program where a user can enter his data during the initial phase. Basically, it is used for simulating the interview process for the cold start users.

7.3 DATASET USED

The dataset used for carrying out the experiments is MovieLens-100k. It contains 1, 00,000 ratings across a rating matrix of 943 users and 1682 movies. Each user has rated at least 20 ratings. Further the ratings range from 1 to 5.

For sparsity reduction, this dataset requires a lot of time, hence we have reduced the dataset by selecting the users who have rated at least 50 items and items that have been rated by at least 100 users. By applying this constraint in R, we got a dataset with 563 users and 334 items with 55653 ratings.

In Fig. 7.1, we present first 5 rows and first 5 columns of this dataset:

The NA values represent blank ratings or no ratings.

	[,1]	[,2]	[,3]	[,4]	[,5]
[1,]	5	3	3	4	1
[2,]	4	NA	NA	NA	NA
[3,]	NA	NA	NA	NA	NA
[4,]	4	3	NA	NA	NA
[5,]	4	NA	NA	2	4

Figure 7.1 First 5 rows and columns of the dataset

Finally, for presenting items to the users during the interview process of cold start users, we use the original dataset only.

7.4 RESULTS

We calculate the Mean Absolute Error (MAE) of the system in both the cases i.e. when shilling users are not present and when shilling users are present in the system and removed using PCA algorithm. The train sets and test sets are taken in standard 80:20 ratio.

1. **Without Attack**

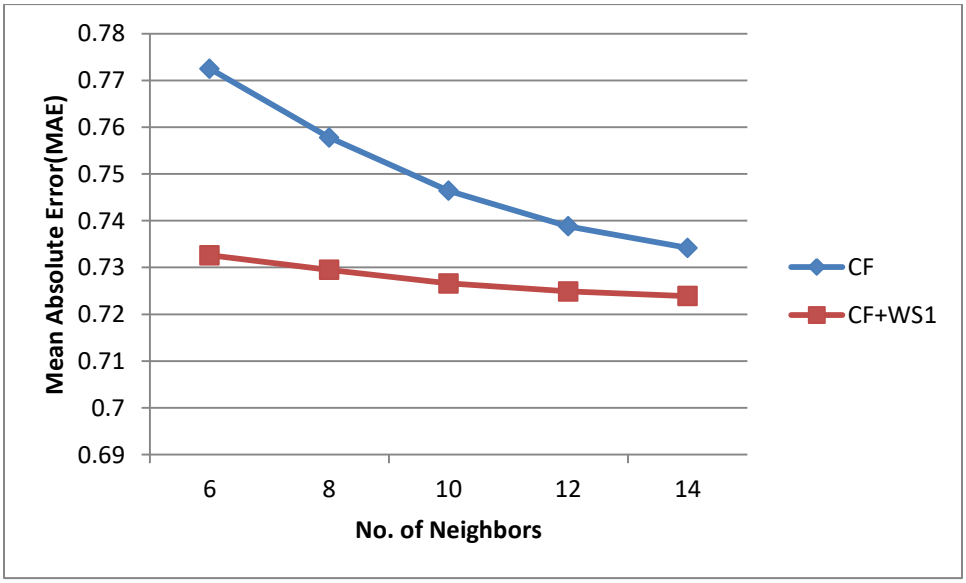


Figure 7.2 Comparative results without shilling attack

Here CF means User based collaborative filtering technique and CF+WS1 means sparsity reduction is done using weighted slope one algorithm before applying collaborative filtering.

2. **With Attack**

Attack size: 10%, filler size: 50%; i.e. 56 shilling users were injected into the system using average attack model. 31 users were correctly recognized by the PCA algorithm. After removal of all the users detected by PCA as shilling users; MAE calculation was done in a similar manner as the previous case.

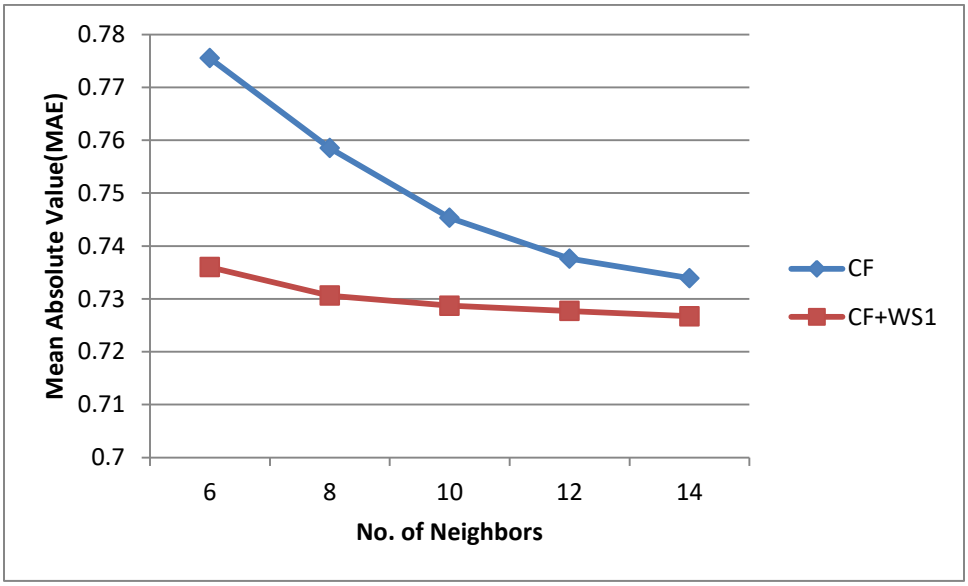


Figure 7.3 Comparative results with shilling attack

Similar Notations as described before are used. The specified range of neighbors is selected because out of this range; sparsity reduction will lead to overfitting and decrease in accuracy.

The items that are presented to the cold start users under different strategies for MovieLens 100k dataset are listed below:

1. Popularity Strategy

Number	Name of the Movie(Year)	Frequency
50	Star Wars (1977)	583
258	Contact (1997)	509
100	Fargo (1996)	508
181	Return of the Jedi (1983)	507
294	Liar Liar (1997)	485

Table 7.1 List of movies using popularity strategy

2. Entropy Strategy

Number	Name of the Movie(Year)	Entropy
990	Cats Don't Dance (1997)	2.3043
640	Cook the Thief His Wife & Her Lover, The (1989)	2.2970
219	Nightmare on Elm Street, A (1984)	2.2908
1038	Grease 2 (1982)	2.2783
324	Lost Highway (1997)	2.2746

Table 7.2 List of movies using entropy strategy

3. HELF

Number	Name of the Movie(Year)	HELF
294	Liar Liar (1997)	0.9066
288	Scream (1996)	0.9062
286	English Patient, The (1996)	0.9057
121	Independence Day (ID4) (1996)	0.8996
748	Saint, The (1997)	0.886

Table 7.3 List of movies using HELF strategy

4. Genre-based HELF (Proposed Technique)

Genre = Action

Number	Name of the Movie(Year)	HELF
121	Independence Day (ID4) (1996)	0.8996
748	Saint, The (1997)	0.886
118	Twister (1996)	0.8732
300	Air Force One (1997)	0.8681
323	Dante's Peak (1997)	0.8578

Table 7.4 List of movies using Genre based HELF strategy (Action)

Genre = Adventure

Number	Name of the Movie(Year)	HELF
118	Twister (1996)	0.8732
151	Willy Wonka and the Chocolate Factory (1971)	0.8695
271	Starship Troopers (1997)	0.8559
405	Mission: Impossible (1996)	0.8542
117	Rock, The (1996)	0.8515

Table 7.5 List of movies using Genre based HELF strategy (Adventure)

Hence, in this way, the proposed system alleviates the three types of attacks i.e. shilling attack, sparsity problem and cold start user problem.

CHAPTER 8: CONCLUSION AND FUTURE WORK

8.1 CONCLUSION

With increasing information availability online, there is a vital need for building efficient recommender systems so that users have a restricted number of options that are relevant and in accordance with their preferences. Collaborative filtering that is the most popular technique for RS produces recommendations based on the similarity with the other users/items in the system. In this thesis, we presented a framework and complete description of the various approaches to collaborative filtering. We also presented a comprehensive literature survey of the improvements to collaborative filtering that will help any researcher working in this area to get an idea about past and present improvements to the conventional algorithm.

We also analyzed the various issues/ challenges to collaborative filtering and developed a system that alleviates three types of attacks i.e. shilling attacks, sparsity problem and cold start problem simultaneously by utilizing various techniques like PCA, weighted slope one technique, etc.

The proposed system shows increase in the efficiency and effectiveness of the current recommender systems by producing quality recommendations to the users.

8.2 FUTURE SCOPE

Context information can be added to the recommendations to increase their effectiveness, e.g. In case of music recommender system, a particular music clip can be played depending on different times of the day. Similarly, we can include trust information as well during computation of similarities between the users.

Hence, by incorporating more information about the users depending on the type of application for which the recommendations are to be done, we can improve the quality of recommendations.

REFERENCES

[1] Adibi, P. and Ladani, B.T., 2013, May. A collaborative filtering recommender system based on user's time pattern activity. In *Information and Knowledge Technology (IKT), 2013 5th Conference on* (pp. 252-257). IEEE.

[2] Alhijawi, B. and Kilani, Y., 2016, June. Using genetic algorithms for measuring the similarity values between users in collaborative filtering recommender systems. In *Computer and Information Science (ICIS), 2016 IEEE/ACIS 15th International Conference on* (pp. 1-6). IEEE.

[3] Alqadah, F., Reddy, C.K., Hu, J. and Alqadah, H.F., 2015. Biclustering neighborhood-based collaborative filtering method for top-n recommender systems. *Knowledge and Information Systems*, *44*(2), pp.475-491.

[4] Ayyaz, S. and Qamar, U., 2017, March. Improving collaborative filtering by selecting an effective user neighborhood for recommender systems. In *Industrial Technology (ICIT), 2017 IEEE International Conference on*(pp. 1244-1249). IEEE.

[5] "Cold Start Problem" [online], Available: https://en.wikipedia.org/wiki/Cold_start_(computing)

[6] "Collaborative Filtering" [online], Available: https://en.wikipedia.org/wiki/Collaborative_filtering

[7] Cong Li and Zhigang Luo "Detection of Shilling Attacks in Collaborative Filtering Recommender Systems", IEEE International Conference of Soft Computing and Pattern Recognition (SoCPaR), 2011

[8] Dakhel, G.M. and Mahdavi, M., 2011, December. A new collaborative filtering algorithm using K-means clustering and neighbors' voting. In *Hybrid Intelligent Systems (HIS), 2011 11th International Conference on* (pp. 179-184). IEEE.

[9] Deng, Z.J., Zhang, F. and Wang, S.P., 2016, July. Shilling attack detection in collaborative filtering recommender system by PCA detection and perturbation. In *Wavelet Analysis and Pattern Recognition (ICWAPR), 2016 International Conference on* (pp. 213-218). IEEE.

[10] Du, Y., Du, X. and Huang, L., 2016. Improve the collaborative filtering recommender system performance by trust network construction. *Chinese Journal of Electronics*, *25*(3), pp.418-423.

[11] Faridani, V., Jalali, M. and Jahan, M.V., 2017. Collaborative filtering-based recommender systems by effective trust. *International Journal of Data Science and Analytics*, *3*(4), pp.297-307.

[12] Ghaznavi, F. and Alizadeh, S.H., 2017, April. Assessing usage of negative similarity and distrust information in CF-based recommender system. In *Artificial Intelligence and Robotics (IRANOPEN), 2017* (pp. 132-138). IEEE.

[13] Gong, S. and Cheng, G., 2008, December. Mining user interest change for improving collaborative filtering. In *Intelligent Information Technology Application, 2008. IITA'08. Second International Symposium On* (Vol. 3, pp. 24-27). IEEE.

[14] Gopalachari, M.V. and Sammulal, P., 2014, December. Personalized collaborative filtering recommender system using domain knowledge. In *Computer and Communications Technologies (ICCCT), 2014 International Conference on* (pp. 1-6). IEEE.

[15] Kaleroun, A. and Batra, S., 2014, August. Collaborating trust and item-prediction with ant colony for recommendation. In *Contemporary Computing (IC3), 2014 Seventh International Conference on* (pp. 334-339). IEEE.

[16] Katarya, R. and Verma, O.P., 2016. A collaborative recommender system enhanced with particle swarm optimization technique. *Multimedia Tools and Applications*, *75*(15), pp.9225-9239.

[17] Kumar, A. and Sharma, A., 2013. Alleviating sparsity and scalability issues in collaborative filtering based recommender systems. In *Proceedings of the International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA)* (pp. 103-112). Springer, Berlin, Heidelberg.

[18] Leskovec, J., Rajaraman, A. and Ullman, J.D., 2014. *Mining of massive datasets*. Cambridge university press.

[19] Liang, H., Xu, Y., Li, Y. and Nayak, R., 2008, December. Collaborative filtering recommender systems using tag information. In *Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT'08. IEEE/WIC/ACM International Conference on* (Vol. 3, pp. 59-62). IEEE.

[20] Liu, H., Hu, Z., Mian, A., Tian, H. and Zhu, X., 2014. A new user similarity model to improve the accuracy of collaborative filtering. *Knowledge-Based Systems*, *56*, pp.156-166.

[21] Mehta, B., 2007, July. Unsupervised shilling detection for collaborative filtering. In *AAAI* (pp. 1402-1407).

[22] Nadimi-Shahraki, M.H. and Bahadorpour, M., 2014. Cold-start problem in collaborative recommender systems: efficient methods based on ask-to-rate technique. *Journal of computing and information technology*, *22*(2), pp.105-113.

[23] Nasiri, M. and Minaei, B., 2016. Increasing prediction accuracy in collaborative filtering with initialized factor matrices. *The Journal of Supercomputing*, *72*(6), pp.2157-2169.

[24] Pujahari, A. and Padmanabhan, V., 2015, December. Group Recommender Systems: Combining user-user and item-item Collaborative filtering techniques. In *Information Technology (ICIT), 2015 International Conference on* (pp. 148-152). IEEE.

[25] Sharma, R., Gopalani, D. and Meena, Y., 2017, February. Collaborative filtering-based recommender system: Approaches and research challenges. In *Computational Intelligence & Communication Technology (CICT), 2017 3rd International Conference on* (pp. 1-6). IEEE.

[26] Te Braak, P., Abdullah, N. and Xu, Y., 2009, September. Improving the performance of collaborative filtering recommender systems through user profile clustering. In *Web Intelligence and Intelligent Agent Technologies, 2009. WI-IAT'09. IEEE/WIC/ACM International Joint Conferences on*(Vol. 3, pp. 147-150). IEEE.

[27] Tewari, A.S. and Barman, A.G., 2016, December. Collaborative book recommendation system using trust based social network and association rule mining. In *Contemporary Computing and Informatics (IC3I), 2016 2nd International Conference on* (pp. 85-88). IEEE.

[28] Wang, P. and Ye, H., 2009, April. A personalized recommendation algorithm combining slope one scheme and user based collaborative filtering. In *Industrial and Information Systems, 2009. IIS'09. International Conference on* (pp. 152-154). IEEE.

[29] Xie, F., Chen, Z., Shang, J., Huang, W. and Li, J., 2015, March. Item similarity learning methods for collaborative filtering recommender systems. In *Advanced Information Networking and Applications (AINA), 2015 IEEE 29th International Conference on* (pp. 896-903). IEEE.

[30] Yang, Z., Wu, B., Zheng, K., Wang, X. and Lei, L., 2016. A survey of collaborative filtering-based recommender systems for mobile Internet applications. *IEEE Access*, *4*, pp.3273-3287.

[31] Zou, J. and Fekri, F., 2013, October. A belief propagation approach for detecting shilling attacks in collaborative filtering. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management* (pp. 1837-1840). ACM.

APPENDIX A: CODE SNIPPETS

GENRE BASED HELF STRATEGY FOR COLD START USERS

```
## Cold start users
## Harmonic mean of Entropy and Logarithm of Frequency
if(!"recommenderlab" %in% rownames(installed.packages())){
  install.packages("recommenderlab")}
library("recommenderlab")
#install.packages("ggplot2")
library("ggplot2")
library("Matrix")

set.seed(1)

### From Grouplense Website
data <- read.csv("ratings.csv")

R <- matrix(nrow = 943, ncol = 1682 )
R[1:2,1:2]

i <- 0
for ( i in 1:100000){
  R[data$USER[i],data$ITEM[i]] = data$RATING[i]
}
dim(R)
# 943 users and 1682 movies

###genre matrix
genre <- read.csv("item_genre.csv")
```

```
genre[1:5,1:5]
dim(genre)

g <- 3 ## concerned genre
n <- 0 ## number of items
items <- 1:1682

for (i in 1:1682){  ### extract items of that particular genre
  if (genre[i,g] == 1){
    n <- n + 1
    items[n] <- i
  }
}
items[1:n]
n

####  number of ratings of each item/ popularity/ frequency
count_item <- 1:1682
for (i in 1:1682){
  count_item[i] <- 0
}
count_item[1:5]

for (j in 1:1682){
  for (i in 1:943){
    if (is.na(R[i,j]) == TRUE){
      next;
    }
```

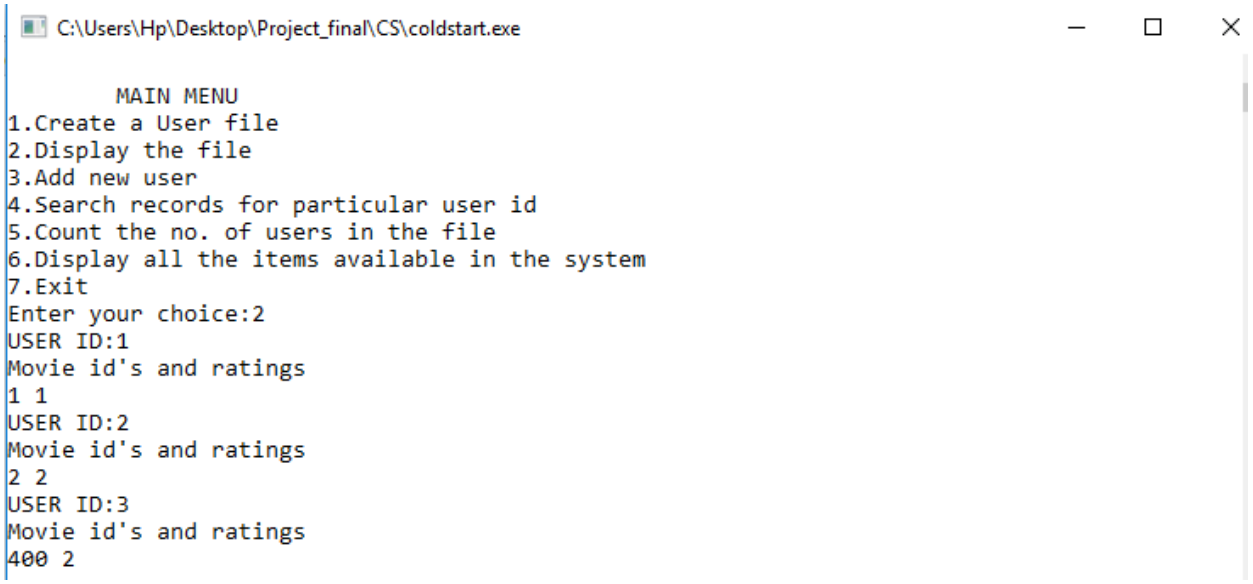
```
    count_item[j] <- count_item[j] + 1
  }
}

count_item[4]

### normalized logarithm of frequency
lf <- 1:n
for (i in 1:n ){
  lf[i] <- log2(count_item[items[i]]) / log2(943)
}
lf

#### calculation of entropy
entropy <- 1:n
p <- 1:5  #proportion
val <- 1:5 #no of one type of ratings

for (i in 1:n){

  entropy[i] = 0
  for (j in 1:5){
    val[j] = 0
  }
  #calculate no of one type of ratings
  for (k in 1:943){
    if (is.na(R[k,items[i]]) == TRUE){
      next;
```

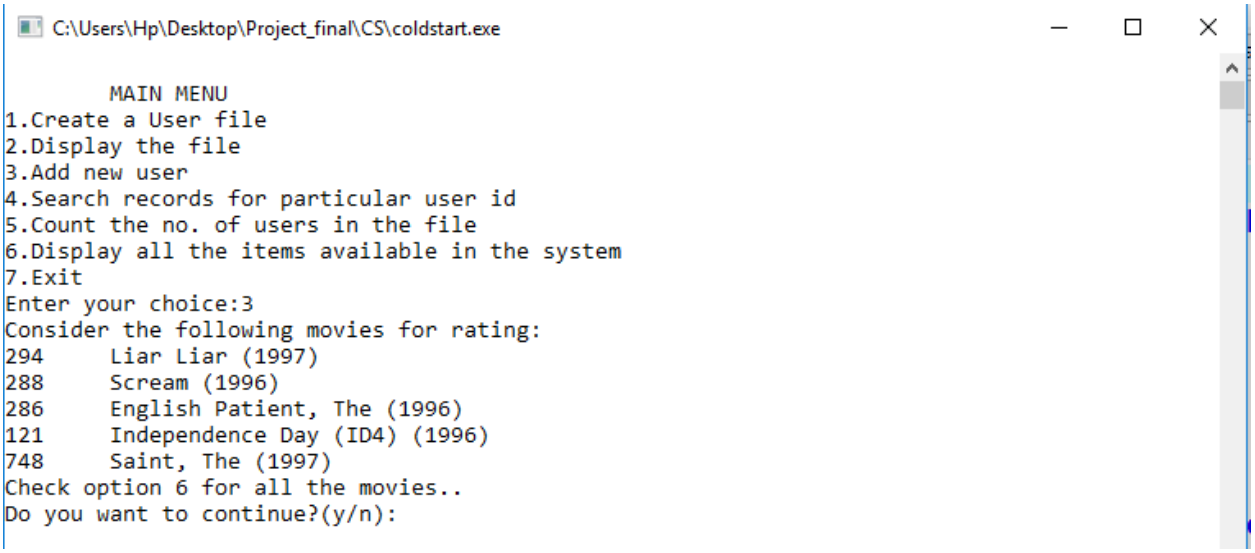
```
    }
    val[R[k,items[i]]] =  val[R[k,items[i]]] + 1
  }
  #calculate proportion and add in entropy
  for (j in 1:5){
    p[j] = val[j] / count_item[items[i]]
    if (p[j] != 0){
      entropy[i] <- entropy[i] + (p[j]*log2(p[j]))
    }
  }
  entropy[i] = -entropy[i]
}
entropy

#normalize
for (i in 1:n){
  entropy[i] = entropy[i]/log2(5)
}
entropy
### CALCULATION OF HELF
helf <- 1:n

for (i in 1:n){
  if (entropy[i] == 0 && lf[i] == 0){
    helf[i] = 0
    next;
  }
  helf[i] <- (2*lf[i]*entropy[i]) / (lf[i] + entropy[i])
```

```
}
help

use <- order(help, decreasing = TRUE)
use
items[use[1:5]]
help[use[1:5]]
```

APPENDIX B: SNAPSHOTS OF THE SYSTEM

MENU DRIVEN PROGRAM FOR NEW USERS





APPENDIX C: PUBLICATION

Manchanda, S. and Sethi, M., 2018, March. A study of collaborative filtering based recommender systems. In *Computing for Sustainable Global Development (INDIACom), 2018 5nd International Conference on* (pp. 198-204). IEEE.