BTP 16im30034

by Yash Agarwal

Submission date: 03-Dec-2019 04:23PM (UTC+0530)
Submission ID: 1223180501
File name: BTP_check.docx (240.83K)
Word count: 3940
Character count: 22720

Abstract

Recommender Systems have become an essential part of various online applications by providing higher customer satisfaction through personalised recommendations. Most of the research in this domain focuses on improving the recommendation methods for higher accuracy values but the inconsistencies or the noise is often ignored. There exist mainly two types of noise in recommender systems: malicious and non-malicious or natural noise. The main reason behind natural noise existence is faulty user behaviour i.e. careless/erroneous preference selection. Malicious noise arises when deliberate attempts are made to tamper the output results in some manner. We believe that both classes of noise are important and can adversely affect recommendations. Therefore, we propose to simultaneously handle both malicious and natural noises.

For the detection of natural noise, we characterize users and items based on their ratings into different classes and identify a rating as noise if it contravenes user or item tendencies. We correct these identified naturally noisy ratings by prediction using collaborative filtering approaches. For the malicious noise, we use a value-based neighbour selection which selects neighbours for active users in user-based collaborative filtering recommender systems under shilling attack. We provide an empirical evaluation of our approach for validation.

Contents

1. Introduction

Today, Recommender Systems have become a significant part of most of the commercial websites such as Amazon, Flipkart, Netflix, IMDB and LinkedIn. These websites are incorporating recommender systems in their mainframe systems to provide personalized user experiences in order to serve them better and increase overall sales and revenue. They recommend items or products that might be interesting for a user. As the amount of data being generated is increasing with every second, the scope for recommender systems is also widening to guide users through this massive amount of data. The problem or rather confusion due to the availability of hundreds of options for products to perform a task is increasing and recommender systems can prove to be very useful in solving that.

The most famous version of Recommender Systems (RS) [4], learns from the user preferences about an existing group of products and predicts the ratings of the non-existing products. With this goal, many applications have been built to recommend different types of items like movies, TV series, products and consumer goods, news, research papers, and many more. These applications stretch to wide domains like e-commerce, e-learning, e-services, tourism, and social media [5,6].

The most used methodologies in the recommender systems are the content-based and collaborative filtering ones. Content-based recommender systems suggest items with similar features to those that the user selected earlier in the past. Whereas, collaborative filtering recommender systems recommend items that other similar users liked in the past. Collaborative filtering systems can generate recommendations just using information about users' preferences regarding a set of items, and due to their simplicity, they have become popular nowadays.

In these recent years, Collaborative filtering has remained a popular topic of research in the community. But, most of the research has been focused on the improvement of the basic recommendation algorithms like neighbourhood-based user-user and item-item approaches, dimensionality reduction approaches, probabilistic, MDP-based, graph-based, rule-based methods, and their hybridization for better accuracy results.

It is often assumed that in the RS, ratings provided by the users which act as the main source of information for the recommendation algorithms are free of irregularities unlike the classical data mining processes where we spend a significant amount of time in data pre-processing to clean the data. Recently Amatriain et al. [7,9] explained in his paper how the users could be inconsistent while giving the ratings or may illicitly provide false ratings, exposing the recommender systems data to inconsistencies. We address these inconsistencies as noise in recommender system databases.

Mahony et al. [8] in their paper titled 'Detecting Noise in Recommender System Databases' have classified these noises into two categories: Natural (Non-Malicious) and Malicious noise. They state that the main reason behind the natural noise is the imperfect user behaviour (e.g. erroneous/careless preference selection) and the different rating collection processes that are being employed. Whereas the malicious noise is injected deliberately by malicious users who

seek improper benefits through recommendations. They either promote their own products to active users or demote competitors' products. These users are known as the shilling attackers and their activities are known as shilling attacks [10].

In the research done so far on the consideration of noise in recommender systems, most of the works fall in either of the above mentioned two categories but no one to the best of our knowledge has tried to consider both malicious and natural noise simultaneously. In this contribution, we have tried to detect and correct both the types of noises and later compare the results with the classical recommender systems without the noise rectification to check the correctness of the hypothesis. We have used Movielens dataset to perform the analysis.

2. Literature Review

With a large number of developments in the field of Recommender Systems and their improvement, millions of consumers are getting motivated to go for online shopping and movie streaming, etc at a lower searching costs. The user review data is also increasing with every second. However, many of the recommender systems face problems of noisy ratings by users. Sometimes these noisy ratings are naturally induced by the misbehaviour of the user who is providing the rating while, sometimes malicious users intentionally hamper with the ratings in order to promote their own product or demote others'.

In this section, we have given an overview of the collaborative filtering approach and recent works related to the handling of noise are discussed. First, we discuss about the natural noise and then about the malicious attacks. Later, we discuss the research gap and how our approach can be effective in dealing with noise in recommender systems.

2.1. Collaborative Filtering Recommender Systems:

Collaborative filtering recommender systems (CFRS) suggest items to the users based on previous preferences of items rated by all the users. CF is a technique to recommend items based on similarity values. There are two types of collaborative filtering methods [12]:

2.1.1. User-based CF (memory-based):

In User-based collaborative filtering approach, items are predicted to the target user that are already of interest for other users similar to the target user. First, the algorithm tries to find the user's neighbour based on user similarities and then combines the neighbour user's rating score by using supervised learning like KNN or Bayesian network or unsupervised learning techniques like clustering.

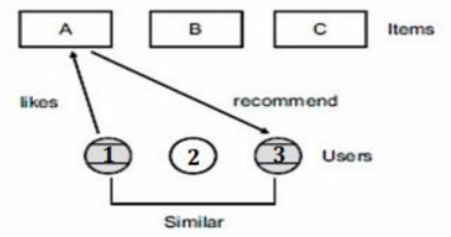

Fig 1. User-based Collaborative Filtering

2.1.2. Item-based CF (model-based):

It is similar to the user-based collaborative filtering if we consider the use of user-rating score but instead of nearest neighbours, it looks for a set of items. The target user has already rated items and this algorithm computes how similar items are to the target item under recommendation. Afterwards, it combines the customer's previous preferences based on these item similarities. In item-based CF, users' preference data can be collected in 2 ways, by explicitly given user ratings or by implicitly analysed user's purchase records or click-through rate.

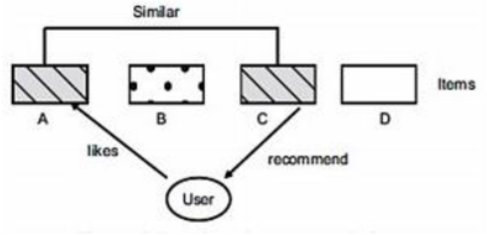

Fig 2. Item-based Collaborative Filtering

In our study, we have used these two classical approaches of CF for evaluation and analysis.

2.2. Natural Noise in Recommender Systems

Natural noise refers to the inconsistencies that are incorporated into the user-item rating matrix unintentionally by the users. Sometimes, users give faulty ratings because of lack of interest or bad mood, etc. This leads to inconsistencies in the rating matrix which affects the performance of the recommender system.

Amatriain et al. [7,9] explains how consideration of natural noise and its correction is important for recommender systems and it leads to the improvement in accuracy of the recommendation algorithm. O' Mahony et al. [8] in his paper uses comparison of user predicted preference for an item with rating of that item. He uses a set of genuine user profiles manually obtained for predicting these ratings and removes the ratings whose difference exceeds some threshold. However, this method always removes the highly deviating ratings which can be outliers also.

Li [15] suggested a method of detecting natural noise by assuming that the ratings given by a user on nearly correlated items should have similar scores. He identified users that had a high noise degree and removed these users from the dataset, increasing the accuracy of CFRS. This approach is limited to user level only and better approaches are needed that consider noise on all levels.

Toledo et al. [1] detects inconsistencies in the recommender system by characterizing users, items and ratings by their profiles into separate classes and then looking for contradictions among those profiles. He followed a re-prediction approach using Pearson's correlation coefficient to correct the identified noisy ratings. Yera et al. [14] used a fuzzy model along with re-prediction method for handling natural noise. Researchers have also used re-prediction, fuzzy profiling, global and local noise management approaches in group recommender systems.

Choudhary et al. [13] used a similar approach as of Toledo et al [1] for detection and correction of noisy ratings and generated recommendations using dice similarity coefficient between the active users along with optimized weights using particle swarm optimization (PSO). This method has limitations when the dataset is sparse and the optimal choice of PSO parameters is difficult.

In most of the re-prediction approaches the evaluation has been done on dense datasets. But, in sparse dataset, the Pearson's correlation coefficient and neighbourhood search method for prediction doesn't always give correct results. Therefore, instead of re-prediction, we follow a re-classification approach for users, items and ratings and then look for contradictions between those classes to detect noisy ratings.

2.3. Malicious Noise in Recommender Systems

Most of the CFRS rely on the opinions of the users for the items or the rating given by them and are vulnerable to the shilling attacks which can be manipulated to increase/decrease the sale of a target item by recommending it more/less. These shilling attacks are becoming a great threat to the recommender systems as they not only influence the performance of the recommender system but also, reduces the trust of the customer in the recommendation platform by misleading them.

There are many types of solutions to tackle shilling attacks for the CFRS and the most famous is by the detection of fake user profile. In this, we detect the malicious user by noticing the features of attack types. But, most of the models are limited to particular attack types only and majority of the methods detected anomaly user rather than intentional attacker. We propose to follow the detection of anomaly item directly as followed by Xia et al [21], which in turn is similar to detecting items attacked by fake users. The approach is based on the assumption that the item's intrinsic quality follows a uniform distribution and the rating distribution of items remain stable in absence of shilling attacks. In this way, this approach is independent of the attack types.

There are mainly two types of shilling attacks considering the attack intention: push attacks and nuke attacks [22]. Attacks whose objective is to increase the sales of the target items are called push attacks whereas the attacks whose objective is to decrease or reduce the sales of the target items are called nuke attacks.

2.4. Research Gap

Most of the research has either been done on either natural noise handling or malicious noise attacks but in our research, we try to consider and remove both the types of noise and then compare the results. We propose to use a re-classification approach instead of a re-prediction approach as recommended by Toledo et al [1]. For malicious attacks, we think to implement a dynamic time interval segmentation technique to detect the anomaly against shilling attacks by looking for abnormalities in item profiles. The same approach was followed by Xia et al [21].

To the best of our knowledge, no one has tried to remove both these types of noise in CFRS using individually focused methods for each type of noise. However, this report only contains the results and analysis for natural noise handling only, the shilling attack part is still to be implemented and evaluated on Movie Lens Dataset.

3. Handling Natural Noise in CFRS

We use the user-item rating values to detect and correct natural noise in recommender systems, in this way no additional information about users and items is needed. Users give noisy ratings either unintentionally (bad mood or lack of interest, etc) or intentionally (special cases) but these ratings may change his user profile and can generate recommendations which are not suitable for him.

Our approach for dealing with natural noise contains two steps:

1) Noisy ratings detection: Classification of user and item profiles into 4 different classes based on the ratings. Ratings themselves are classified as weak, average or strong and then we look for intra-class contradictions.

2) Noise Correction: The identified noisy ratings are corrected with the thresholds of the inter-class boundaries. Bag et al. [11] suggest following this instead of re-prediction.

3.1. Noisy Ratings Detection:

To identify the noisy ratings, users and items are characterized into weak, average, strong and uncertain classes based on the constraints presented in Table 1 and 2. Toledo et al. suggested thresholds to distinguish between the weak, average and strong ratings [1]. ku, vu, ki and vi are the threshold boundaries of user-classes and item-classes between weak-average and average-strong respectively.

Classes for users	Criteria
Weak user (Uw)	$\|Uw\| \geq \|Ua\| + \|Us\|$
Average user (Ua)	$\|Ua\| \geq \|Uw\| + \|Us\|$
Strong user (Us)	$\|Us\| \geq \|Ua\| + \|Uw\|$
Variable user (Uv)	*Otherwise*

Table 1. Method for Users classification

Classes for items	Criteria
Weak item (Iw)	$\|Iw\| \geq \|Ia\| + \|Is\|$
Average item (Ia)	$\|Ia\| \geq \|Iw\| + \|Is\|$

Strong item (Is)	$\|Is\| \geq \|Ia\| + \|Iw\|$
Variable item (Iv)	*Otherwise*

Table 2. Method for Items classification

So, first the user-ratings are classified into weak, average or strong based on these threshold boundaries and then the cardinalities of each classes for users and items are compared to classify them as described in the above tables. The complete algorithms can be seen as follows:

Algorithm 1: Classification of User's classes

Input : *User - item rating metric of a sparse and noise dataset*
Output : *Set of classified user's classes*
1 : $U_W = (\ ), U_a = (\ ), U_S = (\ ), U_V = (\ )$
2 : for *each user u* ***do***
3 : for *each item i* ***do***
*/ * k_u and v_u are the threshold boundaries of* user's *classes between weak – average and average – strong respectively * /*
4 : if $r(u,i) < k_u$ ***then***
5 : *add* $r(u,i)$ *to the set of* U_W
6 : else if $r(u,i) \geq k_u$ *and* $r(u,i) < v_u$ ***then***
7 : *add* $r(u,i)$ *to the set of* U_a
8 : else
9 : *add* $r(u,i)$ *to the set of* U_S
10 : end if
11 : end for
12 : end for
13 : for *each user u* ***do***
14 : if $\|U_W\| \geq \|U_a\| + \|U_S\|$ ***then***
15 : U_W *belongs to weak user's class*
16 : else if $\|U_a\| \geq \|U_W\| + \|U_S\|$ ***then***
17 : U_a *belongs to average user's class*
18 : else if $\|U_S\| \geq \|U_a\| + \|U_W\|$ ***then***
19 : U_S *belongs to strong user's class*
20 : else
21 : U_V *belongs to variable user's class*
22 : end if
23 : end for

Algorithm 2: Classification of Item's classes

Input : *User - item rating metric of a sparse and noise dataset*
Output : *Set of classified items - classes*
1 : $I_W = (\ ), I_a = (\ ), I_S = (\ ), I_V = (\ )$
3 : for *each item i* ***do***
2 : for *each user u* ***do***
*/ * k_i and v_i are the threshold boundaries of* item's *classes between weak – average and average – strong respectively * /*
4 : if $r(u,i) < k_i$ ***then***
5 : *add* $r(u,i)$ *to the set of* I_W
6 : else if $r(u,i) \geq k_i$ *and* $r(u,i) < v_i$ ***then***
7 : *add* $r(u,i)$ *to the set of* I_a
8 : else
9 : *add* $r(u,i)$ *to the set of* I_S
10 : end if
11 : end for
12 : end for
13 : for *each user i* ***do***
14 : if $\|I_W\| \geq \|I_a\| + \|I_S\|$ ***then***
15 : U_W *belongs to weak item's class*
16 : else if $\|I_a\| \geq \|I_W\| + \|I_S\|$ ***then***
17 : U_a *belongs to average item's class*
18 : else if $\|I_S\| \geq \|I_a\| + \|I_W\|$ ***then***
19 : I_S *belongs to strong item's class*
20 : else
21 : I_V *belongs to variable item's class*
22 : end if
23 : end for

Now, the next step is correction of these identified natural noise.

3.2. Noise Correction:

For the correction of natural noise, we look for the contradictions among the user, item and rating classes. If both the user and item classes are weak but the rating is strong, then replace the noisy rating with the threshold of weak-average class. If both user and item belong to average classes and the rating is either weak or strong then it is replaced with the mean of thresholds for weak-average and average-strong classes. At last, if both the user and item

belong to strong classes and the rating is weak then it is replaced with the threshold of average-strong class.

In this way, the noise correction is done and after that, the unknown ratings are predicted using user-user collaborative filtering and item-item collaborative filtering approaches using cosine similarities. The further analysis and evaluation are explained in the next section.

Algorithm 3: Noisy rating correction

Input : *User - item rating matrix of a sparse dataset including natural noise, classified user - item rating matrix*
Output : *Updated rating matrix after correcting the natural noise*
1 : $U_w = (\ ), I_w = (\ ), U_a = (\ ), I_a = (\ ), U_s = (\ ), I_s = (\ )$
2 : for *each rating r(u, i)* ***do***
3 : *if both u and i belongs to weak class & r(u, i) ≥ k* ***then***
4 : replace r(u,i) with k
5 : ***else if*** *both u and i belongs to average class, and r(u, i) < k or r(u, i) ≥ v* ***then***
6 : replace r(u,i) with (k+u)/2
7 : ***else if*** *both u and i belongs to strong class and r(u, i) < v* ***then***
8 : replace r(u,i) with v
9 : ***else***
10 : *unchanged*
11 : ***end if***
12 : end for

4. Validation Testbed

In the following sub-sections, the validation testbed for the experimental analysis is explained which includes the details of dataset and evaluation parameters.

4.1. Dataset

We have used **MovieLens** dataset**,** a very famous dataset in recommender system research. It is collected from the Group Lens data store [2]. The dataset contains a total of 1 million ratings given by 943 users and for 1682 items. The ratings range on a discrete scale from 1 to 5, where a rating of 1 indicates the lowest possible rating meaning user dislikes the movie and rating of 2 indicates user slightly prefers the movie, 3 indicates neutral behaviour and 4,5 represent strong preferences.

We follow the protocol suggested by Gunawardana and Shani in [3] for generating the training and test datasets. It proposes to select few users from the original dataset and then randomly hide few items for each user. The hidden items will represent the test set and the remaining ratings will be training set. While calculating the accuracy of the recommender system we only compare these ratings in the test set and the predicted ratings.

4.2. Prediction and Evaluation Metrics

We have used User based collaborative filtering and Item-based collaborative filtering for prediction and cosine similarity for generating the similarity between users and items. The algorithms' quality is calculated using Mean Squared Error (MSE) for all the predictions made in the test set.

$$sim(u,u') = cos(\theta) = \frac{\mathbf{r}_u \dot{\mathbf{r}}_{u'}}{|\mathbf{r}_u||\mathbf{r}_{u'}|} = \sum_i \frac{r_{ui} r_{u'i}}{\sqrt{\sum_i r_{ui}^2}\sqrt{\sum_i r_{u'i}^2}}$$

Fig 3. Cosine similarity formula

For user-based collaborative filtering, we predict that a user's u's rating for item i is given by the weighted sum of all other users' ratings for item i where the weighting is the cosine similarity between each user and the input user u.

$$\hat{r}_{ui} = \sum_{u'} sim(u,u') r_{u'i}$$

We must also normalize by the number of ru'i ratings:

$$\hat{r}_{ui} = \frac{\sum_{u'} sim(u,u') r_{u'i}}{\sum_{u'} |sim(u,u')|}$$

Fig 4. Rating Prediction formula

For item-based collaborative filtering, instead of user-user similarity values we use item-item similarities. We can attempt to improve our prediction MSE by only considering the top k users who are most similar to the input user (or, similarly, the top k items). We try to find the optimum value of k by trying out different values and then analysing the test errors.

5. Results

k	Noise Corrected User-based CF train MSE	Simple User-based CF train MSE	Noise Corrected User-based CF test MSE	Simple User-based CF test MSE
5	1.821979	1.875663	8.469821	8.952164
10	2.638975	2.691344	7.404366	7.721665
15	3.05267	3.121662	7.013152	7.271317
20	3.322753	3.396814	6.783077	7.023413
25	3.52128	3.609356	6.641782	6.883827
30	3.687529	3.779436	6.565137	6.800327
35	3.830741	3.922667	6.519378	6.74686
40	3.955262	4.052634	6.49333	6.718763
45	4.064607	4.166282	6.485966	6.696686
50	4.163331	4.26887	6.48514	6.681174
55	4.252474	4.36372	6.476548	6.67671
60	4.333391	4.450926	6.474075	6.67367
65	4.408836	4.531853	6.478096	6.678984
70	4.481774	4.608197	6.485592	6.686241
75	4.547994	4.680004	6.491986	6.694332

80	4.614255	4.747234	6.500494	6.70396
85	4.674508	4.810772	6.510176	6.710295
90	4.732572	4.872641	6.518374	6.717036
95	4.790864	4.932101	6.529863	6.73387
100	4.843891	4.988904	6.538085	6.745619

Table 3. User-based Collaborative Filtering Mean Squared Errors for Noise Corrected and Normal scenarios on training and test datasets for different k (top-k) values

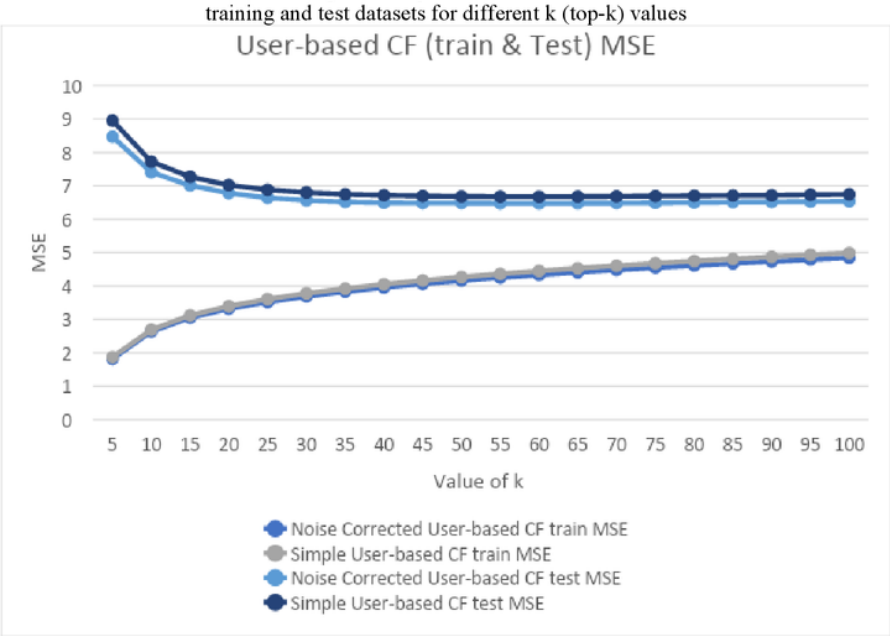

Fig 5. User-based CF Training and Test Errors for different k values

k	Noise Corrected Item-based CF train MSE	Simple Item-based CF train MSE	Noise Corrected Item-based CF test MSE	Simple Item-based CF test MSE
5	1.613517	1.691159	8.26332	8.745066
10	2.290939	2.365884	7.570153	7.930925
15	2.652164	2.749226	7.436371	7.796198
20	2.900794	2.99845	7.437813	7.774712
25	3.095299	3.183482	7.473872	7.812725
30	3.253437	3.348762	7.525974	7.893234
35	3.393325	3.488442	7.608783	7.96932
40	3.51117	3.605437	7.67508	8.052454
45	3.614892	3.712047	7.743558	8.131801
50	3.712686	3.81038	7.814856	8.209364
55	3.804354	3.901119	7.888687	8.280585
60	3.888919	3.985525	7.961512	8.354438
65	3.967432	4.070067	8.031378	8.430379
70	4.0418	4.146834	8.092228	8.495702
75	4.112744	4.220293	8.159288	8.558617
80	4.179169	4.290909	8.215012	8.623238
85	4.244049	4.358464	8.273361	8.685048
90	4.307907	4.427405	8.330321	8.747409
95	4.369562	4.492336	8.384995	8.806533
100	4.430462	4.555911	8.439378	8.864087

Table 4. Item-based Collaborative Filtering Mean Squared Errors for Noise Corrected and Normal scenarios on training and test datasets for different k (Top-k) values

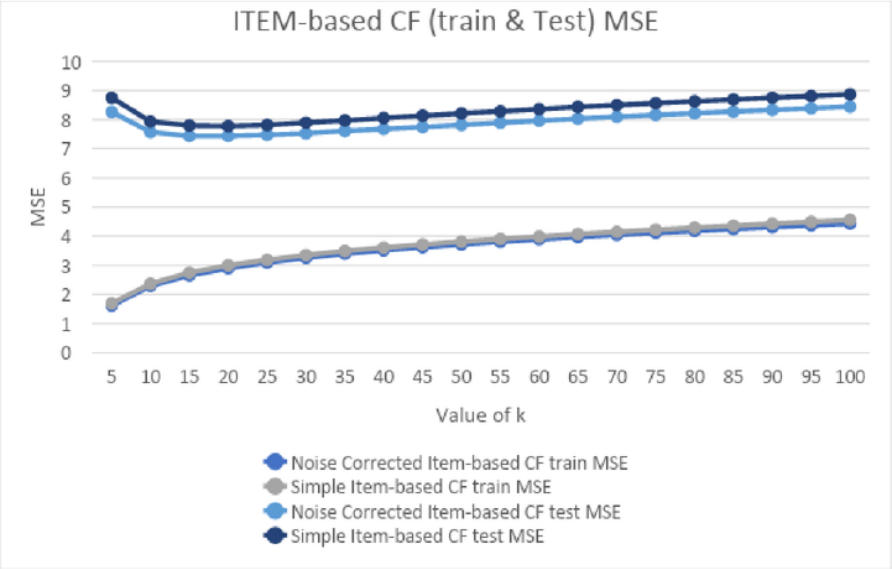

Fig 6. Item-based CF Training and Test Errors for different k values

6. Conclusion and Future Prospect (Handling Shilling Attacks in CFRS)

Till now, in this research, we explained the re-classification based approach for Natural Noise detection and correction. We used User-User and Item-Item based collaborative filtering approaches for making predictions for the unknown rating values using the modified and corrected user-item rating matrix which is free from any natural noise.

In the future, we plan to apply the malicious noise handling approaches as followed by Xia et al [21] and Cai et al [22]. We plan to follow a dynamic time interval segmentation technique based item anomaly detection to tackle the problem of shilling attacks. Using this approach can make the process independent of the attack type and looks for abnormalities in item profiles. After handling the shilling attacks, we will have a robust recommender model free of both natural and malicious noise and we expect to increase the accuracy in this way. In the end, we can also perform a detailed accuracy analysis using multiple datasets in order to validate the results thoroughly.

7. References

[1] R.Y. Toledo, Y.C. Mota, L. Martínez, Correcting noisy ratings in collaborative recommender systems, Knowledge-Based Systems 76 (2015) 96–108, https://doi.org/10.1016/j.knosys.2014.12.011.

[2] GroupLens, MovieLens datasets, https://grouplens.org/datasets/movielens, (2017), Accessed date: 30 December 2018.

[3] A. Gunawardana, Shani, G., "A survey of accuracy evaluation metrics of recommendation tasks," Journal of Machine Learning Research, vol. 10, pp. 2935-2962, 2009.

[4] G. Adomavicius, Tuzhilin, A., "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," IEEE Transactions on Knowledge and Data Engineering, vol. 17, pp. 734-749, 2005.

[5] K. J. B. Schafer, J.A., Riedl, J., "E-commerce recommendation applications," Data Mining and Knowledge Discovery, vol. 5, pp. 115-153, 2001.

[6] E. J. Castellano, Martínez, L. , "A Web-Decision Support System based on Collaborative Filtering for Academic Orientation. Case Study of the Spanish Secondary School.," Journal of Universal Computer Science, vol. 15, pp. 2786-2807, 2009.

[7] X. Amatriain, Pujol, J., Oliver, N. , "I like it... I like it not: Evaluating user ratings noise in recommender systems," presented at the 17th International Conference on User Modeling, Adaptation and Personalization (UMAP), 2009.

[8] M. P. O'Mahony, Hurley, N.J., Silvestre, G.C. , "Detecting noise in recommender system databases.," presented at the 11th ACM International Conference on Intelligent Users Interfaces (IUI) 2006.

[9] X. Amatriain, Pujol, J., Tintarev, N., Oliver, N., "Rate it again: Increasing recommendation accuracy by user re-rating.," presented at the 3rd ACM International Conference on Recommender Systems (RecSys), 2009.

[10] P.A. Chirita, W. Nejdl, C. Zamfir, Preventing shilling attacks in online recommender systems, Proceedings of the 7th Annual ACM International Workshop on Web Information and Data Management, NY, 2005.

[11] S. Bag, S. Kumar, A. Awasthi, M.K. Tiwari, A noise correction-based approach to support a recommender system in a highly sparse rating environment, Decision Support Systems 118 (2019).

[12] Lee, Yunkyoung, "RECOMMENDATION SYSTEM USING COLLABORATIVE FILTERING" (2015). Master's Projects. 439. DOI: https://doi.org/10.31979/etd.5c62-ve53

[13] Priyankar Choudhary,Vibhor Kant, Pragya Dwivedi, Handling Natural Noise in Multi Criteria Recommender System utilizing effective similarity measure and Particle Swarm Optimization, 7th International Conference on Advances in Computing & Communications, ICACC-2017, 22-24 August 2017, Cochin, India.

[14] Raciel Yera, Jorge Castro, Luis Martínez, A fuzzy model for managing natural noise in recommender systems, Applied Soft Computing 40 (2016).

[15] B. Li, Chen, L., Xingquan, Z., Chengqi, Z., "Noisy but non-malicious user detection in social recommender systems," World Wide Web, 2012.

[16] https://www.ethanrosenthal.com/2015/11/02/intro-to-collaborative-filtering/

[17] https://towardsdatascience.com/various-implementations-of-collaborative-filtering-100385c6dfe0

[18] http://files.grouplens.org/papers/FnT%20CF%20Recsys%20Survey.pdf

[19] https://towardsdatascience.com/collaborative-filtering-and-embeddings-part-1-63b00b9739ce

[20]https://medium.com/sfu-big-data/recommendation-systems-user-based-collaborative-filtering-using-n-nearest-neighbors-bf7361dc24e0

[21] Hui Xia, Bin Fang, Min Gao, Hui Ma, Yuanyan Tang, Jing Wen, A novel item anomaly detection approach against shilling attacks in collaborative recommendation systems using the dynamic time interval segmentation technique, Information Sciences 306 (2015).

[22] Yuanfeng Cai, Dan Zhu, Trustworthy and profit: A new value-based neighbor selection method in recommender systems under shilling attacks, Decision Support Systems 124 (2019).

BTP 16im30034

ORIGINALITY REPORT

30%	14%	25%	10%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Sujoy Bag, Susanta Kumar, Anjali Awasthi, Manoj Kumar Tiwari. "A noise correction-based approach to support a recommender system in a highly sparse rating environment", Decision Support Systems, 2019 Publication	4%
2	Raciel Yera Toledo, Yailé Caballero Mota, Luis Martínez. "Correcting noisy ratings in collaborative recommender systems", Knowledge-Based Systems, 2015 Publication	3%
3	scholarworks.sjsu.edu Internet Source	3%
4	www.ethanrosenthal.com Internet Source	2%
5	Hui Xia, Bin Fang, Min Gao, Hui Ma, Yuanyan Tang, Jing Wen. "A novel item anomaly detection approach against shilling attacks in collaborative recommendation systems using the dynamic time interval segmentation	2%

technique", Information Sciences, 2015
Publication

6	sinbad2.ujaen.es Internet Source	2%
7	Yuanfeng Cai, Dan Zhu. "Trustworthy and profit: A new value-based neighbor selection method in recommender systems under shilling attacks", Decision Support Systems, 2019 Publication	1%
8	Toledo, Raciel Yera, Yailé Caballero Mota, and Luis Martínez. "Correcting noisy ratings in collaborative recommender systems", Knowledge-Based Systems, 2015. Publication	1%
9	dl.acm.org Internet Source	1%
10	Priyankar Choudhary, Vibhor Kant, Pragya Dwivedi. "Handling Natural Noise in Multi Criteria Recommender System utilizing effective similarity measure and Particle Swarm Optimization", Procedia Computer Science, 2017 Publication	1%
11	Submitted to UI, Springfield Student Paper	1%
12	Submitted to University of Sheffield	

	Student Paper	1%
13	"Recommender Systems Handbook", Springer Nature, 2015 Publication	1%
14	Raciel Yera, Luis Martínez. "A recommendation approach for programming online judges supported by data preprocessing techniques", Applied Intelligence, 2017 Publication	<1%
15	Submitted to CSU, San Jose State University Student Paper	<1%
16	link.springer.com Internet Source	<1%
17	Recommender Systems, 2016. Publication	<1%
18	Submitted to Middle East Technical University Student Paper	<1%
19	Lecture Notes in Computer Science, 2012. Publication	<1%
20	Advances in Intelligent Systems and Computing, 2014. Publication	<1%
21	Submitted to University of Wolverhampton Student Paper	<1%

22	interactivesystems.info Internet Source	<1%
23	Liangmin Guo, Jiakun Liang, Ying Zhu, Yonglong Luo, Liping Sun, Xiaoyao Zheng. "Collaborative filtering recommendation based on trust and emotion", Journal of Intelligent Information Systems, 2018 Publication	<1%
24	open.uct.ac.za Internet Source	<1%
25	Lecture Notes in Computer Science, 2015. Publication	<1%
26	Submitted to King's College Student Paper	<1%
27	Raciel Yera Toledo, Yaile Caballero Mota, Milton Garcia Borroto. "A Regularity-Based Preprocessing Method for Collaborative Recommender Systems", Journal of Information Processing Systems, 2013 Publication	<1%
28	Adomavicius, G., and YoungOk Kwon. "Improving Aggregate Recommendation Diversity Using Ranking-Based Techniques", IEEE Transactions on Knowledge and Data Engineering, 2012. Publication	<1%

29	AKIHIRO YAMASHITA, HIDENORI KAWAMURA, KEIJI SUZUKI. "ADAPTIVE FUSION METHOD FOR USER-BASED AND ITEM-BASED COLLABORATIVE FILTERING", Advances in Complex Systems, 2011 Publication	<1%
30	Hongyun Cai, Fuzhi Zhang. "An Unsupervised Method for Detecting Shilling Attacks in Recommender Systems by Mining Item Relationship and Identifying Target Items", The Computer Journal, 2019 Publication	<1%
31	Submitted to Napier University Student Paper	<1%
32	Aristomenis S. Lampropoulos, George A. Tsihrintzis. "Chapter 2 A Survey of Approaches to Designing Recommender Systems", Springer Science and Business Media LLC, 2013 Publication	<1%
33	Submitted to University of Southampton Student Paper	<1%
34	HaoIrwinMichael R. MaKingLyu. "Effective missing data prediction for collaborative filtering", Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR 07 SIGIR 07, 2007	<1%

Publication

35	Bin Li, Ling Chen, Xingquan Zhu, Chengqi Zhang. "Noisy but non-malicious user detection in social recommender systems", World Wide Web, 2012 Publication	<1%
36	Gawesh Jawaheer, Peter Weller, Patty Kostkova. "Modeling User Preferences in Recommender Systems", ACM Transactions on Interactive Intelligent Systems, 2014 Publication	<1%
37	Submitted to Associatie K.U.Leuven Student Paper	<1%
38	Guénolé C.M. Silvestre. "Detecting noise in recommender system databases", Proceedings of the 11th international conference on Intelligent user interfaces - IUI 06 IUI 06, 2006 Publication	<1%

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		