

Research Journal of Pharmaceutical, Biological and Chemical Sciences

Assessment of Unfair User Rating in Multidisciplinary System.

Mareeswari V*, and Bala Prasanth P.

School of Information Technology and Engineering, VIT University, Vellore-632014.

ABSTRACT

The concept of E-Commerce is another milestone in the world of shopping of any products. As well as, in the medical field, ratings and reviews will assist to connecting the patients with right doctors and improvement in hospitals. Since the recommender system helps in predicting the likeliness of user towards any product. But the limitation it faces is the unfairness in the rating given by the users intentionally or unintentionally. This affects the result of prediction mechanisms. In recent years, several attack detection algorithms have been proposed to handle this issue. Unfortunately, their applications are restricted by various constraints. The proposed system will reduce the unfair rating problem based on the algorithm that detects malicious attackers and removes their ratings. This algorithm ensures that the recommendations given are reliable. This work can be applied for any multidisciplinary application based on the rating system.

Keywords: Unfair rating, beta protection, recommendation, attack detection.

**Corresponding author*

INTRODUCTION

Different software systems often need to exchange data with each other, and a Web service is a method of communication that allows two software systems to exchange this data over the internet. A recommender system is an extensive class of Web applications that involve predicting user responses to options based on the user preference in the past. They are a useful alternative to search algorithms since it helps the users discover items they might not have found by themselves. They are categorized into two:

- Content-based systems examine properties of the items recommended.
- Collaborative filtering systems recommend items based on similarity measures between users and/or items. The items recommended to a user are those preferred by similar users.

The recommender system is now getting vulnerable because of the problems faced by the filtering approaches. Taking this as advantage attackers injects malicious profiles to affect the actual rating and also affects the recommendations results. In the recent past several algorithms have been designed to detect the malicious attacks. The algorithms face lots of issues such as the inability to handle missing values in the user-item matrix, high false alarm rate, low detection rate, and some algorithms expect a balanced number of normal and malicious ratings to solve them but this scenario cannot be applied in the reality. Some algorithms are well on paperwork but fail to solve the issues in real time. The Beta protection solves these issues with the help of three phase algorithm.

The numbers of transactions that happen over the internet are growing exponentially each day. While picking a product/service users to have a lot of options. So selecting the appropriate product/service is really difficult for a user. Customer's decision to purchase a particular product solely depends on the rating available for the product. Now a lot of attackers are altering the rating to promote or demote the product's reputation. So now the correctness of rating is put to doubt. In order to filters, malicious contents a lot of algorithms are put forth. This paper implements an algorithm to improve the correctness of the ratings given in web services.

RELATED WORKS

The online rating systems prevailing in the market allows us to rate a product, service or other users. These ratings are manipulated by the attackers to promote or de-motivate a value of product/service/user. To identify the attackers as the first step that has to be done as generating real world data. To create data sets this project works on various algorithms. There is a rating challenge, new rating aggregation algorithm, unfair rating generators and attack models. These modules help us to generate a data set that is not available now. Because the above modules ensure on the fact that the data set that comes out is close to actual data in the ratings.

The rating is generated by the user as part of competition [8]. In this, the user is asked to provide a malicious rating in the system without being detected. Then the rater is awarded based the rate of success. In [1] a detection algorithm (signal based filtering) was adopted to pick the malicious raters. After the competition many observations were made: The attacks were mostly straight forward. The detector was able to pick the attack profiles easily. Only a few attackers adjusted their ratings manually and attacked the system. They were successful in achieving their objectives. But the above-mentioned categories of attackers were few in numbers.

A collaborative recommender using any of the common algorithms can be exploited by attackers without a great degree of knowledge of the system. There have been some recent research efforts aimed at detecting and reducing the effects of profile injection attacks. Several metrics for analyzing rating patterns of malicious users and algorithms designed specifically for detecting such attack profiles have been introduced. Developing a multi-strategy approach [2] to attack defense which including supervised and unsupervised classification approaches, time-series analysis, vulnerability analysis, and anomaly detection.

Attacks on recommender systems can affect the quality of the prediction for many users, resulting in decreasing overall user satisfaction with the system. The paper [3] proposed and investigated the use of statistical metrics for detecting patterns of shilling attackers in a recommender system. They also proposed the

use of an additional metric, RDMA, to measure a user's disagreement with the other users in the database, weighted by the inverse rating frequency of her rated items. They intend to further improve RDMA, as well as study other rating patterns for attackers.

It [4] proposes to use Neyman-Pearson statistical detection to identify attack profiles and show how to statistically model the standard attacks that have been proposed to date. This analysis shows that the success of the PCA-based detector is largely due to the unrealistic manner in which items are rated in attack profiles of standard attacks. With this realization, they show how it is possible to create an effective attack which is undetectable by the PCA detector with a simple modification of the Average attack. To address the detection of such obfuscated attacks, their model as an attacked dataset as a multivariate Gaussian mixture model and design supervised and unsupervised Neyman-Pearson detectors based on this model. These detectors significantly out-perform the PCA detector on obfuscated attacks.

The possibility of obfuscating attacks has been considered previously, where three different obfuscation strategies were proposed. These strategies represent ad hoc modifications of the standard attack models that an attacker might employ in order to avoid detection. No guiding principle is employed in the design of these obfuscations. An effective obfuscation strategy must try to minimize the statistical differences between genuine and attack profiles. They have seen that an attack profile that is undetectable by the PCA detector must reduce the differences in filter selection that are keys to the success of this detector.

In [4] have discussed profile injection attacks and developed statistical models for their detection. It is clear that attack and detection can result in an arms race scenario between the system manager and attacker. It is important to know the limits of this arms race and work an interesting theory. Nevertheless, their work shows that the optimal attacks are not random, but exploit explicit vulnerabilities in the recommendation algorithms. It is possible [5] to mount successful attacks against collaborative recommender systems without substantial knowledge of the system or users. The examination of the segment attack, a very effective reduced-knowledge attack, also demonstrated the vulnerability of the item-based algorithm, which was previously thought to be relatively robust. User's trust in a recommender system will, in general, be affected by many factors, and the trustworthiness of a system, its ability to earn and deserve that trust, is likewise a multifaceted problem. However, an important contributor to users' trust will be their perception that the recommender system really does what it claims to do, which is to represent evenhandedly the tastes of a large cross-section of users, rather than to serve the ends of a few unscrupulous attackers.

While current detection algorithms are able to use certain characteristics of shilling profiles to detect them, they suffer from low precision and require a large amount of training data. The authors [6] provide an in-depth analysis of shilling profiles and describe new approaches to detect malicious collaborative filtering profiles. In particular, it exploits the similarity structure in shilling user profiles to separate them from normal user profiles using unsupervised dimensionality reduction. It presents two detection algorithms; one based on PCA, while the other uses PLSA. Clearly, our work is only a starting point for building highly robust recommender systems. One direction would be developing algorithms, which treat attack profiles differently from noisy data and also use some distinctive features based on voting patterns for further increasing the precision of shilling detection. More theoretical work on the stability of popular recommender algorithms will add significantly to the state-of-the-art in personalization.

PROPOSED SYSTEM

The Beta protection algorithm comprises of three phases. In each phase, the abnormal users are being identified. Once all the phases are over then there will be abnormal users detected from each phase. If a user is identified as an abnormal user in at least two of the three phases then the user is declared as a malicious user and his/ her ratings will be removed from the overall system.

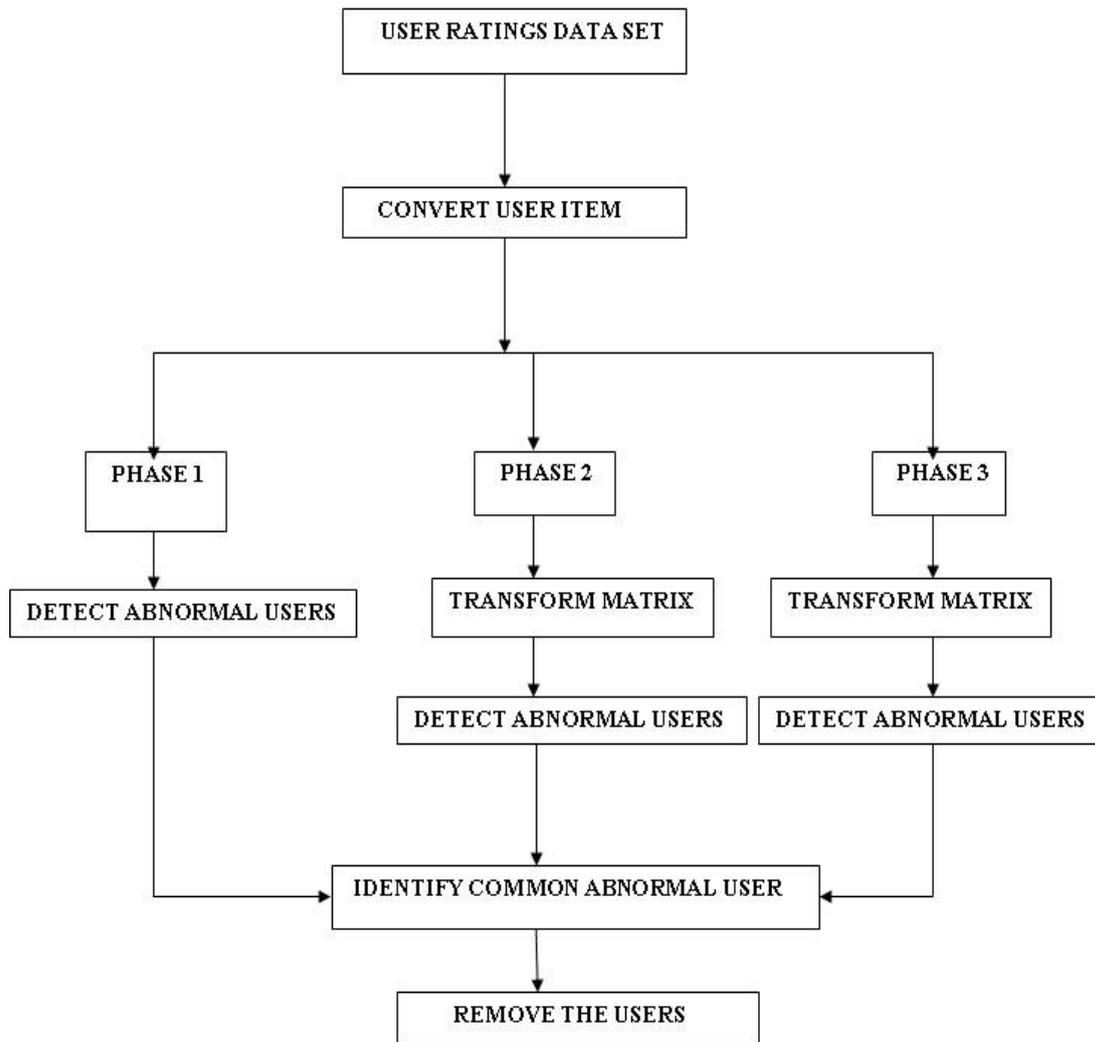


Figure 1 Beta Protection

Phase 1

In this phase, the alpha and beta values are calculated from adding the number of non-zero entries and the number of zero entries respectively. Then by using the Beta Distribution formula for two distinct parameters $F(p)$ will be calculated. Then expected value $E(p)$ is also calculated from the matrix. Then as mentioned in the algorithm, substitute the value of $Q_f=0.000064$ in the following formula:

$$Q_f = \int_{B_n}^1 f(p) \dots(1)$$

From that equation, identify the B_n value and if $B_n > E(p)$, then the user is consider as abnormal.

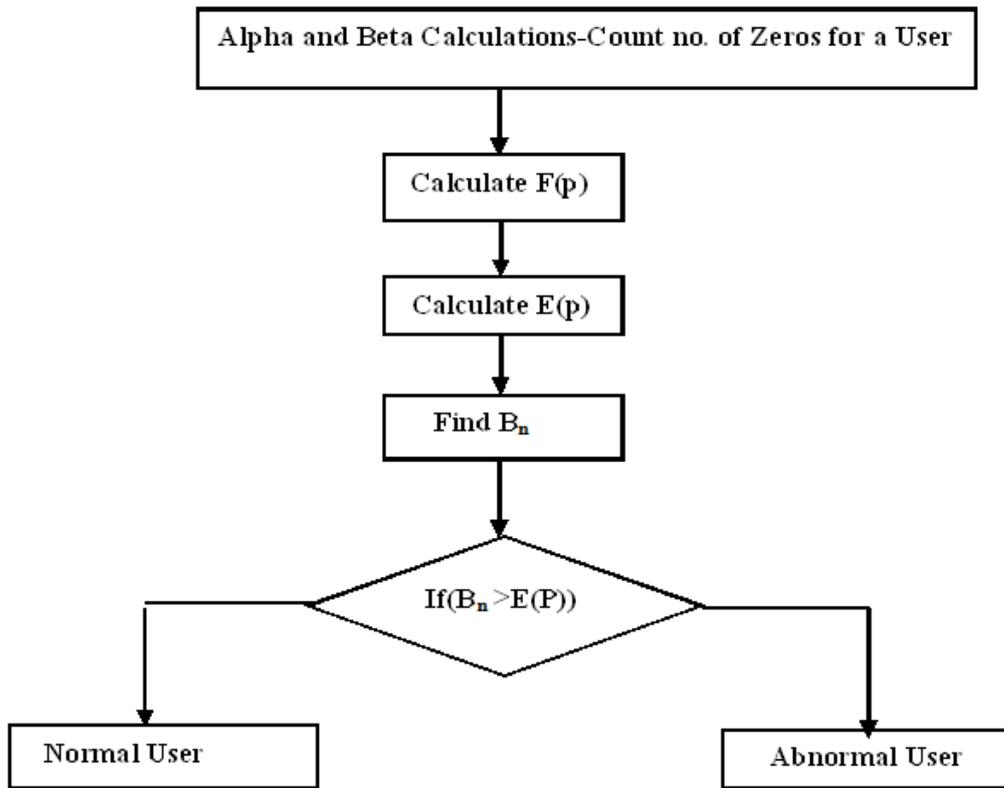


Figure 2 Phase 1

Phase 2

The phase two starts with the transforming the actual matrix into a new matrix with entries of values as m, n, l, h and 0. In this phase, the values of $F(p)$ and $E(p)$ are identified. Then as mentioned in the algorithm, substitute the value of $Q_s = 0.00142$ in the following formula:

$$Q_s = \int_0^{B_{lower}} f(p) \dots (2)$$

From that equation, identify the B_n value and if the value of $E(p) < B_{lower}$, then the user is judged abnormal.

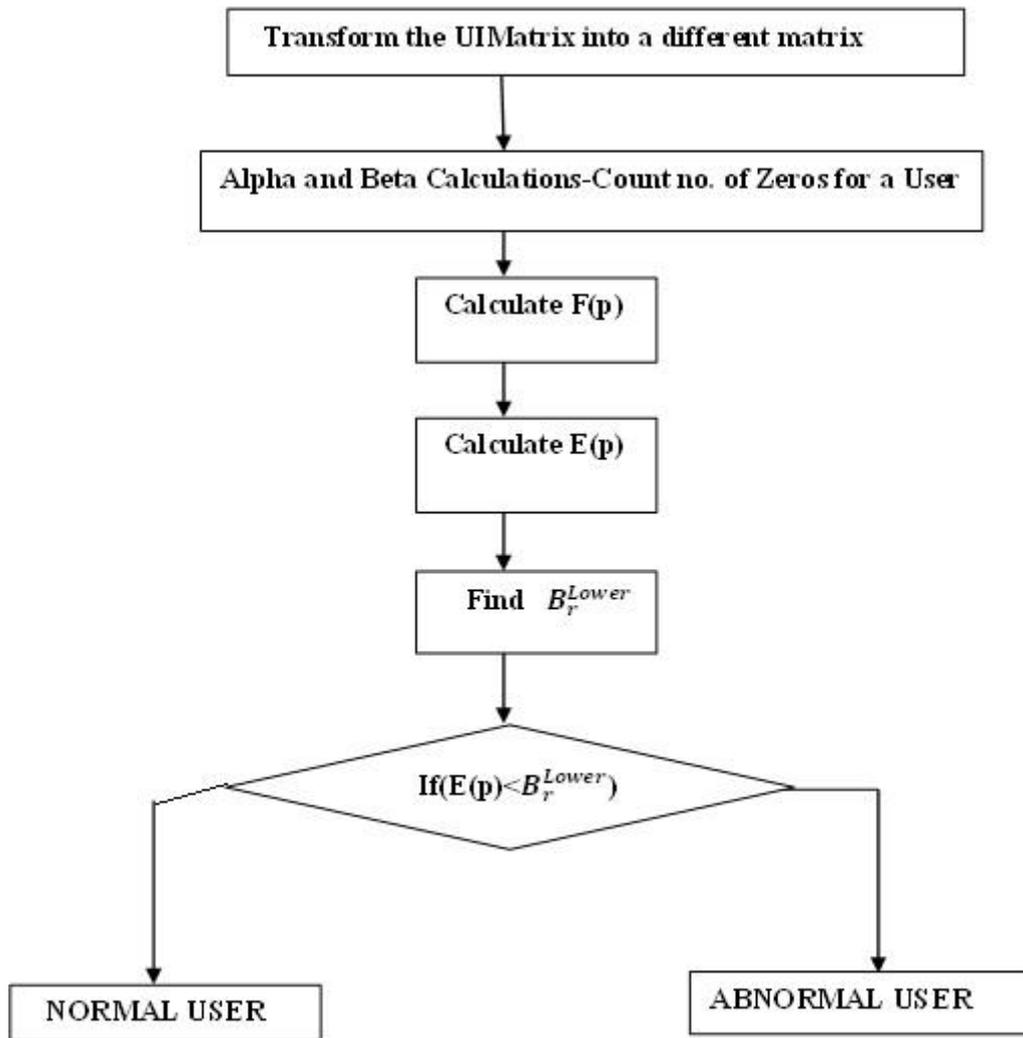


Figure 3 Phase2

Phase 3

The phase two starts with the transforming the actual matrix into a new matrix with entries of values as m ,n, l, h and 0.In this phase, the values of F(p) and the values of E(p) are identified. Then as mentioned in the algorithm, substitute the value of $Q_s = 0.00142$ in the following formula:

$$Q_s = \int_{B_{upper}}^1 f(p) \dots(3)$$

From that equation, identify the B_n value and if $B_{upper} < E(p)$, then the user is judged abnormal.

After the completion of three phases, if a user is identified as an abnormal user in at least two of the three phases then the user is declared as a malicious user and his/ her ratings will be removed from the overall system.

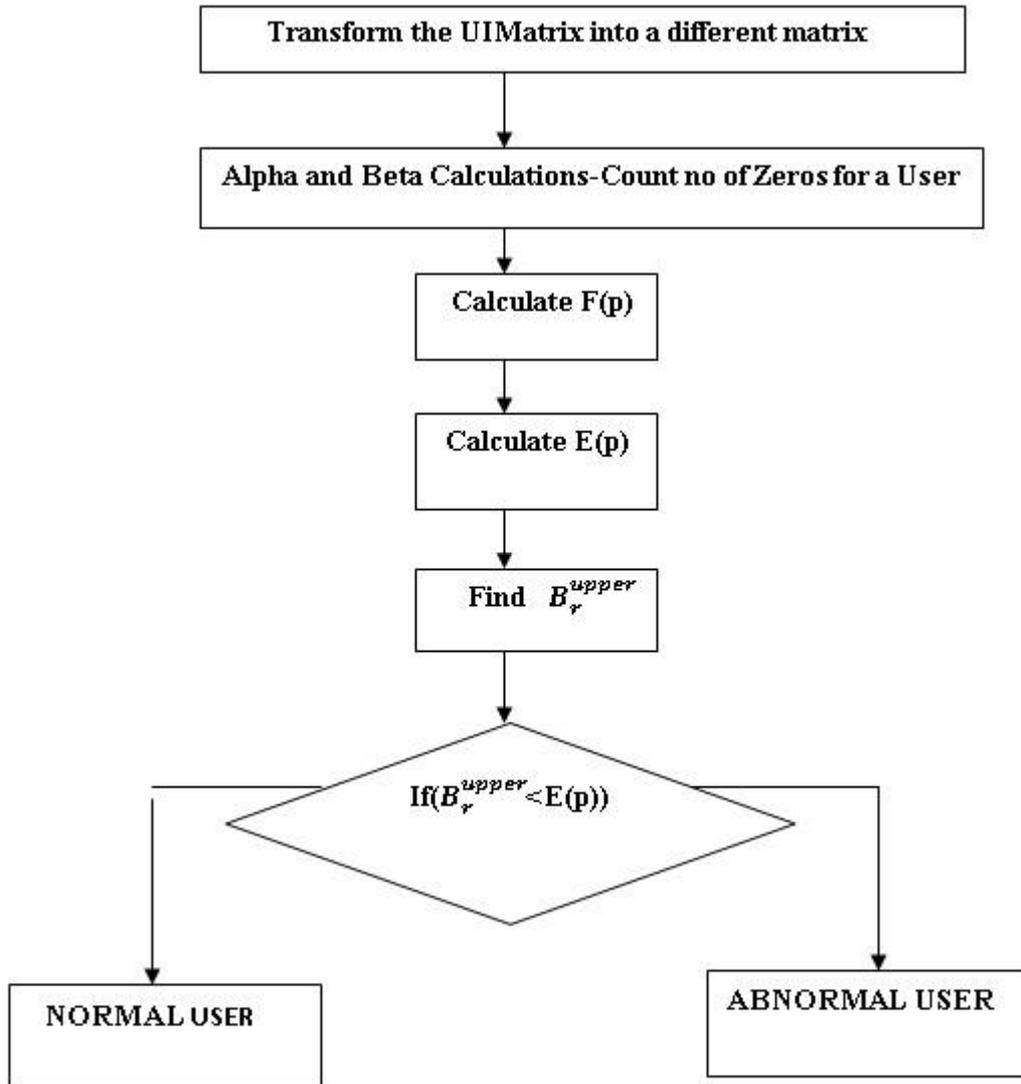
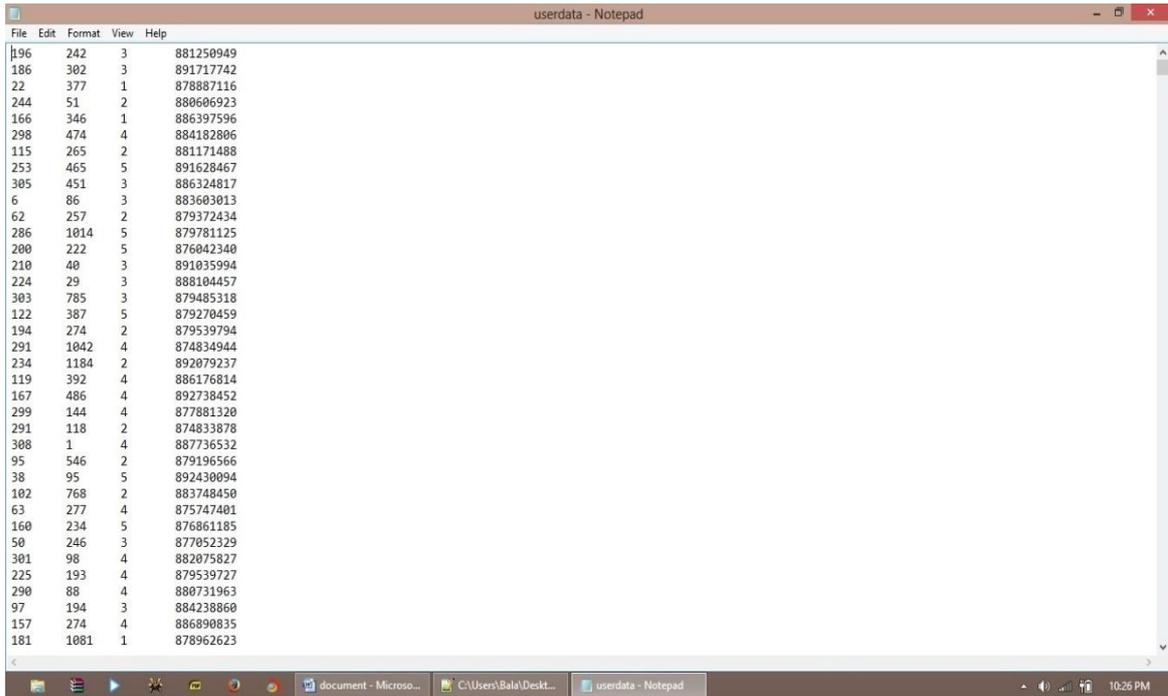


Figure 4 Phase3

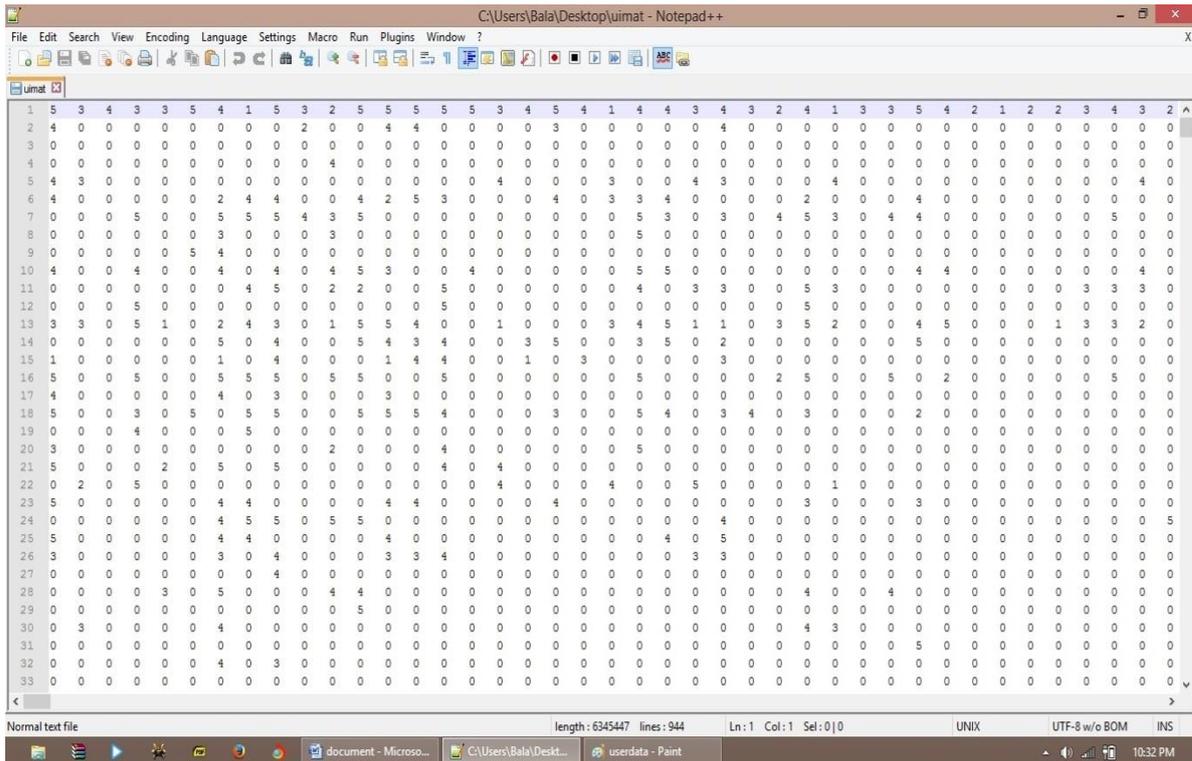
RESULTS AND DISCUSSION

This work has implemented using Java Programming. In figure 5 shows that the input file contains 100000 ratings for 1682 Items rated by 943 users. The figure 6 shows that the input file has converted in the form UI Matrix. In figure 7 shows that UI matrix has converted to transform matrix with the value of n, l, m and h based on the rating. The figures 8,9 and 10 shows that the result of Phase1, Phase2, and Phase3. Finally the abnormal and normal users are concluded by beta protection and displayed in figure 11.



File	Edit	Format	View	Help
196	242	3	881250949	
186	302	3	891717742	
22	377	1	878887116	
244	51	2	880606923	
166	346	1	886397596	
298	474	4	884182806	
115	265	2	881171488	
253	465	5	891628467	
305	451	3	886324817	
6	86	3	883603013	
62	257	2	879372434	
286	1014	5	879781125	
200	222	5	876042340	
210	40	3	891035994	
224	29	3	888104457	
303	785	3	879485318	
122	387	5	879270459	
194	274	2	879539794	
291	1042	4	874834944	
234	1184	2	892079237	
119	392	4	886176814	
167	486	4	892738452	
299	144	4	877881320	
291	118	2	874833878	
308	1	4	887736532	
95	546	2	879196566	
38	95	5	892430094	
102	768	2	883748450	
63	277	4	875747401	
160	234	5	876861185	
50	246	3	877052329	
301	98	4	882075827	
225	193	4	879539727	
290	88	4	880731963	
97	194	3	884238860	
157	274	4	886890835	
181	1081	1	878962623	

Figure 5 USERDATA INPUT FILE



File	Edit	Search	View	Encoding	Language	Settings	Macro	Run	Plugins	Window	?																																			
1	5	3	4	3	3	5	4	1	5	3	2	5	5	5	5	5	3	4	5	4	1	4	4	3	4	3	2	4	1	3	3	5	4	2	1	2	2	3	4	3	2					
2	4	0	0	0	0	0	0	0	0	0	2	0	0	4	4	0	0	0	3	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
4	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
5	4	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	3	0	0	0	0	4	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	
6	4	0	0	0	0	0	2	4	4	0	0	4	2	5	3	0	0	0	4	0	3	3	4	0	0	0	0	2	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	
7	0	0	0	5	0	0	5	5	5	4	3	5	0	0	0	0	0	0	0	0	0	0	0	0	5	3	0	3	0	4	5	3	0	4	4	0	0	0	0	0	0	0	5	0	0	
8	0	0	0	0	0	3	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
9	0	0	0	0	5	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
10	4	0	0	4	0	0	4	0	4	0	4	5	3	0	0	4	0	0	0	5	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	
11	0	0	0	0	0	0	4	5	0	2	2	0	0	5	0	0	0	0	0	4	0	3	3	0	0	5	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	3	3	0	
12	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
13	3	3	0	5	1	0	2	4	3	0	1	5	5	4	0	0	1	0	0	3	4	5	1	1	0	3	5	2	0	0	4	5	0	0	0	0	1	3	3	2	0	0	0	0		
14	0	0	0	0	0	5	0	4	0	0	5	4	3	4	0	0	3	5	0	3	5	0	2	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
15	1	0	0	0	0	0	1	0	4	0	0	1	4	0	0	1	0	3	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	5	0	0	5	0	0	5	5	0	5	5	0	0	5	0	0	0	0	0	5	0	0	0	2	5	0	0	5	0	2	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	
17	4	0	0	0	0	0	4	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
18	5	0	0	3	0	5	0	5	5	0	5	5	5	4	0	0	3	0	0	5	4	0	0	3	4	0	3	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
19	0	0	0	4	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
20	3	0	0	0	0	0	0	0	0	0	2	0	0	0	0	4	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	5	0	0	0	2	0	5	0	5	0	0	0	0	0	4	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	2	0	5	0	0	0	0	0	0	0	0	0	0	0	4	0	0	4	0	0	4	0	0	5	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	5	0	0	0	0	0	4	4	0	0	0	4	4	0	0	0	4	0	0	4	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	4	5	5	0	5	5	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0
25	5	0	0	0	0	0	4	4	0	0	0	4	0	0	0	0	0	0	0	0	0	0	4	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
26	3	0	0	0	0	3	0	4	0	0	0	3	3	4	0	0	0	0	0	0	0	0	0	3	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
28	0	0	0	0	3	0	5	0	0	0	4	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	3	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	4	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 6 UIMATRIX INPUT FILE

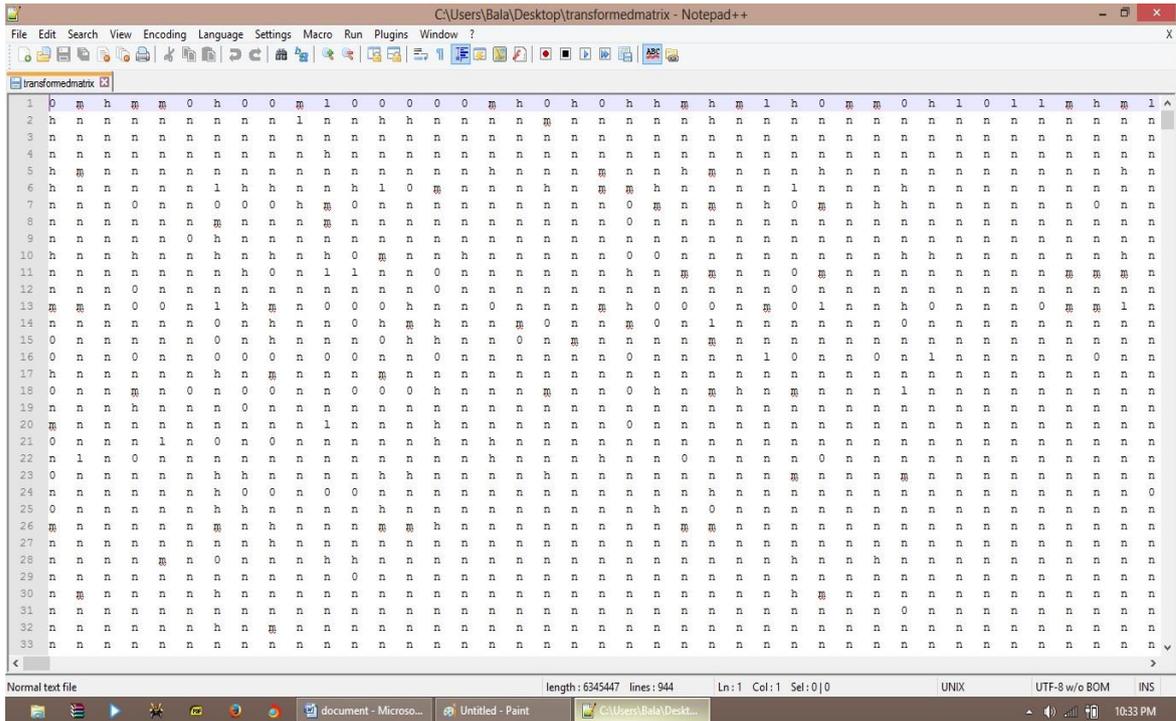


Figure 7 TRANSFORMED UIMATRIX

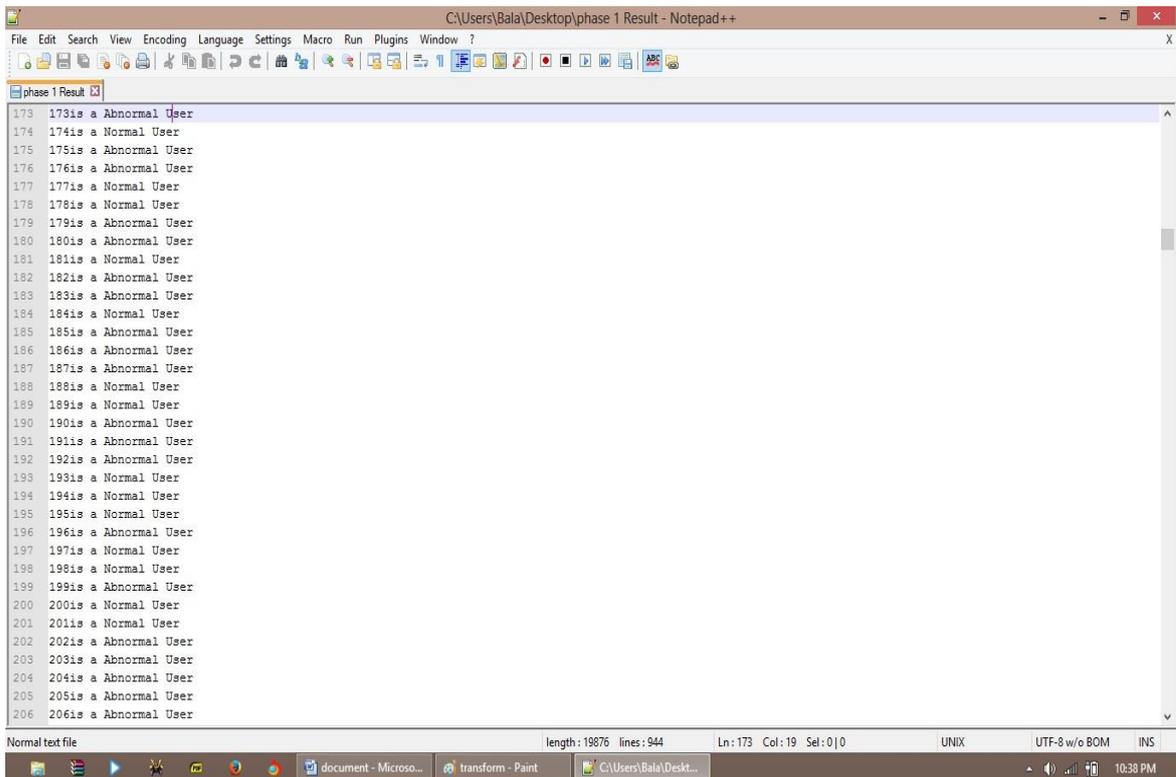
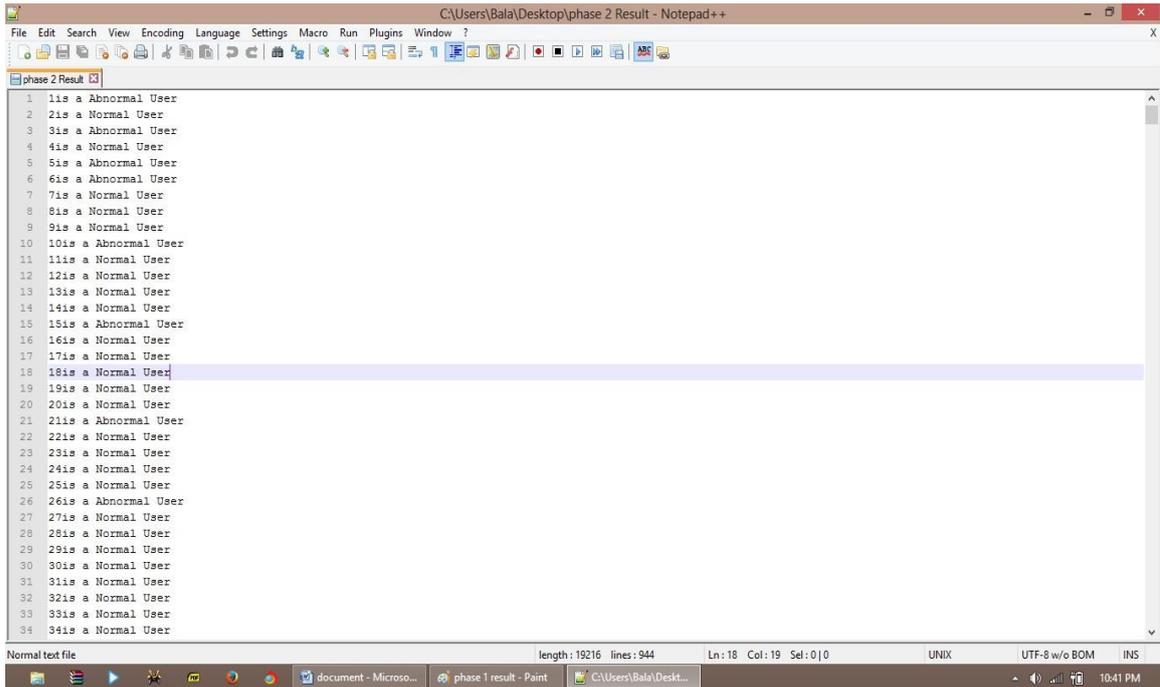


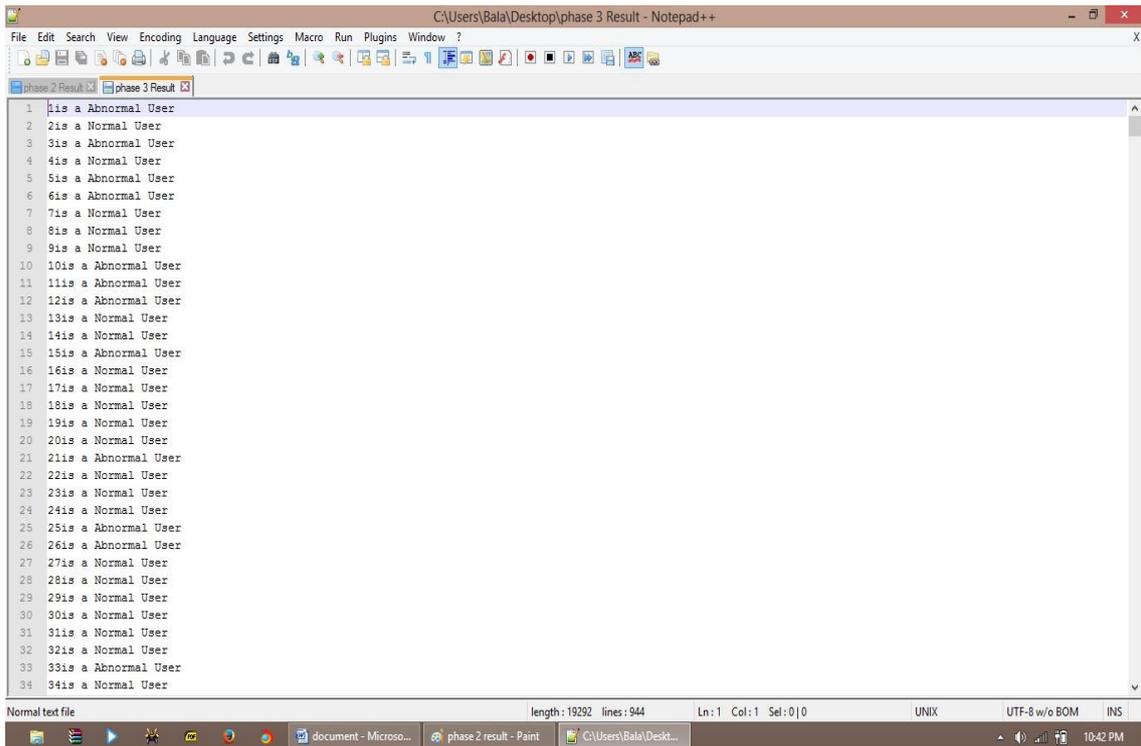
Figure 8 PHASE 1 RESULT



A screenshot of a Notepad++ window titled "C:\Users\Bala\Desktop\phase 2 Result - Notepad++". The window contains a list of 34 lines of text, each representing a user status. The status is either "Abnormal User" or "Normal User". The status alternates between "Abnormal" and "Normal" for every two lines, starting with "Abnormal" on line 1. The status for line 18 is highlighted in blue. The status bar at the bottom indicates "length: 19216 lines: 944", "Ln: 18 Col: 19 Sel: 0 | 0", "UNIX", and "UTF-8 w/o BOM INS".

```
1 11s a Abnormal User
2 21s a Normal User
3 31s a Abnormal User
4 41s a Normal User
5 51s a Abnormal User
6 61s a Abnormal User
7 71s a Normal User
8 81s a Normal User
9 91s a Normal User
10 101s a Abnormal User
11 111s a Normal User
12 121s a Normal User
13 131s a Normal User
14 141s a Normal User
15 151s a Abnormal User
16 161s a Normal User
17 171s a Normal User
18 181s a Normal User
19 191s a Normal User
20 201s a Normal User
21 211s a Abnormal User
22 221s a Normal User
23 231s a Normal User
24 241s a Normal User
25 251s a Normal User
26 261s a Abnormal User
27 271s a Normal User
28 281s a Normal User
29 291s a Normal User
30 301s a Normal User
31 311s a Normal User
32 321s a Normal User
33 331s a Normal User
34 341s a Normal User
```

Figure 9 PHASE 2 RESULT



A screenshot of a Notepad++ window titled "C:\Users\Bala\Desktop\phase 3 Result - Notepad++". The window contains a list of 34 lines of text, each representing a user status. The status is either "Abnormal User" or "Normal User". The status alternates between "Abnormal" and "Normal" for every two lines, starting with "Abnormal" on line 1. The status for line 18 is highlighted in blue. The status bar at the bottom indicates "length: 19292 lines: 944", "Ln: 1 Col: 1 Sel: 0 | 0", "UNIX", and "UTF-8 w/o BOM INS".

```
1 11s a Abnormal User
2 21s a Normal User
3 31s a Abnormal User
4 41s a Normal User
5 51s a Abnormal User
6 61s a Abnormal User
7 71s a Normal User
8 81s a Normal User
9 91s a Normal User
10 101s a Abnormal User
11 111s a Abnormal User
12 121s a Abnormal User
13 131s a Normal User
14 141s a Normal User
15 151s a Abnormal User
16 161s a Normal User
17 171s a Normal User
18 181s a Normal User
19 191s a Normal User
20 201s a Normal User
21 211s a Abnormal User
22 221s a Normal User
23 231s a Normal User
24 241s a Normal User
25 251s a Abnormal User
26 261s a Abnormal User
27 271s a Normal User
28 281s a Normal User
29 291s a Normal User
30 301s a Normal User
31 311s a Normal User
32 321s a Normal User
33 331s a Abnormal User
34 341s a Normal User
```

Figure 10 PHASE 3 RESULT

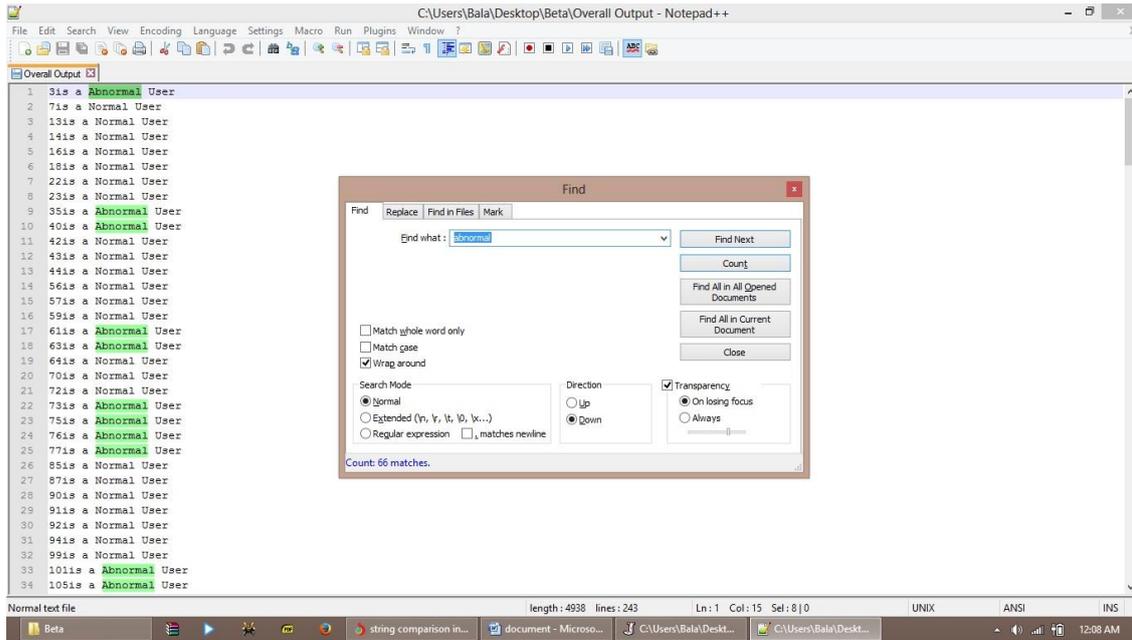


Figure 11 FINAL RESULT

Prior Probability

While calculating F(p) for a particular user, identify appropriate prior probability (p(x)) value in order to make the readings work fine. This probability always affects the way in which the final outcome that is been generated. The formula to identify f(p) is given by:

$$F(p | \alpha, \beta) = \frac{\text{gamma}(\alpha + \beta)}{\text{gamma}(\alpha) * \text{gamma}(\beta)} * p(x)^{\alpha - 1} * (1 - p(x))^{\beta - 1} \dots (4)$$

While refining the values of p(x)-prior probability, find that assigning yield good results p(x)=0.9. The following table will show the results that are been recorded.

Table 1: The different range of P(x) value and corresponding result

Sl.No	P(x) Value	Abnormal Users Detected
1	0.9	66 users
2	0.8	39 users
3	0.7	25 users
4	0.6	33 users

This table suggests that the ratings detected when the p(x) is 0.9 yields good result while comparing the other results.

Transformed Matrix

The transformed matrix from phase2 will have alpha and beta that is being calculated with respect to “l” as 2 and “h” as 4. This is because, exempts the least possible rating and highest ratings. If changes the alpha and beta calculation, then there will be the difference in the outcome.

The alpha and beta values are used in the calculations of equation (4) and in finding E(p).

$$E(p) = \alpha / (\alpha + \beta) \dots (5)$$

Table 2: The different alpha and beta values (phase 2) and its corresponding result

S. No	Alpha and Beta in Transformed Matrix	Output
1	Beta as 2 and Alpha as 4	66 Users
2	Beta as 3 and Alpha as 5	307 Users

The output suggests that assuming the value of alpha and beta as 2 and 4 yields effective results compared to the values of alpha -3 and beta -5 in the overall result.

CONCLUSION

The experimental results obtained from the study suggests that the fine tuning the values of prior probability, alpha and beta will always have greater effects on the overall outcome. This system detects the malicious users quite well. The important factor that comes into play is the quantile value that helps us identifying the value of B_n . This work proves that improve detection rate and reduce false alarm rate, track attack with lesser failure rate, ability to solve missing value problem and alter the benchmark values if necessary.

The system works fine while working on few thousands of data. But now the amount data that is being created is very large. So this system can be put use in real time once the range of data in which it works is increased. This is possible when some other factors are included in the system. This work can be applied for any multidisciplinary application.

REFERENCES

- [1] Chen-Yao Chung, Ping-Yu Hsu , Shih-Hsiang Huang, βP : *A novel approach to filter out malicious rating profiles from recommender systems*, C.-Y. Chung et al. Decision Support Systems, (2013).
- [2] Yefai yang, Lindsay Sun, Jin Ren and Qing Yang ,*Building trust in online rating through signal modeling*, 27th International Conference on Distributed Computing Systems Workshops (ICDCSW),2007.
- [3] R. Burke, B. Mobasher, C. Williams, R. Bhaumik, *Profile Injection Attack Detection for securing Collaborative Recommender Systems*, KDD '06 Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM Press, 2006.
- [4] P.A. Chirita, W. Nejdl, C. Zamfir, *Preventing shilling attacks in online recommender systems*, WIDM '05 Proceedings of the 7th Annual ACM International Workshop on Web Information and Data Management, ACM Press, 2005.
- [5] N.J. Hurley, Z. Cheng, M. Zhang, *Statistical attack detection*, RecSys '09 Proceedings of the third ACM conference on Recommender systems, ACM Press, 2009.
- [6] B. Mobasher, R. Burke, B. Bhaumil, C. Williams, *Towards trustworthy recommender systems: an analysis of attack models and algorithm robustness*, ACM Transactions on Internet Technology (2007).
- [7] Bhaskar Mehta, Wolfgang Nejdl, *Unsupervised strategies for shilling detection and robust collaborative filtering*, User Model (2009)
- [8] V.Mareeswari, E.Sathiyamoorthy, *A Survey on Trust in Semantic Web Services*, International Journal of Scientific & Engineering Research, Volume 3, Issue 2, Feb (2012)